IBM Library Server

Title: *COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide*
Document Number: *SC26-8071-01*
Build Date: *05/26/98 04:08:30* *Build Version:* *1.2*
Book Path: *C:\IBMZLIB\BOO\igyvi002.boo*

CONTENTS Table of Contents

IBM Library Server

CONTENTS Table of Contents

EDITION Edition Notice

Second Edition (June 1998)

This edition applies to:

IBM COBOL for VSE/ESA Version 1 Release 1 Modification 1 (Program Number 5686-068)

IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA Version 1 Release 1 (Program Number 5686-MLE)

and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader's comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation, W92/H3
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

IBM Library Server

CONTENTS Table of Contents

IBM Library Server

Title: *COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide*
Document Number: *SC26-8071-01*
Build Date: *05/26/98 04:08:30* *Build Version:* *1.2*
Book Path: *C:\IBMZLIB\BOO\igyvi002.boo*

COVER Book Cover

COBOL for VSE/ESA
VisualAge COBOL Millennium Language Extensions
for VSE/ESA

Installation and Customization Guide

Document Number SC26-8071-01

Program Number
5686-068
5686-MLE

File Number S370-34

IBM Library Server

Notices

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" in topic FRONT 1.

EDITION Edition Notice

Second Edition (June 1998)

This edition applies to:

IBM COBOL for VSE/ESA Version 1 Release 1 Modification 1 (Program Number 5686-068)

IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA Version 1 Release 1 (Program Number 5686-MLE)

and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader's comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation, W92/H3
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

IBM Library Server

CONTENTS Table of Contents

IBM Library Server

Tables

IBM Library Server

Figures

IBM Library Server

FRONT_1 Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Subtopics:

IBM Library Server

FRONT_1.1 Programming Interface Information

This *Installation and Customization Guide* documents information NOT intended to be used as Programming Interfaces of IBM COBOL for VSE/ESA or IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA.

IBM Library Server

FRONT_1.2 Trademarks

The following terms are trademarks of the IBM Corporation in the United States and/or other countries or both:

BookManager	IBM	Systems Application Architecture
CICS	S/370	VisualAge
CICS/VSE	SAA	VSE/ESA
DFSORT	SQL/DS	

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

IBM Library Server

FRONT_2 About This Book

This book is for systems programmers who are responsible for installing and customizing either or both of the products:

IBM COBOL for VSE/ESA
IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

for their location. This book provides information needed to plan for, install, and customize COBOL/VSE and VisualAge COBOL Millennium Language Extensions (MLE) for VSE/ESA under VSE/ESA. In addition, this book may help you to assess the value of COBOL/VSE and VisualAge COBOL MLE for VSE to your organization.

In this book, the generic term operating system is used when referring to VSE/ESA.

You should have a knowledge of COBOL/VSE and of your system's operating environment to use this book and ensure a successful installation of COBOL/VSE and VisualAge COBOL MLE for VSE.

Important

COBOL/VSE is distributed as two offerings:

- A full function offering, which includes Debug Tool/VSE
- An alternate function offering, which does not include Debug Tool/VSE

This book does **not** provide information needed to plan for, install, or customize Debug Tool/VSE. For information about installing Debug Tool/VSE, see *Debug Tool/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

FRONT_2.1 How to Read the Syntax Diagrams

Throughout this book, syntax for the compiler options is described using the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following table shows the meaning of symbols at the beginning and end of syntax diagram lines.

Symbol	**Indicates**
>>-	the syntax diagram starts here
->	the syntax diagram is continued on the next line
>-	the syntax diagram is continued from the previous line
-><	the syntax diagram ends here

Diagrams of syntactical units other than complete statements start with the >- symbol and end with the -> symbol.

- Required items appear on the horizontal line (the main path).

```
>>__STATEMENT__required item_____________________________________><
```

- Optional items appear below the main path.

```
>>__STATEMENT__ _______________ _____________________________________><
              |_optional item_|
```

- When you can choose from two or more items, they appear vertically in a stack.

If you **must** choose one of the items, one item of the stack appears on the main path.

```
>>__STATEMENT__ _required choice 1_ ____________________________><
              |_required choice 2_|
```

If choosing one of the items is optional, the entire stack appears below the main path.

```
>>__STATEMENT__ ___________________ _____________________________><
              |_optional choice 1_|
              |_optional choice 2_|
```

- An arrow returning to the left above the main line indicates an item

that can be repeated.

```
          <_,_____________
>>__STATEMENT____repeatable item_|_____________________________________><
```

A repeat arrow above a stack indicates that you can make more than one choice from the stacked items, or repeat a single choice.

- Keywords appear in uppercase letters (for example, PRINT). They must be spelled exactly as shown. Variables appear in all lowercase letters (for example, item). They represent user-supplied names or values.

- If punctuation marks, parentheses, arithmetic operators, or such symbols are shown, they must be entered as part of the syntax.

- Use at least one blank or comma to separate parameters.

IBM Library Server

FRONT_2.2 Using the Macro Planning Worksheets

The planning worksheets in this book will help you prepare to customize COBOL/VSE. By completing them, you will be able to easily identify those options that you wish to change from the IBM-supplied default values. You might also wish to use the worksheets as a source from which to actually customize the IBM-supplied default values.

The headings in each worksheet may differ slightly from each other. Refer to the following list of definitions for an explanation of each specific column heading. Worksheets are located on topics 1.2.2.2.1 and 1.2.3.21.1.

Option
The OPTION column identifies the options contained within a specific installation macro. This column represents the options exactly as they are in the macro.

Fixed
The FIXED column is used to identify the options that can not be overridden by an application programmer. Enter an asterisk [*] into the **Enter * for Fixed** only for those options that you want to be fixed.

Selection
The SELECTION column is for you to identify the value associated with each option. In the space provided, enter the value you want to assign to each option. Use the topic reference column to locate the specific information about the option that will assist you in selecting the appropriate value.

IBM-Supplied Default
The IBM-SUPPLIED DEFAULT column identifies the value supplied for the specified installation macro if the option is not altered. If the IBM-supplied default is identical to the value you desire for installation, you need not modify that option within that specific macro.

Syntax Description
The SYNTAX DESCRIPTION column identifies the topic in which you will find the syntax diagram for and more specific information about the option.

Once you have completed the worksheets, identify those options that are different from the IBM-supplied defaults. These are the items you must code in the installation macros. The worksheet entries have been positioned such that the order of the entries will remain consistent with the actual coding semantics.

IBM Library Server

FRONT_3 Summary of Changes

This section lists the major changes that have been made to the IBM COBOL for VSE/ESA product and this manual since Release 1. Technical changes are marked in the text by a change bar in the left margin.

Subtopics:

IBM Library Server

FRONT_3.1 Major Changes to COBOL/VSE

Subtopics:

IBM Library Server

| FRONT_3.1.1 Service Update to Release 1, June 1998

| • When used in conjunction with IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA, support for automatic date processing (century windowing for dates containing 2-digit years).

| • New language elements in support of automatic date processing:

| - DATE FORMAT clause in data description entries

| - Intrinsic functions:

| - DATEVAL
| - UNDATE
| - YEARWINDOW

| • New compiler options in support of automatic date processing:

| - DATEPROC/NODATEPROC
| - YEARWINDOW

| • INTDATE, added as an installation option by PTF, is now a standard compiler option, not just an installation option.

| • New date intrinsic functions to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - DATE-TO-YYYYMMDD
| - DAY-TO-YYYYDDD
| - YEAR-TO-YYYY

| • Extension of the ACCEPT statement to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - ACCEPT FROM DATE YYYYMMDD
| - ACCEPT FROM DAY YYYYDDD

IBM Library Server

FRONT_3.2 Major Changes to the Documentation

Subtopics:

IBM Library Server

| FRONT_3.2.1 Second Edition, June 1998

| • Information needed to plan for and install IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA has been included in this manual.

| • Information about the following new COBOL/VSE installation options has been added:

| - DATEPROC
| - INTDATE
| - YRWINDOW

IBM Library Server

1.0 Planning for, Installing, Customizing, and Maintaining COBOL/VSE

Subtopics:

IBM Library Server

1.1 Chapter 1. Planning to Install COBOL/VSE

This chapter provides information for planning the installation of COBOL/VSE. For information about planning the installation of VisualAge COBOL MLE for VSE, see "Planning for, Installing, and Maintaining VisualAge COBOL MLE for VSE" in topic 2.0. This chapter includes information on:

- What you receive with COBOL/VSE
- Ensuring that you have the required machine configuration
- Ensuring that you have enough storage for COBOL/VSE
- Choosing required and optional programs
- Considerations before installing

Subtopics:

IBM Library Server

1.1.1 What You Receive with COBOL/VSE

COBOL/VSE is distributed in two forms:

- full function offering, including Debug Tool/VSE
- alternate function offering

The COBOL/VSE licensed program package is distributed on tape containing the necessary material for installation.

Product Identification

Table 1. IBM COBOL for VSE/ESA Components and Component Level Codes (CLC)

Component ID	CLC	Component Description
5686-068-00	18M	COBOL/VSE Base
5686-068-01	18N	COBOL/VSE US English Language Feature
5686-068-02	18O	COBOL/VSE Japanese Language Feature

Basic Machine-Readable Material: You receive COBOL/VSE basic machine-readable material on one of these distribution media:

Table 2. Basic Material: Program Tape(s)

Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
6250 tape	5801 5811 5821	1	COBOL/VSE Release 1	unlabeled
3480 cart.	5802 5812 5822	1	COBOL/VSE Release 1	unlabeled
1/4" tape	5804 5814 5824	1	COBOL/VSE Release 1	unlabeled

The distribution media contains all the programs and data you need to install COBOL/VSE.

Table 3. Program Tapes: File Content

File	Description
1	Header file containing the COBOL/VSE copyright statement
2	Backup file ID COB.BASE...1.1.0 followed by an MSHP System History File
3	COBOL/VSE product with US uppercase English
4	Header file containing the COBOL/VSE copyright statement
5	Backup file ID COB.ENU....1.1.0 followed by an MSHP System History File
6	COBOL/VSE US English language feature
7	Header file containing the COBOL/VSE copyright statement
8	Backup file ID COB.JPN....1.1.0 followed by an MSHP System History File
9	COBOL/VSE Japanese language feature
10	Null (Tape Mark)
11	End of backup (EOB) record
12	Null (Tape Mark)

IBM Library Server

1.1.2 Optional Material

There is no optional machine-readable material for COBOL/VSE.

IBM Library Server

1.1.3 Basic Unlicensed Publications

The table below lists basic unlicensed publications for COBOL/VSE. When you order COBOL/VSE, you receive one of each of these publications.

Table 4. COBOL/VSE Unlicensed Publications

Publication Title	Order Number
COBOL/VSE Licensed Program Specifications	GC26-8069
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

1.1.4 Optional Unlicensed Publications

The table below lists optional unlicensed publications for COBOL/VSE. You can order one free copy of an optional unlicensed publication by specifying its feature number.

Table 5. IBM COBOL for VSE/ESA Publications

Publication	Order Number	Feature number
COBOL/VSE Diagnosis Guide	SC26-8528	7170
COBOL/VSE Language Reference	SC26-8073	7167
COBOL/VSE Reference Summary	SX26-3834	7169
COBOL/VSE Migration Guide	GC26-8070	7168
COBOL/VSE Programming Guide	SC26-8072	7166
COBOL/VSE General Information	GC26-8068	7165

IBM Library Server

1.1.5 Ensuring That You Have the Required Machine Configuration

COBOL/VSE can be installed and run in a virtual storage environment on system configurations that support the operating system releases listed in "Choosing Required and Optional Licensed Programs" in topic 1.1.7.

The following table lists the machine requirements for installing and using COBOL/VSE.

Table 6. COBOL/VSE Hardware Requirements

Item	Machine Requirements
Installation	1. A system processor that can support the required operating system. 2. Enough storage on DASD to hold the COBOL/VSE program product. See "Ensuring That You Have Enough Storage for COBOL/VSE" in topic 1.1.6 for minimum requirements.
Compilation	1. Any processor that supports the required operating system. 2. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is required only when the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD). 3. The SYSADAT file is required when the ADATA option is specified. 4. I/O devices for use by the compiler.
Run-Time Machine Requirements	1. Any processor supported by the required operating system. 2. I/O devices used by the object program during the run.
Devices Supported	1. All IBM devices supported by Sequential Access Method (SAM) and Virtual Storage Access Method (VSAM) under VSE/ESA can be used by object programs produced by the COBOL/VSE compiler when used with the LE/VSE Library. 2. All IBM devices supported under CICS are supported by COBOL/VSE when running under CICS.

IBM Library Server

1.1.6 Ensuring That You Have Enough Storage for COBOL/VSE

COBOL/VSE requirements for virtual and auxiliary storage vary, depending on whether you are installing the product, compiling a program, or running a program. The sections that follow describe how COBOL/VSE uses storage.

For information on run-time storage needs, see *LE/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

1.1.6.1 DASD Storage Required for Installation

When installing COBOL/VSE, you must provide DASD storage for:

- VSE Librarian library
- System history file

Table 7 lists the minimum DASD storage that COBOL/VSE requires for a VSE Librarian library. This does not include storage for other licensed programs installed in the library.

The MSHP System History File for the COBOL/VSE component requires about one cylinder of 3380 or equivalent DASD space.

Allow 10% to 15% extra storage for future enhancements and service updates.

Table 7. Minimum library storage required

LIBR BLKS(1)	3350 CYL	3375 CYL	3380 CYL	3390 CYL	FBA 9345 CYL	FBA BLKS
6700	21	23	15	14	16	13330

Note:

1. One library block equals 1 kilobyte (1024 bytes)

1.1.6.2 Storage Required for Compilation

The COBOL/VSE compiler requires a minimum of 760K bytes of virtual storage.

The virtual storage required depends on the size and content of your COBOL program. The COBOL/VSE compiler is designed to take advantage of systems with large virtual storage.

The following auxiliary storage is required by the COBOL/VSE compiler:

1. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07

2. IJSYS05 work file for the LIB compiler option

3. The SYSADAT file for the ADATA compiler option

4. SYSLST, SYSLNK and SYSPCH, if the appropriate options are in effect and if these files are going to be routed to a DASD, tape, or other auxiliary storage

5. SYSIPT input containing the COBOL program

1.1.7 Choosing Required and Optional Licensed Programs

COBOL/VSE runs under VSE/ESA with the required licensed programs listed in Table 8 and optional licensed programs listed in Table 9. **All licensed programs should be installed with the minimum release listed or with any subsequent release.**

Table 8. Required Licensed Programs for COBOL/VSE

Required Licensed Program	Minimum Release	Program Number
One of:		
VSE/ESA	Version 1 Release 4	5750-ACD
VSE/ESA	Version 2 Release 1	5690-VSE
LE/VSE(1)	Release 4	5686-094
High Level Assembler/MVS & VM & VSE	Release 1	5696-234

Note:

1. If you want to run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option), the PTFs that resolve the APARs shown Table 25 in topic 2.1.2.3 must be applied to the LE/VSE components you have installed.

Table 9. Optional Licensed Programs Supported by COBOL/VSE

Optional Licensed Program	Minimum Release	Program Number
BookManager Read	Release 2 is required to view softcopy documentation	73F6-023
CICS/VSE	Version 2 Release 3	5686-026
DFSORT/VSE(1)	Version 3 Release 1	5746-SM3
DL/I DOS/VS	Release 10	5746-XX1
DOS/VS Sort/Merge (VSE 1.4 only)	Version 2 Release 5	5746-SM2
SQL/DS	Version 3 Release 4	5688-103

Note:

1. If you want to sort dates with two-digit years using the century windowing capability provided by DFSORT/VSE, you require DFSORT Version 3 Release 2 with PTF UN99635, or a later release of DFSORT/VSE.

Subtopics:

IBM Library Server

1.1.7.1 National Language Support

Two language features are available with COBOL/VSE: the Japanese Language Feature and the US English Language Feature (mixed-case English). In order to receive compiler error messages and listing headers in Japanese, mixed-case English, or both, you must order the appropriate language feature(s).

Uppercase US English is supplied on the base tape, and does not require a language feature.

IBM Library Server

1.1.8 Considerations Before Installing

Before you begin installing COBOL/VSE, review the following considerations.

Subtopics:

IBM Library Server

1.1.8.1 Choosing the Language Feature You Want

In addition to uppercase English compiler messages, COBOL/VSE provides two optional language features, the Japanese Language Feature and the US English Language Feature (mixed-case English). Before you begin to install COBOL/VSE, you should determine if you want to install either or both of these language features, or if you will use the uppercase English compiler messages supplied in the base product.

To install the US English Language Feature use COB.ENU....1.1.0, and to install the Japanese Language Feature use COB.JPN....1.1.0.

1.1.8.2 Understanding the Installation Tools

- **MSHP Definition for COBOL/VSE:** When this product is prepared for initial shipping it will be from a VSE/ESA Version 1 Release 4 or later release environment. The product is shipped with an MSHP history file for each component of COBOL/VSE.

- **Maintain System History Program (MSHP):** MSHP helps you to control and record the installation of, and changes to, system software in a VSE operating environment. To install and maintain COBOL/VSE you must use MSHP, which records all system maintenance details in the system history file. MSHP also applies preventive and corrective service, through the use of functional control statements. For detailed information about MSHP, see *VSE/ESA System Control Statements*.

- **VSE Librarian program (LIBR):** Ensure that you are familiar with the VSE Librarian program (LIBR) which will be used to define the libraries in which COBOL/VSE will be installed.

IBM Library Server

1.1.8.3 If VS COBOL II is already Installed

Some modules supplied with COBOL/VSE have the same names as modules in the VS COBOL II sublibrary. If VS COBOL II is already installed on your system, COBOL/VSE cannot be installed in the same sublibrary.

IBM Library Server

1.1.8.4 Checking Service Updates

Before installing COBOL/VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 10. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLVSE110	06818M
COBOLVSE110	06818N
COBOLVSE110	06818O

1.2 Chapter 2. Planning to Customize COBOL/VSE

This chapter provides the following information for planning the customization of COBOL/VSE:

- Making changes after installation--why customize?
- Planning to modify compiler option default values
- Planning to place compiler phases in the Shared Virtual Area (SVA)
- Planning to create an additional reserved word table

These topics provide the information you need to plan the customization of COBOL/VSE. For information on how to make the changes to COBOL/VSE that are covered in this chapter, see Chapter 4, "Customizing COBOL/VSE" in topic 1.4.

This chapter also contains worksheets that help you to plan modifications to the IBM-supplied default values within macros. An explanation of the planning worksheets in this book is in "Using the Macro Planning Worksheets" in topic FRONT_2.2.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

IBM Library Server

1.2.1 Making Changes after Installation--Why Customize?

When you install COBOL/VSE you receive IBM-supplied defaults for compiler options and phases, and the reserved word table. You might want to customize COBOL/VSE to meet the application programming needs at your site.

After COBOL/VSE is installed, you can make the following changes:

- Modify the compiler option default values.
- Modify the compiler phases residency values.
- Make the compiler options fixed.
- Create additional reserved word tables.

The sections that follow discuss the planning you must do to customize COBOL/VSE for your site.

IBM Library Server

1.2.2 Planning to Modify Compiler Option Default Values

Compiler option defaults, and the required load locations for compiler phases, are set in the IGYCOPT macro as shown in Table 11 in topic 1.2.2.2.1 and Table 13 in topic 1.2.3.21.1. The default options module, IGYCDOPT, is link-edited with AMODE (31) and RMODE (ANY) during installation.

The IGYCOPT macro has a dual purpose: it allows you to select and fix the compiler options defaults, and to specify which compiler phases are in the SVA. This means that you can accept the IBM-supplied compiler option values you receive when you install COBOL/VSE, or you can modify them to suit your site. You can also choose whether or not your application programmers will be able to replace these options.

Subtopics:

IBM Library Server

1.2.2.1 Why Make Compiler Options Fixed?

COBOL/VSE can aid in setting up your site's programming standards. For example, many sites select APOST or QUOTE as their preferred compiler option for literal delimiters, but have no easy way to enforce its use. In COBOL/VSE, the IGYCOPT macro can be used to specify that an option may not be changed or replaced at compile time; that is, the option is fixed. At compile time, an attempt to replace a fixed option will not be allowed and will result in a diagnostic message with a nonzero compiler return code.

When certain options are fixed for consistent use, there might be times when special conditions require the ability to bypass such a fixed option. This can be done by assembling a temporary copy of the IGYCOPT macro with different parameters. At compile time, by selectively using a LIBDEF JCL statement specifying a sublibrary containing the required IGYCDOPT phase, you can bypass the fixed option. For example, if you select the OPT (OPTIMIZE) option to be fixed (indicating you always want the COBOL compiler to generate optimized object code), and then need to exempt an application from this requirement, you must reassemble the IGYCOPT macro after removing the asterisk parameter from the option. Then place the resulting IGYCDOPT phase in a temporary sublibrary to be accessed using the LIBDEF JCL statement at compile time.

Subtopics:

IBM Library Server

1.2.2.2 Syntax Format for Modifying Compiler Options and Phases

If you plan to modify compiler option values and compiler phases, use the IGYCOPT syntax format shown in Figure 1. The IBM-supplied default values are shown both on the planning worksheets and immediately following the syntax diagrams.

Compiler options and phases, and their defaults, are discussed in the following sections. You should review these options and phases, and their default values to determine the values most suitable for your applications.

IGYCOPT format

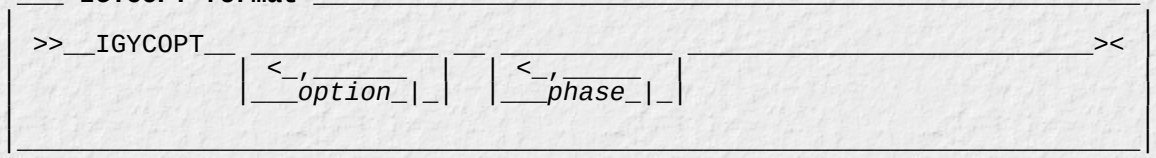


Figure 1. Syntax Format for IGYCOPT Compiler Options and Phases Macro

Subtopics:

- 1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

IBM Library Server

1.2.3 Planning to Place Compiler Phases in the SVA

You might want to make some phases resident in the SVA. in order to minimize the search (path length) for them when the COBOL/VSE compiler is run, or to allow the compiler phases to be shared.

Subtopics:

1.2.3.1 Why Place the Compiler Phases in SVA

All compiler phases with the exception of the run dump phases (IGYCRDPR and IGYCRDSC) and the reserved word utility (IGY8RWTU) are eligible for placement in the SVA. The SVA (shared virtual area) is an area of storage that is the same for each partition. Because it is the same space for all users, information stored there can be shared and does not have to be loaded into the partition GETVIS. By sharing the information, more space is made available for the compiler work area.

All compiler phases except those listed in Table 12 have RMODE(ANY) and AMODE(ANY). When phases are loaded into the SVA, VSE will load RMODE(24) phases in low SVA (SVA-24) and RMODE(31) phases in high SVA (SVA-31).

Table 12. RMODE and AMODE Exceptions for Compiler Phases

Phase	RMODE	AMODE
IGYCLIB0	24	ANY
IGYCRCTL	24	ANY
IGYCSIMD	24	ANY
IGYCRWTU	24	24

The IGYCOPT macro indicates where each compiler phase will be loaded--either inside (IN) or outside (OUT) the partition GETVIS. By placing compiler phases in the SVA, the compiler has more storage available for the user's program.

If you indicate that a phase will not reside in the partition GETVIS, you must ensure that you actually place the phase in the SVA. (This information is used by the compiler to determine how much storage to leave for the system to load compiler phases in the partition GETVIS). For a description of how to place the phase in the SVA, see *VSE/ESA System Control Statements*.

The four phases recommended to be placed in the SVA area are:

IGYCRCTL because it is resident in the user partition throughout compilation

IGYCSIMD because it is resident in the user partition throughout compilation

IGYCPGEN because it is one of the largest compiler phases

IGYCSCAN because it is one of the largest compiler phases

Select any or all compiler phases to be placed in the SVA based on frequency of concurrent use and phase size. If your site seldom uses the compiler, there might be no advantage to installing any phases in the SVA. However, if there are frequent compilations and enough SVA storage is available, making the entire compiler resident might be advantageous. If enough SVA storage is not available, priority should then be given to IGYCRCTL and IGYCSIMD, the two phases that are always resident in the user partition during compilation, and to IGYCPGEN and IGYCSCAN, the largest compiler phases.

Another advantage of placing compiler phases in the SVA is that, at compile time, the initialization logic allocates, in the user partition, a storage block large enough to contain the largest phase ***not*** resident in the SVA. Minimizing the space allocation for any given user partition size means more space for the compilation process (that is, larger programs can be compiled within a given user partition), and possibly a more efficient compile. The IGYCPGEN and IGYCSCAN compiler phases are about 250K bytes larger than the next largest compiler phase.

Some phases supplied with COBOL/VSE have the same names as phases supplied with VS COBOL II. Therefore, if you decide to place any COBOL/VSE phases in the SVA, you should ensure that there are no VS COBOL II phases in the SVA at the same time.

IBM Library Server

1.2.3.2 Compiler Phases and Their Defaults

You can indicate where each compiler phase will be loaded in relation to the partition by specifying either IN or OUT. See "Why Place the Compiler Phases in SVA" in topic 1.2.3.1, for more information on why you might or might not want to change these defaults.

IN indicates that the compiler phase will be loaded into the partition from a sublibrary available at compile time. The compiler will allow storage for the phase from the value specified in the SIZE option.

Even though IN is specified for a compiler phase, it still can be placed into the SVA. However, the compiler control phase will ensure that the main storage area reserved for compiler phases is large enough to contain the largest phase for which IN is specified. This will cause some storage to be unused.

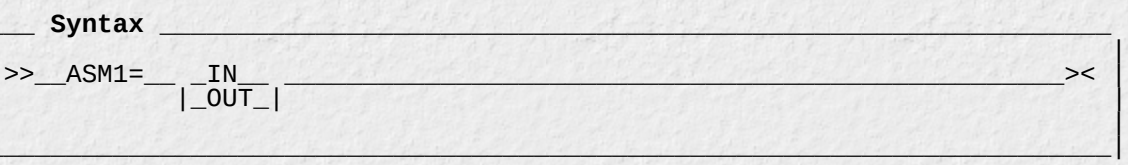
OUT indicates that the compiler phase will not be loaded into the partition from the sublibrary, and therefore must reside in the SVA.

IBM Library Server

1.2.3.3 IGYCASM1

IGYCASM1 is the Assembly 1 phase. It determines the object module storage, allocates the permanent and temporary registers, and optimizes addressability for data and procedure references. It also creates object text for data areas.

Syntax

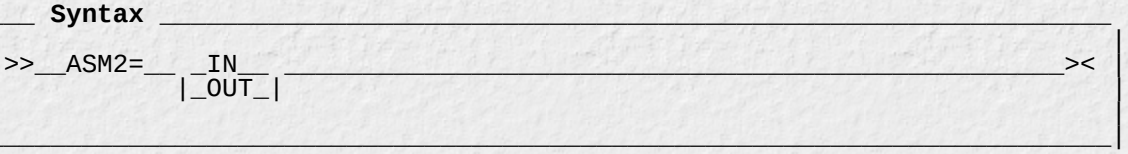


IBM Library Server

1.2.3.4 IGYCASM2

IGYCASM2 is the Assembly 2 phase. It completes preparation of the object program and creates object text, listings, punch data sets, and tables for the debugging feature.

Syntax



IBM Library Server

1.2.3.5 IGYCDIAG

IGYCDIAG is the diagnostic phase that processes E-form text and generates compiler diagnostics for source program errors. It includes IGYCDIAG plus the following message modules: IGYCxx$D, IGYCxx$1, IGYCxx$2, IGYCxx$3, IGYCxx$4, IGYCxx$5, and IGYCxx$8, where xx is EN, UE, or JA.

Syntax

IBM Library Server

1.2.3.6 IGYCDMAP

IGYCDMAP is the mapping phase. It prepares text for output requested by the MAP option.

Syntax

IBM Library Server

1.2.3.7 IGYCFGEN

IGYCFGEN is the file generation phase. It generates the control blocks for the FDs and SDs defined in the program.

Syntax

IBM Library Server

1.2.3.8 IGYCINIT

IGYCINIT is the initialization phase. It does housekeeping to prepare for running of the processing phases.

Syntax

IBM Library Server

1.2.3.9 IGYCLIBO

IGYCLIBO is the copy, system dependent phase. It provides system dependent services for COPY processing.

Syntax

IBM Library Server

1.2.3.10 IGYCLIBR

IGYCLIBR is the copy phase. It processes library source text and checks the syntax of the COPY, BASIS, and REPLACE statements.

Syntax

IBM Library Server

1.2.3.11 IGYCLSTR

IGYCLSTR is the source listing phase. It prints the source listing with embedded cross-reference and diagnostic information.

Syntax

IBM Library Server

1.2.3.12 IGYCMSGT

IGYCMSGT represents the header text table and diagnostic message level tables. There is no IGYCMSGT phase. To place the header text table and diagnostic message level tables in the SVA, specify the following phases: IGYCxx$R, IGYCLVL0, IGYCLVL1, IGYCLVL2, IGYCLVL3, and IGYCLVL8, where xx is EN, UE, or JA.

Syntax

IBM Library Server

1.2.3.13 IGYCOPTM

IGYCOPTM is the optimizer phase. It restructures the PERFORM statements and eliminates duplicate calculations.

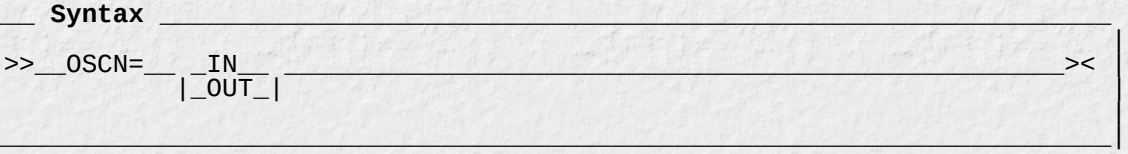
Syntax

IBM Library Server

1.2.3.14 IGYCOSCN

IGYCOSCN is the option scanning phase. It determines the default options, processes the EXEC PARM options, and processes the PROCESS (CBL) statements.

Syntax



IBM Library Server

1.2.3.15 IGYCPGEN

IGYCPGEN is the procedure generation phase. It supplies code for all procedure source verbs.

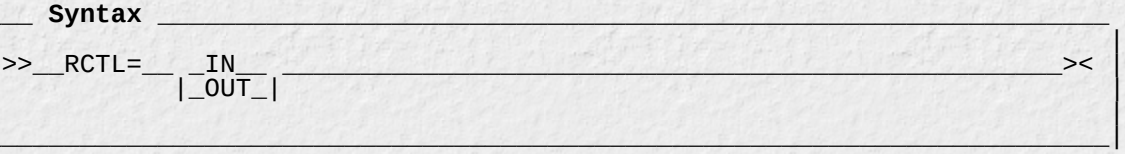
Syntax

IBM Library Server

1.2.3.16 IGYCRCTL

IGYCRCTL is the resident control phase. It establishes the size of compiler common and working storage, and performs initialization of program common storage.

Syntax



IBM Library Server

1.2.3.17 IGYCRWT

IGYCRWT is the normal reserved word table.

Syntax

IBM Library Server

1.2.3.18 IGYCSAW

IGYCSAW is the Systems Application Architecture (SAA) reserved word table.

Syntax

IBM Library Server

1.2.3.19 IGYCSCAN

IGYCSCAN is the scanning phase. It performs syntax and semantic analysis of the source program and translates the source to intermediate text.

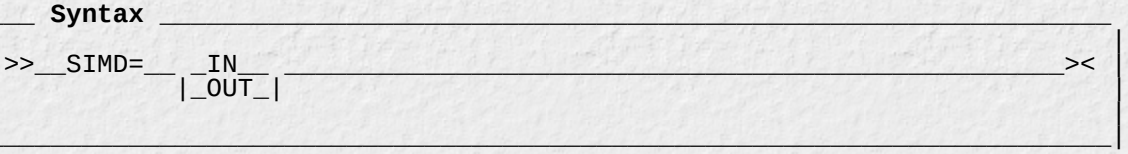
Syntax

IBM Library Server

1.2.3.20 IGYCSIMD

IGYCSIMD is the system interface phase for the COBOL/VSE compiler. This phase is called by all other compiler phases to perform system-dependent functions.

Syntax



IBM Library Server

1.2.3.21 IGYCXREF

IGYCXREF is the cross-reference phase. It sorts user-names and procedure-names in EBCDIC collating sequence.

Syntax

Subtopics:

- 1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

IBM Library Server

1.2.4 Planning to Create an Additional Reserved Word Table

You are provided with a default reserved word table when you install COBOL/VSE. A CICS-specific reserved word table is provided as an alternate reserved word table (See "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2). You can create additional reserved word tables after installation. (During compilation, the value of the WORD compiler option determines which reserved word table is used).

Subtopics:

IBM Library Server

1.2.4.1 Why Create Additional Reserved Word Tables?

You can create additional reserved word tables to do the following:

- Translate the reserved words into another language, such as French or German.
- Prevent application programmers from using a particular COBOL/VSE instruction, such as GO TO.
- Control the use of nested programs.
- Flag COBOL verbs without a run-time function in CICS, such as READ/WRITE.

IBM Library Server

1.2.4.2 Controlling Use of Nested Programs

You can allow source programs to access all VS COBOL II Release 3.0 and later NOCMPR2 features, except those related to nested programs, by modifying the reserved word table. This is accomplished through the INFO and RSTR control statements.

IBM Library Server

1.2.4.3 Reserved Word Tables Supplied with COBOL/VSE

Three reserved word tables come on the installation tape: the default reserved word table, the CICS reserved word table, and the SAA reserved word table.

Subtopics:

IBM Library Server

1.2.5 COBOL/VSE Compiler Options

This section describes those compiler options whose default values you can change. The notes that accompany some of the descriptions provide additional information about these options, such as how they interact with other options during compilation. This information may assist you in making decisions regarding the default values appropriate for your installation. For more information on using the compiler options, see the *COBOL/VSE Programming Guide*.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

IBM Library Server

1.2.5.1 Specifying COBOL Compiler Options

When you specify compiler options in the IGYCOPT macro, both the option name and its value must be specified in uppercase. If the option name is not specified in uppercase, both the option name and its value will be ignored and the default value will be used. No error message will be issued. If only the option value is not in uppercase, an error message will be issued indicating that an invalid option value has been specified.

IBM Library Server

1.2.5.2 Options in Support of the COBOL 85 Standard

The following option values are required to conform to the COBOL 85 Standard:

ADV=YES	FLAGMIG=NO	NUMPROC=NOPFD or MIG
CMPR2=NO	FLAGSAA=NO	SEQ=NO
DATEPROC=NO	INTDATE=ANSI	TRUNC=STD
DBCS=NO	LIB=YES	WORD=NO
DYNAM=YES	LITCHAR=QUOTE	ZWB=YES
FASTSRT=NO	NUM=NO	

Note: The term "COBOL 85 Standard" is used in this book to refer to the combination of the following standards:

1. ISO 1989:1985, Programming Languages - COBOL.

   ISO 1989/Amendment 1, Programming Languages - COBOL - Amendment 1: Intrinsic Function Module.

2. X3.23-1985, American National Standard for Information Systems - Programming Language - COBOL.

   X3.23a-1989, American National Standard for Information Systems - Programming Language - Intrinsic Function Module for COBOL.

IBM Library Server

1.2.5.3 Conflicting Compiler Options

Specifying certain compiler option values can create conflicts with other compiler options. Table 14 will help you resolve possible conflicts between compiler options. If you do specify conflicting compiler option values, you will receive a nonzero return code when attempting to assemble the IGYCOPT macro.

Table 14. Conflicting Compiler Options

Compiler Option	Conflicts with:
CMPR2=NO	FLAGMIG=YES
CMPR2=YES	DATEPROC=YES DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
COMPILE=NOC	LIST=YES OFFSET=YES OPT=STD or OPT=FULL TEST=(other than NO) VBREF=YES
DATEPROC=YES	CMPR2=YES FLAGSAA=YES FLAGSTD=(other than NO)
DBCS=YES	CMPR2=YES FLAGMIG=YES FLAGSTD=(other than NO)
FLAGMIG=YES	CMPR2=NO DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
FLAGSAA=YES	ADV=NO CMPR2=YES DATEPROC=(other than NO) DYNAM=NO FLAGMIG=YES FLAGSTD=(other than NO) LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
FLAGSTD=(other than NO)	ADV=NO CMPR2=YES DATEPROC=(other than NO) DBCS=YES DYNAM=NO FLAGMIG=YES FLAGSAA=YES LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD

	SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
LIST=YES	OFFSET=YES
OFFSET=YES	LIST=YES
OPT=STD or OPT=FULL	TEST=(other than NONE for hook location)
TEST=(other than NONE for hook-location suboption)	OPT=STD or OPT=FULL
WORD=xxxx	FLAGSTD=(other than NO)

IBM Library Server

1.2.5.4 Compiler Options Syntax and Descriptions

The following syntax diagrams describe each modifiable compiler option. The text below each diagram describes the effect of selecting a specific parameter.

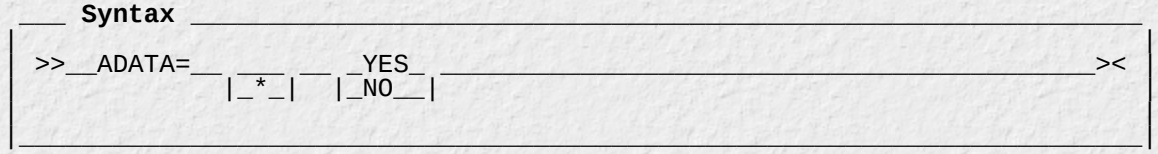
Notice the absence of the DUMP option. Unless changed at compile time, DUMP is always set to NODUMP. This option is not for general use; it is only used at the request of an IBM representative.

Note: The DECK and OBJECT compiler options may only be specified at compile time, and are therefore not described in the following sections. For more information, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.5 ADATA

Syntax



Default
ADATA=NO

YES
Specifies that COBOL/VSE associated data is to be collected and placed in the Associated Data file defined by the SYSADAT DLBL statement.

NO
Specifies that no associated data is to be collected.

Notes:

1. The ADATA option can only be specified at invocation via the option list on the PARM field of JCL.

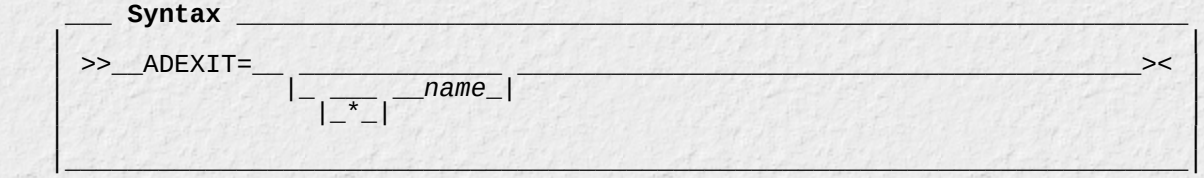
2. Selection of the Japanese language option might result in DBCS characters written records in the Associated Data file.

3. Specification of NOCOMPILE(W|E|S) might stop compilation prematurely, resulting in a loss of specific Associated Data Records.

4. Specification of INEXIT will prohibit identification of the compilation source file for the Associated Data file.

IBM Library Server

1.2.5.6 ADEXIT



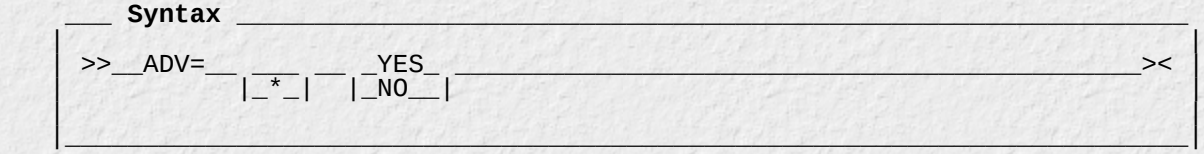
Default
No exit specified.

name
Identifies the name of a module, to be used with the EXIT option, that will be loaded and called by the compiler to monitor the associated data records written by the compiler to the Associated Data file. For more specific information, see the *COBOL/VSE Programming Guide*

Note: This is a read-only exit.

IBM Library Server

1.2.5.7 ADV



Default
ADV=YES

YES
Instructs the compiler to add one byte to the record length for the printer control character. This option may be useful to programmers who use WRITE...ADVANCING in their source files. The first character of the record does **not** have to be explicitly reserved by the programmer.

NO
Instructs the compiler not to adjust the record length for WRITE...ADVANCING. The compiler uses the first character of the specified record area to place the printer control character. The application programmer must ensure that the record description allows for this additional byte.

Notes:

1. ADV=YES conforms to the COBOL 85 Standard. With ADV=YES, the record length on the physical device is one byte larger than the record description length in the source program.

2. If the record length for the output file is not defined in the source code, COBOL ensures that it is appropriately set.

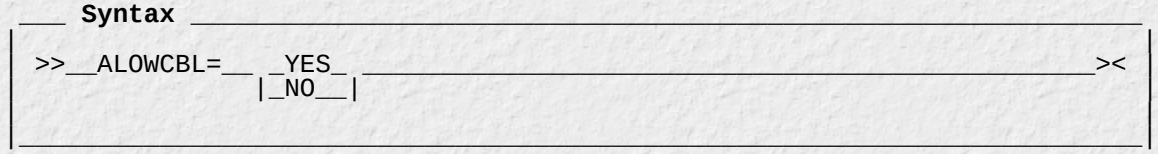
3. If ADV=YES is specified, and the record length for the output file has been defined in the source code, the programmer ***must*** specify the record description length as one byte larger than the source program record description. The programmer ***must*** also specify the block size in correct multiples of the larger record size.

4. If the LINAGE clause is specified in a file description (FD), the compiler treats that file as if ADV=YES had been specified.

IBM Library Server

1.2.5.8 ALOWCBL

Syntax



Default
ALOWCBL=YES

YES
Instructs the compiler to allow the use of the PROCESS (or CBL) statements in COBOL programs.

NO
Instructs the compiler to diagnose the use of PROCESS (or CBL) statements in a program as an error.

Notes:

1. ALOWCBL cannot be replaced at compile time because it cannot be included in the PROCESS (or CBL) statement.

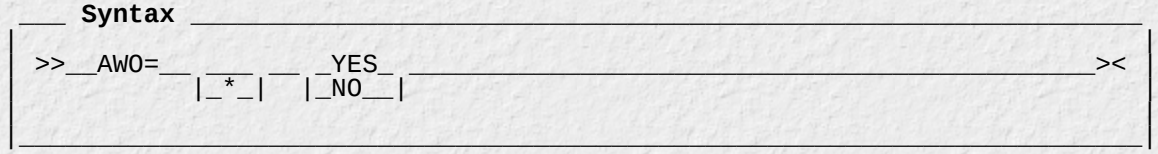
2. The PROCESS (or CBL) statement is used to specify compiler option parameters within source programs. If your installation requirements do not allow compiler options to be specified in a source program, specify ALOWCBL=NO.

3. You must specify ALOWCBL=YES for application programs that run under CICS.

IBM Library Server

1.2.5.9 AWO

Syntax



Default
AWO=NO

YES
Activates the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

Performance Consideration: Using AWO=YES generally results in fewer calls to Data Management Services for run-time files when handling I/O.

NO
Does not activate the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

IBM Library Server

1.2.5.10 BUF

Syntax

Default
BUF=4K

integer
Specifies the amount of dynamic storage, in bytes, to be allocated to each compiler work file buffer. The minimum value is 256 bytes.

Performance Consideration: Using BUF=integer generally improves compile-time performance by reducing the number of I/Os to the work files.

integerK
Specifies the amount of dynamic storage to be allocated to buffers in increments of 1K (1024) bytes.

Notes:

1. BUF and SIZE values are used by the compiler in determining how much storage is to be used during compilation. The amount allocated to the buffers is included in the amount of main storage available to the compiler for the SIZE option.

2. BUF cannot exceed the track capacity for the device used, nor can it exceed the maximum allowed by data management services.

IBM Library Server

1.2.5.11 CMPR2

Syntax

Default
CMPR2=NO

YES
Causes the compiler to generate code that is run-time compatible with valid VS COBOL II Release 2 programs. See the *COBOL/VSE Migration Guide* for details of language elements that are sensitive to CMPR2.

NO
Causes the compiler to generate code that may not be run-time compatible with valid VS COBOL II Release 2 programs for some COBOL 85 standard language constructs.

Notes:

1. Existing applications which rely on such matters as the format of the listing file, mapping of message numbers to messages, and output of the TERM option may need to be converted to COBOL/VSE.

2. Functions are **NOT** allowed under CMPR2. The word 'function' may be a dataname under CMPR2, but under NOCMPR2 it is considered a reserved word.

3. New language extensions for defining and manipulating procedural pointers are only supported with NOCMPR2.

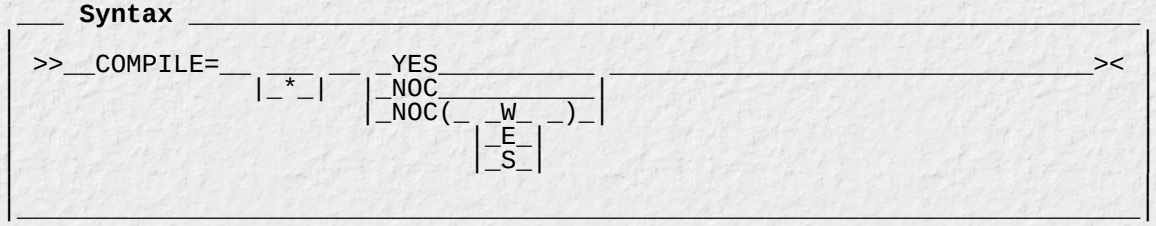
4. CMPR2=YES cannot be specified with certain other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflicting compiler options.

5. CMPR2=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.12 COMPILE

Syntax



Default
COMPILE=NOC(S)

YES
Indicates that you want full compilation, including diagnostics and object code.

NOC
Indicates that you want only a syntax check.

NOC(W)
NOC(E)
NOC(S)
Specifies an error message level:

W for warning

E for error

S for severe

When an error of the specified level or of a more severe level occurs, compilation stops, and only syntax checking is done for the remainder of the compilation.

Notes:

1. If the following compiler options are explicitly or implicitly specified in a program, COMPILE=NOC issues a warning message during compile time:

LIST=YES	TEST=(other than NO)
OFFSET=YES	VBREF=YES
OPT=STD or OPT=FULL	

Specifying these options together with COMPILE=NOC, while attempting

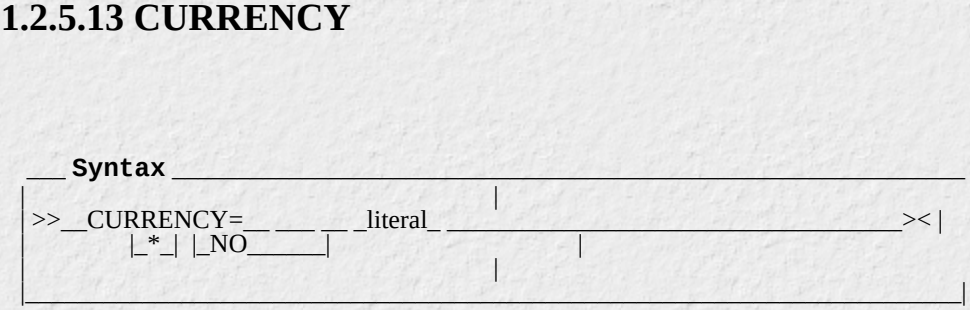
to assemble the customization macro, will result in a nonzero return code.

2. Specifying NOCOMPILE may affect the Associated Data file by stopping compilation prematurely, resulting in loss of specific messages.

IBM Library Server

1.2.5.13 CURRENCY

Syntax



The COBOL default currency symbol is the dollar sign ($). The CURRENCY option allows you to define an alternate default currency symbol.

Default
CURRENCY=NO

literal
Represents the default currency symbol you want to use in your program.

The literal must be a nonnumeric literal representing a one-byte EBCDIC character that **must not** be any of the following:

- Digits zero (0) through nine (9)
- Uppercase alphabetic characters: A B C D P R S V X Z
- Lowercase alphabetic characters a through z
- The space
- Special characters: * + - / , . ; ( ) = '
- Uppercase alphabetic character G, if the COBOL program defines a DBCS item with the PICTURE symbol G. The PICTURE clause will be invalid for that DBCS item because the symbol G is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character N, if the COBOL program defines a DBCS item with the PICTURE symbol N. The PICTURE clause will be invalid for that DBCS item because the symbol N is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character E, if the COBOL program defines an external floating point item. The PICTURE clause will be invalid for the external floating-point item because the symbol E is considered to be a currency symbol in the PICTURE clause.

The literal (including hex literal) syntax rules are as follows:

- The literal delimiters may be either quotes or apostrophes regardless of any option setting for literal delimiters.

- When an apostrophe (') is to be the currency sign, the embedded apostrophe must be "doubled". That is, two apostrophes must be coded to represent one apostrophe within the literal. For example:

  "" or """"

- The format for a hex literal specification is as follows:

  X'H1H2' or X"H1H2"

  where H1H2 is a valid hexadecimal value representing a one-byte EBCDIC character conforming to the rules for the currency sign literal as described above. Alphabetic characters in the hex literal must be in uppercase.

  Note: Hex values of X'20 or X'21' are not allowed.

NO
indicates that no alternate default currency sign is provided by way of the CURRENCY option, and the dollar sign will be used as the default currency sign for the program if the CURRENCY option is not specified at compile time.

The value NO provides the same results for the source program as omitting the CURRENCY SIGN clause in the COBOL source program.

Notes:

1. You can use the CURRENCY option as an alternative to the CURRENCY SIGN clause (which is specified in the COBOL source program) for selecting the currency symbol you will use in the PICTURE clause of your COBOL program.

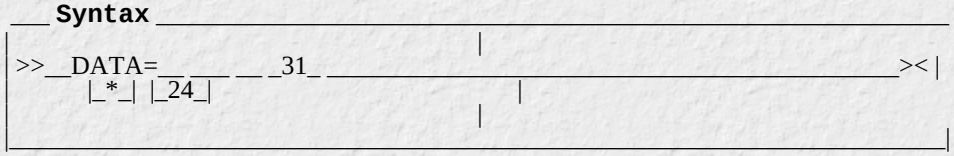
2. When both the CURRENCY option and the CURRENCY SIGN clause are used in a program, the symbol specified in the CURRENCY SIGN clause will be considered the currency symbol in a PICTURE clause when that symbol is used (even if the CURRENCY option is fixed {*}).

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.14 DATA

Syntax



Default
DATA=31

31
Causes user data areas, such as working storage and FD record areas, to be allocated from unrestricted storage in storage acquired by a GETVIS with the LOC=ANY option. Specifying this option can result in storage being acquired in virtual addresses either above or below 16 megabytes. The operating system generally satisfies the request with space in virtual addresses above 16 megabytes, if it is available.

24
Causes user data areas to be allocated in virtual addresses below 16 megabytes in storage acquired by a GETVIS with the LOC=BELOW option.

Specify DATA=24 for programs running in 31-bit mode that are passing data parameters to programs in 24-bit mode. This includes the following cases:

- An AMODE(31) program is passing items in its working storage to an AMODE(24) program.
- An AMODE(31) program is passing, by reference, data items received from its caller to an AMODE(24) program. DATA=24 is required even when the data received is below the 16-megabyte line.

Otherwise, the data may not be addressable by the called program.

Notes:

1. When a program is compiled with the RENT option and run in 31-bit mode, the DATA option controls how dynamic storage is acquired.
2. This option has no effect on a program compiled with the NORENT option, or a program run in 24-bit mode.
3. This option is ignored at run time on a 24-bit mode system, but is always present in the object module. If the module is subsequently moved, without recompilation, to an extended architecture system and run in 31-bit mode, the DATA option will control how dynamic storage

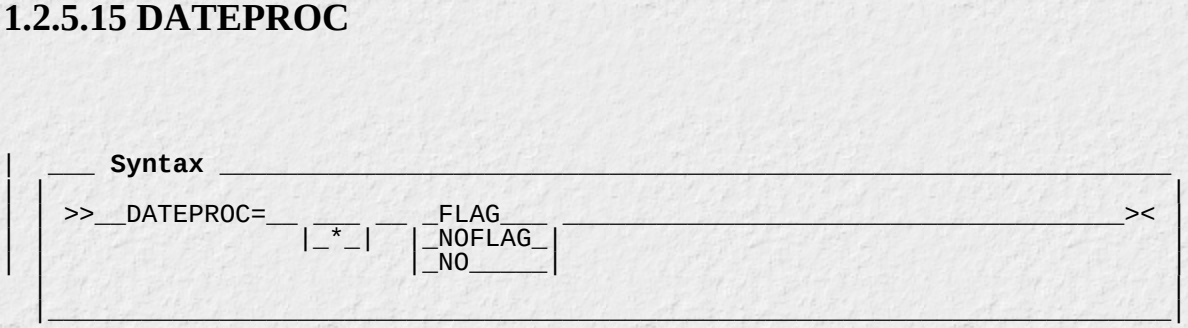
is acquired.

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.15 DATEPROC

Syntax



The DATEPROC option determines whether the compiler will perform date processing using the DATE FORMAT clause and other language constructs. IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA must be installed to specify anything other than DATEPROC=NO.

Default
DATEPROC=NO

FLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. In addition, specifying DATEPROC=FLAG instructs the compiler to flag, either with an information-level message or a warning-level message as appropriate, each statement that uses or is affected by date processing.

NOFLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. Statements that use or are affected by date processing are not flagged with information-level or warning-level messages.

NO
Instructs the compiler to treat the DATE FORMAT clause as comments and to disable automatic date processing. In the case of the new intrinsic functions, specifying DATEPROC=NO instructs the compiler to generate object code that returns a default value whenever a new intrinsic function is used.

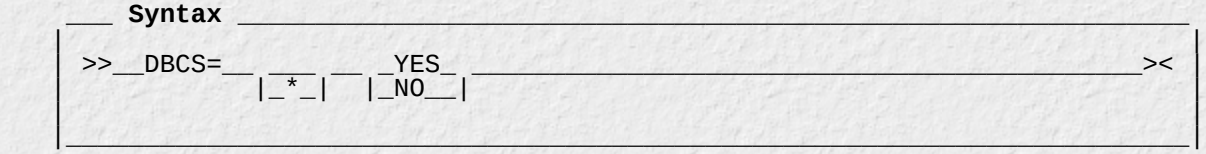
Notes:

1. Error-level and severe-level messages are issued regardless of whether DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. DATEPROC=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.16 DBCS



Default
DBCS=NO

YES
Instructs the compiler to recognize X'0E' and X'0F' in a nonnumeric literal and to treat them as Shift-Out and Shift-In control characters for delimiting DBCS data.

NO
Specifies the compiler is not to recognize X'0E' and X'0F' in a nonnumeric literal.

Notes:

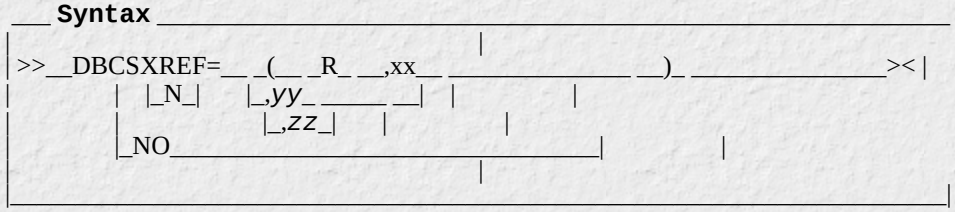
1. The presence of DBCS data inside the nonnumeric literal may cause the compiler to disallow certain uses of that literal. For example, DBCS characters are not allowed as program names or file system-names.

2. DBCS=NO supports the COBOL 85 standard.

3. DBCS=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

1.2.5.17 DBCSXREF

Syntax



Default
DBCSXREF=NO

NO
Specifies that no ordering program will be used for cross-reference of DBCS names. If the XREF phase is specified, a DBCS names cross-reference listing will be provided based on their physical order in the program.

R
Specifies that the DBCS Ordering Support Program (DBCSOS) will be loaded into the partition.

N
Specifies that the DBCSOS will be loaded into shared storage.

xx
Names a phase of the relevant ordering program to produce DBCS cross references. It must be 8 characters in length.

yy
Names an ordering type. It must be 2 characters in length. The default ordering type defined by the specified ordering program will occur if this parameter is omitted.

zz
Names the encode table used by the specified ordering type. It must be 8 characters in length. The default encode table associated with the particular ordering type will occur if this parameter is omitted.

Notes:

1. The DBCS Ordering Support Program (DBCSOS) must be installed to specify anything other than DBCSXREF=NO. DBCSOS is not available for use under VSE, therefore you should specify DBCSXREF=NO.

2. Specifying both XREFOPT=NO and DBCSXREF with an ordering program will result in a nonzero return code while attempting to assemble the customization macro.

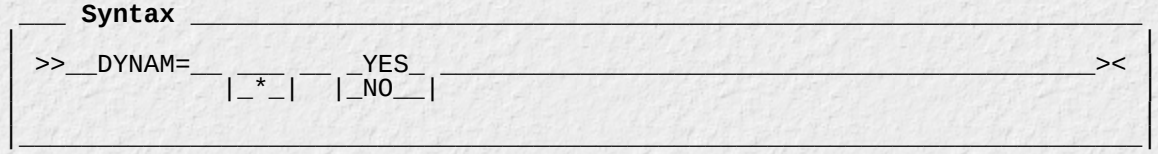
3. The assembly process will terminate when validation diagnoses:

   - Characters other than 'R' and 'N'
   - An invalid parameter length
   - Missing parameters after a comma
   - Missing 'yy' when 'zz' is specified

 *Copyright IBM Corp. 1983,1998*

1.2.5.18 DYNAM

Syntax



Default
DYNAM=NO

YES
Causes subprograms that are invoked through the CALL literal statement to be dynamically loaded.

Performance Consideration: Using DYNAM=YES eases subprogram maintenance since the application will not have to be relink-edited if the subprogram is changed. However, individual applications may experience some performance degradation due to a longer path length, but overall system performance may be slightly improved.

NO
Causes the text files of subprograms called with a CALL literal statement to be included with the calling program into a single phase.

Notes:

1. DYNAM=YES is used in support of the COBOL 85 Standard.

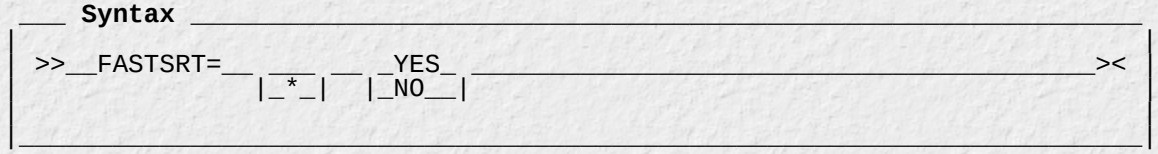
2. The DYNAM option has no effect on the CALL identifier statement at compile time. The CALL identifier statement always compiles to a dynamic call.

| 3. Do not specify DYNAM=YES for applications running under CICS.

IBM Library Server

1.2.5.19 FASTSRT

Syntax



Default
FASTSRT=NO

YES
Specifies that DFSORT/VSE or a comparable product (for example, Sort/Merge II), is to perform I/O when using either the USING or GIVING option.

Performance Consideration: Using FASTSRT=YES eliminates the overhead, in terms of CPU time, of returning to COBOL/VSE after each record is processed. However, there are restrictions you must follow if you choose to use this option. (For a detailed description of the restrictions, see *COBOL/VSE Programming Guide*.)

NO
Specifies that COBOL/VSE does the I/O for the sort/merge.

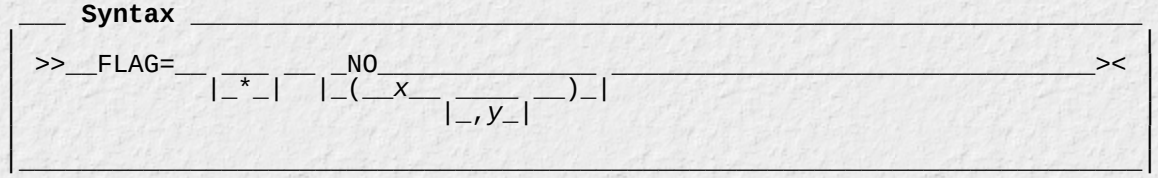
Notes:

1. If FASTSRT is in effect at compile time, the compiler verifies that the FASTSRT interface can be used for all restrictions except: (1) those requiring use of a device other than a direct-access device for sort work files; and (2) if appropriate, the BLKSIZE parameter of the JCL DLBL statement for the input/output file must match the File Description (FD) of the file.

   If FASTSRT cannot be used, the compiler generates a diagnostic message and prevents the sort program from performing I/O when using either the USING or GIVING options. Therefore it may be to your advantage to specify YES as the default.

IBM Library Server

1.2.5.20 FLAG



Default
FLAG=(I)

Note: The second severity level used in this syntax must be equal to or higher than the first.

x
I|W|E|S|U

Specifies that errors at or above the severity level specified are to be flagged and written at the end of the source listing.

ID	Type	Return Code
I	Information	0
W	Warning	4
E	Error	8
S	Severe error	12
U	Unrecoverable error	16

y
I|W|E|S|U

The optional second severity level specifies the level of syntax messages to be embedded in the source listing in addition to being at the end of the listing.

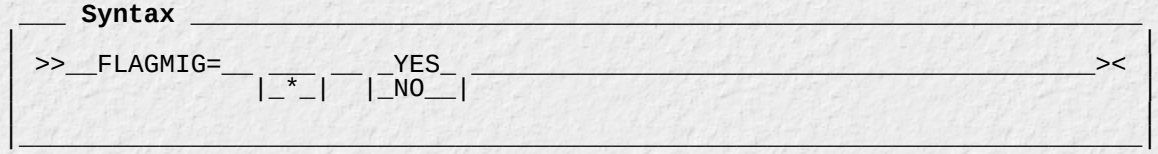
NO
Indicates that no error messages are flagged.

Note: If the messages are to be embedded, SOURCE must be specified at compile time. Embedded messages enhance productivity because they are placed after the referenced source statement.

IBM Library Server

1.2.5.21 FLAGMIG

Syntax



Default
FLAGMIG=NO

YES
Flags those VS COBOL II Release 2 language elements which might have different semantics in COBOL/VSE.

NO
Does not flag those VS COBOL II Release 2 language elements which have different semantics in COBOL/VSE.

Notes:

1. FLAGMIG=YES must be specified with CMPR2=YES to take advantage of this option.

2. FLAGMIG=YES has no effect when specified with CMPR2=NO.

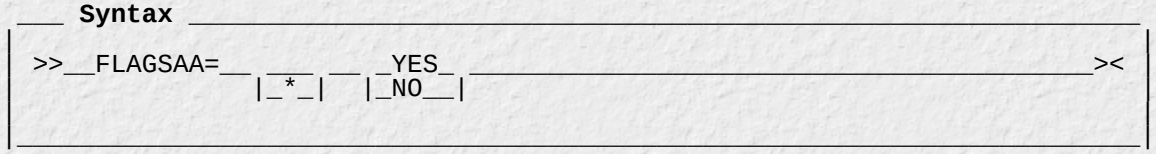
3. FLAGSAA=YES, FLAGSTD=YES, and DBCS=YES will all replace the FLAGMIG option.

4. FLAGMIG=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.22 FLAGSAA

Syntax



Default
FLAGSAA=NO

YES
Flags language elements which inhibit portability across Systems Application Architecture (SAA) COBOL systems.

NO
Does not flag language elements which inhibit portability across SAA COBOL systems.

Notes:

1. FLAGSAA=YES issues a warning (W) message to identify all SAA nonportable items.

   Specifying the following options together with FLAGSAA=YES, while attempting to assemble the customization macro, will result in a nonzero return code:

   CMPR2=YES
   FLAGSTD
   FLAGMIG

2. If the following compiler options are explicitly or implicitly specified in a program, FLAGSAA=YES issues a warning message during compile time:

   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DYNAM=NO
   FLAGMIG=YES
   FLAGSTD=(other than NO)
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT
   WORD=(other than NO or RWT)
   ZWB=NO

3. The FLAGSAA option identifies COBOL/VSE reserved words that are outside the current SAA specifications. Furthermore, the FLAGSAA option also identifies all COBOL words that are not in COBOL/VSE, but are reserved by other IBM SAA compilers.

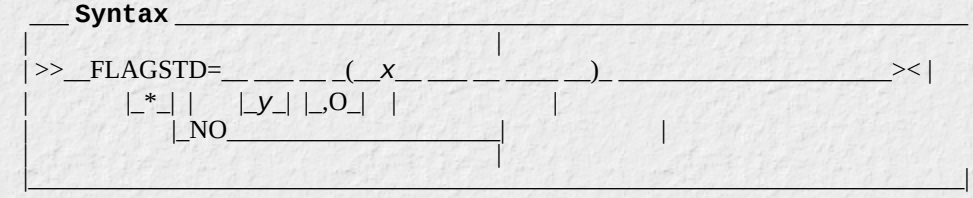
4. FLAGSAA=NO supports the COBOL 85 standard.

5. FLAGSAA can be in conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.23 FLAGSTD

Syntax



Default
FLAGSTD=NO

x
Can be M, I, or H to specify flagging for a FIPS (Federal Information Processing Standard) COBOL subset or standard.

M = ANS minimum subset of Standard COBOL.
I = ANS intermediate subset, containing those additional intermediate subset language elements which are not part of the ANS minimum subset.
H = ANS high subset, containing those additional high subset language elements which are not part of the ANS intermediate subset.

y
Can be any one or two combinations of D, N, or S to further define the level of flagging produced.

D Specifies ANS Debug module Level 1
N Specifies ANS Segmentation Module Level 1
S Specifies ANS Segmentation Module Level 2 (S is a superset of N.)

O
Specifies that obsolete language elements occurring in any of the above sets will be flagged.

NO
Specifies that no FIPS flagging will be accomplished.

Notes:

1. When FIPS flagging is specified, language syntax, such as the DATE FORMAT clause, used by the COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option) is flagged as nonconforming nonstandard IBM extensions to the COBOL 85 Standard.

2. When FIPS flagging is specified, informational messages in the source

program listing will identify:

- Whether the language element is obsolete, nonconforming standard, or nonconforming nonstandard (language elements that are both obsolete and nonconforming are flagged as obsolete only)
- The clause, statement, or header containing the nonconforming or obsolete syntax
- The source program line and an indication of the starting column within that line
- The level or optional module to which the language element belongs

3. FIPS flagging is suppressed when either:

   - Any error diagnosed as level E or higher occurs
   - The ABBR function of the WORD option is used, because standards conformance requires the standard set of reserved words

4. The FLAGSTD option can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

5. If the following compiler options are explicitly or implicitly specified in a program, FLAGSTD=(other than NO) issues a warning message during compile time:

   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DBCS=YES
   DYNAM=NO
   FASTSRT=YES
   FLAGMIG=YES
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT or BIN
   WORD=(other than NO or RWT)
   ZWB=NO

6. Specifying the following options together with FLAGSTD=(other than NO), while attempting to assemble the customization macro, will result in a nonzero return code.

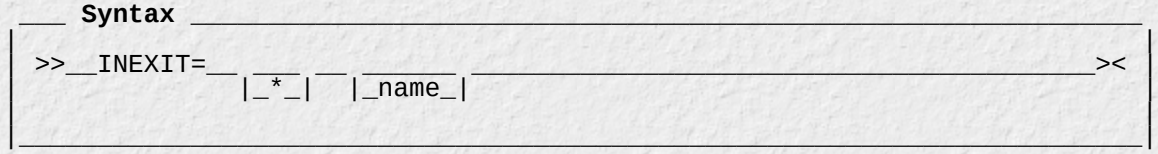
   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DBCS=YES
   DYNAM=NO
   FLAGMIG=YES
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT or BIN
   WORD=(other than NO or RWT)
   ZWB=NO

7. FLAGSTD may affect the Associated Data file by generating error records for FIPS standard conformation messages. Error messages are not guaranteed to be sequential in regard to source record numbers.

IBM Library Server

1.2.5.24 INEXIT

Syntax



Default
No exit specified.

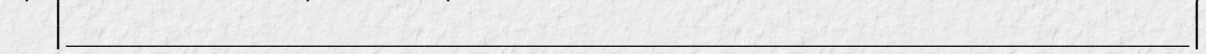
name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain source statements instead of using SYSIPT. When the option is supplied, SYSIPT is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

Note: Specification of INEXIT will prohibit identification of the compilation source file.

IBM Library Server

1.2.5.25 INTDATE

Syntax



Default
INTDATE=ANSI

ANSI
Instructs the compiler to use the ANSI COBOL Standard starting date for Integer date format dates used with date intrinsic functions. Day 1 = Jan 1, 1601.

With INTDATE(ANSI), the date intrinsic functions will return the same results as in COBOL/VSE without PTF UQ04360.

LILIAN
Instructs the compiler to use the Language Environment Lilian starting date for integer date format dates used with date intrinsic functions. Day 1 = Oct 15, 1582.

With INTDATE(LILIAN), the date intrinsic functions will return results compatible with the LE/VSE date callable services. These results will be different than in COBOL/VSE without PTF UQ04360.

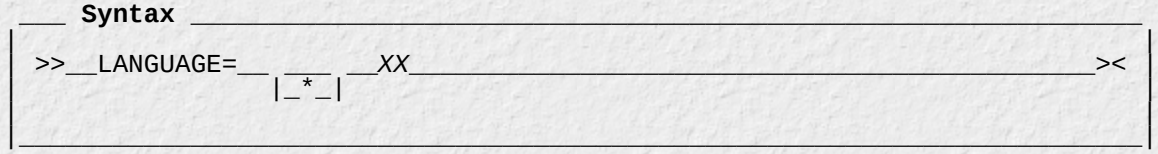
Notes:

1. When INTDATE(LILIAN) is in effect, CEECBLDY will not be useable since you will have no way to turn an ANSI integer into a meaningful date using either intrinsic functions or callable services. If you code a CALL literal statement with CEECBLDY as the target of the call with INTDATE(LILIAN) in effect, the compiler will diagnose this and convert the call target to CEEDAYS.

2. If you set your installation option to INTDATE(LILIAN), you should recompile all of your COBOL/VSE programs that use Intrinsic Functions to ensure that all of your code will be using the LILIAN integer date standard. This method is the safest, because you can store integer dates, pass them between programs, and even pass them from PL/I to COBOL to C programs and have no problems.

1.2.5.26 LANGUAGE

Syntax



Default
LANGUAGE=UE

XX
Specifies the language for compiler output messages. Entries for this parameter may be selected from the following list:

Table 15. Entries for the LANGUAGE compiler option

Entry	Language
EN or **ENGLISH**	Mixed case US English
JA, **JP**, or **JAPANESE**	Japanese
UE or **UENGLISH**	Uppercase US English

Notes:

1. The LANGUAGE option name must consist of at least the first two identifying characters. Other characters following the first two identifiers may be used, however only the first two will be used to determine the language name.

2. This compiler option does not affect the language in which run-time messages are displayed. For more information on run-time options and messages, refer to the *LE/VSE Programming Guide*.

3. Some printers use only uppercase and may not accept output in mixed-case (LANGUAGE=ENGLISH).

4. To specify the Japanese language option, the Japanese National Language Feature must be installed.

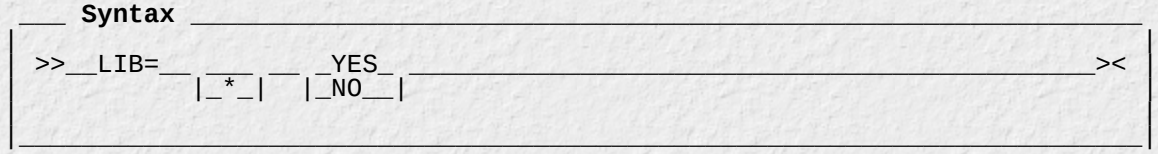
5. To specify the English language option (mixed-case English), the US English Language Feature must be installed.

6. If your installation provides a language other than those listed above, and you choose it to be your installation's default, you must specify at least the first two characters of the language name. The first two characters must be alphanumeric.

IBM Library Server

1.2.5.27 LIB

Syntax



Default
LIB=NO

YES
Indicates that the source program contains COPY or BASIS statements.

NO
Indicates that the source program does not contain COPY or BASIS statements.

Notes:

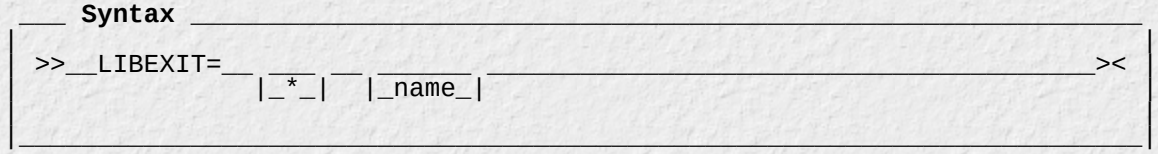
1. LIB=YES conforms to the COBOL 85 Standard.

2. LIB=YES is used when compiling a program containing COPY or BASIS statements. Specifying LIB=YES when there are no COPY or BASIS statements in a source program results in an unnecessary invocation of the COPY processing phase, unnecessary allocation of a buffer for the Librarian, and an unnecessary pass of the source text. However, compilation results are not affected.

IBM Library Server

1.2.5.28 LIBEXIT

Syntax



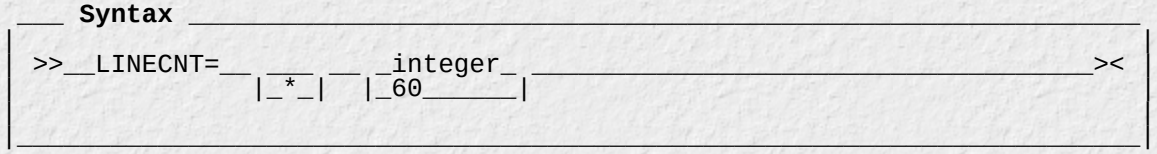
Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain copy statements instead of using the VSE Librarian to read the library search chain. When the option is supplied, the VSE Librarian is not called. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.29 LINECNT

Syntax



Default
LINECNT=60

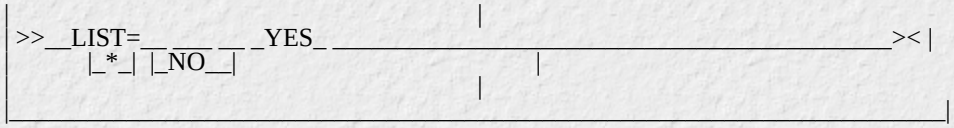
integer
Specifies the number of lines to be printed on each page of the compiler source code listing. Three of the lines are used to generate headings. For example, if you specify LINECNT=60, 57 lines of source code are printed on each page of the output listing, and 3 lines are used for headings.

| **Note:** The LINECNT installation option is equivalent to the LINECOUNT
| compile-time option.

IBM Library Server

1.2.5.30 LIST

Syntax



Default
NO

YES
Produces a listing that includes:

- The assembler-language expansion of source code
- Information about working storage
- Global tables
- Literal pools

NO
Suppresses this listing.

Notes:

1. Unless the installation default value of the LIST option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LISTX option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE LIST option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

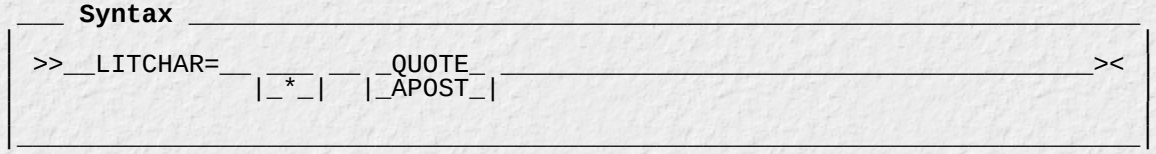
2. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code and an error message during assembly of the customization macro.

3. LIST=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.31 LITCHAR

Syntax



Default
LITCHAR=QUOTE

QUOTE
Specifies that double quotation marks (") are used to delineate literals and in the generation of figurative constants.

APOST
Specifies that an apostrophe (') is used to delineate literals and in the generation of figurative constants.

Notes:

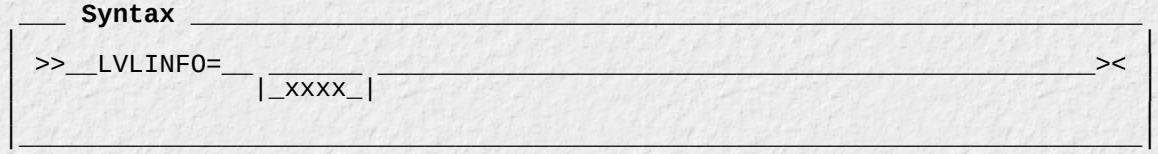
1. LITCHAR=QUOTE is used in support of the COBOL 85 Standard.

2. If you have an existing source library, determine whether QUOTE or APOST is used consistently.

IBM Library Server

1.2.5.32 LVLINFO

Syntax



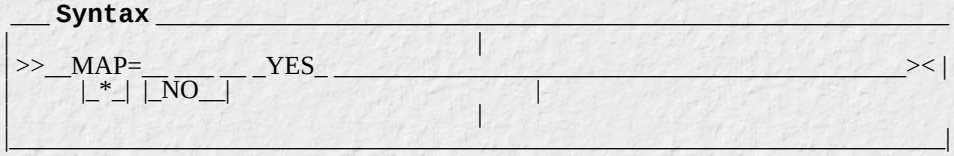
Default
No characters are specified.

xxxx
Identifies the one to four alphanumeric characters that will be inserted into the listing header following the Release number (the last four bytes of the signature area). This option may be used to identify "compiler level" information within the listing header.

IBM Library Server

1.2.5.33 MAP

Syntax



Default
MAP=NO

YES
Maps items declared in the Data Division. Map output includes:

- Data Division map
- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled without the RENT compiler option

NO
Mapping is not performed.

IBM Library Server

1.2.5.34 NAME

Syntax

Default
NAME=NO

NOALIAS
When used in conjunction with specific JCL options performs the following:

1. When used with LINK or CATAL, NOALIAS precedes each object deck written to SYSLNK with a linkage editor PHASE statement (PHASE phasename,*). The phase name (phasename) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

2. When used with DECK, NOALIAS precedes each object deck written to SYSPCH with a VSE Librarian CATALOG statement (CATALOG modname.OBJ,REPLACE=YES). The module name (modname) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

ALIAS
Specifying ALIAS has the same effect as NOALIAS.

NO
Does not produce linkage editor PHASE statements or VSE Librarian CATALOG statements.

Notes:

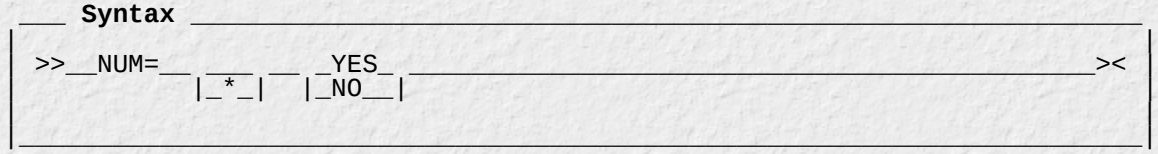
1. The NAME option allows you to create multiple modules in a program library with a single batch compilation. This can be useful for dynamic calls, for example.

2. NAME=NOALIAS or NAME=ALIAS supports the COBOL 85 Standard.

IBM Library Server

1.2.5.35 NUM

Syntax



Default
NUM=NO

YES
Uses the line numbers from the source program rather than compiler-generated line numbers for error messages and procedure maps.

NO
Uses the line numbers generated from the compiler for error messages and procedure maps.

Notes:

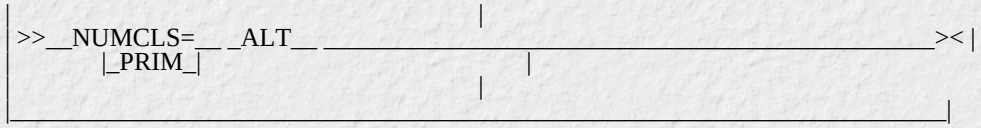
1. If COPY statements are used in a program and NUM=YES is in effect, the source program line numbers and the COPY member line numbers must be coordinated.

2. NUM=NO supports the COBOL 85 Standard.

IBM Library Server

1.2.5.36 NUMCLS

Syntax



Default
NUMCLS=PRIM

ALT
Used to specify the sign representations that are recognized as valid by the numeric class test for data items that are defined:

- As signed (with an "S" in the PICTURE clause)
- Using DISPLAY or COMPUTATIONAL-3 (packed-decimal)
- Without the SEPARATE phrase on any SIGN clause

Processing with ALT accepts hexadecimal A through F as valid.

PRIM
Processing with PRIM accepts only hexadecimal C, D, and F as valid.

Notes:

1. The numeric class test is affected by how both the NUMPROC and the NUMCLS options are specified. The NUMCLS option is effective only for NUMPROC=MIG or NUMPROC=NOPFD. NUMPROC=PFD specifies more strict rules for valid sign configuration.

1.2.5.37 NUMPROC

Syntax

Default
NUMPROC=NOPFD

MIG
Aids in migrating DOS/VS COBOL application programs to COBOL/VSE. Processing with MIG:

- Uses existing signs for comparison and arithmetic operations
- Generates preferred signs for the results of MOVE and arithmetic operations (These results meet the criteria for using NUMPROC=PFD.)
- Performs numeric rather than logical comparisons

NOPFD
Repairs signs on input. After repair is performed, the signs meet the criteria for NUMPROC=PFD.

PFD
Optimizes the generated code, especially when OPT=YES is specified. No explicit sign repair is performed. Note that NUMPROC=PFD has stringent criteria to produce correct results. To use NUMPROC=PFD:

- The sign position of unsigned numeric items must be X'F'.
- The sign position of signed numeric items must be either X'C' if positive or zero, or must be X'D' if negative.
- The sign position of separately signed numeric items must either be '+' if positive or zero, or '-' if negative.

Elementary MOVE and arithmetic statements in COBOL/VSE always generate results with these preferred signs, however group MOVEs and redefinitions may produce nonconforming results. The numeric class test may be used for verification. With NUMPROC=PFD, a numeric item will fail the numeric class test if the signs do not meet the preferred sign criteria.

Performance Consideration: Using NUMPROC=PFD will generate significantly more efficient code for numeric comparisons. For most references to COMP-3 and DISPLAY numeric data items, using NUMPROC=MIG and NUMPROC=NOPFD will generate extra code because of sign "fix-up" processing. This extra code may also inhibit some other types of optimization.

Before setting this option, please consult with your application programmers to determine the effect on the application program's output.

Notes:

1. NUMPROC=NOPFD or NUMPROC=MIG supports the COBOL 85 Standard.

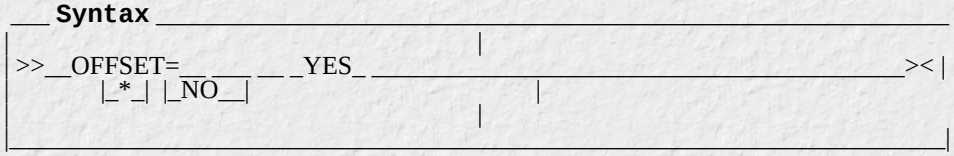
2. NUMPROC=NOPFD and NUMPROC=PFD are equivalent to the VS COBOL II Release 2 options PFDSGN=NO and PFDSGN=YES, respectively.

3. Both the NUMPROC and NUMCLS options affect the numeric class test. With NUMPROC=MIG or NUMPROC=NOPFD, the results of the numeric class test are controlled by how NUMCLS is set. When NUMPROC=PFD, a data item must meet the preferred sign criteria to be considered numeric.

IBM Library Server

1.2.5.38 OFFSET

Syntax



Default
OFFSET=NO

YES
Produces a condensed Procedure Division listing. The procedure portion of the listing will contain line numbers, verb references, and the location of the first instruction generated for each verb. In addition, the following are produced:

- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled with the NORENT compiler option

NO
Does not condense the listing and does not produce the items listed above.

Notes:

1. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code when attempting to assemble the customization macro. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflict resolution.

IBM Library Server

1.2.5.39 OPT

Syntax

Default
OPT=NO

STD
Causes the compiler to generate optimized object code.

FULL
Causes the compiler to generate optimized object code, and to delete unused WORKING-STORAGE data items as well as delete the VALUE clause code that initializes those items.

Performance Consideration: Using OPT=STD or OPT=FULL generally results in more efficient run-time code. This option requires more CPU time for compiles than OPT=NO.

NO
Does not cause the compiler to generate optimized object code.

Notes:

1. You can only specify TEST=(NONE,...) for the hook-location subparameter with OPT=STD or OPT=FULL. Any other combination results in a nonzero return code and an error during installation.

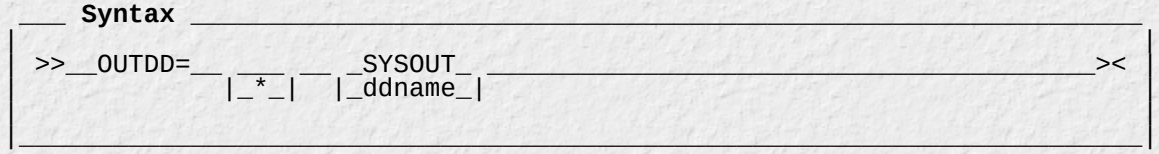
2. Do not use OPT=FULL if your programs depend on 'using' data items that are not referenced in the procedure division, such as adjacent tables, addresses passed to assembler programs, or 'eyecatchers' to identify the begin and end of the WORKING-STORAGE section. See *COBOL/VSE Programming Guide*.

3. Optimization is turned off if an S-level error or higher is flagged by the compiler.

IBM Library Server

1.2.5.40 OUTDD

Syntax



Default
OUTDD=SYSOUT

filename
Specifies the file name of the file to be used for run-time DISPLAY output.

Notes:

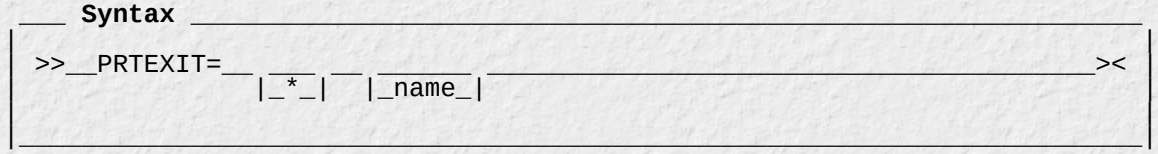
1. See the *LE/VSE Programming Reference* description of the MSGFILE run-time option to see how OUTDD interacts with MSGFILE.

2. Change the default for this option if, at run time, you expect to have a conflict with another product that requires SYSOUT as a file name. Both SYSOUT and SYSLST direct the output to SYSLST.

IBM Library Server

1.2.5.41 PRTEXIT

Syntax



Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it instead of writing to SYSLST. When the option is supplied, SYSLST is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.42 RENT

Syntax

```
>>__RENT=__ ___ __ _YES_ ____________________________________________><
          |_*_|  |_NO__|
```

Default
RENT=NO

YES
Indicates that the object code produced for a COBOL program is to be reentrant.

Using RENT=YES enables the program to be placed in the SVA for running above the 16-megabyte line. However, this option causes the compiler to generate additional code to ensure that the application program is reentrant.

NO
Indicates that the object code produced for a COBOL program is not to be reentrant.

Notes:

1. Programs must be compiled with RENT=YES or RMODE=ANY if they will be run with extended addressing in virtual storage addresses above 16 megabytes.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RENT/NORENT and RMODE compiler options. Valid combinations include:

Table 16. Effect of RENT and RMODE on Residency Mode

RENT/NORENT Setting	RMODE Setting	Residency Mode Assigned
RENT	AUTO	RMODE(ANY)
RENT	ANY	RMODE(ANY)
RENT	24	RMODE(24)
NORENT	AUTO	RMODE(24)
NORENT	ANY	RMODE(ANY)

	NORENT	24	RMODE(24)

IBM Library Server

1.2.5.43 RMODE

Syntax

Default
RMODE=AUTO

AUTO
A program compiled with the RMODE=AUTO option will have residency mode 24 if NORENT is specified, and residency mode ANY if RENT is specified.

24
A program compiled with the RMODE=24 option will have residency mode 24 whether NORENT or RENT is specified.

ANY
A program compiled with the RMODE=ANY option will have residency mode ANY whether NORENT or RENT is specified.

Notes:

1. COBOL/VSE NORENT programs that are required to pass data to programs running in AMODE(24) must either be compiled with the RMODE(24) option, or link-edited with RMODE(24). The data areas for NORENT programs will be above the line or below the line depending on the RMODE of the program, even if DATA(24) has been specified. DATA(24) applies to programs compiled with the RENT option only.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RMODE and RENT/NORENT compiler options. Valid combinations include:

Table 17. Effect of RMODE and RENT/NORENT on Residency Mode

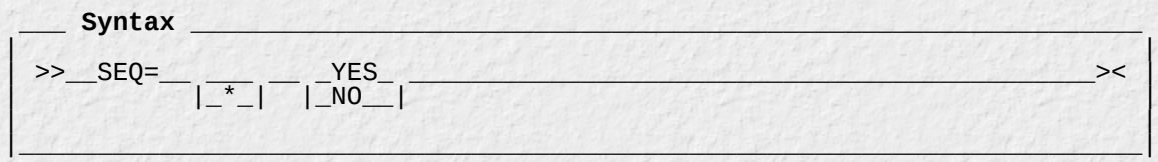
RMODE Setting	RENT/NORENT Setting	Residency Mode Assigned
AUTO	RENT	RMODE(ANY)
AUTO	NORENT	RMODE(24)
ANY	RENT	RMODE(ANY)

ANY	NORENT	RMODE(ANY)
24	RENT	RMODE(24)
24	NORENT	RMODE(24)

IBM Library Server

1.2.5.44 SEQ

Syntax



Default
SEQ=YES

YES
Instructs the compiler to check that the source statements are in ascending alphanumeric order by line number.

NO
Instructs the compiler not to perform sequence checking.

Notes:

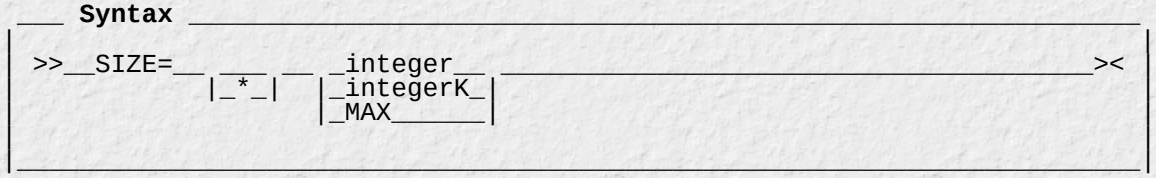
1. If both SEQ and NUM are in effect at compile time, the sequence is checked according to numeric, rather than alphanumeric, collating sequence.

2. SEQ=NO supports the COBOL 85 Standard.

IBM Library Server

1.2.5.45 SIZE

Syntax



Default
SIZE=MAX

integer
Specifies the amount, in bytes, of virtual storage available to the compiler.

The minimum acceptable value is 778240.

integerK
Specifies the amount of virtual storage available to the compiler in 1024-byte (K) increments.

The minimum acceptable value is 760K.

MAX
The compiler will request all available space in the partition GETVIS for use during the compilation. For Extended Architecture, the compiler obtains the largest contiguous block of free storage above the 16-megabyte line.

Notes:

1. Using SIZE=MAX simplifies compiler invocation by eliminating the need to determine a specific value for the SIZE option.

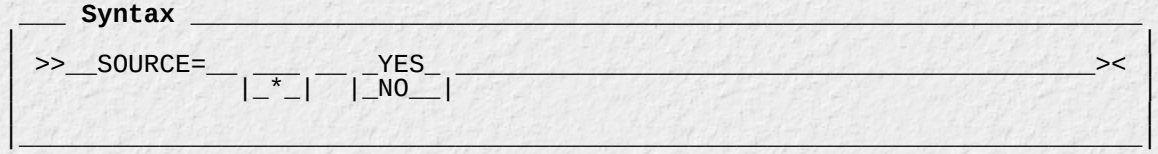
2. Do not use SIZE=MAX if, when you invoke the compiler, you require it to leave a specific amount of unused storage available in the partition.

3. SIZE=MAX in Extended Architecture allows the compiler to obtain all available above-the-line storage in the partition and below-the-line storage for work file buffers and compiler modules that must be below the 16-megabyte line. This occurs unless tuning is done for the Extended Architecture environment. Therefore, you may not want to fix this option as SIZE=MAX at installation.

IBM Library Server

1.2.5.46 SOURCE

Syntax



Default
SOURCE=YES

YES
Indicates that you want a listing of the source statements in the compiler-generated output. This listing also includes any statements embedded by COPY.

NO
Source statements will not appear in the output.

Notes:

1. Unless the installation default value of the SOURCE option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LIST option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE SOURCE option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

2. The SOURCE compiler option must be in effect at compile time if you want embedded messages in the source listing.

IBM Library Server

1.2.5.47 SPACE

Syntax

Default
SPACE=1

1
Indicates that you want single spacing for the source statement listing.

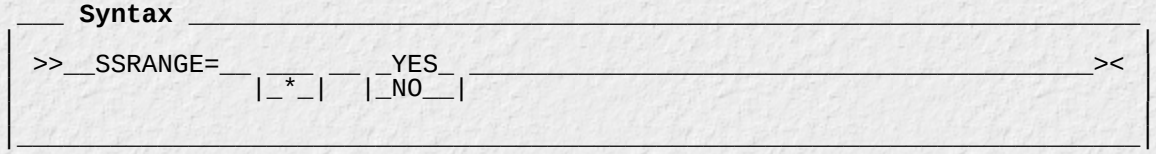
2
Indicates that you want double spacing for the source statement listing.

3
Indicates that you want triple spacing for the source statement listing.

IBM Library Server

1.2.5.48 SSRANGE

Syntax



Default
SSRANGE=NO

YES
At compile time, generates code that checks subscripts, reference modifications, variable-length group ranges, and indexes in the source program at run time to ensure that they do not refer to storage outside the area assigned. It also verifies that a table with ALL subscripting, specified as a function argument, will contain at least one occurrence in the table.

The generated code also checks that a variable-length item does not exceed its defined maximum length as a result of incorrect setting of the OCCURS DEPENDING ON object.

Performance Consideration: If SSRANGE=YES at compile time, object code size will be increased and there will be an increase in run-time overhead to accomplish the range checking.

NO
No code is generated to perform subscript or index checking at run time.

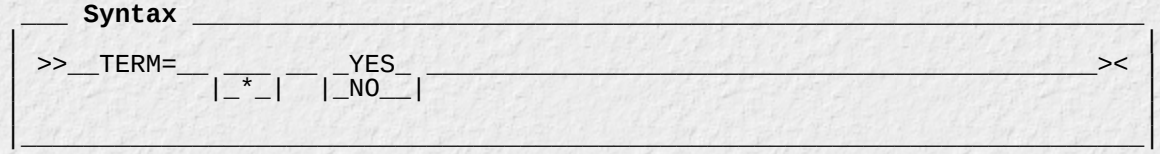
Notes:

1. If the SSRANGE option is in effect at compile time, the range-checking code is generated. Range-checking can be inhibited at run time by specifying the LE/VSE run-time option CHECK(OFF). However, the range-checking code still requires overhead and is dormant within the object code. The range-checking code can then be used optionally to aid in resolving any unexpected errors without recompilation.

IBM Library Server

1.2.5.49 TERM

Syntax



Default
TERM=NO

YES
Specifies that the progress and diagnostic messages are to be sent to the SYSLOG file.

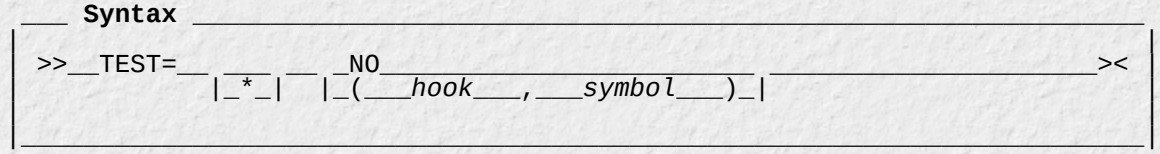
NO
Specifies that no messages are to be sent to SYSLOG.

| **Note:** Unless the installation default value of the TERM option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the TERM option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE TERM option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.50 TEST

Syntax



Default
TEST=NO

Other than NO
Produces object code containing symbol table information that is used by LE/VSE to produce a formatted dump, and hooks for Debug Tool/VSE.

You must specify options for both the hook-location (*hook*) and the symbol table (*symbol*) values.

hook values:

ALL Activates the generation of all compiled-hooks. Hooks will be generated at all statements, all path points, and at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

NONE Suppresses the generation of all compiled-hooks. TEST(NONE) is compatible with the OPT compiler option.

STMT Hooks will be compiled at every statement and label, as well as at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

PATH Hooks will be compiled at all path points, including program entry and exit points.

BLOCK Hooks will be compiled at all program entry and exit points.

symbol values:

SYM Activates inclusion of symbolic dictionary information in your object program.

NOSYM Deactivates inclusion of symbolic dictionary information in your object program.

Performance Consideration: Because TEST=(a hook-location suboption other than NONE) generates additional code, it can cause significant performance degradation at run time when used in a production environment.

NO
Produces object code that does not contain the symbol table information that is used by LE/VSE to produce a formatted dump, and the hooks for Debug Tool/VSE.

Notes:

1. If you specify TEST= (other than NONE for the hook-location suboption), the following option is put into effect at compilation time:

   OPT=NO

2. Only TEST(NONE) is compatible with the OPT compiler option. See "Conflicting Compiler Options" in topic 1.2.5.3 for more conflict resolution information.

3. Programmers might notice an increase in run-time for programs compiled with any hook-location suboption other than NONE in effect.

4. A date processing statement is any statement that references a date field, or any EVALUATE or SEARCH statement WHEN phrase that references a date field.

5. To get the full capability of Debug Tool/VSE, compile your program with TEST(ALL,SYM).

6. For production programs compile your programs with TEST(NONE,NOSYM) if you do not want to increase the size of your production modules, but still get minimum debugging capability.

IBM Library Server

1.2.5.51 TRUNC

Syntax

Default
TRUNC=STD

STD
Controls the way arithmetic fields are shortened during MOVE and arithmetic operations. The TRUNC option applies only to binary (COMP) receiving fields in MOVE statements and in arithmetic expressions. When TRUNC=STD is in effect, the final intermediate result of an arithmetic expression, or of the sending field in the MOVE statement, is shortened to the number of digits in the PICTURE clause of the binary receiving field.

OPT
The compiler assumes that the data conforms to PICTURE and USAGE specifications. The compiler manipulates the result based on the size of the field in storage (halfword or fullword).

TRUNC=OPT should only be specified when data being moved into binary areas will not have a value with larger precision than that defined by the binary item PICTURE clause. Otherwise, truncation of high-order digits may occur. The truncation results are dependent on the particular code sequence generated and may not necessarily be the same in DOS/VS COBOL and VS COBOL II.

BIN
Specifies that:

1. Output binary fields will be shortened only at the S/370* half/full/double word boundaries, rather than at COBOL base 10 picture limits.

2. Input binary fields will be treated as S/370 half/full/double words, and no assumption will be made that the values are limited to those implied by the base 10 PICTURE clause.

3. DISPLAY will convert and output the full content of binary fields with no truncation to the PICTURE description.

Performance Consideration: Using TRUNC=OPT does not generate extra code and generally improves performance. However, both TRUNC=BIN and TRUNC=STD generate extra code whenever a BINARY data item is changed. TRUNC=BIN is usually the slowest of these options.

Notes:

1. TRUNC=STD supports the COBOL 85 Standard.

2. TRUNC=STD and TRUNC=OPT are equivalent to TRUNC=YES and TRUNC=NO (respectively) in VS COBOL II Release 2.

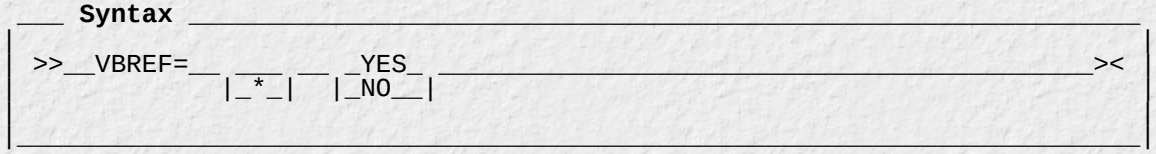
3. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether the COBOL/VSE source programs assume a particular setting for correct running.

4. TRUNC=BIN is the recommended option when interfacing with other products that have S/370-format binary data (such as CICS, SQL/DS*, FORTRAN, and PL/I). This is especially true if there is a possibility of having more than 9 digits in a fullword or more than 4 digits in a halfword.

IBM Library Server

1.2.5.52 VBREF

Syntax



YES
Produces a cross-reference of all verb types in a source program to the line numbers at which they were found. VBREF=YES also produces a summary of how many times each verb was used in the program.

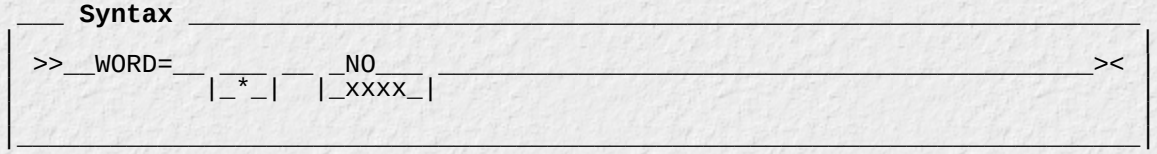
NO
Does not produce a cross-reference or verb summary listing.

Default
VBREF=NO

IBM Library Server

1.2.5.53 WORD

Syntax



Default
WORD=*NO

NO
Indicates that no alternative reserved word table is to be used as the default.

xxxx
Specifies an alternative default reserved word table to be used during compilation. **xxxx** represents the ending characters (may be 1 to 4 characters in length) of the name of the reserved word table to be used. The first 4 characters are IGYC. The last 4 characters may not be any one of the character strings listed below, nor may any of them contain the dollar sign character ($).

ASM1	LIBO	LVL8	RDSC
ASM2	LIBR	OPTM	RWT
DIAG	LSTR	OSCN	SAW
DMAP	LVL0	PGEN	SCAN
DOPT	LVL1	RCTL	SIMD
FGEN	LVL2	RDPR	XREF
INIT	LVL3		

Notes:

1. The default for the WORD option is specified with an asterisk. When the option is installed with the default (WORD=*NO), the application programmer cannot replace the option at compile time to specify an alternative reserved word table.

2. WORD=NO supports the COBOL 85 Standard.

3. Specification of WORD affects the interpretation of input reserved words. System names (such as UPSI and SYSPCH) and the intrinsic function names should not be used as aliases for reserved words. If a function name is specified as an alias via the reserved word table ABBR control-statement, that function name will be recognized and diagnosed by the compiler as a reserved word and the intrinsic function will not be performed.

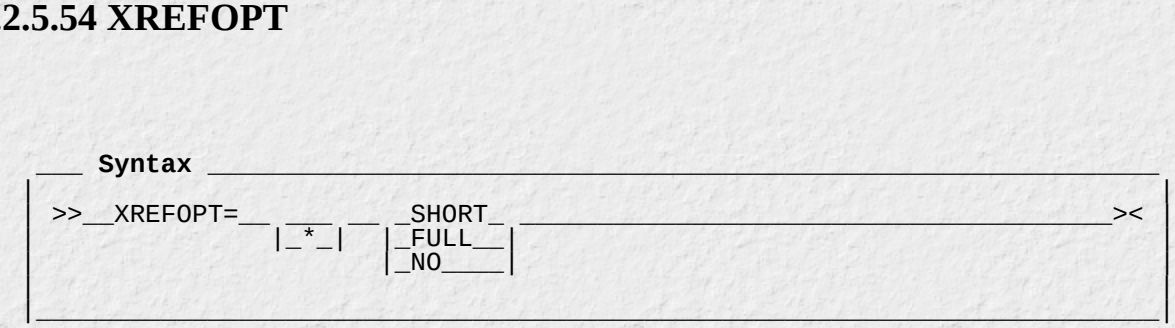
4. Changing the default value of the WORD option could cause compiler option conflicts. See "Conflicting Compiler Options" in topic 1.2.5.3 for information on conflict resolution.

5. A CICS-specific reserved word table is provided as an alternate reserved word table. For a description, see "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2.

IBM Library Server

1.2.5.54 XREFOPT

Syntax



Default
XREFOPT=NO

SHORT
Produces only the explicitly referenced variables in the cross-reference listing.

FULL
Produces both a sorted and embedded cross-reference listing. If SOURCE=YES is also specified, the listing line numbers cross-reference recurrences of particular data-names.

NO
Suppresses the cross-reference listing.

Notes:

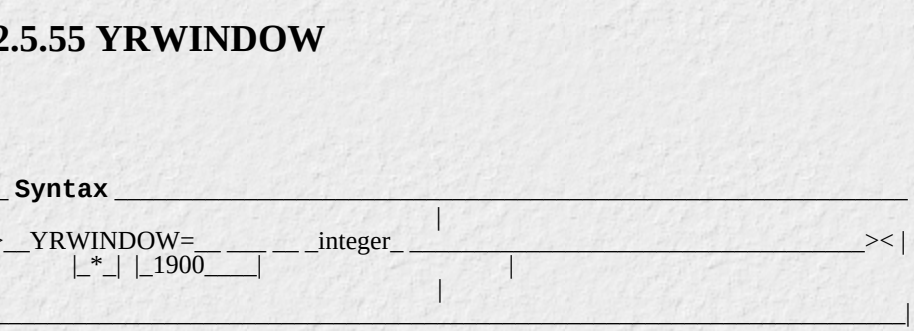
1. The XREFOPT option sets the default value for the compiler option XREF.

Note: Unless the installation default value of the XREFOPT option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the XREF or SXREF option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE XREF option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.55 YRWINDOW

Syntax



Default
YRWINDOW=1900

integer
Specifies the first year of the 100-year window used by COBOL windowed date fields, and may be one of the following:

- An unsigned decimal integer from 1900 through 1999.
- A negative decimal integer from -1 through -99, representing an offset from the current year at run time. The current year is determined for each compilation unit when it is first initialized, or re-initialized after execution of a CANCEL statement that refers to the compilation unit.

Notes:

1. YRWINDOW has no effect unless DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. At run-time, two conditions must be true:

   a. The 100-year window must have its beginning year in the 1900s.

   b. The current year must lie within the 100-year window for the compilation unit.

3. All windowed dates have a year relative to the base year. For example, if the base year were specified as 1965, all windowed year values would be interpreted as years within the 100-year window of 1965 to 2064, inclusive. So, a windowed year value of 67 would represent the year 1967, whereas a windowed year value of 05 would represent the year 2005.

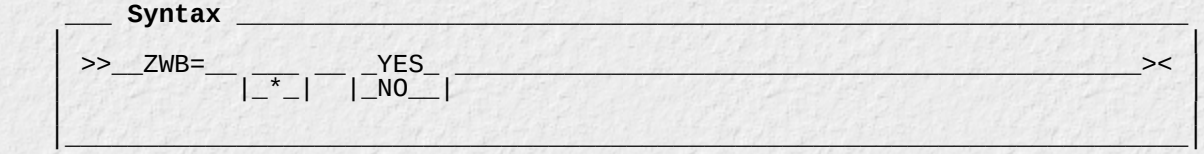
   If the base year were specified as -30, then the window would depend on when the application were run. Running it during 1998 would give a 100-year window of 1968 through 2067. So, a windowed year value of 68 would represent the year 1968, whereas a windowed year value of 67

would represent the year 2067.

4. The YRWINDOW installation option is equivalent to the YEARWINDOW compile-time option.

IBM Library Server

1.2.5.56 ZWB



Default
ZWB=YES

YES
Instructs the compiler to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

NO
Instructs the compiler not to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

Notes:

1. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether your COBOL/VSE source programs assume a particular setting to run correctly.

2. ZWB=YES supports the COBOL 85 Standard.

3. Application programmers use ZWB=NO to test input numeric fields for SPACES.

IBM Library Server

1.3 Chapter 3. Installing COBOL/VSE

This chapter provides the following information:

- Overview of the installation procedure
- Procedure for installing COBOL/VSE

Note: Installing COBOL/VSE requires the use of the Maintain System History Program (MSHP).

Subtopics:

IBM Library Server

1.3.1 Installation Overview

The following table provides an overview of the installation steps.

Step	Description	Topic
1	Backup the Original System	1.3.2.1
2	Allocate Space for the Library	1.3.2.2
3a	Install COBOL/VSE using the Interactive Interface	1.3.2.3.1
3b	Install COBOL/VSE using a batch job	1.3.2.3.2
4	Verify the COBOL/VSE Installation	1.3.2.4

IBM Library Server

1.3.2 Procedure for Installing COBOL/VSE

The following sections detail the installation steps outlined in "Installation Overview."

Subtopics:

IBM Library Server

1.3.2.1 Step 1: Backup the Original System

Make a backup copy of your original system, including the system history file. For more information about backing up your VSE system, see *VSE/ESA System Control Statements*.

1.3.2.2 Step 2: Allocate Space for the Library

By default COBOL/VSE is installed into the PRD2.PROD sublibrary. If you decide to install COBOL/VSE into a different sublibrary proceed with this step. If you decide to install COBOL/VSE in the default sublibrary, proceed to "Step 3a: Install COBOL/VSE using the Interactive Interface" in topic 1.3.2.3.1 or "Step 3b: Install COBOL/VSE using a Batch Job" in topic 1.3.2.3.2. Once you have determined how much space you require, decide where to allocate space for the library. You can allocate space for the library in non-VSAM space or in VSAM space.

- Non-VSAM

  Identify, on the disk volume (or volumes) to be used for the library, suitable areas of free space. To do this, list the volume table of contents (VTOC) of the disk or disks to be used. The sample job shown in Figure 2 illustrates the JCL needed to list the VTOC for the volume with serial number COBVOL.

```
// JOB COBVSE  LIST VTOC for COBVOL
// ASSGN SYS004,DISK,TEMP,VOL=COBVOL,SHR
// ASSGN SYS005,SYSLST
// EXEC LVTOC
/*
/&
```

Figure 2. Listing the Contents of a DASD Volume

- VSAM

  Examine the user catalog to check for enough VSAM space. To do this, use the access method services (IDCAMS) LISTCAT command to provide a report showing the space available. The sample job shown in Figure 3 illustrates the JCL needed to display the space use in the VSAM user catalog supplied with VSE/ESA.

```
// JOB IDCAMS  SHOW USER CATALOG SPACE
// EXEC IDCAMS,SIZE=AUTO
 LISTCAT SPACE ALL CATALOG(VSESP.USER.CATALOG)
/*
/&
```

Figure 3. Listing the Space in a VSAM Catalog

If you intend to use VSAM space for the COBOL/VSE library, see *VSE/ESA Guide to System Functions* for information about how to define the required VSAM cluster. VSE/ESA also provides dialogs (Interactive Interface) for these tasks. Refer to *VSE/ESA Administration* for details.

Once you have determined where to allocate space for the library, you can define the library using the JCL shown in Figure 4.

```
// JOB LIBRDEF CREATE A LIBRARY
// OPTION LOG
*  Label for the Product Library as a Non-VSAM file  1
// DLBL COBVSE,'COBOL.VSE.LIBRARY',99/365,SD
// EXTENT SYS002,volid,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volid,SHR
*  Label for the Product Library as a VSAM file      2
// DLBL COBVSE,'COBOL.VSE.LIBRARY',,VSAM,CAT=catname
* --------------------------------------
*  This step defines the Product Library             3
* --------------------------------------
// EXEC LIBR
 DELETE LIB=COBVSE
 DEFINE LIB=COBVSE,REPLACE=NO
/*
/&
```

Figure 4. Job to Allocate the COBOL/VSE Library Space

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. If you intend to allocate space for the COBOL/VSE library in non-VSAM space, delete the DLBL statement in area **2** of the sample job, and make the following changes in area **1** :

   a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

   b. Change the variable *volid* in the EXTENT statement to the volume serial number of the volume that will hold your library.

   c. Change the variable *rtrk* in the EXTENT ASSGN statements to the start location (track or block number) of the extent.

d. Change the variable *ntrk* in the EXTENT statement to the number of tracks or blocks required for the library.

4. If you intend to allocate space for the COBOL/VSE library in VSAM space, delete the DLBL, EXTENT and ASSGN statements in area **1** of the sample job, and make the following changes in area **2**:

a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

b. Change the variable *catname* to the file name of the VSAM catalog that will own the library. You can omit *catname* (and the CAT keyword and preceding comma) if the library is to be cataloged in the IJSYSUC user catalog.

The Librarian job step in area **3** includes a DELETE statement before the DEFINE statement, so the job can be rerun. This means the following messages will be issued when the job runs for the first time. These may be ignored and the job will continue to allocate the library.

```
L101I  LIBRARY COBVSE DOES NOT EXIST    <----------for Non-VSAM
L255I  LIBRARY COBVSE - OPEN FAILURE    <----------for VSAM
L027I  ABNORMAL END DURING DELETE COMMAND PROCESSING
L113I  RETURN CODE OF DELETE IS 8
```

IBM Library Server

1.3.2.3 Step 3: Install COBOL/VSE

You can use the Interactive Interface (Step 3a) or a batch job (Step 3b) to install COBOL/VSE.

Depending on how you ordered the COBOL/VSE product, you will receive one of the following types of distribution tape:

- A distribution tape containing only the COBOL/VSE product
- A VSE stacked tape containing one or more optional products

The installation methods shown will handle both types of distribution tape.

Continue with either Step 3a or Step 3b.

Subtopics:

1.3.2.4 Step 4: Verify the COBOL/VSE Installation

The installation verification program IGYWEIVP.C is a sample program provided on the distribution tape. It allows you to check that your installation is successful by exercising representative features of COBOL/VSE.

1. Job **CALLIVP1**: The JCL (IGYWEIN1.Z) and the source (IGYWEIVP.C) to run the verification program CALLIVP1 are supplied. They test the compiler and library, and verify the product installation.

   Before you run the JCL from member IGYWEIN1.Z, do the following:

   a. Modify the JOB statement to meet the requirements of your site.

   b. Add POWER JECL statements if wanted.

   c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

   d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

      Six compiler work files are always required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

   e. The JCL provided in IGYWEIN1.Z, without modification to the EXEC statement, compiles, link-edits, and runs the program. If you have not installed LE/VSE you must remove the two steps at the end of the job that link-edit and run the program.

   f. Execute the modified JCL to run the installation verification program.

   After you run job CALLIVP1 you should receive the following messages, printed on SYSLST:

   ```
   ***** START OF CALLIVP1 *****
   ***** CALLIVP1 SUCCESSFUL *****
   ```

2. Job **ERRMSG**: The JCL (IGYWEIN2.Z) and source (IGYWEERM.C) are used to run the verification program ERRMSG. This job will verify the operation of the compiler, and produce a complete list of compiler messages.

   Before you run the JCL from member IGYWEIN2.Z, do the following:

a. Modify the JOB statement to meet the requirements of your site.

b. Add POWER JECL statements if wanted.

c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

   Six compiler work files are required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

e. Execute the modified JCL to run the ERRMSG program.

After you run job ERRMSG, you will receive a listing of COBOL/VSE compiler messages.

IBM Library Server

1.4 Chapter 4. Customizing COBOL/VSE

You can make modifications to COBOL/VSE only after installation of the product is complete. "Making Changes after Installation--Why Customize?" in topic 1.2.1 provides information on what you can modify, and why you might want to customize COBOL/VSE. This chapter tells how to *make* the modifications or where to find the necessary coding information to tailor COBOL/VSE to the needs of your site.

The issues discussed in this chapter are:

- General rules for changing default values
- Modifying compiler options and phases
- Placing COBOL/VSE in the shared virtual area
- Creating additional reserved word tables

To make the modifications to the option macros, you will need to modify and run the sample JCL supplied. Sample JCL for customization is available in the sublibrary for the compiler.

Subtopics:

IBM Library Server

1.4.1 General Rules for Changing Default Values

If you choose to modify the default values for options, follow these assembler coding instructions:

- Code in columns 2 through 71.
- Do not put a comma in front of the first option in your macro.
- Begin any continuation lines in column 16 and have a nonblank character in column 72 of the previous line. You can break the coding after any comma.
- Place an END statement after the macro instruction line or lines.

If you choose to modify only some of the options in the macro, COBOL/VSE will use the defaults supplied by IBM for those not changed. A syntax format and a detailed description of each of these options begin under "Making Changes after Installation--Why Customize?" in topic 1.2.1. For a description of the syntax notation that applies to the macro formats, see "How to Read the Syntax Diagrams" in topic FRONT_2.1.

IBM Library Server

1.4.2 Modifying Compiler Options and Phases

During installation, default compiler option and phase values were stored in the options module IGYCDOPT. To change the compiler-option defaults or tell the compiler where its phases are expected to reside, you should code your IGYCOPT macro calls in the sample JCL supplied in member IGYWEOP1.Z.

Note: If you specify that the compiler phases are to be resident in the SVA, you must ensure that you have placed them there. Failure to do so could result in an abnormal termination of the compiler because enough space was not available for the phase.

Subtopics:

1.4.2.1 Procedure for Modifying Compiler Options

Code your required IGYCOPT macro calls with the sample JCL supplied in member IGYWEOP1.Z. If you choose to modify only some of the compiler default values or phase residency selections, then COBOL/VSE will use the supplied defaults for any unchanged options or phase residency selections. You must observe the macro coding rules described under "General Rules for Changing Default Values" in topic 1.4.1 when coding the macro operands in your job. If you require more than one line for coding your option and phase changes, be sure to observe the line continuation rules. For a detailed description of each of the compiler options and phases in IGYCOPT, see "COBOL/VSE Compiler Options" in topic 1.2.5.

Member IGYWEOP1.Z contains a sample VSE JCL job stream you can use to assemble and link-edit the compiler options module.

1.4.3 Placing COBOL/VSE in the Shared Virtual Area

After planning for the SVA requirements, complete the following procedure according to the planning that has been performed.

To place the COBOL/VSE compiler phases in the SVA, the following steps must be performed:

1. Customize IGYCDOPT to indicate which phases of the compiler are resident in the SVA.

2. Modify the SVA statement of the VSE IPL ASI (Automated System Initialization) procedure to allow space for the compiler phases.

   - Increase the SDL parameter by the number of new phases being added to the SVA for the compiler.

   - Increase the PSIZE parameters for the 24-bit SVA and the 31-bit SVA by the number of K bytes required to contain the new phases being added to the SVA for the compiler.

3. Modify the VSE background ASI (Automated System Initialization) procedure to automatically load the selected compiler phases into the SVA.

   - Modify the ALLOC statements for the partitions to ensure that the remaining SVA space is large enough to contain the selected compiler phases.

   - Modify the LIBDEF PHASE,SEARCH=PRD2.PROD statement preceding the SET SDL statement to include the name of the VSE Librarian sublibrary containing COBOL/VSE.

   - After the SET SDL command add the statement:

     phasename,SVA

     for each compiler phase that is to be loaded into the SVA.

     The sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary contains SVA statements for all the SVA-eligible compiler phase names.

4. Shut down and re-IPL your VSE system

Note: If you do not want to place the compiler phases in the SVA at

IPL-time, you can do this at any time after IPL using the JCL statements supplied in the sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary. You should first customize IGYCDOPT as described in step 1 above. If you do not already have enough space available in the SVA, you should also perform steps 2 and 4 before you attempt to place the compiler phases in the SVA.

Using other languages

Member IGYWESVA.Z contains the uppercase language phases for English as shown:

- IGYCUE$R
- IGYCUE$D
- IGYCUE$0
- IGYCUE$1
- IGYCUE$2
- IGYCUE$3
- IGYCUE$4
- IGYCUE$5
- IGYCUE$8

If you wish to use other languages you should include the following phases:

For mixed-case English:

- IGYCEN$R
- IGYCEN$D
- IGYCEN$0
- IGYCEN$1
- IGYCEN$2
- IGYCEN$3
- IGYCEN$4
- IGYCEN$5
- IGYCEN$8

For Japanese:

- IGYCJA$R
- IGYCJA$D
- IGYCJA$0
- IGYCJA$1
- IGYCJA$2
- IGYCJA$3
- IGYCJA$4
- IGYCJA$5
- IGYCJA$8

For more information on loading phases in the SVA refer to *VSE/ESA System Control Statements* and *VSE/ESA Installation and Service*.

IBM Library Server

1.4.4 Modifying or Creating Additional Reserved Word Tables

The reserved words used by the COBOL/VSE compiler are maintained in a table (IGYCRWT) provided with the product. A CICS-specific reserved word table (IGYCCICS) is provided as an alternate reserved word table. You can change the reserved words by using the reserved word table utility (IGY8RWTU) to create additional reserved word tables.

The reserved word table utility accepts control statements you can use to create a new reserved word table. This new table then contains the reserved words from the table, as supplied by IBM, with all the changes you have applied.

You can make the following changes to the reserved word table:

- Add an alternative form of an existing reserved word.
- Add words to be flagged with an informational message whenever they are used in a program.
- Add words to be flagged with an error message whenever they are used in a program.
- Indicate that words currently flagged with an informational or error message should no longer be flagged.

Each reserved word table you create must have a unique 1- to 4-character identifier. For a list of 1- to 4-character strings that **cannot** be used, see the WORD compiler option in topic 1.2.5.53.

At compile time, the value of the compiler option **WORD(xxxx)** identifies the reserved word table to be used, where **xxxx** is the unique 1- to 4-character identification you specified in the member name IGYCxxxx. You can create multiple reserved word tables, but only one can be specified at compile time.

Note: The total amount of storage used for all entries in a reserved word table should not exceed 1536 bytes or 1.5K bytes.

Subtopics:

1.4.4.1 Procedure for Creating or Modifying a Reserved Word Table

To create or modify a reserved word table you must edit the appropriate source member from the compiler sublibrary:

- Member IGY8RWRD.Z (default reserved word table supplied by IBM)
- Member IGY8CICS.Z (CICS reserved word table supplied by IBM)
- A user member (user-supplied reserved word table)

You must also modify and invoke the appropriate JCL.

The JCL for the default reserved word table is IGYWERWD.Z

The JCL for the CICS reserved word table is IGYWERWC.Z

Your source member should have four parts: Parts I, II, III, and IV. Modify the member and JCL as follows:

1. Make a private copy of the member.
2. Do not edit Part I (all lines up to and including the line with the keyword MOD).
3. Edit Part II by placing asterisks in column 1 of the lines that contain CODASYL reserved words for which you do not want informational messages issued.
4. Edit Part III by placing asterisks in column 1 of the lines that contain obsolete reserved words for which you do not want severe messages issued.
5. Edit Part IV by coding additional reserved word control statements that create the modifications you want, as described under "Coding Control Statements" in topic 1.4.4.1.1.
6. Modify and run the JCL, as discussed under "Modifying and Running JCL to Create a New Reserved Word Table" in topic 1.4.4.5. You will also have to create a unique 1- to 4-character identification for the new reserved word table and supply it in the JCL.

Subtopics:

- 1.4.4.1.3 Coding Operands in Control Statements
- 1.4.4.1.4 Rules for Coding Control Statement Operands

IBM Library Server

1.4.4.2 ABBR Statement

```
ABBR  reserved-word:  user-word [ comments ]
    Defines an alternative symbol for the reserved word specified.  The
    symbol can be used when coding a program.
```

Notes:

1. *User-word* becomes a reserved word and can be used in place of the reserved word specified in this statement.

2. *Reserved-word* remains a reserved word with its original definition.

3. The source listing will show the original source--the symbol as you coded it.

4. The reserved word will be used in compiler output--other listings, some messages, and so forth.

Subtopics:

- 1.4.4.2.1 Example of an ABBR Statement

1.4.4.3 INFO Statement

```
INFO  COBOL-word  [ comments ]
    Specifies a COBOL word that is to be flagged by the compiler.


The messages will be handled as information (I) messages.  The compilation
condition is not changed.
```

Subtopics:

- 1.4.4.3.1 Example of an INFO Statement

IBM Library Server

1.4.4.4 RSTR Statement

```
RSTR  COBOL-word  [ comments ]
    Specifies a COBOL word that cannot be used in a program.
```

Note: The following reserved words may not be restricted:

IDENTIFICATION	FD
ENVIRONMENT	DATA
WORKING-STORAGE	PROCEDURE
DIVISION	SECTION
PROGRAM-ID	

Subtopics:

- 1.4.4.4.1 Examples of RSTR Statements

IBM Library Server

1.4.4.5 Modifying and Running JCL to Create a New Reserved Word Table

The JCL you use to create a new reserved word table contains three steps:

STEP 1 Runs the reserved word table utility with your modified table.

STEP 2 Assembles your modified reserved word table.

STEP 3 Produces a run-time loadable phase from the object module.

After you run the job, a new reserved word table will be created, the sublibrary you specified will contain the new table, and the name of the table will be IGYC plus the 1- to 4-character identification you specified.

Subtopics:

- 1.4.4.5.1 Procedure for Modifying and Running JCL

IBM Library Server

1.5 Chapter 5. Maintaining COBOL/VSE

This chapter describes how to replace or re-install COBOL/VSE, and how to apply service updates to COBOL/VSE. To use the maintenance procedures effectively, you must have already installed COBOL/VSE and any required products.

In addition, this chapter describes how to remove COBOL/VSE.

Subtopics:

IBM Library Server

1.5.1 Reinstalling COBOL/VSE

You do not need to perform all the planning and installation procedures to re-install COBOL/VSE. For example, you might not need to reconsider your storage needs if COBOL/VSE replaces the existing COBOL/VSE sublibrary.

You do not need to remove COBOL/VSE from your system before re-installing it unless you intend to re-install the product in a different sublibrary. In this case you must remove COBOL/VSE from the system history file before re-installation can take place. Figure 10 in topic 1.5.3 shows a job to remove COBOL/VSE from the system history file.

To re-install COBOL/VSE you simply follow the same steps as for installing it. See Chapter 3, "Installing COBOL/VSE" in topic 1.3.

IBM Library Server

1.5.2 Applying Service Updates

You might need to apply maintenance or service updates to COBOL/VSE periodically.

Subtopics:

1.5.2.1 What You Receive

If you report a problem with COBOL/VSE to your IBM Support Center, you will receive a tape containing one or more PTFs (Program Temporary Fix) which have been created to solve your problem.

You may also receive a list of pre-requisite APARs (Authorized Program Analysis Report) or PTFs which should have been applied to your system before applying the current service. These pre-requisite APARs and PTFs may relate to COBOL/VSE or any other licensed product you have installed, including VSE/ESA.

You apply service to COBOL/VSE using either the VSE/ESA Interactive Interface or a batch job.

The following checklist provides a summary of the steps you should use to apply service to COBOL/VSE.

IBM Library Server

1.5.2.2 Checklist for Applying Service

Table 18 lists the steps for installing corrective service on COBOL/VSE. You can use this table as a checklist.

Table 18. Summary of Steps for Installing Service on COBOL/VSE

Step	Description	MSHP Command or Jobname	Topic
1	Check pre-requisite APARs or PTFs	RETRACE	1.5.2.2.1
2	Backup the Original System	----	1.5.2.2.2
3a	Apply Service using the Interactive Interface	INSTALL	1.5.2.2.3
3b	Apply Service using a batch job	INSTALL	1.5.2.2.3
4	Run the Installation Verification Program	IGYWEIVP	1.5.2.2.6

Subtopics:

IBM Library Server

1.5.3 Removing COBOL/VSE

You do not have to remove COBOL/VSE from your system before installing a new version or release.

If you do have to remove COBOL/VSE from your system for any reason at all, you must delete all the COBOL/VSE entries from your sublibrary and remove COBOL/VSE from the system history file. Figure 9 shows the JCL needed to remove COBOL/VSE from the sublibrary.

```
// JOB IGYDELV  Remove COBOL/VSE
* Label for the COBOL/VSE library     1
// EXEC LIBR,SIZE=200K
ACCESS S=PRD2.PROD                   2
DELETE IGY*.*
DELETE $$$CO18M.Z
DELETE $$$CO18N.Z
DELETE $$$CO18O.Z
/*
/&
```

Figure 9. Job to Remove COBOL/VSE from a Sublibrary

If you have installed COBOL/VSE into a sublibrary other than the default (PRD2.PROD):

- Insert the required DLBL, EXTENT and ASSGN information for the COBOL/VSE library in area **1** .
- Change the statement in area **2** .

To remove COBOL/VSE from the system history file, use the REMOVE command of MSHP. The sample job shown in Figure 10 shows the JCL needed to remove COBOL/VSE from the system history file.

```
// JOB IGYDELH  Remove Product
```

```
// EXEC MSHP,SIZE=900K
REMOVE 5686-068-00-18M      1
REMOVE 5686-068-01-18N      2
REMOVE 5686-068-02-18O      2
/*
/&
```

Figure 10. Job to Remove COBOL/VSE from the System History File

Area **1** shows the component ID for the COBOL/VSE base component. This REMOVE statement should be the first because it removes the COBOL compiler itself. See below for the other REMOVE statements.

Areas **2** show the National Language components of COBOL/VSE. Use REMOVE only for those which have been installed. For example, if you have installed the Japanese Language Feature but not the US English Language Feature, you should include only the REMOVE statement for the Japanese National Language Feature.

1.5.4 How to Report a Problem with COBOL/VSE

For information on how to report a problem with COBOL/VSE, see the *COBOL/VSE Diagnosis Guide*.

IBM Library Server

| 2.0 Planning for, Installing, and Maintaining VisualAge COBOL MLE for VSE

Subtopics:

IBM Library Server

| 2.1 Chapter 6. Planning to Install VisualAge COBOL MLE for VSE

| This chapter provides information for planning the installation of
| VisualAge COBOL MLE for VSE. It includes information on:

| - What you receive with VisualAge COBOL MLE for VSE
| - What you need to install VisualAge COBOL MLE for VSE
| - Checking service updates

Subtopics:

IBM Library Server

2.1.1 What You Receive with VisualAge COBOL MLE for VSE

You receive VisualAge COBOL MLE for VSE in one of two forms:

- As an optional program on the VSE/ESA optional program tape
- As a separately-ordered licensed program on a separate distribution tape

When you receive VisualAge COBOL MLE for VSE, either as an optional program, or as a separately-ordered licensed program, you receive:

- Basic machine-readable material
- Basic unlicensed publications
- Optional unlicensed publications (if you specify the required feature numbers when ordering VisualAge COBOL MLE for VSE)

You receive publications in the national language (U.S. English or Japanese) you specify when you order.

Table 19 describes the component supplied by Release 1 of VisualAge COBOL MLE for VSE.

Table 19. VisualAge COBOL MLE for VSE Component and Component Level Code (CLC)

Component ID	CLC	Component Name
5686-MLE-00	1JQ	VA COBOL MLE for VSE

Subtopics:

IBM Library Server

| 2.1.1.1 Distribution Media

| You will receive VisualAge COBOL MLE for VSE on one of the following:

- 9-track magnetic tape written at 6250 BPI
- IBM 3480 or IBM 3490 cartridge
- 4mm DAT cartridge

| The tape or cartridge contains all the programs and data needed for installation.

Subtopics:

IBM Library Server

| 2.1.1.2 Program Documentation

| This section identifies the basic and optional VisualAge COBOL MLE for VSE
| documentation you receive.

Subtopics:

IBM Library Server

| 2.1.2 What You Need to Install VisualAge COBOL MLE for VSE

| The following sections identify the system requirements for installing and
| activating VisualAge COBOL MLE for VSE.

Subtopics:

IBM Library Server

| 2.1.2.1 Machine Requirements

```
| VisualAge COBOL MLE for VSE runs on any hardware configuration that
| supports COBOL/VSE.
```

IBM Library Server

| 2.1.2.2 Operating System Requirements

| VisualAge COBOL MLE for VSE operates under the same operating systems as
| COBOL/VSE. See "Choosing Required and Optional Licensed Programs" in
| topic 1.1.7 for more information.

IBM Library Server

2.1.2.3 Programming Requirements

VisualAge COBOL MLE for VSE has the following programming requirements:

1. COBOL/VSE Version 1 Release 1 must be installed before VisualAge COBOL MLE for VSE is installed. In addition, the PTFs that resolve the APARs shown in Table 24 must be applied to the components of COBOL/VSE you have installed before VisualAge COBOL MLE for VSE is installed.

Table 24. COBOL/VSE Service Requirements

Component Name	APAR
COBOL/VSE Base	PQ13830 PQ13831
COBOL/VSE US English Language Feature	PQ13832
COBOL/VSE Japanese Language Feature	PQ13834

For information about installing COBOL/VSE, see "Planning for, Installing, Customizing, and Maintaining COBOL/VSE" in topic 1.0.

2. LE/VSE Version 1 Release 4 is required to compile and run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option). In addition, the PTFs that resolve the APARs shown in Table 25 must be applied to the components of LE/VSE you have installed.

Table 25. LE/VSE Service Requirements

Component Name	APAR
LE/VSE COBOL-Specific Base	PQ15106
LE/VSE COBOL-Specific Japanese Language Feature	PQ15112
LE/VSE COBOL-Specific CICS	PQ15114

3. If you have Debug Tool/VSE installed on your system, the PTFs that resolve the APARs shown in Table 26 must be applied to the components of Debug Tool/VSE you have installed before you can use Debug Tool/VSE with programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option).

Table 26. Debug Tool/VSE Service Requirements

Component Name	APAR
Debug Tool/VSE Base	PQ15548 PQ15549
Debug Tool/VSE Japanese Language Feature	PQ15550

4.	High Level Assembler for MVS & VM & VSE might be required to apply service.

In addition to the above requirements, VisualAge COBOL MLE for VSE has the same run-time programming requirements as COBOL/VSE. See "Choosing Required and Optional Licensed Programs" in topic 1.1.7 for more information.

IBM Library Server

| 2.1.2.4 DASD Storage Requirements

| VisualAge COBOL MLE for VSE should be installed in the same sublibrary as
| COBOL/VSE and has only minor storage requirements for installation. For
| more information, see "DASD Storage Required for Installation" in
| topic 1.1.6.1.

IBM Library Server

2.1.2.5 Checking Service Updates

Before installing VisualAge COBOL MLE for VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 27. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLMLE	MLE1JQ

IBM Library Server

| 2.2 Chapter 7. Installing VisualAge COBOL MLE for VSE

| This chapter describes the installation method and the step-by-step
| procedures you use to install VisualAge COBOL MLE for VSE from an optional
| program tape or a separate distribution tape.

Subtopics:

IBM Library Server

| 2.2.1 Overview of Installation

| You install this release of VisualAge COBOL MLE for VSE by using the
| Maintain System History Program (MSHP), or by using the VSE/ESA
| Interactive Interface.

Subtopics:

- 2.2.1.1 List of Installation Steps

IBM Library Server

2.2.1.1 List of Installation Steps

Table 28 lists the steps and associated jobs to install VisualAge COBOL MLE for VSE. The remaining sections in this chapter describe what each job does and how to modify it. You can use Table 28 as a checklist.

Table 28. Summary of Steps for Installing VisualAge COBOL MLE for VSE

Step	Description	Installation Job	Topic
1	Read this book to plan the installation.	-	2.2.2
2	Back up the original system.	-	2.2.3
3	Install VisualAge COBOL MLE for VSE.		2.2.4
	Method 1: Install using the VSE/Interactive Interface.	-	2.2.4.1
	Method 2: Install using a batch job.	IGYMLINS	2.2.4.2
4	Verify VisualAge COBOL MLE for VSE installation.	IGYMLIV1	2.2.5
5	Place VisualAge COBOL MLE for VSE in the SVA.	IGYMLSV1 IGYMLSVA	2.2.6

IBM Library Server

| 2.2.2 Step 1: Read this Book and Plan the Installation

| Read this book and familiarize yourself with all the installation material
| and procedures.

Subtopics:

- 2.2.2.1 Plan the Installation

IBM Library Server

2.2.2.1 Plan the Installation

Before you install VisualAge COBOL MLE for VSE, you should do the following:

1. Verify that you have the required machine and software configuration to run VisualAge COBOL MLE for VSE. See "What You Need to Install VisualAge COBOL MLE for VSE" in topic 2.1.2 for more information.

2. Determine which tape volumes you need for the installation.

   - If you order VisualAge COBOL MLE for VSE as a VSE/ESA optional licensed program, you receive one or more tapes that contain the optional licensed programs you ordered. These tapes are called stacked tapes. The external label on each stacked tape is:

     VSE/ESA x.x.x OPTIONAL PROGRAMS n OF x

   - If you order VisualAge COBOL MLE for VSE separately, you receive one non-stacked tape. The external label on this tape is:

     VA CBL MLE V1R1

3. Determine the name of the sublibrary where COBOL/VSE is installed. VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE.

4. Check on the latest service updates needed. For more information, see "Checking Service Updates" in topic 2.1.2.5.

5. Decide whether you are going to install VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface or an MSHP batch job.

6. Plan any necessary changes to the installation jobs described in this chapter.

IBM Library Server

| 2.2.3 Step 2: Back Up the Original System

| Make a backup copy of the library you intend to install VisualAge COBOL
| MLE for VSE into, and the system history file.

| For information about backing up libraries and the system history file,
| see *VSE/ESA System Control Statements*.

IBM Library Server

| 2.2.4 Step 3: Install VisualAge COBOL MLE for VSE

| You can install VisualAge COBOL MLE for VSE using either the VSE/ESA
| Interactive Interface or a batch installation job.

Subtopics:

IBM Library Server

| 2.2.4.1 Method 1: Install VisualAge COBOL MLE for VSE Using the Interactive Interface

| The VSE/ESA Interactive Interface enables you to use dialog requests to
| install VisualAge COBOL MLE for VSE. For more information about
| installing licensed programs using the Interactive Interface, see *VSE/ESA Installation*.

| To install VisualAge COBOL MLE for VSE using the Interactive Interface, do
| the following:

| 1. Logon to the Interactive Interface as the system administrator.

| 2. Mount the VisualAge COBOL MLE for VSE distribution tape on an
| available tape drive.

| 3. In the following menus, specify the items that appear following the
| ===> symbol.

| a. **VSE/ESA FUNCTION SELECTION** menu:

| ===> 1 (Installation)

| b. **INSTALLATION** menu:

| ===> 1 (Install Programs - V2 Format)

| **Note:** You can use option 1 for both a stacked distribution tape
| containing one or more optional licensed programs, and a
| non-stacked tape containing just VisualAge COBOL MLE for VSE.
| Both are in Librarian V2 format.

| c. **INSTALL PROGRAMS - V2 FORMAT** menu:

| ===> 1 (Prepare for Installation)

| d. **PREPARE FOR INSTALLATION** menu:

| ===> cuu (address of drive with distribution tape)

| e. **JOB DISPOSITION** menu:

Make any changes required and press ENTER to submit the job. While the job is running, reply to any system console messages as necessary.

The output listing from this job lists the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifier for the VisualAge COBOL MLE for VSE component has the format COBOLMLE...1.1.0.

f. Check the output from the batch job created by the previous steps to ensure that the install was successful. Once the batch job created by this step has successfully run, the program identifiers of the optional programs on the distribution tape are also automatically entered on the **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu.

g. After the batch job created by the previous step has successfully run, return to the **VSE/ESA FUNCTION SELECTION** menu:

===> 1 (Installation)

h. **INSTALLATION** menu:

===> 1 (Install Programs - V2 Format)

i. **INSTALL PROGRAMS - V2 FORMAT** menu:

===> 2 (Install Program(s) from Tape)

j. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

Enter 1 (install) in the OPT field against the required identifier of the format COBOLMLE...1.1.0, and 2 (skip installation) against any other optional licensed programs you do not intend to install at this time.

VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE. If you did not install COBOL/VSE in the default sublibrary PRD2.PROD, enter the name of the library and sublibrary where COBOL/VSE is installed on this screen.

k. Press PF5 to generate the installation job.

l. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> 1 (Save the list)

m. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> cuu (address of drive with distribution tape)

n. **JOB DISPOSITION** menu:

Make any changes required and press ENTER to submit the job and install VisualAge COBOL MLE for VSE. While the job is running, reply to any system console messages as necessary.

o. Check the output from the batch job created by the previous steps to ensure that the install was successful. If you are installing VisualAge COBOL MLE for VSE from a non-stacked tape, you received the following message:

```
IESI0083I  TAPE IS NOT V2-STACKED
```

This message is for information only and can be ignored. The Librarian RESTORE job ends with a return code of 4. This is not an error. Installation was successful.

Subtopics:

- 2.2.4.1.1 Condition Code and Messages

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

2.2.4.2 Method 2: Install VisualAge COBOL MLE for VSE Using a Batch Job

A sample batch job to install VisualAge COBOL MLE for VSE using MSHP is shown in Figure 11. This sample job uses the MSHP system history file that already exists as part of the VSE system. This system history file may already be defined in the system standard labels; if not, make sure that DLBL and EXTENT job control statements with the necessary information for the system history file, are included in the job stream.

Depending on how you ordered the VisualAge COBOL MLE for VSE licensed program, you will have received a stacked tape containing one or more optional licensed programs, or a non-stacked tape containing only the VisualAge COBOL MLE for VSE licensed program. The JCL shown in Figure 11 will handle both stacked and non-stacked tapes.

To install VisualAge COBOL MLE for VSE using a batch job, create and tailor the job stream shown in Figure 11, mount the distribution tape, and run the installation job.

There are four main sections in the sample job that you will need to tailor to suit the requirements of your installation. The tailoring requirements for each section are discussed in the notes that follow Figure 11.

```
// JOB IGYMLINS
* ------------------------------------------------------------ *
*   LICENSED MATERIALS - PROPERTY OF IBM                       *
*                                                              *
*   5686-MLE                                                   *
*                                                              *
*   (C) COPYRIGHT IBM CORP. 1998. ALL RIGHTS RESERVED.         *
*                                                              *
*   US GOVERNMENT USERS RESTRICTED RIGHTS - USE,               *
*   DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP            *
*   SCHEDULE CONTRACT WITH IBM CORP.                           *
*                                                              *
* ------------------------------------------------------------ *
* -------------------- Description --------------------------- *
*  Name       : IGYMLINS                                      *
*  Description : Sample job to install VISUALAGE COBOL/VSE MLE  *
*                See the Installation & Customization Guide    *
*                for the tailoring requirements.               *
*                                                              *
* ------------------------------------------------------------ *
// OPTION LOG
*
*  Label information for the COBOL/VSE library if required
*
*  ____________________________________________ 1
*
// DLBL COBVSE,'COBVSE.LIBRARY',99/365,SD
// EXTENT SYS002,volser,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volser,SHR
*
*  Assign for the distribution tape.
*
```

```
// ASSGN SYS006,cuu  _____________________________  2
// MTC REW,SYS006
*  ---------------------------------------------------------------
*  Install VisualAge COBOL MLE for VSE from the distribution tape
*  ---------------------------------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'  _________  3
 INSTALL PROD FROMTAPE ID='COBOLMLE...1.1.0' -
   PROD INTO=PRD2.PROD
/*
*
*  Retrace VisualAge COBOL MLE
*
// EXEC MSHP,SIZE=900K  _________________________  4
RETRACE COMPONENT IDENTIFIER=5686-MLE-00
/*
// MTC RUN,SYS006
/*
/&
```

Figure 11. Job to Install VisualAge COBOL MLE for VSE

1 Specify the label information.

You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a library other than the default (PRD2), code the DLBL job control statement (and EXTENT and ASSGN statements if necessary) of the library in which you installed COBOL/VSE.

Usually, you do not need to specify label information for the system history file. Your installation should have a permanent system standard label for this, with IJSYSHF as the filename. (IJSYSHF is the default filename that MSHP looks for in a label statement.)

2 Assign the distribution tape.

Replace *cuu* with the address of the tape drive on which you have mounted the distribution tape.

3 Install VisualAge COBOL MLE for VSE.

This job step invokes MSHP to install VisualAge COBOL MLE for VSE into the sublibrary identified on the INTO operand of the INSTALL statement. You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a sublibrary other than the default PRD2.PROD, change the name of the sublibrary on the INTO operand of each INSTALL statement to the name of the sublibrary into which you installed COBOL/VSE. For example, if you installed COBOL/VSE into sublibrary COBVSE.BASE, change references to PRD2.PROD to COBVSE.BASE.

For more information about the MSHP install options, see *VSE/ESA System Control Statements*.

4 Retrace the VisualAge COBOL MLE for VSE product in the system history file.

This step prints the component records from the system history file for VisualAge COBOL MLE for VSE. Remove this step if an MSHP retrace listing is not required.

If the install job runs successfully, it should finish with a return code of zero.

If you need to rerun this install job, make sure you first restore the system history file, which you should have backed up before you started the install process.

IBM Library Server

| 2.2.5 Step 4: Verify the Installation of VisualAge COBOL MLE for VSE

| After you install VisualAge COBOL MLE for VSE, run sample job IGYMLIV1 to
| verify automatic date processing in COBOL/VSE.

Subtopics:

- 2.2.5.1 Verify Date Processing in COBOL/VSE

IBM Library Server

| 2.2.5.1 Verify Date Processing in COBOL/VSE

| The sample job IGYMLIV1, in the installation sublibrary member IGYMLIV1.Z, compiles, link-edits, and runs the COBOL program CALLIVPM installed from your distribution tape as sublibrary member IGYMLIVP.C. The object code produced by the COBOL/VSE compiler is link-edited and run using the libraries created when LE/VSE was installed. You must install LE/VSE and apply the appropriate PTFs before running IGYMLIV1.

| **Note:** If you installed VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface dialogs, IGYMLIV1 is also available in ICCF library 62.

| **Modifying the JCL for IGYMLIV1**

| 1. Modify the job card as appropriate for your site.

| 2. Add POWER JECL statements if your site requires them.

| 3. If necessary, change the LIBDEF statement to match the sublibraries where you have installed COBOL/VSE, VisualAge COBOL MLE for VSE, and LE/VSE.

| After you modify the IGYMLIV1 job, submit it. The job finishes with a return code of 2 if it runs successfully. The return code of 2 indicates there were unresolved weak external references during the link-edit step. This return code is normal and does not indicate a problem.

| IGYMLIV1 prints the following messages to SYSLST:

```
|    ***** START OF CALLIVPM *****
|    ***** CALLIVPM SUCCESSFUL *****
```

| **Note:** If VisualAge COBOL MLE for VSE was properly installed, IGYMLIV1 should not produce any LE/VSE run-time messages.

IBM Library Server

2.2.6 Step 5: Place VisualAge COBOL MLE for VSE in the SVA

After you have verified the successful installation of VisualAge COBOL MLE for VSE, you might want to place VisualAge COBOL MLE for VSE in the SVA.

Note: Placing VisualAge COBOL MLE for VSE in the SVA must be done by a system administrator.

To assist you in placing the VisualAge COBOL MLE for VSE phase in the SVA, a sample job is provided in member IGYMLSV1.Z. This sample job is shown in Figure 12. See the notes following Figure 12 for information about tailoring the sample job.

```
// JOB IGYMLSV1 LOAD SVA-ELIGIBLE PHASE
*
*  Load SVA
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
IGYCMLE,SVA
/*
/&
```

Figure 12. Job to Place VisualAge COBOL MLE for VSE in the SVA

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

Alternatively, as shown by the sample job in Figure 13, you can place the VisualAge COBOL MLE for VSE phase in the SVA using a load list. To help you do this, the load list $SVAIGYM is provided. You can also specify this load list in the BG startup procedure. If you choose to specify the load list in the BG startup procedure, the sublibrary where you installed VisualAge COBOL MLE for VSE must be in the LIBDEF search chain within the LIBSDL procedure.

```
// JOB IGYMLSVA LOAD SVA-ELIGIBLE PHASES
*
```

```
*  Load SVA using a load list
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
LIST=$SVAIGYM
/*
/&
```

Figure 13. Job to Place VisualAge COBOL MLE for VSE in the SVA Using a Load List

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

For more information about placing phases in the SVA, see *VSE/ESA System Control Statements*.

IBM Library Server

| 2.3 Chapter 8. Maintaining VisualAge COBOL MLE for VSE

| This chapter describes how to replace or reinstall VisualAge COBOL MLE for VSE, and how to apply service updates to VisualAge COBOL MLE for VSE. To effectively use the maintenance procedures, you must have already installed VisualAge COBOL MLE for VSE and any required products.

| In addition, this chapter describes how to remove VisualAge COBOL MLE for VSE from your system.

Subtopics:

IBM Library Server

2.3.1 Reinstalling VisualAge COBOL MLE for VSE

You do not need to remove VisualAge COBOL MLE for VSE from your system before reinstalling VisualAge COBOL MLE for VSE unless you intend to reinstall the product in a different sublibrary from the previous installation. In this case you must remove VisualAge COBOL MLE for VSE from the system history file before you can reinstall it. However, if you are reinstalling in the same sublibrary, you might need to delete VisualAge COBOL MLE for VSE from the sublibrary and release the space to ensure that there is sufficient library space available for the reinstallation.

If you do need to remove VisualAge COBOL MLE for VSE Release 1 from your system, sample jobs, IGYMLDLV.Z and IGYMLDLH.Z are provided in PRD2.PROD to help you do this.

If you installed VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface dialogs, these jobs will also be available in ICCF library 62.

To reinstall VisualAge COBOL MLE for VSE, follow the installation steps described in Chapter 7, "Installing VisualAge COBOL MLE for VSE" in topic 2.2.

IBM Library Server

| 2.3.2 Applying Service Updates

| You might need to apply maintenance or service updates to VisualAge COBOL
| MLE for VSE periodically. There are two types of formally supported
| software fixes. One is the program temporary fix (PTF) applied as
| corrective service or as preventive maintenance. The other is the
| authorized program analysis report (APAR) fix applied as a code
| replacement in a corrective maintenance mode.

Subtopics:

| 2.3.2.1 What You Receive

| If you report a problem with VisualAge COBOL MLE for VSE to your IBM
| Support Center, you will receive a tape containing one or more APARs or
| PTFs which have been created to solve your problem.

| You may also receive a list of prerequisite APARs or PTFs which should
| have been applied to your system before applying the current service.
| These prerequisite APARs or PTFs may relate to VisualAge COBOL MLE for VSE
| or any other licensed product you have installed, including VSE/ESA.

| You apply service to VisualAge COBOL MLE for VSE with MSHP, using either
| the VSE/ESA Interactive Interface or a batch job.

| The following checklist provides a summary of steps you should use to
| apply service to VisualAge COBOL MLE for VSE.

IBM Library Server

2.3.2.2 Checklist for Applying Service

Table 29 lists the steps for installing corrective service on VisualAge COBOL MLE for VSE. You can use Table 29 as a checklist.

Table 29. Summary of Steps for Installing Service on VisualAge COBOL MLE for VSE

Step	Description	MSHP Command or Jobname	Topic
1	Ensure prerequisite APARs or PTFs are applied	IGYMLRTR	2.3.2.3
2	Backup existing system	-	2.3.2.4
3	Apply service	IGYMLSVC	2.3.2.5
4	Run the installation verification programs	IGYMLIV1	2.3.2.6

IBM Library Server

2.3.2.3 Step 1: Check Prerequisite APARs or PTFs

Prerequisite APARs or PTFs are APARs or PTFs that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to VisualAge COBOL MLE for VSE or any licensed program you have installed at your site.

Your IBM Support Center will have given you a list of any relevant prerequisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 14 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYMLRTR Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 14. Job to Retrace APARs and PTFs

Use the listing produced when you run this job to check that you have already applied any prerequisite APARs or PTFs. If you have not, your IBM Support Center will arrange to send them to you and you should apply them before applying other service.

IBM Library Server

| 2.3.2.4 Step 2: Backup Existing System

| Make a backup copy of the library in which VisualAge COBOL MLE for VSE is
| installed, and the system history file. For information about backing up
| libraries and the system history file, see *VSE/ESA System Control
| Statements*.

IBM Library Server

| 2.3.2.5 Step 3: Apply Service

| You can apply service to VisualAge COBOL MLE for VSE from the provided
service tape using either the Interactive Interface dialogs or a batch
job.

| You will receive detailed instructions for applying service with the
service tape.

Subtopics:

IBM Library Server

| 2.3.2.6 Step 4: Run the Installation Verification Program (IVP)

| After you have applied all the files on the service tape, run the
| appropriate installation verification programs to ensure that VisualAge
| COBOL MLE for VSE functions properly. For information about how to run
| the installation verification programs, see "Step 4: Verify the Installation of VisualAge COBOL MLE for VSE" in topic 2.2.5.

IBM Library Server

| 2.3.3 To Report a Problem with VisualAge COBOL MLE for VSE

| Report any difficulties you have using this product to your IBM Support
| Center. Table 30 identifies the component ID and Release Level for
| VisualAge COBOL MLE for VSE.

Table 30. Component IDs

Component Id	Component Description	REL
5686-MLE-00	VA COBOL MLE for VSE	1JQ

IBM Library Server

BACK_1 Bibliography

Subtopics:

IBM Library Server

BACK_1.1 IBM COBOL for VSE/ESA

General Information, GC26-8068

Migration Guide, GC26-8070

Installation and Customization Guide, SC26-8071

Programming Guide, SC26-8072

Language Reference, SC26-8073

Reference Summary, SX26-3834

Diagnosis Guide, SC26-8528

Licensed Program Specifications, GC26-8069

IBM Library Server

BACK_1.2 IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

Installation and Customization Guide, SC26-8071

COBOL Millennium Language Extensions Guide, GC26-9266

Fact Sheet, GC26-9321

Licensed Program Specifications, GC26-9417

IBM Library Server

BACK_1.3 Language Environment Publications

Fact Sheet, GC33-6679

Concepts Guide, GC33-6680

Installation and Customization Guide, SC33-6682

Programming Guide, SC33-6684

Programming Reference, SC33-6685

Debugging Guide and Run-Time Messages, SC33-6681

Licensed Program Specifications, GC33-6683

Run-Time Migration Guide, SC33-6687

Writing Interlanguage Communication Applications, SC33-6686

C Run-Time Library Reference, SC33-6689

C Run-Time Programming Guide, SC33-6688

BACK_1.4 Related Publications

CICS/VSE

Application Programming Guide, SC33-0712

Application Programming Reference, SC33-0713

System Definition and Operations Guide, SC33-0706

Resource Definition (Online), SC33-0708

Debug Tool for VSE/ESA

User's Guide and Reference, SC26-8797

Installation and Customization Guide, SC26-8798

DFSORT for VSE/ESA

Application Programming Guide, SC26-7040

DL/I DOS/VS

Application Programming: High-Level Programming Interface, SH24-5009

Application Programming: CALL and RQDLI Interfaces, SH12-5411

Sort/Merge II

DOS/VS VM/SP Sort/Merge Version 2 Application Programming Guide, SC33-4044

SQL/DS

Application Programming for VSE, SH09-8098

VSE/ESA Version 1 Release 4

Planning, SC33-6503

Installation and Service, SC33-6504

Administration, SC33-6505

Guide to System Functions, SC33-6511

System Control Statements, SC33-6513

System Macros User's Guide, SC33-6515

System Macros Reference, SC33-6516

System Utilities, SC33-6517

Messages and Codes Vol.1 & 2, SC33-6507

VSE/ESA Version 2

System Upgrade and Service, SC33-6602

Planning, SC33-6603

Installation, SC33-6604

Administration, SC33-6605

Guide to System Functions, SC33-6611

System Control Statements, SC33-6613

System Macros User's Guide, SC33-6615

System Macros Reference, SC33-6616

System Utilities, SC33-6617

Messages and Codes Vol. 1, SC33-6698

Messages and Codes Vol. 2, SC33-6699

VSE/ESA VSAM

VSE/VSAM Commands and Macros, SC33-6532

VSE/VSAM User's Guide, SC33-6535

IBM Library Server

BACK_1.5 Softcopy Publications

These collection kit contains the COBOL/VSE and LE/VSE-conforming language product publications:

VSE Collection, SK2T-0060

IBM Library Server

Index

A

B

C

D

E

F

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Y

Z

IBM Library Server

BACK_2 We'd Like to Hear from You

COBOL for VSE/ESA
VisualAge COBOL Millennium Language Extensions
for VSE/ESA
Installation and Customization Guide

Publication No. SC26-8071-01

Please use one of the following ways to send us your comments about this book:

- Mail--Print and use the Readers' Comments form on the next page. To print the form, select **Print** or **Copy** from the **Services** pull-down menu. Enter *COMMENTS* as the topic to be printed or copied. Mail the completed form to:

  IBM Corporation
  555 Bailey Avenue, W92/H3
  P.O. Box 49023
  San Jose, CA 95161-9023

  If you are sending the form from a country other than the United States, give it to your local IBM branch office or IBM representative for mailing.

- Fax--Print and use the Readers' Comments form on the next page and fax it to this U.S. number: 800-426-7773. To print the form, follow the instructions under "Mail."

- Electronic mail--Use one of the following network IDs:

  - IBMMail: USIB2VVG at IBMMAIL
  - IBMLink: COBPUBS at STLVM27
  - Internet: COBPUBS@VNET.IBM.COM

  Be sure to include the following with your comments:

  - Title and publication number of this book
  - Your name, address, and telephone number if you would like a reply

Your comments should pertain only to the information in this book and the way the information is presented. To request additional publications, or to comment on other IBM information or the function of IBM products, please give your comments to your IBM representative or to your IBM authorized remarketer.

IBM may use or distribute your comments without obligation.

IBM Library Server

COMMENTS Readers' Comments

COBOL for VSE/ESA
VisualAge COBOL Millennium Language Extensions
for VSE/ESA
Installation and Customization Guide

Publication No. SC26-8071-01

How satisfied are you with the information in this book?

Legend:
1 Very satisfied
2 Satisfied
3 Neutral
4 Dissatisfied
5 Very dissatisfied

Please circle the number that corresponds to the level of your satisfaction.

Technically accurate	1	2	3	4	5
Complete	1	2	3	4	5
Easy to find	1	2	3	4	5
Easy to understand	1	2	3	4	5
Well organized	1	2	3	4	5
Applicable to your tasks	1	2	3	4	5
Grammatically correct and consistent	1	2	3	4	5
Graphically well designed	1	2	3	4	5
Overall satisfaction	1	2	3	4	5

May we contact you to discuss your comments? Yes No

Name . . . . . _____
Company or Organization _____
Address . . . . . _____
_____
_____
Phone No. . . . . _____

IBM Library Server

EDITION Edition Notice

Second Edition (June 1998)

This edition applies to:

IBM COBOL for VSE/ESA Version 1 Release 1 Modification 1 (Program Number 5686-068)

IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA Version 1 Release 1 (Program Number 5686-MLE)

and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader's comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation, W92/H3
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

IBM Library Server

Notices

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" in topic FRONT 1.

IBM Library Server

Notices

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" in topic FRONT 1.

IBM Library Server

Title: *COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide*
Document Number: *SC26-8071-01*
Build Date: *05/26/98 04:08:30* *Build Version:* *1.2*
Book Path: *C:\IBMZLIB\BOO\igyvi002.boo*

COVER Book Cover

COBOL for VSE/ESA
VisualAge COBOL Millennium Language Extensions
for VSE/ESA

Installation and Customization Guide

Document Number SC26-8071-01

Program Number
5686-068
5686-MLE

File Number S370-34

IBM Library Server

Title: *COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide*
Document Number: *SC26-8071-01*
Build Date: *05/26/98 04:08:30* ***Build Version:*** *1.2*
Book Path: *C:\IBMZLIB\BOO\igyvi002.boo*

COVER Book Cover

COBOL for VSE/ESA
VisualAge COBOL Millennium Language Extensions
for VSE/ESA

Installation and Customization Guide

Document Number SC26-8071-01

Program Number
5686-068
5686-MLE

File Number S370-34

IBM Library Server

CONTENTS Table of Contents

1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

The following worksheet will help you plan and code the compiler options portion of the IGYCOPT macro. To complete the worksheet, fill in the 'Enter * for Fixed' and the 'Enter Selection' columns.

The IGYCOPT macro worksheet also includes a section for compiler phases. That section of the worksheet can be found in "IGYCOPT Macro Worksheet for Compiler Phases" in topic 1.2.3.21.1, following the discussion of compiler phases.

Notes:

1. Coding the asterisk [*], when modifying a compiler option default value, indicates that the option is to be fixed and cannot be replaced by an application programmer.

| 2. The ALOWCBL, DBCSXREF, LVLINFO, and NUMCLS options cannot be overridden at compile time. The 'Enter * for Fixed' worksheet entries for these options is therefore blank.

3. The IBM-supplied default value for ADEXIT, INEXIT, LVLINFO, LIBEXIT, and PRTEXIT is null. The 'IBM-Supplied Default' worksheet entries for these options are therefore blank.

4. The DUMP compiler option cannot be set through the IGYCOPT macro. Unless changed at compile time, DUMP is always set to NODUMP.

Table 11. IGYCOPT Worksheet for Options

Compiler Option	Enter * for Fixed	Enter Selection	IBM-Supplied Default	Syntax Description
ADATA=	____	_____	*NO*	Topic 1.2.5.5
ADEXIT=	____	_____	Not specified	Topic 1.2.5.6
ADV=	____	_____	*YES*	Topic 1.2.5.7
ALOWCBL=		_____	*YES*	Topic 1.2.5.8
AWO=	____	_____	*NO*	Topic 1.2.5.9
BUF=	____	_____	*4K*	Topic 1.2.5.10
CMPR2=	____	_____	*NO*	Topic 1.2.5.11
COMPILE=	____	_____	*NOC(S)*	Topic 1.2.5.12
CURRENCY=	____	_____	*NO*	Topic 1.2.5.13
DATA=	____	_____	*31*	Topic 1.2.5.14

DATEPROC=	____	_____	*NO*	Topic 1.2.5.15
DBCS=	____	_____	*NO*	Topic 1.2.5.16
DBCSXREF=		_____	*NO*	Topic 1.2.5.17
DYNAM=	____	_____	*NO*	Topic 1.2.5.18
FASTSRT=	____	_____	*NO*	Topic 1.2.5.19
FLAG=	____	_____	*(I)*	Topic 1.2.5.20
FLAGMIG=	____	_____	*NO*	Topic 1.2.5.21
FLAGSAA=	____	_____	*NO*	Topic 1.2.5.22
FLAGSTD=	____	_____	*NO*	Topic 1.2.5.23
INTDATE=	____	_____	*ANSI*	Topic 1.2.5.25
INEXIT=	____	_____	Not specified	Topic 1.2.5.24
LANGUAGE=	____	_____	*UE*	Topic 1.2.5.26
LIB=	____	_____	*NO*	Topic 1.2.5.27
LIBEXIT=	____	_____	Not specified	Topic 1.2.5.28
LINECNT=	____	_____	*60*	Topic 1.2.5.29
LIST=	____	_____	*NO*	Topic 1.2.5.30
LITCHAR=	____	_____	*QUOTE*	Topic 1.2.5.31
LVLINFO=		_____	Not specified	Topic 1.2.5.32
MAP=	____	_____	*NO*	Topic 1.2.5.33
NAME=	____	_____	*NO*	Topic 1.2.5.34
NUM=	____	_____	*NO*	Topic 1.2.5.35
NUMCLS=		_____	*PRIM*	Topic 1.2.5.36
NUMPROC=	____	_____	*NOPFD*	Topic 1.2.5.37
OFFSET=	____	_____	*NO*	Topic 1.2.5.38
OPT=	____	_____	*NO*	Topic 1.2.5.39
OUTDD=	____	_____	*SYSOUT*	Topic 1.2.5.40
PRTEXIT=	____	_____	Not specified	Topic 1.2.5.41
RENT=	____	_____	*NO*	Topic 1.2.5.42
RMODE=	____	_____	*AUTO*	Topic 1.2.5.43
SEQ=	____	_____	*YES*	Topic 1.2.5.44
SIZE=	____	_____	*MAX*	Topic 1.2.5.45
SOURCE=	____	_____	*YES*	Topic 1.2.5.46
SPACE=	____	_____	*1*	Topic 1.2.5.47
SSRANGE=	____	_____	*NO*	Topic 1.2.5.48
TERM=	____	_____	*NO*	Topic 1.2.5.49
TEST=	____	_____	*NO*	Topic 1.2.5.50
TRUNC=	____	_____	*STD*	Topic 1.2.5.51
VBREF=	____	_____	*NO*	Topic 1.2.5.52

	WORD=	____	_____	*NO	Topic 1.2.5.53
	XREFOPT=	____	_____	NO	Topic 1.2.5.54
|	YRWINDOW=	____	_____	1900	Topic 1.2.5.55
	ZWB=	____	_____	YES	Topic 1.2.5.56

IBM Library Server

1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

The following worksheet will help you plan and code the phases portion of the IGYCOPT macro. Circle the value you plan to assign to each phase. For more information on the values that can be assigned to each phase, see "Compiler Phases and Their Defaults" in topic 1.2.3.2.

Note: All phase defaults are initially set to *IN*.

Table 13. IGYCOPT Macro Worksheet for Compiler Phases

Phase	Circle Selection	Syntax Description
ASM1=	*IN* / OUT	Topic 1.2.3.3
ASM2=	*IN* / OUT	Topic 1.2.3.4
DIAG=	*IN* / OUT	Topic 1.2.3.5
DMAP=	*IN* / OUT	Topic 1.2.3.6
FGEN=	*IN* / OUT	Topic 1.2.3.7
INIT=	*IN* / OUT	Topic 1.2.3.8
LIBO=	*IN* / OUT	Topic 1.2.3.9
LIBR=	*IN* / OUT	Topic 1.2.3.10
LSTR=	*IN* / OUT	Topic 1.2.3.11
MSGT=	*IN* / OUT	Topic 1.2.3.12
OPTM=	*IN* / OUT	Topic 1.2.3.13
OSCN=	*IN* / OUT	Topic 1.2.3.14
PGEN=	*IN* / OUT	Topic 1.2.3.15
RCTL=	*IN* / OUT	Topic 1.2.3.16
RWT=	*IN* / OUT	Topic 1.2.3.17
SAW=	*IN* / OUT	Topic 1.2.3.18
SCAN=	*IN* / OUT	Topic 1.2.3.19
SIMD=	*IN* / OUT	Topic 1.2.3.20
XREF=	*IN* / OUT	Topic 1.2.3.21

IBM Library Server

2.1.1.1.1 Basic Machine-Readable Material

Table 20 describes the separately-ordered licensed program distribution tape.

Table 20. Basic Material: Separately-Ordered Distribution Tape

National Language	Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
US English(1)	6250 tape	5801	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5802	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6300	1 of 1	VA CBL MLE V1R1	unlabeled
Japanese(1)	6250 tape	5811	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5812	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6301	1 of 1	VA CBL MLE V1R1	unlabeled

Note:

1. There is no difference between the program tapes for the US English Language Feature and the Japanese Language Feature. The different feature numbers are used to specify the national language of the program documentation.

The file content of the distribution tape you receive depends upon the

form in which you receive VisualAge COBOL MLE for VSE. If you receive VisualAge COBOL MLE for VSE on a VSE/ESA optional program tape, there might be other licensed programs on the tape. If you ordered VisualAge COBOL MLE for VSE separately, the distribution tape will contain only VisualAge COBOL MLE for VSE. Table 21 describes the file content of the separately-ordered licensed program distribution tape.

Table 21. File Content: Separately-Ordered Licensed Program Distribution Tape

File	Description
1	Header file containing the VisualAge COBOL MLE for VSE copyright statement
2	Backup file ID COBOLMLE...1.1.0. followed by an MSHP System History File
3	VisualAge COBOL MLE for VSE product
4	Tape Mark
5	End of backup (EOB) record
6	Tape Mark

IBM Library Server

2.1.1.2.1 Basic Unlicensed Program Publications

Table 22 identifies the basic program publications for VisualAge COBOL MLE for VSE. Unless otherwise noted, you receive one copy of each of these publications when you receive the basic materials for VisualAge COBOL MLE for VSE. For additional copies, contact your IBM representative.

Table 22. Basic VisualAge COBOL MLE for VSE Documentation

Title	Order Number
VisualAge COBOL MLE for VSE Licensed Program Specifications	GC26-9417
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

2.1.1.2.3 Optional Documentation

Table 23. Optional VisualAge COBOL MLE for VSE Documentation

Title	Order Number
COBOL/VSE Language Reference	SC26-8073
COBOL/VSE Programming Guide	SC26-8072
COBOL Millennium Language Extensions Guide	GC26-9266

IBM Library Server

Tables

IBM Library Server

1.3.2.3.2 Step 3b: Install COBOL/VSE using a Batch Job

The batch installation job for installing COBOL/VSE uses the MSHP system history file that already exists in the VSE system. This file may already be defined in the system standard labels. If it is not defined, make sure that appropriate DLBL and EXTENT statements are included in the job stream.

Figure 5 shows sample JCL for installing COBOL/VSE which will handle both types of distribution tape.

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. Tailor the areas in the chosen sample job that are flagged with reference keys, such as **1** . Information on how to tailor these areas is provided following Figure 5.

Mount the distribution tape and run the installation job.

```
// JOB INSTALL 5686-068 COBOL/VSE COMPILER
*  Label for the Product Library                    1
*  Assign the install tape as SYS006                2
// ASSGN SYS006,cuu
// MTC REW,SYS006
* ----------------------------------------
*  This step installs the distribution tape
*  using the VSE system history file                3
* ----------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'
INSTALL PRODUCT FROMTAPE ID='COB.BASE...1.1.0' -
         PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.ENU....1.1.0' -    4
         PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.JPN....1.1.0' -    4
         PRODUCTION  INTO=PRD2.PROD
/*
* ------------------------------------
*  This step lists the Product Library              5
* ------------------------------------
// EXEC LIBR
 LISTDIR SUBLIB=PRD2.PROD -
     OUTPUT=NORMAL -
     UNIT=SYSLST
/*
* ------------------------------------
*  Retrace the COBOL/VSE product                    6
```

```
* ------------------------------------
// EXEC MSHP,SIZE=900K
RETRACE COMPONENT IDENTIFIER=5686-068-00
RETRACE COMPONENT IDENTIFIER=5686-068-01
RETRACE COMPONENT IDENTIFIER=5686-068-02
/*
// MTC  RUN,SYS006
/*
/&
```

Figure 5. Job to Install COBOL/VSE Compiler (5686-068)

Specify the Label Information: In area **1** , if you are installing COBOL/VSE in a sublibrary other than the default, insert the DLBL, EXTENT and ASSGN information as specified in Figure 4 in topic 1.3.2.2. The library name must match the name used in the allocation job in Figure 4 in topic 1.3.2.2.

Assign the Distribution Tape: Assign the distribution tape in area **2** to logical unit SYS006. Replace *cuu* with the address of the tape drive on which to mount the distribution tape.

Install COBOL/VSE: Area **3** of the job executes MSHP to install COBOL/VSE into the VSE Librarian sublibraries identified by the INTO operands of the INSTALL statements. If you are installing COBOL/VSE into sublibraries other than the installation default, change the names of the sublibraries specified on the INTO operands of the INSTALL statements.

For more information about the install options, see *VSE/ESA System Control Statements*.

National Language Feature: Choose the national language feature or features you want. Do this by examining the areas indicated by **4** . Remove the language or languages that you do not want.

COB.ENU....1.1.0 US English Language Feature (mixed-case)

COB.JPN....1.1.0 Japanese Language Feature

List the Directory Entries: The step in area **5** lists the directory entries of the VSE Librarian sublibraries where COBOL/VSE was installed. Remove this step if a directory list is not required.

If you are installing COBOL/VSE into sublibraries other than the installation default, change the name of the sublibrary specified on the SUBLIB operand of the LISTDIR statement.

The COBOL/VSE entries have a three character prefix of *IGY*, except for $$$CO18M.Z, $$$CO18N.Z, and $$$CO18O.Z.

Retrace the COBOL/VSE product in the system history file: The final step in area **6** of the job prints the component records from the system history file for COBOL/VSE. Remove this step if you do not want a retrace listing.

Note: If you use this step, you may remove the RETRACE statements that are not required.
Here is what the three RETRACE COMPONENT statements represent:

5686-068-00 Retracing the COBOL/VSE base component

5686-068-01 Retracing the COBOL/VSE US English Language Feature

5686-068-02 Retracing the COBOL/VSE Japanese Language Feature

If this job has to be rerun, remember first to restore the system history file, which should have been backed up before running this installation job, and to rerun the library allocation step, if applicable.

1.4.4.1.1 Coding Control Statements

To create a reserved word table, you must understand the syntax rules for the control statements and for the operands within the control statements.

Figure 6 illustrates the format for coding reserved word processor control statements.

```
ABBR   reserved-word: user-word [comments]
     [reserved-word: user-word [comments]]

.
.
.
INFO   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
RSTR   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
```

Figure 6. Syntax Format for Reserved Word Processor Control Statements

As shown in Figure 6, keywords you can use are:

ABBR to specify an alternative form of an existing reserved word

INFO to specify words that are to be flagged with an informational message whenever they are used in a program

RSTR to specify words that are to be flagged with an error message whenever a program employs them

Note: All words you identify with the control statement keywords INFO and RSTR will be flagged with a message in the source listing of the COBOL/VSE program that uses them. Words that are abbreviated will not be flagged in the source listing unless you have also specified them on the INFO or RSTR control statements.

IBM Library Server

1.5.2.2.1 Step 1. Check Pre-requisite APARs or PTFs

Pre-requisite APARs or PTFs are those that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to COBOL/VSE or any licensed program you have installed.

Your IBM Support Center will have given you a list of any relevant pre-requisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 7 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYRETR  Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 7. Job to Retrace APARs and PTFs

Use the resulting listing to check that you have already applied any pre-requisite APARs or PTFs. If you have not, your IBM Support Center will send them to you, and you should apply them before applying any other service.

1.5.2.2.5 Step 3b: Apply Service Using a Batch Job

The batch job to apply service to COBOL/VSE uses the MSHP system history file where COBOL/VSE was installed.

A sample job to apply service using the Maintain System History Program (MSHP) is shown in Figure 8. For more information on MSHP see *VSE/ESA System Control Statements*.

```
// JOB IGYAPP   Apply Service
// ASSGN SYS006,cuu              1
// EXEC MSHP,SIZE=900K
INSTALL SERVICE FROMTAPE         2
/*
/&
```

Figure 8. Job to Apply Service

In area **1** change *cuu* to the address of the tape drive where you have mounted the service tape.

Area **2** shows the MSHP statement to install service from a tape. The information in the system history file will direct MSHP to apply the service to the sublibrary in which COBOL/VSE is installed.

| 2.3.2.5.2 Method 2: Apply Service Using a Batch Job

| The batch job to apply service to VisualAge COBOL MLE for VSE uses the
| MSHP system history file to determine where VisualAge COBOL MLE for VSE
| was installed.

| A sample job to apply service using the Maintain System History Program
| (MSHP) is shown in Figure 15. For more information on MSHP see *VSE/ESA*
| *System Control Statements*.

```
| // JOB IGYMLSVC
| // ASSGN SYS006,cuu _____________________________ 1
| // EXEC MSHP,SIZE=900K
| INSTALL SERVICE FROMTAPE  _____________________ 2
| /*
| /&
```

| Figure 15. Job to Apply Service

| **1** Specify the address of the tape drive.

| Change *cuu* to the address of the tape drive where you have mounted
| the service tape.

| **2** Install the service from tape.

| This MSHP statement indicates that the service will be installed from
| tape.

| The information in the system history file will direct MSHP to apply
| the service to the sublibrary in which VisualAge COBOL MLE for VSE is
| installed. You do not need to supply this information.

IBM Library Server

Tables

IBM Library Server

Figures

FRONT_1.1 Programming Interface Information

This *Installation and Customization Guide* documents information NOT intended to be used as Programming Interfaces of IBM COBOL for VSE/ESA or IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA.

IBM Library Server

FRONT_1.2 Trademarks

The following terms are trademarks of the IBM Corporation in the United States and/or other countries or both:

BookManager	IBM	Systems Application Architecture
CICS	S/370	VisualAge
CICS/VSE	SAA	VSE/ESA
DFSORT	SQL/DS	

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

IBM Library Server

Figures

IBM Library Server

FRONT_1 Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Subtopics:

FRONT_1 Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Subtopics:

IBM Library Server

FRONT_1.1 Programming Interface Information

This *Installation and Customization Guide* documents information NOT intended to be used as Programming Interfaces of IBM COBOL for VSE/ESA or IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA.

IBM Library Server

FRONT_1.1 Programming Interface Information

This *Installation and Customization Guide* documents information NOT intended to be used as Programming Interfaces of IBM COBOL for VSE/ESA or IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA.

IBM Library Server

FRONT_1.2 Trademarks

The following terms are trademarks of the IBM Corporation in the United States and/or other countries or both:

BookManager	IBM	Systems Application Architecture
CICS	S/370	VisualAge
CICS/VSE	SAA	VSE/ESA
DFSORT	SQL/DS	

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

IBM Library Server

FRONT_2.1 How to Read the Syntax Diagrams

Throughout this book, syntax for the compiler options is described using the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following table shows the meaning of symbols at the beginning and end of syntax diagram lines.

Symbol	**Indicates**
>>-	the syntax diagram starts here
->	the syntax diagram is continued on the next line
>-	the syntax diagram is continued from the previous line
-><	the syntax diagram ends here

Diagrams of syntactical units other than complete statements start with the >- symbol and end with the -> symbol.

- Required items appear on the horizontal line (the main path).

```
>>__STATEMENT__required item_____________________________________><
```

- Optional items appear below the main path.

```
>>__STATEMENT__ _______________ ____________________________________><
              |_optional item_|
```

- When you can choose from two or more items, they appear vertically in a stack.

If you **must** choose one of the items, one item of the stack appears on the main path.

```
>>__STATEMENT__ _required choice 1_ ____________________________><
              |_required choice 2_|
```

If choosing one of the items is optional, the entire stack appears below the main path.

```
>>__STATEMENT__ _____________________ ____________________________><
              |_optional choice 1_|
              |_optional choice 2_|
```

- An arrow returning to the left above the main line indicates an item

that can be repeated.

```
          <_,_____________
>>__STATEMENT____repeatable item_|_____________________________________><
```

A repeat arrow above a stack indicates that you can make more than one choice from the stacked items, or repeat a single choice.

- Keywords appear in uppercase letters (for example, PRINT). They must be spelled exactly as shown. Variables appear in all lowercase letters (for example, item). They represent user-supplied names or values.

- If punctuation marks, parentheses, arithmetic operators, or such symbols are shown, they must be entered as part of the syntax.

- Use at least one blank or comma to separate parameters.

IBM Library Server

FRONT_2.2 Using the Macro Planning Worksheets

The planning worksheets in this book will help you prepare to customize COBOL/VSE. By completing them, you will be able to easily identify those options that you wish to change from the IBM-supplied default values. You might also wish to use the worksheets as a source from which to actually customize the IBM-supplied default values.

The headings in each worksheet may differ slightly from each other. Refer to the following list of definitions for an explanation of each specific column heading. Worksheets are located on topics 1.2.2.2.1 and 1.2.3.21.1.

Option
The OPTION column identifies the options contained within a specific installation macro. This column represents the options exactly as they are in the macro.

Fixed
The FIXED column is used to identify the options that can not be overridden by an application programmer. Enter an asterisk [*] into the **Enter * for Fixed** only for those options that you want to be fixed.

Selection
The SELECTION column is for you to identify the value associated with each option. In the space provided, enter the value you want to assign to each option. Use the topic reference column to locate the specific information about the option that will assist you in selecting the appropriate value.

IBM-Supplied Default
The IBM-SUPPLIED DEFAULT column identifies the value supplied for the specified installation macro if the option is not altered. If the IBM-supplied default is identical to the value you desire for installation, you need not modify that option within that specific macro.

Syntax Description
The SYNTAX DESCRIPTION column identifies the topic in which you will find the syntax diagram for and more specific information about the option.

Once you have completed the worksheets, identify those options that are different from the IBM-supplied defaults. These are the items you must code in the installation macros. The worksheet entries have been positioned such that the order of the entries will remain consistent with the actual coding semantics.

IBM Library Server

FRONT_1.2 Trademarks

The following terms are trademarks of the IBM Corporation in the United States and/or other countries or both:

BookManager	IBM	Systems Application Architecture
CICS	S/370	VisualAge
CICS/VSE	SAA	VSE/ESA
DFSORT	SQL/DS	

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

IBM Library Server

FRONT_2 About This Book

This book is for systems programmers who are responsible for installing and customizing either or both of the products:

IBM COBOL for VSE/ESA
IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

for their location. This book provides information needed to plan for, install, and customize COBOL/VSE and VisualAge COBOL Millennium Language Extensions (MLE) for VSE/ESA under VSE/ESA. In addition, this book may help you to assess the value of COBOL/VSE and VisualAge COBOL MLE for VSE to your organization.

In this book, the generic term operating system is used when referring to VSE/ESA.

You should have a knowledge of COBOL/VSE and of your system's operating environment to use this book and ensure a successful installation of COBOL/VSE and VisualAge COBOL MLE for VSE.

Important

COBOL/VSE is distributed as two offerings:

- A full function offering, which includes Debug Tool/VSE
- An alternate function offering, which does not include Debug Tool/VSE

This book does **not** provide information needed to plan for, install, or customize Debug Tool/VSE. For information about installing Debug Tool/VSE, see *Debug Tool/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

FRONT_2 About This Book

This book is for systems programmers who are responsible for installing and customizing either or both of the products:

IBM COBOL for VSE/ESA
IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

for their location. This book provides information needed to plan for, install, and customize COBOL/VSE and VisualAge COBOL Millennium Language Extensions (MLE) for VSE/ESA under VSE/ESA. In addition, this book may help you to assess the value of COBOL/VSE and VisualAge COBOL MLE for VSE to your organization.

In this book, the generic term operating system is used when referring to VSE/ESA.

You should have a knowledge of COBOL/VSE and of your system's operating environment to use this book and ensure a successful installation of COBOL/VSE and VisualAge COBOL MLE for VSE.

Important

COBOL/VSE is distributed as two offerings:

- A full function offering, which includes Debug Tool/VSE
- An alternate function offering, which does not include Debug Tool/VSE

This book does **not** provide information needed to plan for, install, or customize Debug Tool/VSE. For information about installing Debug Tool/VSE, see *Debug Tool/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

FRONT_2.1 How to Read the Syntax Diagrams

Throughout this book, syntax for the compiler options is described using the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following table shows the meaning of symbols at the beginning and end of syntax diagram lines.

Symbol	**Indicates**
>>-	the syntax diagram starts here
->	the syntax diagram is continued on the next line
>-	the syntax diagram is continued from the previous line
-><	the syntax diagram ends here

Diagrams of syntactical units other than complete statements start with the >- symbol and end with the -> symbol.

- Required items appear on the horizontal line (the main path).

```
>>__STATEMENT__required item_____________________________________><
```

- Optional items appear below the main path.

```
>>__STATEMENT__ _______________ _____________________________________><
              |_optional item_|
```

- When you can choose from two or more items, they appear vertically in a stack.

If you **must** choose one of the items, one item of the stack appears on the main path.

```
>>__STATEMENT__ _required choice 1_ ____________________________><
              |_required choice 2_|
```

If choosing one of the items is optional, the entire stack appears below the main path.

```
>>__STATEMENT__ _____________________ _____________________________><
              |_optional choice 1_|
              |_optional choice 2_|
```

- An arrow returning to the left above the main line indicates an item

that can be repeated.

```
          <_,_____________
>>__STATEMENT____repeatable item_|_____________________________________><
```

A repeat arrow above a stack indicates that you can make more than one choice from the stacked items, or repeat a single choice.

- Keywords appear in uppercase letters (for example, PRINT). They must be spelled exactly as shown. Variables appear in all lowercase letters (for example, item). They represent user-supplied names or values.

- If punctuation marks, parentheses, arithmetic operators, or such symbols are shown, they must be entered as part of the syntax.

- Use at least one blank or comma to separate parameters.

IBM Library Server

FRONT_2.1 How to Read the Syntax Diagrams

Throughout this book, syntax for the compiler options is described using the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following table shows the meaning of symbols at the beginning and end of syntax diagram lines.

Symbol	**Indicates**
>>-	the syntax diagram starts here
->	the syntax diagram is continued on the next line
>-	the syntax diagram is continued from the previous line
-><	the syntax diagram ends here

Diagrams of syntactical units other than complete statements start with the >- symbol and end with the -> symbol.

- Required items appear on the horizontal line (the main path).

```
>>__STATEMENT__required item_____________________________________><
```

- Optional items appear below the main path.

```
>>__STATEMENT__ _______________ ____________________________________><
              |_optional item_|
```

- When you can choose from two or more items, they appear vertically in a stack.

If you **must** choose one of the items, one item of the stack appears on the main path.

```
>>__STATEMENT__ _required choice 1_ ____________________________><
              |_required choice 2_|
```

If choosing one of the items is optional, the entire stack appears below the main path.

```
>>__STATEMENT__ ___________________ ____________________________><
              |_optional choice 1_|
              |_optional choice 2_|
```

- An arrow returning to the left above the main line indicates an item

that can be repeated.

```
          <_,_____________
>>__STATEMENT____repeatable item_|_____________________________________><
```

A repeat arrow above a stack indicates that you can make more than one choice from the stacked items, or repeat a single choice.

- Keywords appear in uppercase letters (for example, PRINT). They must be spelled exactly as shown. Variables appear in all lowercase letters (for example, item). They represent user-supplied names or values.

- If punctuation marks, parentheses, arithmetic operators, or such symbols are shown, they must be entered as part of the syntax.

- Use at least one blank or comma to separate parameters.

IBM Library Server

FRONT_2.2 Using the Macro Planning Worksheets

The planning worksheets in this book will help you prepare to customize COBOL/VSE. By completing them, you will be able to easily identify those options that you wish to change from the IBM-supplied default values. You might also wish to use the worksheets as a source from which to actually customize the IBM-supplied default values.

The headings in each worksheet may differ slightly from each other. Refer to the following list of definitions for an explanation of each specific column heading. Worksheets are located on topics 1.2.2.2.1 and 1.2.3.21.1.

Option
The OPTION column identifies the options contained within a specific installation macro. This column represents the options exactly as they are in the macro.

Fixed
The FIXED column is used to identify the options that can not be overridden by an application programmer. Enter an asterisk [*] into the **Enter * for Fixed** only for those options that you want to be fixed.

Selection
The SELECTION column is for you to identify the value associated with each option. In the space provided, enter the value you want to assign to each option. Use the topic reference column to locate the specific information about the option that will assist you in selecting the appropriate value.

IBM-Supplied Default
The IBM-SUPPLIED DEFAULT column identifies the value supplied for the specified installation macro if the option is not altered. If the IBM-supplied default is identical to the value you desire for installation, you need not modify that option within that specific macro.

Syntax Description
The SYNTAX DESCRIPTION column identifies the topic in which you will find the syntax diagram for and more specific information about the option.

Once you have completed the worksheets, identify those options that are different from the IBM-supplied defaults. These are the items you must code in the installation macros. The worksheet entries have been positioned such that the order of the entries will remain consistent with the actual coding semantics.

IBM Library Server

FRONT_3.1 Major Changes to COBOL/VSE

Subtopics:

IBM Library Server

FRONT_3.2 Major Changes to the Documentation

Subtopics:

IBM Library Server

FRONT_2.2 Using the Macro Planning Worksheets

The planning worksheets in this book will help you prepare to customize COBOL/VSE. By completing them, you will be able to easily identify those options that you wish to change from the IBM-supplied default values. You might also wish to use the worksheets as a source from which to actually customize the IBM-supplied default values.

The headings in each worksheet may differ slightly from each other. Refer to the following list of definitions for an explanation of each specific column heading. Worksheets are located on topics 1.2.2.2.1 and 1.2.3.21.1.

Option
The OPTION column identifies the options contained within a specific installation macro. This column represents the options exactly as they are in the macro.

Fixed
The FIXED column is used to identify the options that can not be overridden by an application programmer. Enter an asterisk [*] into the **Enter * for Fixed** only for those options that you want to be fixed.

Selection
The SELECTION column is for you to identify the value associated with each option. In the space provided, enter the value you want to assign to each option. Use the topic reference column to locate the specific information about the option that will assist you in selecting the appropriate value.

IBM-Supplied Default
The IBM-SUPPLIED DEFAULT column identifies the value supplied for the specified installation macro if the option is not altered. If the IBM-supplied default is identical to the value you desire for installation, you need not modify that option within that specific macro.

Syntax Description
The SYNTAX DESCRIPTION column identifies the topic in which you will find the syntax diagram for and more specific information about the option.

Once you have completed the worksheets, identify those options that are different from the IBM-supplied defaults. These are the items you must code in the installation macros. The worksheet entries have been positioned such that the order of the entries will remain consistent with the actual coding semantics.

IBM Library Server

FRONT_3 Summary of Changes

This section lists the major changes that have been made to the IBM COBOL for VSE/ESA product and this manual since Release 1. Technical changes are marked in the text by a change bar in the left margin.

Subtopics:

IBM Library Server

| FRONT_3.1.1 Service Update to Release 1, June 1998

| • When used in conjunction with IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA, support for automatic date processing (century windowing for dates containing 2-digit years).

| • New language elements in support of automatic date processing:

| - DATE FORMAT clause in data description entries

| - Intrinsic functions:

| - DATEVAL
| - UNDATE
| - YEARWINDOW

| • New compiler options in support of automatic date processing:

| - DATEPROC/NODATEPROC
| - YEARWINDOW

| • INTDATE, added as an installation option by PTF, is now a standard compiler option, not just an installation option.

| • New date intrinsic functions to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - DATE-TO-YYYYMMDD
| - DAY-TO-YYYYDDD
| - YEAR-TO-YYYY

| • Extension of the ACCEPT statement to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - ACCEPT FROM DATE YYYYMMDD
| - ACCEPT FROM DAY YYYYDDD

IBM Library Server

FRONT_3 Summary of Changes

This section lists the major changes that have been made to the IBM COBOL for VSE/ESA product and this manual since Release 1. Technical changes are marked in the text by a change bar in the left margin.

Subtopics:

IBM Library Server

FRONT_3.1 Major Changes to COBOL/VSE

Subtopics:

IBM Library Server

FRONT_3.1 Major Changes to COBOL/VSE

Subtopics:

IBM Library Server

| FRONT_3.1.1 Service Update to Release 1, June 1998

| • When used in conjunction with IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA, support for automatic date processing (century windowing for dates containing 2-digit years).

| • New language elements in support of automatic date processing:

| - DATE FORMAT clause in data description entries

| - Intrinsic functions:

| - DATEVAL
| - UNDATE
| - YEARWINDOW

| • New compiler options in support of automatic date processing:

| - DATEPROC/NODATEPROC
| - YEARWINDOW

| • INTDATE, added as an installation option by PTF, is now a standard compiler option, not just an installation option.

| • New date intrinsic functions to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - DATE-TO-YYYYMMDD
| - DAY-TO-YYYYDDD
| - YEAR-TO-YYYY

| • Extension of the ACCEPT statement to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - ACCEPT FROM DATE YYYYMMDD
| - ACCEPT FROM DAY YYYYDDD

IBM Library Server

| FRONT_3.2.1 Second Edition, June 1998

| • Information needed to plan for and install IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA has been included in this manual.

| • Information about the following new COBOL/VSE installation options has been added:

| - DATEPROC
| - INTDATE
| - YRWINDOW

IBM Library Server

| FRONT_3.1.1 Service Update to Release 1, June 1998

| • When used in conjunction with IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA, support for automatic date processing (century windowing for dates containing 2-digit years).

| • New language elements in support of automatic date processing:

| - DATE FORMAT clause in data description entries

| - Intrinsic functions:

| - DATEVAL
| - UNDATE
| - YEARWINDOW

| • New compiler options in support of automatic date processing:

| - DATEPROC/NODATEPROC
| - YEARWINDOW

| • INTDATE, added as an installation option by PTF, is now a standard compiler option, not just an installation option.

| • New date intrinsic functions to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - DATE-TO-YYYYMMDD
| - DAY-TO-YYYYDDD
| - YEAR-TO-YYYY

| • Extension of the ACCEPT statement to cover the recommendation in the *Working Draft for Proposed Revision of ISO 1989:1985 Programming Language COBOL*:

| - ACCEPT FROM DATE YYYYMMDD
| - ACCEPT FROM DAY YYYYDDD

IBM Library Server

FRONT_3.2 Major Changes to the Documentation

Subtopics:

- FRONT 3.2.1 Second Edition, June 1998

IBM Library Server

FRONT_3.2 Major Changes to the Documentation

Subtopics:

- FRONT 3.2.1 Second Edition, June 1998

IBM Library Server

| FRONT_3.2.1 Second Edition, June 1998

| • Information needed to plan for and install IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA has been included in this manual.

| • Information about the following new COBOL/VSE installation options has been added:

| - DATEPROC
| - INTDATE
| - YRWINDOW

IBM Library Server

1.1 Chapter 1. Planning to Install COBOL/VSE

This chapter provides information for planning the installation of COBOL/VSE. For information about planning the installation of VisualAge COBOL MLE for VSE, see "Planning for, Installing, and Maintaining VisualAge COBOL MLE for VSE" in topic 2.0. This chapter includes information on:

- What you receive with COBOL/VSE
- Ensuring that you have the required machine configuration
- Ensuring that you have enough storage for COBOL/VSE
- Choosing required and optional programs
- Considerations before installing

Subtopics:

1.2 Chapter 2. Planning to Customize COBOL/VSE

This chapter provides the following information for planning the customization of COBOL/VSE:

- Making changes after installation--why customize?
- Planning to modify compiler option default values
- Planning to place compiler phases in the Shared Virtual Area (SVA)
- Planning to create an additional reserved word table

These topics provide the information you need to plan the customization of COBOL/VSE. For information on how to make the changes to COBOL/VSE that are covered in this chapter, see Chapter 4, "Customizing COBOL/VSE" in topic 1.4.

This chapter also contains worksheets that help you to plan modifications to the IBM-supplied default values within macros. An explanation of the planning worksheets in this book is in "Using the Macro Planning Worksheets" in topic FRONT_2.2.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

IBM Library Server

1.3 Chapter 3. Installing COBOL/VSE

This chapter provides the following information:

- Overview of the installation procedure
- Procedure for installing COBOL/VSE

Note: Installing COBOL/VSE requires the use of the Maintain System History Program (MSHP).

Subtopics:

IBM Library Server

1.4 Chapter 4. Customizing COBOL/VSE

You can make modifications to COBOL/VSE only after installation of the product is complete. "Making Changes after Installation--Why Customize?" in topic 1.2.1 provides information on what you can modify, and why you might want to customize COBOL/VSE. This chapter tells how to *make* the modifications or where to find the necessary coding information to tailor COBOL/VSE to the needs of your site.

The issues discussed in this chapter are:

- General rules for changing default values
- Modifying compiler options and phases
- Placing COBOL/VSE in the shared virtual area
- Creating additional reserved word tables

To make the modifications to the option macros, you will need to modify and run the sample JCL supplied. Sample JCL for customization is available in the sublibrary for the compiler.

Subtopics:

IBM Library Server

1.5 Chapter 5. Maintaining COBOL/VSE

This chapter describes how to replace or re-install COBOL/VSE, and how to apply service updates to COBOL/VSE. To use the maintenance procedures effectively, you must have already installed COBOL/VSE and any required products.

In addition, this chapter describes how to remove COBOL/VSE.

Subtopics:

IBM Library Server

| FRONT_3.2.1 Second Edition, June 1998

| • Information needed to plan for and install IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA has been included in this manual.

| • Information about the following new COBOL/VSE installation options has been added:

| - DATEPROC
| - INTDATE
| - YRWINDOW

IBM Library Server

1.0 Planning for, Installing, Customizing, and Maintaining COBOL/VSE

Subtopics:

IBM Library Server

1.1.1 What You Receive with COBOL/VSE

COBOL/VSE is distributed in two forms:

- full function offering, including Debug Tool/VSE
- alternate function offering

The COBOL/VSE licensed program package is distributed on tape containing the necessary material for installation.

Product Identification

Table 1. IBM COBOL for VSE/ESA Components and Component Level Codes (CLC)

Component ID	CLC	Component Description
5686-068-00	18M	COBOL/VSE Base
5686-068-01	18N	COBOL/VSE US English Language Feature
5686-068-02	18O	COBOL/VSE Japanese Language Feature

Basic Machine-Readable Material: You receive COBOL/VSE basic machine-readable material on one of these distribution media:

Table 2. Basic Material: Program Tape(s)

Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
6250 tape	5801 5811 5821	1	COBOL/VSE Release 1	unlabeled
3480 cart.	5802 5812 5822	1	COBOL/VSE Release 1	unlabeled
1/4" tape	5804 5814 5824	1	COBOL/VSE Release 1	unlabeled

The distribution media contains all the programs and data you need to install COBOL/VSE.

Table 3. Program Tapes: File Content

File	Description
1	Header file containing the COBOL/VSE copyright statement
2	Backup file ID COB.BASE...1.1.0 followed by an MSHP System History File
3	COBOL/VSE product with US uppercase English
4	Header file containing the COBOL/VSE copyright statement
5	Backup file ID COB.ENU....1.1.0 followed by an MSHP System History File
6	COBOL/VSE US English language feature
7	Header file containing the COBOL/VSE copyright statement
8	Backup file ID COB.JPN....1.1.0 followed by an MSHP System History File
9	COBOL/VSE Japanese language feature
10	Null (Tape Mark)
11	End of backup (EOB) record
12	Null (Tape Mark)

IBM Library Server

1.1.2 Optional Material

There is no optional machine-readable material for COBOL/VSE.

IBM Library Server

1.1.3 Basic Unlicensed Publications

The table below lists basic unlicensed publications for COBOL/VSE. When you order COBOL/VSE, you receive one of each of these publications.

Table 4. COBOL/VSE Unlicensed Publications

Publication Title	**Order Number**
COBOL/VSE Licensed Program Specifications	GC26-8069
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

1.1.4 Optional Unlicensed Publications

The table below lists optional unlicensed publications for COBOL/VSE. You can order one free copy of an optional unlicensed publication by specifying its feature number.

Table 5. IBM COBOL for VSE/ESA Publications

Publication	Order Number	Feature number
COBOL/VSE Diagnosis Guide	SC26-8528	7170
COBOL/VSE Language Reference	SC26-8073	7167
COBOL/VSE Reference Summary	SX26-3834	7169
COBOL/VSE Migration Guide	GC26-8070	7168
COBOL/VSE Programming Guide	SC26-8072	7166
COBOL/VSE General Information	GC26-8068	7165

IBM Library Server

1.1.5 Ensuring That You Have the Required Machine Configuration

COBOL/VSE can be installed and run in a virtual storage environment on system configurations that support the operating system releases listed in "Choosing Required and Optional Licensed Programs" in topic 1.1.7.

The following table lists the machine requirements for installing and using COBOL/VSE.

Table 6. COBOL/VSE Hardware Requirements

Item	Machine Requirements
Installation	1. A system processor that can support the required operating system. 2. Enough storage on DASD to hold the COBOL/VSE program product. See "Ensuring That You Have Enough Storage for COBOL/VSE" in topic 1.1.6 for minimum requirements.
Compilation	1. Any processor that supports the required operating system. 2. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is required only when the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD). 3. The SYSADAT file is required when the ADATA option is specified. 4. I/O devices for use by the compiler.
Run-Time Machine Requirements	1. Any processor supported by the required operating system. 2. I/O devices used by the object program during the run.
Devices Supported	1. All IBM devices supported by Sequential Access Method (SAM) and Virtual Storage Access Method (VSAM) under VSE/ESA can be used by object programs produced by the COBOL/VSE compiler when used with the LE/VSE Library. 2. All IBM devices supported under CICS are supported by COBOL/VSE when running under CICS.

IBM Library Server

1.1.6 Ensuring That You Have Enough Storage for COBOL/VSE

COBOL/VSE requirements for virtual and auxiliary storage vary, depending on whether you are installing the product, compiling a program, or running a program. The sections that follow describe how COBOL/VSE uses storage.

For information on run-time storage needs, see *LE/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

1.1.7 Choosing Required and Optional Licensed Programs

COBOL/VSE runs under VSE/ESA with the required licensed programs listed in Table 8 and optional licensed programs listed in Table 9. **All licensed programs should be installed with the minimum release listed or with any subsequent release.**

Table 8. Required Licensed Programs for COBOL/VSE

Required Licensed Program	Minimum Release	Program Number
One of:		
VSE/ESA	Version 1 Release 4	5750-ACD
VSE/ESA	Version 2 Release 1	5690-VSE
LE/VSE(1)	Release 4	5686-094
High Level Assembler/MVS & VM & VSE	Release 1	5696-234

Note:

1. If you want to run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option), the PTFs that resolve the APARs shown Table 25 in topic 2.1.2.3 must be applied to the LE/VSE components you have installed.

Table 9. Optional Licensed Programs Supported by COBOL/VSE

Optional Licensed Program	Minimum Release	Program Number
BookManager Read	Release 2 is required to view softcopy documentation	73F6-023
CICS/VSE	Version 2 Release 3	5686-026
DFSORT/VSE(1)	Version 3 Release 1	5746-SM3
DL/I DOS/VS	Release 10	5746-XX1
DOS/VS Sort/Merge (VSE 1.4 only)	Version 2 Release 5	5746-SM2
SQL/DS	Version 3 Release 4	5688-103

Note:

1. If you want to sort dates with two-digit years using the century windowing capability provided by DFSORT/VSE, you require DFSORT Version 3 Release 2 with PTF UN99635, or a later release of DFSORT/VSE.

Subtopics:

IBM Library Server

1.1.8 Considerations Before Installing

Before you begin installing COBOL/VSE, review the following considerations.

Subtopics:

IBM Library Server

1.0 Planning for, Installing, Customizing, and Maintaining COBOL/VSE

Subtopics:

IBM Library Server

1.1 Chapter 1. Planning to Install COBOL/VSE

This chapter provides information for planning the installation of COBOL/VSE. For information about planning the installation of VisualAge COBOL MLE for VSE, see "Planning for, Installing, and Maintaining VisualAge COBOL MLE for VSE" in topic 2.0. This chapter includes information on:

- What you receive with COBOL/VSE
- Ensuring that you have the required machine configuration
- Ensuring that you have enough storage for COBOL/VSE
- Choosing required and optional programs
- Considerations before installing

Subtopics:

IBM Library Server

1.1 Chapter 1. Planning to Install COBOL/VSE

This chapter provides information for planning the installation of COBOL/VSE. For information about planning the installation of VisualAge COBOL MLE for VSE, see "Planning for, Installing, and Maintaining VisualAge COBOL MLE for VSE" in topic 2.0. This chapter includes information on:

- What you receive with COBOL/VSE
- Ensuring that you have the required machine configuration
- Ensuring that you have enough storage for COBOL/VSE
- Choosing required and optional programs
- Considerations before installing

Subtopics:

IBM Library Server

1.1.1 What You Receive with COBOL/VSE

COBOL/VSE is distributed in two forms:

- full function offering, including Debug Tool/VSE
- alternate function offering

The COBOL/VSE licensed program package is distributed on tape containing the necessary material for installation.

Product Identification

Table 1. IBM COBOL for VSE/ESA Components and Component Level Codes (CLC)

Component ID	CLC	Component Description
5686-068-00	18M	COBOL/VSE Base
5686-068-01	18N	COBOL/VSE US English Language Feature
5686-068-02	18O	COBOL/VSE Japanese Language Feature

Basic Machine-Readable Material: You receive COBOL/VSE basic machine-readable material on one of these distribution media:

Table 2. Basic Material: Program Tape(s)

Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
6250 tape	5801 5811 5821	1	COBOL/VSE Release 1	unlabeled
3480 cart.	5802 5812 5822	1	COBOL/VSE Release 1	unlabeled
1/4" tape	5804 5814 5824	1	COBOL/VSE Release 1	unlabeled

The distribution media contains all the programs and data you need to install COBOL/VSE.

Table 3. Program Tapes: File Content

File	Description
1	Header file containing the COBOL/VSE copyright statement
2	Backup file ID COB.BASE...1.1.0 followed by an MSHP System History File
3	COBOL/VSE product with US uppercase English
4	Header file containing the COBOL/VSE copyright statement
5	Backup file ID COB.ENU....1.1.0 followed by an MSHP System History File
6	COBOL/VSE US English language feature
7	Header file containing the COBOL/VSE copyright statement
8	Backup file ID COB.JPN....1.1.0 followed by an MSHP System History File
9	COBOL/VSE Japanese language feature
10	Null (Tape Mark)
11	End of backup (EOB) record
12	Null (Tape Mark)

IBM Library Server

1.1.1 What You Receive with COBOL/VSE

COBOL/VSE is distributed in two forms:

- full function offering, including Debug Tool/VSE
- alternate function offering

The COBOL/VSE licensed program package is distributed on tape containing the necessary material for installation.

Product Identification

Table 1. IBM COBOL for VSE/ESA Components and Component Level Codes (CLC)

Component ID	CLC	Component Description
5686-068-00	18M	COBOL/VSE Base
5686-068-01	18N	COBOL/VSE US English Language Feature
5686-068-02	18O	COBOL/VSE Japanese Language Feature

Basic Machine-Readable Material: You receive COBOL/VSE basic machine-readable material on one of these distribution media:

Table 2. Basic Material: Program Tape(s)

Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
6250 tape	5801 5811 5821	1	COBOL/VSE Release 1	unlabeled
3480 cart.	5802 5812 5822	1	COBOL/VSE Release 1	unlabeled
1/4" tape	5804 5814 5824	1	COBOL/VSE Release 1	unlabeled

The distribution media contains all the programs and data you need to install COBOL/VSE.

Table 3. Program Tapes: File Content

File	Description
1	Header file containing the COBOL/VSE copyright statement
2	Backup file ID COB.BASE...1.1.0 followed by an MSHP System History File
3	COBOL/VSE product with US uppercase English
4	Header file containing the COBOL/VSE copyright statement
5	Backup file ID COB.ENU....1.1.0 followed by an MSHP System History File
6	COBOL/VSE US English language feature
7	Header file containing the COBOL/VSE copyright statement
8	Backup file ID COB.JPN....1.1.0 followed by an MSHP System History File
9	COBOL/VSE Japanese language feature
10	Null (Tape Mark)
11	End of backup (EOB) record
12	Null (Tape Mark)

1.1.2 Optional Material

There is no optional machine-readable material for COBOL/VSE.

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.1.2 Optional Material

There is no optional machine-readable material for COBOL/VSE.

IBM Library Server

1.1.3 Basic Unlicensed Publications

The table below lists basic unlicensed publications for COBOL/VSE. When you order COBOL/VSE, you receive one of each of these publications.

Table 4. COBOL/VSE Unlicensed Publications

Publication Title	**Order Number**
COBOL/VSE Licensed Program Specifications	GC26-8069
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

1.1.3 Basic Unlicensed Publications

The table below lists basic unlicensed publications for COBOL/VSE. When you order COBOL/VSE, you receive one of each of these publications.

Table 4. COBOL/VSE Unlicensed Publications

Publication Title	Order Number
COBOL/VSE Licensed Program Specifications	GC26-8069
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

1.1.4 Optional Unlicensed Publications

The table below lists optional unlicensed publications for COBOL/VSE. You can order one free copy of an optional unlicensed publication by specifying its feature number.

Table 5. IBM COBOL for VSE/ESA Publications

Publication	Order Number	Feature number
COBOL/VSE Diagnosis Guide	SC26-8528	7170
COBOL/VSE Language Reference	SC26-8073	7167
COBOL/VSE Reference Summary	SX26-3834	7169
COBOL/VSE Migration Guide	GC26-8070	7168
COBOL/VSE Programming Guide	SC26-8072	7166
COBOL/VSE General Information	GC26-8068	7165

IBM Library Server

1.1.4 Optional Unlicensed Publications

The table below lists optional unlicensed publications for COBOL/VSE. You can order one free copy of an optional unlicensed publication by specifying its feature number.

Table 5. IBM COBOL for VSE/ESA Publications

Publication	Order Number	Feature number
COBOL/VSE Diagnosis Guide	SC26-8528	7170
COBOL/VSE Language Reference	SC26-8073	7167
COBOL/VSE Reference Summary	SX26-3834	7169
COBOL/VSE Migration Guide	GC26-8070	7168
COBOL/VSE Programming Guide	SC26-8072	7166
COBOL/VSE General Information	GC26-8068	7165

1.1.5 Ensuring That You Have the Required Machine Configuration

COBOL/VSE can be installed and run in a virtual storage environment on system configurations that support the operating system releases listed in "Choosing Required and Optional Licensed Programs" in topic 1.1.7.

The following table lists the machine requirements for installing and using COBOL/VSE.

Table 6. COBOL/VSE Hardware Requirements

Item	Machine Requirements
Installation	1. A system processor that can support the required operating system. 2. Enough storage on DASD to hold the COBOL/VSE program product. See "Ensuring That You Have Enough Storage for COBOL/VSE" in topic 1.1.6 for minimum requirements.
Compilation	1. Any processor that supports the required operating system. 2. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is required only when the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD). 3. The SYSADAT file is required when the ADATA option is specified. 4. I/O devices for use by the compiler.
Run-Time Machine Requirements	1. Any processor supported by the required operating system. 2. I/O devices used by the object program during the run.
Devices Supported	1. All IBM devices supported by Sequential Access Method (SAM) and Virtual Storage Access Method (VSAM) under VSE/ESA can be used by object programs produced by the COBOL/VSE compiler when used with the LE/VSE Library. 2. All IBM devices supported under CICS are supported by COBOL/VSE when running under CICS.

IBM Library Server

1.1.6.1 DASD Storage Required for Installation

When installing COBOL/VSE, you must provide DASD storage for:

- VSE Librarian library
- System history file

Table 7 lists the minimum DASD storage that COBOL/VSE requires for a VSE Librarian library. This does not include storage for other licensed programs installed in the library.

The MSHP System History File for the COBOL/VSE component requires about one cylinder of 3380 or equivalent DASD space.

Allow 10% to 15% extra storage for future enhancements and service updates.

Table 7. Minimum library storage required

LIBR BLKS(1)	3375 3350 CYL	3390 CYL	3390 CYL	FBA CYL	FBA 9345 CYL	LIBR BLKS
6700	21	23	15	14	16	13330

Note:

1. One library block equals 1 kilobyte (1024 bytes)

1.1.6.2 Storage Required for Compilation

The COBOL/VSE compiler requires a minimum of 760K bytes of virtual storage.

The virtual storage required depends on the size and content of your COBOL program. The COBOL/VSE compiler is designed to take advantage of systems with large virtual storage.

The following auxiliary storage is required by the COBOL/VSE compiler:

1. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07

2. IJSYS05 work file for the LIB compiler option

3. The SYSADAT file for the ADATA compiler option

4. SYSLST, SYSLNK and SYSPCH, if the appropriate options are in effect and if these files are going to be routed to a DASD, tape, or other auxiliary storage

5. SYSIPT input containing the COBOL program

1.1.5 Ensuring That You Have the Required Machine Configuration

COBOL/VSE can be installed and run in a virtual storage environment on system configurations that support the operating system releases listed in "Choosing Required and Optional Licensed Programs" in topic 1.1.7.

The following table lists the machine requirements for installing and using COBOL/VSE.

Table 6. COBOL/VSE Hardware Requirements

Item	Machine Requirements
Installation	1. A system processor that can support the required operating system. 2. Enough storage on DASD to hold the COBOL/VSE program product. See "Ensuring That You Have Enough Storage for COBOL/VSE" in topic 1.1.6 for minimum requirements.
Compilation	1. Any processor that supports the required operating system. 2. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is required only when the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD). 3. The SYSADAT file is required when the ADATA option is specified. 4. I/O devices for use by the compiler.
Run-Time Machine Requirements	1. Any processor supported by the required operating system. 2. I/O devices used by the object program during the run.
Devices Supported	1. All IBM devices supported by Sequential Access Method (SAM) and Virtual Storage Access Method (VSAM) under VSE/ESA can be used by object programs produced by the COBOL/VSE compiler when used with the LE/VSE Library. 2. All IBM devices supported under CICS are supported by COBOL/VSE when running under CICS.

1.1.6 Ensuring That You Have Enough Storage for COBOL/VSE

COBOL/VSE requirements for virtual and auxiliary storage vary, depending on whether you are installing the product, compiling a program, or running a program. The sections that follow describe how COBOL/VSE uses storage.

For information on run-time storage needs, see *LE/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

1.1.6 Ensuring That You Have Enough Storage for COBOL/VSE

COBOL/VSE requirements for virtual and auxiliary storage vary, depending on whether you are installing the product, compiling a program, or running a program. The sections that follow describe how COBOL/VSE uses storage.

For information on run-time storage needs, see *LE/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

1.1.6.1 DASD Storage Required for Installation

When installing COBOL/VSE, you must provide DASD storage for:

- VSE Librarian library
- System history file

Table 7 lists the minimum DASD storage that COBOL/VSE requires for a VSE Librarian library. This does not include storage for other licensed programs installed in the library.

The MSHP System History File for the COBOL/VSE component requires about one cylinder of 3380 or equivalent DASD space.

Allow 10% to 15% extra storage for future enhancements and service updates.

Table 7. Minimum library storage required

LIBR BLKS(1)	3375 3350 CYL	3390 CYL	3390 CYL	FBA CYL	FBA 9345 CYL	LIBR BLKS
6700	21	23	15	14	16	13330

Note:

1. One library block equals 1 kilobyte (1024 bytes)

IBM Library Server

1.1.6.1 DASD Storage Required for Installation

When installing COBOL/VSE, you must provide DASD storage for:

- VSE Librarian library
- System history file

Table 7 lists the minimum DASD storage that COBOL/VSE requires for a VSE Librarian library. This does not include storage for other licensed programs installed in the library.

The MSHP System History File for the COBOL/VSE component requires about one cylinder of 3380 or equivalent DASD space.

Allow 10% to 15% extra storage for future enhancements and service updates.

Table 7. Minimum library storage required

LIBR		3375		3390		FBA	
BLKS(1)	3350 CYL	CYL	3380 CYL	CYL	9345 CYL	BLKS	
6700	21	23	15	14	16	13330	

Note:

1. One library block equals 1 kilobyte (1024 bytes)

IBM Library Server

1.1.6.2 Storage Required for Compilation

The COBOL/VSE compiler requires a minimum of 760K bytes of virtual storage.

The virtual storage required depends on the size and content of your COBOL program. The COBOL/VSE compiler is designed to take advantage of systems with large virtual storage.

The following auxiliary storage is required by the COBOL/VSE compiler:

1. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07

2. IJSYS05 work file for the LIB compiler option

3. The SYSADAT file for the ADATA compiler option

4. SYSLST, SYSLNK and SYSPCH, if the appropriate options are in effect and if these files are going to be routed to a DASD, tape, or other auxiliary storage

5. SYSIPT input containing the COBOL program

IBM Library Server

1.1.7.1 National Language Support

Two language features are available with COBOL/VSE: the Japanese Language Feature and the US English Language Feature (mixed-case English). In order to receive compiler error messages and listing headers in Japanese, mixed-case English, or both, you must order the appropriate language feature(s).

Uppercase US English is supplied on the base tape, and does not require a language feature.

1.1.6.2 Storage Required for Compilation

The COBOL/VSE compiler requires a minimum of 760K bytes of virtual storage.

The virtual storage required depends on the size and content of your COBOL program. The COBOL/VSE compiler is designed to take advantage of systems with large virtual storage.

The following auxiliary storage is required by the COBOL/VSE compiler:

1. Six compiler work files: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07

2. IJSYS05 work file for the LIB compiler option

3. The SYSADAT file for the ADATA compiler option

4. SYSLST, SYSLNK and SYSPCH, if the appropriate options are in effect and if these files are going to be routed to a DASD, tape, or other auxiliary storage

5. SYSIPT input containing the COBOL program

1.1.7 Choosing Required and Optional Licensed Programs

COBOL/VSE runs under VSE/ESA with the required licensed programs listed in Table 8 and optional licensed programs listed in Table 9. **All licensed programs should be installed with the minimum release listed or with any subsequent release.**

Table 8. Required Licensed Programs for COBOL/VSE

Required Licensed Program	Minimum Release	Program Number
One of:		
VSE/ESA	Version 1 Release 4	5750-ACD
VSE/ESA	Version 2 Release 1	5690-VSE
LE/VSE(1)	Release 4	5686-094
High Level Assembler/MVS & VM & VSE	Release 1	5696-234

Note:

1. If you want to run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option), the PTFs that resolve the APARs shown Table 25 in topic 2.1.2.3 must be applied to the LE/VSE components you have installed.

Table 9. Optional Licensed Programs Supported by COBOL/VSE

Optional Licensed Program	Minimum Release	Program Number
BookManager Read	Release 2 is required to view softcopy documentation	73F6-023
CICS/VSE	Version 2 Release 3	5686-026
DFSORT/VSE(1)	Version 3 Release 1	5746-SM3
DL/I DOS/VS	Release 10	5746-XX1
DOS/VS Sort/Merge (VSE 1.4 only)	Version 2 Release 5	5746-SM2
SQL/DS	Version 3 Release 4	5688-103

Note:

1. If you want to sort dates with two-digit years using the century windowing capability provided by DFSORT/VSE, you require DFSORT Version 3 Release 2 with PTF UN99635, or a later release of DFSORT/VSE.

Subtopics:

IBM Library Server

1.1.7 Choosing Required and Optional Licensed Programs

COBOL/VSE runs under VSE/ESA with the required licensed programs listed in Table 8 and optional licensed programs listed in Table 9. **All licensed programs should be installed with the minimum release listed or with any subsequent release.**

Table 8. Required Licensed Programs for COBOL/VSE

Required Licensed Program	Minimum Release	Program Number
One of:		
VSE/ESA	Version 1 Release 4	5750-ACD
VSE/ESA	Version 2 Release 1	5690-VSE
LE/VSE(1)	Release 4	5686-094
High Level Assembler/MVS & VM & VSE	Release 1	5696-234

Note:

1. If you want to run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option), the PTFs that resolve the APARs shown Table 25 in topic 2.1.2.3 must be applied to the LE/VSE components you have installed.

Table 9. Optional Licensed Programs Supported by COBOL/VSE

Optional Licensed Program	Minimum Release	Program Number
BookManager Read	Release 2 is required to view softcopy documentation	73F6-023
CICS/VSE	Version 2 Release 3	5686-026
DFSORT/VSE(1)	Version 3 Release 1	5746-SM3
DL/I DOS/VS	Release 10	5746-XX1
DOS/VS Sort/Merge (VSE 1.4 only)	Version 2 Release 5	5746-SM2
SQL/DS	Version 3 Release 4	5688-103

Note:

1. If you want to sort dates with two-digit years using the century windowing capability provided by DFSORT/VSE, you require DFSORT Version 3 Release 2 with PTF UN99635, or a later release of DFSORT/VSE.

Subtopics:

IBM Library Server

1.1.7.1 National Language Support

Two language features are available with COBOL/VSE: the Japanese Language Feature and the US English Language Feature (mixed-case English). In order to receive compiler error messages and listing headers in Japanese, mixed-case English, or both, you must order the appropriate language feature(s).

Uppercase US English is supplied on the base tape, and does not require a language feature.

IBM Library Server

1.1.8.1 Choosing the Language Feature You Want

In addition to uppercase English compiler messages, COBOL/VSE provides two optional language features, the Japanese Language Feature and the US English Language Feature (mixed-case English). Before you begin to install COBOL/VSE, you should determine if you want to install either or both of these language features, or if you will use the uppercase English compiler messages supplied in the base product.

To install the US English Language Feature use COB.ENU....1.1.0, and to install the Japanese Language Feature use COB.JPN....1.1.0.

1.1.8.2 Understanding the Installation Tools

- **MSHP Definition for COBOL/VSE:** When this product is prepared for initial shipping it will be from a VSE/ESA Version 1 Release 4 or later release environment. The product is shipped with an MSHP history file for each component of COBOL/VSE.

- **Maintain System History Program (MSHP):** MSHP helps you to control and record the installation of, and changes to, system software in a VSE operating environment. To install and maintain COBOL/VSE you must use MSHP, which records all system maintenance details in the system history file. MSHP also applies preventive and corrective service, through the use of functional control statements. For detailed information about MSHP, see *VSE/ESA System Control Statements*.

- **VSE Librarian program (LIBR):** Ensure that you are familiar with the VSE Librarian program (LIBR) which will be used to define the libraries in which COBOL/VSE will be installed.

1.1.8.3 If VS COBOL II is already Installed

Some modules supplied with COBOL/VSE have the same names as modules in the VS COBOL II sublibrary. If VS COBOL II is already installed on your system, COBOL/VSE cannot be installed in the same sublibrary.

IBM Library Server

1.1.8.4 Checking Service Updates

Before installing COBOL/VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 10. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLVSE110	06818M
COBOLVSE110	06818N
COBOLVSE110	06818O

IBM Library Server

1.1.7.1 National Language Support

Two language features are available with COBOL/VSE: the Japanese Language Feature and the US English Language Feature (mixed-case English). In order to receive compiler error messages and listing headers in Japanese, mixed-case English, or both, you must order the appropriate language feature(s).

Uppercase US English is supplied on the base tape, and does not require a language feature.

IBM Library Server

1.1.8 Considerations Before Installing

Before you begin installing COBOL/VSE, review the following considerations.

Subtopics:

IBM Library Server

1.1.8 Considerations Before Installing

Before you begin installing COBOL/VSE, review the following considerations.

Subtopics:

IBM Library Server

1.1.8.1 Choosing the Language Feature You Want

In addition to uppercase English compiler messages, COBOL/VSE provides two optional language features, the Japanese Language Feature and the US English Language Feature (mixed-case English). Before you begin to install COBOL/VSE, you should determine if you want to install either or both of these language features, or if you will use the uppercase English compiler messages supplied in the base product.

To install the US English Language Feature use COB.ENU....1.1.0, and to install the Japanese Language Feature use COB.JPN....1.1.0.

1.1.8.1 Choosing the Language Feature You Want

In addition to uppercase English compiler messages, COBOL/VSE provides two optional language features, the Japanese Language Feature and the US English Language Feature (mixed-case English). Before you begin to install COBOL/VSE, you should determine if you want to install either or both of these language features, or if you will use the uppercase English compiler messages supplied in the base product.

To install the US English Language Feature use COB.ENU....1.1.0, and to install the Japanese Language Feature use COB.JPN....1.1.0.

IBM Library Server

1.1.8.2 Understanding the Installation Tools

- **MSHP Definition for COBOL/VSE:** When this product is prepared for initial shipping it will be from a VSE/ESA Version 1 Release 4 or later release environment. The product is shipped with an MSHP history file for each component of COBOL/VSE.

- **Maintain System History Program (MSHP):** MSHP helps you to control and record the installation of, and changes to, system software in a VSE operating environment. To install and maintain COBOL/VSE you must use MSHP, which records all system maintenance details in the system history file. MSHP also applies preventive and corrective service, through the use of functional control statements. For detailed information about MSHP, see *VSE/ESA System Control Statements*.

- **VSE Librarian program (LIBR):** Ensure that you are familiar with the VSE Librarian program (LIBR) which will be used to define the libraries in which COBOL/VSE will be installed.

1.1.8.2 Understanding the Installation Tools

- **MSHP Definition for COBOL/VSE:** When this product is prepared for initial shipping it will be from a VSE/ESA Version 1 Release 4 or later release environment. The product is shipped with an MSHP history file for each component of COBOL/VSE.

- **Maintain System History Program (MSHP):** MSHP helps you to control and record the installation of, and changes to, system software in a VSE operating environment. To install and maintain COBOL/VSE you must use MSHP, which records all system maintenance details in the system history file. MSHP also applies preventive and corrective service, through the use of functional control statements. For detailed information about MSHP, see *VSE/ESA System Control Statements*.

- **VSE Librarian program (LIBR):** Ensure that you are familiar with the VSE Librarian program (LIBR) which will be used to define the libraries in which COBOL/VSE will be installed.

IBM Library Server

1.1.8.3 If VS COBOL II is already Installed

Some modules supplied with COBOL/VSE have the same names as modules in the VS COBOL II sublibrary. If VS COBOL II is already installed on your system, COBOL/VSE cannot be installed in the same sublibrary.

1.1.8.3 If VS COBOL II is already Installed

Some modules supplied with COBOL/VSE have the same names as modules in the VS COBOL II sublibrary. If VS COBOL II is already installed on your system, COBOL/VSE cannot be installed in the same sublibrary.

IBM Library Server

1.1.8.4 Checking Service Updates

Before installing COBOL/VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 10. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLVSE110	06818M
COBOLVSE110	06818N
COBOLVSE110	06818O

IBM Library Server

1.2.1 Making Changes after Installation--Why Customize?

When you install COBOL/VSE you receive IBM-supplied defaults for compiler options and phases, and the reserved word table. You might want to customize COBOL/VSE to meet the application programming needs at your site.

After COBOL/VSE is installed, you can make the following changes:

- Modify the compiler option default values.
- Modify the compiler phases residency values.
- Make the compiler options fixed.
- Create additional reserved word tables.

The sections that follow discuss the planning you must do to customize COBOL/VSE for your site.

IBM Library Server

1.2.2 Planning to Modify Compiler Option Default Values

Compiler option defaults, and the required load locations for compiler phases, are set in the IGYCOPT macro as shown in Table 11 in topic 1.2.2.2.1 and Table 13 in topic 1.2.3.21.1. The default options module, IGYCDOPT, is link-edited with AMODE (31) and RMODE (ANY) during installation.

The IGYCOPT macro has a dual purpose: it allows you to select and fix the compiler options defaults, and to specify which compiler phases are in the SVA. This means that you can accept the IBM-supplied compiler option values you receive when you install COBOL/VSE, or you can modify them to suit your site. You can also choose whether or not your application programmers will be able to replace these options.

Subtopics:

1.2.3 Planning to Place Compiler Phases in the SVA

You might want to make some phases resident in the SVA. in order to minimize the search (path length) for them when the COBOL/VSE compiler is run, or to allow the compiler phases to be shared.

Subtopics:

IBM Library Server

1.2.4 Planning to Create an Additional Reserved Word Table

You are provided with a default reserved word table when you install COBOL/VSE. A CICS-specific reserved word table is provided as an alternate reserved word table (See "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2). You can create additional reserved word tables after installation. (During compilation, the value of the WORD compiler option determines which reserved word table is used).

Subtopics:

IBM Library Server

1.2.5 COBOL/VSE Compiler Options

This section describes those compiler options whose default values you can change. The notes that accompany some of the descriptions provide additional information about these options, such as how they interact with other options during compilation. This information may assist you in making decisions regarding the default values appropriate for your installation. For more information on using the compiler options, see the *COBOL/VSE Programming Guide*.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

IBM Library Server

1.1.8.4 Checking Service Updates

Before installing COBOL/VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 10. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLVSE110	06818M
COBOLVSE110	06818N
COBOLVSE110	06818O

1.2 Chapter 2. Planning to Customize COBOL/VSE

This chapter provides the following information for planning the customization of COBOL/VSE:

- Making changes after installation--why customize?
- Planning to modify compiler option default values
- Planning to place compiler phases in the Shared Virtual Area (SVA)
- Planning to create an additional reserved word table

These topics provide the information you need to plan the customization of COBOL/VSE. For information on how to make the changes to COBOL/VSE that are covered in this chapter, see Chapter 4, "Customizing COBOL/VSE" in topic 1.4.

This chapter also contains worksheets that help you to plan modifications to the IBM-supplied default values within macros. An explanation of the planning worksheets in this book is in "Using the Macro Planning Worksheets" in topic FRONT_2.2.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

IBM Library Server

1.2 Chapter 2. Planning to Customize COBOL/VSE

This chapter provides the following information for planning the customization of COBOL/VSE:

- Making changes after installation--why customize?
- Planning to modify compiler option default values
- Planning to place compiler phases in the Shared Virtual Area (SVA)
- Planning to create an additional reserved word table

These topics provide the information you need to plan the customization of COBOL/VSE. For information on how to make the changes to COBOL/VSE that are covered in this chapter, see Chapter 4, "Customizing COBOL/VSE" in topic 1.4.

This chapter also contains worksheets that help you to plan modifications to the IBM-supplied default values within macros. An explanation of the planning worksheets in this book is in "Using the Macro Planning Worksheets" in topic FRONT_2.2.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

IBM Library Server

1.2.1 Making Changes after Installation--Why Customize?

When you install COBOL/VSE you receive IBM-supplied defaults for compiler options and phases, and the reserved word table. You might want to customize COBOL/VSE to meet the application programming needs at your site.

After COBOL/VSE is installed, you can make the following changes:

- Modify the compiler option default values.
- Modify the compiler phases residency values.
- Make the compiler options fixed.
- Create additional reserved word tables.

The sections that follow discuss the planning you must do to customize COBOL/VSE for your site.

IBM Library Server

1.2.2.1 Why Make Compiler Options Fixed?

COBOL/VSE can aid in setting up your site's programming standards. For example, many sites select APOST or QUOTE as their preferred compiler option for literal delimiters, but have no easy way to enforce its use. In COBOL/VSE, the IGYCOPT macro can be used to specify that an option may not be changed or replaced at compile time; that is, the option is fixed. At compile time, an attempt to replace a fixed option will not be allowed and will result in a diagnostic message with a nonzero compiler return code.

When certain options are fixed for consistent use, there might be times when special conditions require the ability to bypass such a fixed option. This can be done by assembling a temporary copy of the IGYCOPT macro with different parameters. At compile time, by selectively using a LIBDEF JCL statement specifying a sublibrary containing the required IGYCDOPT phase, you can bypass the fixed option. For example, if you select the OPT (OPTIMIZE) option to be fixed (indicating you always want the COBOL compiler to generate optimized object code), and then need to exempt an application from this requirement, you must reassemble the IGYCOPT macro after removing the asterisk parameter from the option. Then place the resulting IGYCDOPT phase in a temporary sublibrary to be accessed using the LIBDEF JCL statement at compile time.

Subtopics:

- 1.2.2.1.1 Sample Installation Job

IBM Library Server

1.2.2.2 Syntax Format for Modifying Compiler Options and Phases

If you plan to modify compiler option values and compiler phases, use the IGYCOPT syntax format shown in Figure 1. The IBM-supplied default values are shown both on the planning worksheets and immediately following the syntax diagrams.

Compiler options and phases, and their defaults, are discussed in the following sections. You should review these options and phases, and their default values to determine the values most suitable for your applications.

IGYCOPT format

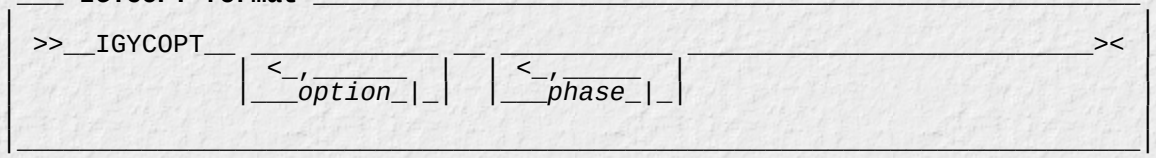


Figure 1. Syntax Format for IGYCOPT Compiler Options and Phases Macro

Subtopics:

- 1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

IBM Library Server

1.2.1 Making Changes after Installation--Why Customize?

When you install COBOL/VSE you receive IBM-supplied defaults for compiler options and phases, and the reserved word table. You might want to customize COBOL/VSE to meet the application programming needs at your site.

After COBOL/VSE is installed, you can make the following changes:

- Modify the compiler option default values.
- Modify the compiler phases residency values.
- Make the compiler options fixed.
- Create additional reserved word tables.

The sections that follow discuss the planning you must do to customize COBOL/VSE for your site.

IBM Library Server

1.2.2 Planning to Modify Compiler Option Default Values

Compiler option defaults, and the required load locations for compiler phases, are set in the IGYCOPT macro as shown in Table 11 in topic 1.2.2.2.1 and Table 13 in topic 1.2.3.21.1. The default options module, IGYCDOPT, is link-edited with AMODE (31) and RMODE (ANY) during installation.

The IGYCOPT macro has a dual purpose: it allows you to select and fix the compiler options defaults, and to specify which compiler phases are in the SVA. This means that you can accept the IBM-supplied compiler option values you receive when you install COBOL/VSE, or you can modify them to suit your site. You can also choose whether or not your application programmers will be able to replace these options.

Subtopics:

IBM Library Server

1.2.2.1.1 Sample Installation Job

COBOL/VSE provides a sample installation job, which can be modified and then used to change compiler option defaults. The job provides an example of how to change the IBM-supplied compiler option defaults.

IGYWEOP1.Z
This sample installation job can be used to change the IBM-supplied defaults.

This job is located in the COBOL sublibrary PRD2.PROD (if the default sublibrary is used).

IBM Library Server

1.2.2 Planning to Modify Compiler Option Default Values

Compiler option defaults, and the required load locations for compiler phases, are set in the IGYCOPT macro as shown in Table 11 in topic 1.2.2.2.1 and Table 13 in topic 1.2.3.21.1. The default options module, IGYCDOPT, is link-edited with AMODE (31) and RMODE (ANY) during installation.

The IGYCOPT macro has a dual purpose: it allows you to select and fix the compiler options defaults, and to specify which compiler phases are in the SVA. This means that you can accept the IBM-supplied compiler option values you receive when you install COBOL/VSE, or you can modify them to suit your site. You can also choose whether or not your application programmers will be able to replace these options.

Subtopics:

IBM Library Server

1.2.2.1.1 Sample Installation Job

COBOL/VSE provides a sample installation job, which can be modified and then used to change compiler option defaults. The job provides an example of how to change the IBM-supplied compiler option defaults.

IGYWEOP1.Z
This sample installation job can be used to change the IBM-supplied defaults.

This job is located in the COBOL sublibrary PRD2.PROD (if the default sublibrary is used).

1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

The following worksheet will help you plan and code the compiler options portion of the IGYCOPT macro. To complete the worksheet, fill in the 'Enter * for Fixed' and the 'Enter Selection' columns.

The IGYCOPT macro worksheet also includes a section for compiler phases. That section of the worksheet can be found in "IGYCOPT Macro Worksheet for Compiler Phases" in topic 1.2.3.21.1, following the discussion of compiler phases.

Notes:

1. Coding the asterisk [*], when modifying a compiler option default value, indicates that the option is to be fixed and cannot be replaced by an application programmer.

| 2. The ALOWCBL, DBCSXREF, LVLINFO, and NUMCLS options cannot be overridden at compile time. The 'Enter * for Fixed' worksheet entries for these options is therefore blank.

3. The IBM-supplied default value for ADEXIT, INEXIT, LVLINFO, LIBEXIT, and PRTEXIT is null. The 'IBM-Supplied Default' worksheet entries for these options are therefore blank.

4. The DUMP compiler option cannot be set through the IGYCOPT macro. Unless changed at compile time, DUMP is always set to NODUMP.

Table 11. IGYCOPT Worksheet for Options

Compiler Option	Enter * for Fixed	Enter Selection	IBM-Supplied Default	Syntax Description
ADATA=	____	_____	*NO*	Topic 1.2.5.5
ADEXIT=	____	_____	Not specified	Topic 1.2.5.6
ADV=	____	_____	*YES*	Topic 1.2.5.7
ALOWCBL=		_____	*YES*	Topic 1.2.5.8
AWO=	____	_____	*NO*	Topic 1.2.5.9
BUF=	____	_____	*4K*	Topic 1.2.5.10
CMPR2=	____	_____	*NO*	Topic 1.2.5.11
COMPILE=	____	_____	*NOC(S)*	Topic 1.2.5.12
CURRENCY=	____	_____	*NO*	Topic 1.2.5.13
DATA=	____	_____	*31*	Topic 1.2.5.14

DATEPROC=	____	_____	*NO*	Topic 1.2.5.15
DBCS=	____	_____	*NO*	Topic 1.2.5.16
DBCSXREF=		_____	*NO*	Topic 1.2.5.17
DYNAM=	____	_____	*NO*	Topic 1.2.5.18
FASTSRT=	____	_____	*NO*	Topic 1.2.5.19
FLAG=	____	_____	*(I)*	Topic 1.2.5.20
FLAGMIG=	____	_____	*NO*	Topic 1.2.5.21
FLAGSAA=	____	_____	*NO*	Topic 1.2.5.22
FLAGSTD=	____	_____	*NO*	Topic 1.2.5.23
INTDATE=	____	_____	*ANSI*	Topic 1.2.5.25
INEXIT=	____	_____	Not specified	Topic 1.2.5.24
LANGUAGE=	____	_____	*UE*	Topic 1.2.5.26
LIB=	____	_____	*NO*	Topic 1.2.5.27
LIBEXIT=	____	_____	Not specified	Topic 1.2.5.28
LINECNT=	____	_____	*60*	Topic 1.2.5.29
LIST=	____	_____	*NO*	Topic 1.2.5.30
LITCHAR=	____	_____	*QUOTE*	Topic 1.2.5.31
LVLINFO=		_____	Not specified	Topic 1.2.5.32
MAP=	____	_____	*NO*	Topic 1.2.5.33
NAME=	____	_____	*NO*	Topic 1.2.5.34
NUM=	____	_____	*NO*	Topic 1.2.5.35
NUMCLS=		_____	*PRIM*	Topic 1.2.5.36
NUMPROC=	____	_____	*NOPFD*	Topic 1.2.5.37
OFFSET=	____	_____	*NO*	Topic 1.2.5.38
OPT=	____	_____	*NO*	Topic 1.2.5.39
OUTDD=	____	_____	*SYSOUT*	Topic 1.2.5.40
PRTEXIT=	____	_____	Not specified	Topic 1.2.5.41
RENT=	____	_____	*NO*	Topic 1.2.5.42
RMODE=	____	_____	*AUTO*	Topic 1.2.5.43
SEQ=	____	_____	*YES*	Topic 1.2.5.44
SIZE=	____	_____	*MAX*	Topic 1.2.5.45
SOURCE=	____	_____	*YES*	Topic 1.2.5.46
SPACE=	____	_____	*1*	Topic 1.2.5.47
SSRANGE=	____	_____	*NO*	Topic 1.2.5.48
TERM=	____	_____	*NO*	Topic 1.2.5.49
TEST=	____	_____	*NO*	Topic 1.2.5.50
TRUNC=	____	_____	*STD*	Topic 1.2.5.51
VBREF=	____	_____	*NO*	Topic 1.2.5.52

	WORD=	____	_____	*NO	Topic 1.2.5.53
	XREFOPT=	____	_____	NO	Topic 1.2.5.54
|	YRWINDOW=	____	_____	1900	Topic 1.2.5.55
	ZWB=	____	_____	YES	Topic 1.2.5.56

IBM Library Server

1.2.2.1.1 Sample Installation Job

COBOL/VSE provides a sample installation job, which can be modified and then used to change compiler option defaults. The job provides an example of how to change the IBM-supplied compiler option defaults.

IGYWEOP1.Z
This sample installation job can be used to change the IBM-supplied defaults.

This job is located in the COBOL sublibrary PRD2.PROD (if the default sublibrary is used).

IBM Library Server

1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

The following worksheet will help you plan and code the compiler options portion of the IGYCOPT macro. To complete the worksheet, fill in the 'Enter * for Fixed' and the 'Enter Selection' columns.

The IGYCOPT macro worksheet also includes a section for compiler phases. That section of the worksheet can be found in "IGYCOPT Macro Worksheet for Compiler Phases" in topic 1.2.3.21.1, following the discussion of compiler phases.

Notes:

1. Coding the asterisk [*], when modifying a compiler option default value, indicates that the option is to be fixed and cannot be replaced by an application programmer.

| 2. The ALOWCBL, DBCSXREF, LVLINFO, and NUMCLS options cannot be overridden at compile time. The 'Enter * for Fixed' worksheet entries for these options is therefore blank.

3. The IBM-supplied default value for ADEXIT, INEXIT, LVLINFO, LIBEXIT, and PRTEXIT is null. The 'IBM-Supplied Default' worksheet entries for these options are therefore blank.

4. The DUMP compiler option cannot be set through the IGYCOPT macro. Unless changed at compile time, DUMP is always set to NODUMP.

Table 11. IGYCOPT Worksheet for Options

Compiler Option	Enter * for Fixed	Enter Selection	IBM-Supplied Default	Syntax Description
ADATA=	____	_____	*NO*	Topic 1.2.5.5
ADEXIT=	____	_____	Not specified	Topic 1.2.5.6
ADV=	____	_____	*YES*	Topic 1.2.5.7
ALOWCBL=		_____	*YES*	Topic 1.2.5.8
AWO=	____	_____	*NO*	Topic 1.2.5.9
BUF=	____	_____	*4K*	Topic 1.2.5.10
CMPR2=	____	_____	*NO*	Topic 1.2.5.11
COMPILE=	____	_____	*NOC(S)*	Topic 1.2.5.12
CURRENCY=	____	_____	*NO*	Topic 1.2.5.13
DATA=	____	_____	*31*	Topic 1.2.5.14

DATEPROC=			*NO*	Topic 1.2.5.15
DBCS=			*NO*	Topic 1.2.5.16
DBCSXREF=			*NO*	Topic 1.2.5.17
DYNAM=			*NO*	Topic 1.2.5.18
FASTSRT=			*NO*	Topic 1.2.5.19
FLAG=			*(I)*	Topic 1.2.5.20
FLAGMIG=			*NO*	Topic 1.2.5.21
FLAGSAA=			*NO*	Topic 1.2.5.22
FLAGSTD=			*NO*	Topic 1.2.5.23
INTDATE=			*ANSI*	Topic 1.2.5.25
INEXIT=			Not specified	Topic 1.2.5.24
LANGUAGE=			*UE*	Topic 1.2.5.26
LIB=			*NO*	Topic 1.2.5.27
LIBEXIT=			Not specified	Topic 1.2.5.28
LINECNT=			*60*	Topic 1.2.5.29
LIST=			*NO*	Topic 1.2.5.30
LITCHAR=			*QUOTE*	Topic 1.2.5.31
LVLINFO=			Not specified	Topic 1.2.5.32
MAP=			*NO*	Topic 1.2.5.33
NAME=			*NO*	Topic 1.2.5.34
NUM=			*NO*	Topic 1.2.5.35
NUMCLS=			*PRIM*	Topic 1.2.5.36
NUMPROC=			*NOPFD*	Topic 1.2.5.37
OFFSET=			*NO*	Topic 1.2.5.38
OPT=			*NO*	Topic 1.2.5.39
OUTDD=			*SYSOUT*	Topic 1.2.5.40
PRTEXIT=			Not specified	Topic 1.2.5.41
RENT=			*NO*	Topic 1.2.5.42
RMODE=			*AUTO*	Topic 1.2.5.43
SEQ=			*YES*	Topic 1.2.5.44
SIZE=			*MAX*	Topic 1.2.5.45
SOURCE=			*YES*	Topic 1.2.5.46
SPACE=			*1*	Topic 1.2.5.47
SSRANGE=			*NO*	Topic 1.2.5.48
TERM=			*NO*	Topic 1.2.5.49
TEST=			*NO*	Topic 1.2.5.50
TRUNC=			*STD*	Topic 1.2.5.51
VBREF=			*NO*	Topic 1.2.5.52

	WORD=	____	_____	*NO	Topic 1.2.5.53
	XREFOPT=	____	_____	NO	Topic 1.2.5.54
|	YRWINDOW=	____	_____	1900	Topic 1.2.5.55
	ZWB=	____	_____	YES	Topic 1.2.5.56

1.2.3.1 Why Place the Compiler Phases in SVA

All compiler phases with the exception of the run dump phases (IGYCRDPR and IGYCRDSC) and the reserved word utility (IGY8RWTU) are eligible for placement in the SVA. The SVA (shared virtual area) is an area of storage that is the same for each partition. Because it is the same space for all users, information stored there can be shared and does not have to be loaded into the partition GETVIS. By sharing the information, more space is made available for the compiler work area.

All compiler phases except those listed in Table 12 have RMODE(ANY) and AMODE(ANY). When phases are loaded into the SVA, VSE will load RMODE(24) phases in low SVA (SVA-24) and RMODE(31) phases in high SVA (SVA-31).

Table 12. RMODE and AMODE Exceptions for Compiler Phases

Phase	RMODE	AMODE
IGYCLIB0	24	ANY
IGYCRCTL	24	ANY
IGYCSIMD	24	ANY
IGYCRWTU	24	24

The IGYCOPT macro indicates where each compiler phase will be loaded--either inside (IN) or outside (OUT) the partition GETVIS. By placing compiler phases in the SVA, the compiler has more storage available for the user's program.

If you indicate that a phase will not reside in the partition GETVIS, you must ensure that you actually place the phase in the SVA. (This information is used by the compiler to determine how much storage to leave for the system to load compiler phases in the partition GETVIS). For a description of how to place the phase in the SVA, see *VSE/ESA System Control Statements*.

The four phases recommended to be placed in the SVA area are:

IGYCRCTL because it is resident in the user partition throughout compilation

IGYCSIMD because it is resident in the user partition throughout compilation

IGYCPGEN because it is one of the largest compiler phases

IGYCSCAN because it is one of the largest compiler phases

Select any or all compiler phases to be placed in the SVA based on frequency of concurrent use and phase size. If your site seldom uses the compiler, there might be no advantage to installing any phases in the SVA. However, if there are frequent compilations and enough SVA storage is available, making the entire compiler resident might be advantageous. If enough SVA storage is not available, priority should then be given to IGYCRCTL and IGYCSIMD, the two phases that are always resident in the user partition during compilation, and to IGYCPGEN and IGYCSCAN, the largest compiler phases.

Another advantage of placing compiler phases in the SVA is that, at compile time, the initialization logic allocates, in the user partition, a storage block large enough to contain the largest phase ***not*** resident in the SVA. Minimizing the space allocation for any given user partition size means more space for the compilation process (that is, larger programs can be compiled within a given user partition), and possibly a more efficient compile. The IGYCPGEN and IGYCSCAN compiler phases are about 250K bytes larger than the next largest compiler phase.

Some phases supplied with COBOL/VSE have the same names as phases supplied with VS COBOL II. Therefore, if you decide to place any COBOL/VSE phases in the SVA, you should ensure that there are no VS COBOL II phases in the SVA at the same time.

IBM Library Server

1.2.3.2 Compiler Phases and Their Defaults

You can indicate where each compiler phase will be loaded in relation to the partition by specifying either IN or OUT. See "Why Place the Compiler Phases in SVA" in topic 1.2.3.1, for more information on why you might or might not want to change these defaults.

IN indicates that the compiler phase will be loaded into the partition from a sublibrary available at compile time. The compiler will allow storage for the phase from the value specified in the SIZE option.

Even though IN is specified for a compiler phase, it still can be placed into the SVA. However, the compiler control phase will ensure that the main storage area reserved for compiler phases is large enough to contain the largest phase for which IN is specified. This will cause some storage to be unused.

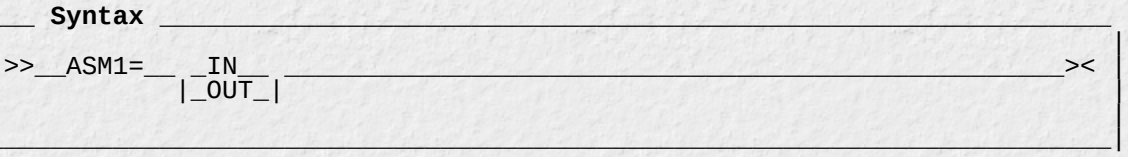
OUT indicates that the compiler phase will not be loaded into the partition from the sublibrary, and therefore must reside in the SVA.

IBM Library Server

1.2.3.3 IGYCASM1

IGYCASM1 is the Assembly 1 phase. It determines the object module storage, allocates the permanent and temporary registers, and optimizes addressability for data and procedure references. It also creates object text for data areas.

Syntax



IBM Library Server

1.2.3.4 IGYCASM2

IGYCASM2 is the Assembly 2 phase. It completes preparation of the object program and creates object text, listings, punch data sets, and tables for the debugging feature.

Syntax

IBM Library Server

1.2.3.5 IGYCDIAG

IGYCDIAG is the diagnostic phase that processes E-form text and generates compiler diagnostics for source program errors. It includes IGYCDIAG plus the following message modules: IGYCxx$D, IGYCxx$1, IGYCxx$2, IGYCxx$3, IGYCxx$4, IGYCxx$5, and IGYCxx$8, where xx is EN, UE, or JA.

Syntax

IBM Library Server

1.2.3.6 IGYCDMAP

IGYCDMAP is the mapping phase. It prepares text for output requested by the MAP option.

Syntax

IBM Library Server

1.2.3.7 IGYCFGEN

IGYCFGEN is the file generation phase. It generates the control blocks for the FDs and SDs defined in the program.

Syntax

IBM Library Server

1.2.3.8 IGYCINIT

IGYCINIT is the initialization phase. It does housekeeping to prepare for running of the processing phases.

Syntax

IBM Library Server

1.2.3.9 IGYCLIBO

IGYCLIBO is the copy, system dependent phase. It provides system dependent services for COPY processing.

Syntax

IBM Library Server

1.2.3.10 IGYCLIBR

IGYCLIBR is the copy phase. It processes library source text and checks the syntax of the COPY, BASIS, and REPLACE statements.

Syntax

IBM Library Server

1.2.3.11 IGYCLSTR

IGYCLSTR is the source listing phase. It prints the source listing with embedded cross-reference and diagnostic information.

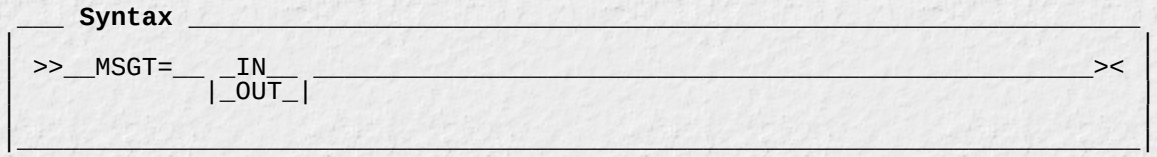
Syntax

IBM Library Server

1.2.3.12 IGYCMSGT

IGYCMSGT represents the header text table and diagnostic message level tables. There is no IGYCMSGT phase. To place the header text table and diagnostic message level tables in the SVA, specify the following phases: IGYCxx$R, IGYCLVL0, IGYCLVL1, IGYCLVL2, IGYCLVL3, and IGYCLVL8, where xx is EN, UE, or JA.

Syntax



IBM Library Server

1.2.3.13 IGYCOPTM

IGYCOPTM is the optimizer phase. It restructures the PERFORM statements and eliminates duplicate calculations.

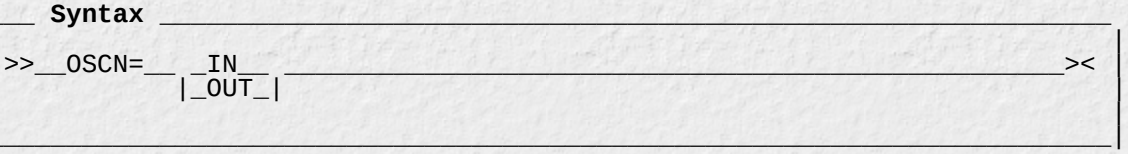
Syntax

IBM Library Server

1.2.3.14 IGYCOSCN

IGYCOSCN is the option scanning phase. It determines the default options, processes the EXEC PARM options, and processes the PROCESS (CBL) statements.

Syntax



IBM Library Server

1.2.3.15 IGYCPGEN

IGYCPGEN is the procedure generation phase. It supplies code for all procedure source verbs.

Syntax

IBM Library Server

1.2.3.16 IGYCRCTL

IGYCRCTL is the resident control phase. It establishes the size of compiler common and working storage, and performs initialization of program common storage.

Syntax

IBM Library Server

1.2.3.17 IGYCRWT

IGYCRWT is the normal reserved word table.

Syntax

IBM Library Server

1.2.3.18 IGYCSAW

IGYCSAW is the Systems Application Architecture (SAA) reserved word table.

Syntax

IBM Library Server

1.2.3.19 IGYCSCAN

IGYCSCAN is the scanning phase. It performs syntax and semantic analysis of the source program and translates the source to intermediate text.

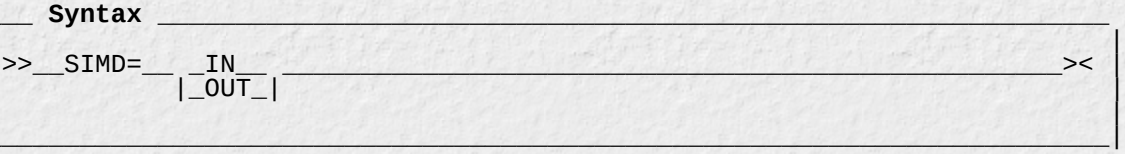
Syntax

IBM Library Server

1.2.3.20 IGYCSIMD

IGYCSIMD is the system interface phase for the COBOL/VSE compiler. This phase is called by all other compiler phases to perform system-dependent functions.

Syntax



IBM Library Server

1.2.3.21 IGYCXREF

IGYCXREF is the cross-reference phase. It sorts user-names and procedure-names in EBCDIC collating sequence.

Syntax

Subtopics:

- 1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

The following worksheet will help you plan and code the compiler options portion of the IGYCOPT macro. To complete the worksheet, fill in the 'Enter * for Fixed' and the 'Enter Selection' columns.

The IGYCOPT macro worksheet also includes a section for compiler phases. That section of the worksheet can be found in "IGYCOPT Macro Worksheet for Compiler Phases" in topic 1.2.3.21.1, following the discussion of compiler phases.

Notes:

1. Coding the asterisk [*], when modifying a compiler option default value, indicates that the option is to be fixed and cannot be replaced by an application programmer.

| 2. The ALOWCBL, DBCSXREF, LVLINFO, and NUMCLS options cannot be overridden at compile time. The 'Enter * for Fixed' worksheet entries for these options is therefore blank.

3. The IBM-supplied default value for ADEXIT, INEXIT, LVLINFO, LIBEXIT, and PRTEXIT is null. The 'IBM-Supplied Default' worksheet entries for these options are therefore blank.

4. The DUMP compiler option cannot be set through the IGYCOPT macro. Unless changed at compile time, DUMP is always set to NODUMP.

Table 11. IGYCOPT Worksheet for Options

Compiler Option	Enter * for Fixed	Enter Selection	IBM-Supplied Default	Syntax Description
ADATA=	____	_____	*NO*	Topic 1.2.5.5
ADEXIT=	____	_____	Not specified	Topic 1.2.5.6
ADV=	____	_____	*YES*	Topic 1.2.5.7
ALOWCBL=		_____	*YES*	Topic 1.2.5.8
AWO=	____	_____	*NO*	Topic 1.2.5.9
BUF=	____	_____	*4K*	Topic 1.2.5.10
CMPR2=	____	_____	*NO*	Topic 1.2.5.11
COMPILE=	____	_____	*NOC(S)*	Topic 1.2.5.12
CURRENCY=	____	_____	*NO*	Topic 1.2.5.13
DATA=	____	_____	*31*	Topic 1.2.5.14

DATEPROC=	____	_____	*NO*	Topic 1.2.5.15
DBCS=	____	_____	*NO*	Topic 1.2.5.16
DBCSXREF=		_____	*NO*	Topic 1.2.5.17
DYNAM=	____	_____	*NO*	Topic 1.2.5.18
FASTSRT=	____	_____	*NO*	Topic 1.2.5.19
FLAG=	____	_____	*(I)*	Topic 1.2.5.20
FLAGMIG=	____	_____	*NO*	Topic 1.2.5.21
FLAGSAA=	____	_____	*NO*	Topic 1.2.5.22
FLAGSTD=	____	_____	*NO*	Topic 1.2.5.23
INTDATE=	____	_____	*ANSI*	Topic 1.2.5.25
INEXIT=	____	_____	Not specified	Topic 1.2.5.24
LANGUAGE=	____	_____	*UE*	Topic 1.2.5.26
LIB=	____	_____	*NO*	Topic 1.2.5.27
LIBEXIT=	____	_____	Not specified	Topic 1.2.5.28
LINECNT=	____	_____	*60*	Topic 1.2.5.29
LIST=	____	_____	*NO*	Topic 1.2.5.30
LITCHAR=	____	_____	*QUOTE*	Topic 1.2.5.31
LVLINFO=		_____	Not specified	Topic 1.2.5.32
MAP=	____	_____	*NO*	Topic 1.2.5.33
NAME=	____	_____	*NO*	Topic 1.2.5.34
NUM=	____	_____	*NO*	Topic 1.2.5.35
NUMCLS=		_____	*PRIM*	Topic 1.2.5.36
NUMPROC=	____	_____	*NOPFD*	Topic 1.2.5.37
OFFSET=	____	_____	*NO*	Topic 1.2.5.38
OPT=	____	_____	*NO*	Topic 1.2.5.39
OUTDD=	____	_____	*SYSOUT*	Topic 1.2.5.40
PRTEXIT=	____	_____	Not specified	Topic 1.2.5.41
RENT=	____	_____	*NO*	Topic 1.2.5.42
RMODE=	____	_____	*AUTO*	Topic 1.2.5.43
SEQ=	____	_____	*YES*	Topic 1.2.5.44
SIZE=	____	_____	*MAX*	Topic 1.2.5.45
SOURCE=	____	_____	*YES*	Topic 1.2.5.46
SPACE=	____	_____	*1*	Topic 1.2.5.47
SSRANGE=	____	_____	*NO*	Topic 1.2.5.48
TERM=	____	_____	*NO*	Topic 1.2.5.49
TEST=	____	_____	*NO*	Topic 1.2.5.50
TRUNC=	____	_____	*STD*	Topic 1.2.5.51
VBREF=	____	_____	*NO*	Topic 1.2.5.52

	WORD=	____	_____	*NO	Topic 1.2.5.53
	XREFOPT=	____	_____	NO	Topic 1.2.5.54
|	YRWINDOW=	____	_____	1900	Topic 1.2.5.55
	ZWB=	____	_____	YES	Topic 1.2.5.56

IBM Library Server

1.2.3 Planning to Place Compiler Phases in the SVA

You might want to make some phases resident in the SVA. in order to minimize the search (path length) for them when the COBOL/VSE compiler is run, or to allow the compiler phases to be shared.

Subtopics:

IBM Library Server

1.2.3 Planning to Place Compiler Phases in the SVA

You might want to make some phases resident in the SVA. in order to minimize the search (path length) for them when the COBOL/VSE compiler is run, or to allow the compiler phases to be shared.

Subtopics:

1.2.3.1 Why Place the Compiler Phases in SVA

All compiler phases with the exception of the run dump phases (IGYCRDPR and IGYCRDSC) and the reserved word utility (IGY8RWTU) are eligible for placement in the SVA. The SVA (shared virtual area) is an area of storage that is the same for each partition. Because it is the same space for all users, information stored there can be shared and does not have to be loaded into the partition GETVIS. By sharing the information, more space is made available for the compiler work area.

All compiler phases except those listed in Table 12 have RMODE(ANY) and AMODE(ANY). When phases are loaded into the SVA, VSE will load RMODE(24) phases in low SVA (SVA-24) and RMODE(31) phases in high SVA (SVA-31).

Table 12. RMODE and AMODE Exceptions for Compiler Phases

Phase	RMODE	AMODE
IGYCLIB0	24	ANY
IGYCRCTL	24	ANY
IGYCSIMD	24	ANY
IGYCRWTU	24	24

The IGYCOPT macro indicates where each compiler phase will be loaded--either inside (IN) or outside (OUT) the partition GETVIS. By placing compiler phases in the SVA, the compiler has more storage available for the user's program.

If you indicate that a phase will not reside in the partition GETVIS, you must ensure that you actually place the phase in the SVA. (This information is used by the compiler to determine how much storage to leave for the system to load compiler phases in the partition GETVIS). For a description of how to place the phase in the SVA, see *VSE/ESA System Control Statements*.

The four phases recommended to be placed in the SVA area are:

IGYCRCTL because it is resident in the user partition throughout compilation

IGYCSIMD because it is resident in the user partition throughout compilation

IGYCPGEN because it is one of the largest compiler phases

IGYCSCAN because it is one of the largest compiler phases

Select any or all compiler phases to be placed in the SVA based on frequency of concurrent use and phase size. If your site seldom uses the compiler, there might be no advantage to installing any phases in the SVA. However, if there are frequent compilations and enough SVA storage is available, making the entire compiler resident might be advantageous. If enough SVA storage is not available, priority should then be given to IGYCRCTL and IGYCSIMD, the two phases that are always resident in the user partition during compilation, and to IGYCPGEN and IGYCSCAN, the largest compiler phases.

Another advantage of placing compiler phases in the SVA is that, at compile time, the initialization logic allocates, in the user partition, a storage block large enough to contain the largest phase ***not*** resident in the SVA. Minimizing the space allocation for any given user partition size means more space for the compilation process (that is, larger programs can be compiled within a given user partition), and possibly a more efficient compile. The IGYCPGEN and IGYCSCAN compiler phases are about 250K bytes larger than the next largest compiler phase.

Some phases supplied with COBOL/VSE have the same names as phases supplied with VS COBOL II. Therefore, if you decide to place any COBOL/VSE phases in the SVA, you should ensure that there are no VS COBOL II phases in the SVA at the same time.

1.2.3.1 Why Place the Compiler Phases in SVA

All compiler phases with the exception of the run dump phases (IGYCRDPR and IGYCRDSC) and the reserved word utility (IGY8RWTU) are eligible for placement in the SVA. The SVA (shared virtual area) is an area of storage that is the same for each partition. Because it is the same space for all users, information stored there can be shared and does not have to be loaded into the partition GETVIS. By sharing the information, more space is made available for the compiler work area.

All compiler phases except those listed in Table 12 have RMODE(ANY) and AMODE(ANY). When phases are loaded into the SVA, VSE will load RMODE(24) phases in low SVA (SVA-24) and RMODE(31) phases in high SVA (SVA-31).

Table 12. RMODE and AMODE Exceptions for Compiler Phases

Phase	RMODE	AMODE
IGYCLIB0	24	ANY
IGYCRCTL	24	ANY
IGYCSIMD	24	ANY
IGYCRWTU	24	24

The IGYCOPT macro indicates where each compiler phase will be loaded--either inside (IN) or outside (OUT) the partition GETVIS. By placing compiler phases in the SVA, the compiler has more storage available for the user's program.

If you indicate that a phase will not reside in the partition GETVIS, you must ensure that you actually place the phase in the SVA. (This information is used by the compiler to determine how much storage to leave for the system to load compiler phases in the partition GETVIS). For a description of how to place the phase in the SVA, see *VSE/ESA System Control Statements*.

The four phases recommended to be placed in the SVA area are:

IGYCRCTL because it is resident in the user partition throughout compilation

IGYCSIMD because it is resident in the user partition throughout compilation

IGYCPGEN because it is one of the largest compiler phases

IGYCSCAN because it is one of the largest compiler phases

Select any or all compiler phases to be placed in the SVA based on frequency of concurrent use and phase size. If your site seldom uses the compiler, there might be no advantage to installing any phases in the SVA. However, if there are frequent compilations and enough SVA storage is available, making the entire compiler resident might be advantageous. If enough SVA storage is not available, priority should then be given to IGYCRCTL and IGYCSIMD, the two phases that are always resident in the user partition during compilation, and to IGYCPGEN and IGYCSCAN, the largest compiler phases.

Another advantage of placing compiler phases in the SVA is that, at compile time, the initialization logic allocates, in the user partition, a storage block large enough to contain the largest phase ***not*** resident in the SVA. Minimizing the space allocation for any given user partition size means more space for the compilation process (that is, larger programs can be compiled within a given user partition), and possibly a more efficient compile. The IGYCPGEN and IGYCSCAN compiler phases are about 250K bytes larger than the next largest compiler phase.

Some phases supplied with COBOL/VSE have the same names as phases supplied with VS COBOL II. Therefore, if you decide to place any COBOL/VSE phases in the SVA, you should ensure that there are no VS COBOL II phases in the SVA at the same time.

IBM Library Server

1.2.3.2 Compiler Phases and Their Defaults

You can indicate where each compiler phase will be loaded in relation to the partition by specifying either IN or OUT. See "Why Place the Compiler Phases in SVA" in topic 1.2.3.1, for more information on why you might or might not want to change these defaults.

IN indicates that the compiler phase will be loaded into the partition from a sublibrary available at compile time. The compiler will allow storage for the phase from the value specified in the SIZE option.

Even though IN is specified for a compiler phase, it still can be placed into the SVA. However, the compiler control phase will ensure that the main storage area reserved for compiler phases is large enough to contain the largest phase for which IN is specified. This will cause some storage to be unused.

OUT indicates that the compiler phase will not be loaded into the partition from the sublibrary, and therefore must reside in the SVA.

IBM Library Server

1.2.3.2 Compiler Phases and Their Defaults

You can indicate where each compiler phase will be loaded in relation to the partition by specifying either IN or OUT. See "Why Place the Compiler Phases in SVA" in topic 1.2.3.1, for more information on why you might or might not want to change these defaults.

IN indicates that the compiler phase will be loaded into the partition from a sublibrary available at compile time. The compiler will allow storage for the phase from the value specified in the SIZE option.

Even though IN is specified for a compiler phase, it still can be placed into the SVA. However, the compiler control phase will ensure that the main storage area reserved for compiler phases is large enough to contain the largest phase for which IN is specified. This will cause some storage to be unused.

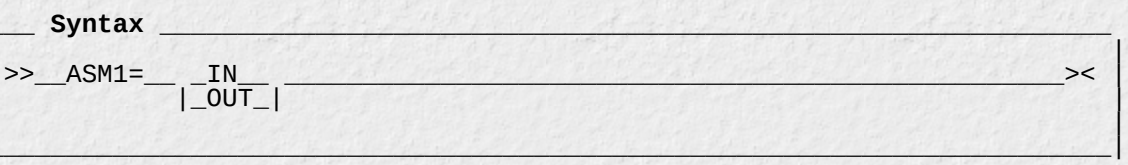
OUT indicates that the compiler phase will not be loaded into the partition from the sublibrary, and therefore must reside in the SVA.

IBM Library Server

1.2.3.3 IGYCASM1

IGYCASM1 is the Assembly 1 phase. It determines the object module storage, allocates the permanent and temporary registers, and optimizes addressability for data and procedure references. It also creates object text for data areas.

Syntax

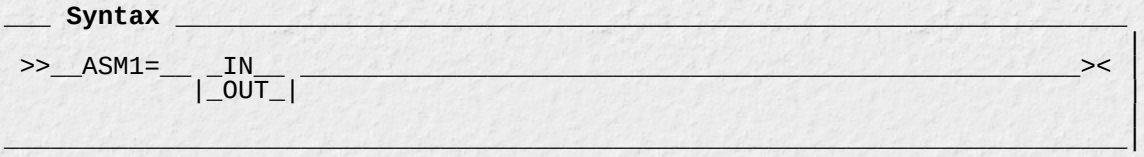


IBM Library Server

1.2.3.3 IGYCASM1

IGYCASM1 is the Assembly 1 phase. It determines the object module storage, allocates the permanent and temporary registers, and optimizes addressability for data and procedure references. It also creates object text for data areas.

Syntax

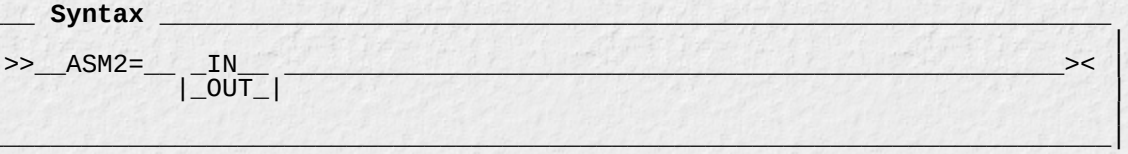


IBM Library Server

1.2.3.4 IGYCASM2

IGYCASM2 is the Assembly 2 phase. It completes preparation of the object program and creates object text, listings, punch data sets, and tables for the debugging feature.

Syntax

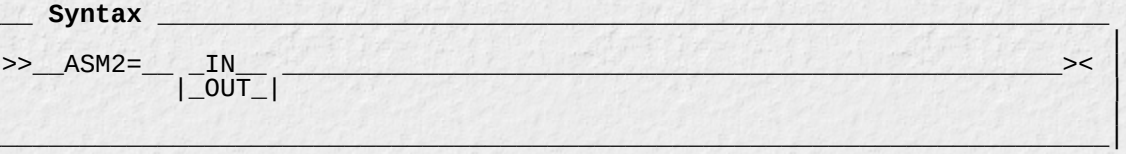


IBM Library Server

1.2.3.4 IGYCASM2

IGYCASM2 is the Assembly 2 phase. It completes preparation of the object program and creates object text, listings, punch data sets, and tables for the debugging feature.

Syntax

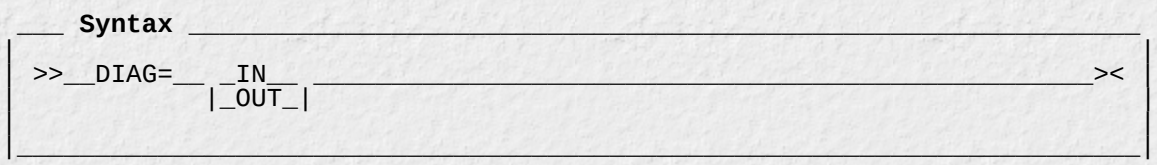


IBM Library Server

1.2.3.5 IGYCDIAG

IGYCDIAG is the diagnostic phase that processes E-form text and generates compiler diagnostics for source program errors. It includes IGYCDIAG plus the following message modules: IGYCxx$D, IGYCxx$1, IGYCxx$2, IGYCxx$3, IGYCxx$4, IGYCxx$5, and IGYCxx$8, where xx is EN, UE, or JA.

Syntax

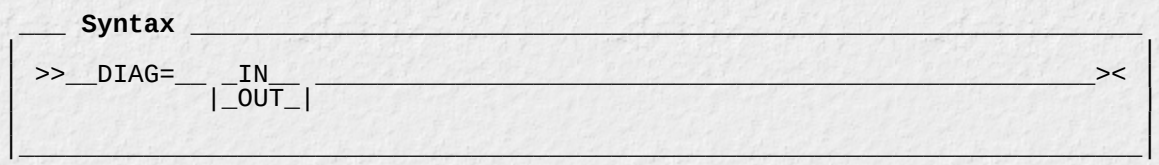


IBM Library Server

1.2.3.5 IGYCDIAG

IGYCDIAG is the diagnostic phase that processes E-form text and generates compiler diagnostics for source program errors. It includes IGYCDIAG plus the following message modules: IGYCxx$D, IGYCxx$1, IGYCxx$2, IGYCxx$3, IGYCxx$4, IGYCxx$5, and IGYCxx$8, where xx is EN, UE, or JA.

Syntax



IBM Library Server

1.2.3.6 IGYCDMAP

IGYCDMAP is the mapping phase. It prepares text for output requested by the MAP option.

Syntax

IBM Library Server

1.2.3.6 IGYCDMAP

IGYCDMAP is the mapping phase. It prepares text for output requested by the MAP option.

Syntax

IBM Library Server

1.2.3.7 IGYCFGEN

IGYCFGEN is the file generation phase. It generates the control blocks for the FDs and SDs defined in the program.

Syntax

IBM Library Server

1.2.3.7 IGYCFGEN

IGYCFGEN is the file generation phase. It generates the control blocks for the FDs and SDs defined in the program.

Syntax

IBM Library Server

1.2.3.8 IGYCINIT

IGYCINIT is the initialization phase. It does housekeeping to prepare for running of the processing phases.

Syntax

IBM Library Server

1.2.3.8 IGYCINIT

IGYCINIT is the initialization phase. It does housekeeping to prepare for running of the processing phases.

Syntax

IBM Library Server

1.2.3.9 IGYCLIBO

IGYCLIBO is the copy, system dependent phase. It provides system dependent services for COPY processing.

Syntax

IBM Library Server

1.2.3.9 IGYCLIBO

IGYCLIBO is the copy, system dependent phase. It provides system dependent services for COPY processing.

Syntax

IBM Library Server

1.2.3.10 IGYCLIBR

IGYCLIBR is the copy phase. It processes library source text and checks the syntax of the COPY, BASIS, and REPLACE statements.

Syntax

IBM Library Server

1.2.3.10 IGYCLIBR

IGYCLIBR is the copy phase. It processes library source text and checks the syntax of the COPY, BASIS, and REPLACE statements.

Syntax

IBM Library Server

1.2.3.11 IGYCLSTR

IGYCLSTR is the source listing phase. It prints the source listing with embedded cross-reference and diagnostic information.

Syntax

IBM Library Server

1.2.3.11 IGYCLSTR

IGYCLSTR is the source listing phase. It prints the source listing with embedded cross-reference and diagnostic information.

Syntax

IBM Library Server

1.2.3.12 IGYCMSGT

IGYCMSGT represents the header text table and diagnostic message level tables. There is no IGYCMSGT phase. To place the header text table and diagnostic message level tables in the SVA, specify the following phases: IGYCxx$R, IGYCLVL0, IGYCLVL1, IGYCLVL2, IGYCLVL3, and IGYCLVL8, where xx is EN, UE, or JA.

Syntax

IBM Library Server

1.2.3.12 IGYCMSGT

IGYCMSGT represents the header text table and diagnostic message level tables. There is no IGYCMSGT phase. To place the header text table and diagnostic message level tables in the SVA, specify the following phases: IGYCxx$R, IGYCLVL0, IGYCLVL1, IGYCLVL2, IGYCLVL3, and IGYCLVL8, where xx is EN, UE, or JA.

Syntax

IBM Library Server

1.2.3.13 IGYCOPTM

IGYCOPTM is the optimizer phase. It restructures the PERFORM statements and eliminates duplicate calculations.

Syntax

IBM Library Server

1.2.3.13 IGYCOPTM

IGYCOPTM is the optimizer phase. It restructures the PERFORM statements and eliminates duplicate calculations.

Syntax

IBM Library Server

1.2.3.14 IGYCOSCN

IGYCOSCN is the option scanning phase. It determines the default options, processes the EXEC PARM options, and processes the PROCESS (CBL) statements.

Syntax

IBM Library Server

1.2.3.14 IGYCOSCN

IGYCOSCN is the option scanning phase. It determines the default options, processes the EXEC PARM options, and processes the PROCESS (CBL) statements.

Syntax

IBM Library Server

1.2.3.15 IGYCPGEN

IGYCPGEN is the procedure generation phase. It supplies code for all procedure source verbs.

Syntax

IBM Library Server

1.2.3.15 IGYCPGEN

IGYCPGEN is the procedure generation phase. It supplies code for all procedure source verbs.

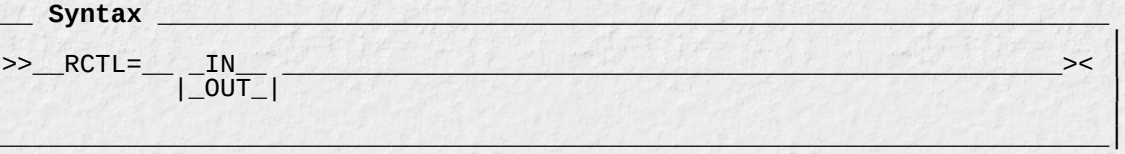
Syntax

IBM Library Server

1.2.3.16 IGYCRCTL

IGYCRCTL is the resident control phase. It establishes the size of compiler common and working storage, and performs initialization of program common storage.

Syntax



IBM Library Server

1.2.3.16 IGYCRCTL

IGYCRCTL is the resident control phase. It establishes the size of compiler common and working storage, and performs initialization of program common storage.

Syntax

IBM Library Server

1.2.3.17 IGYCRWT

IGYCRWT is the normal reserved word table.

Syntax

IBM Library Server

1.2.3.17 IGYCRWT

IGYCRWT is the normal reserved word table.

Syntax

IBM Library Server

1.2.3.18 IGYCSAW

IGYCSAW is the Systems Application Architecture (SAA) reserved word table.

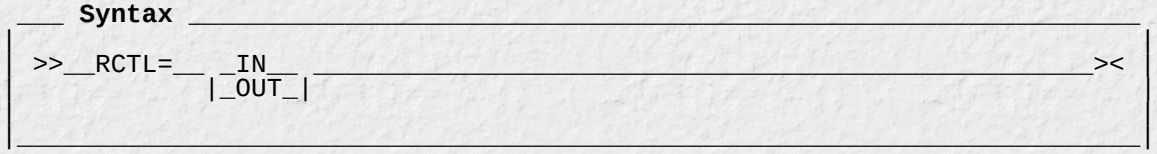
Syntax

IBM Library Server

1.2.3.18 IGYCSAW

IGYCSAW is the Systems Application Architecture (SAA) reserved word table.

Syntax



IBM Library Server

1.2.3.19 IGYCSCAN

IGYCSCAN is the scanning phase. It performs syntax and semantic analysis of the source program and translates the source to intermediate text.

Syntax

IBM Library Server

1.2.3.19 IGYCSCAN

IGYCSCAN is the scanning phase. It performs syntax and semantic analysis of the source program and translates the source to intermediate text.

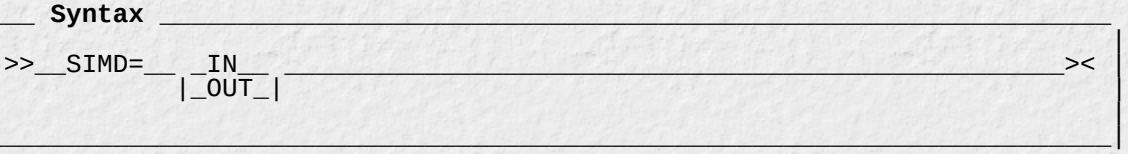
Syntax

IBM Library Server

1.2.3.20 IGYCSIMD

IGYCSIMD is the system interface phase for the COBOL/VSE compiler. This phase is called by all other compiler phases to perform system-dependent functions.

Syntax



IBM Library Server

1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

The following worksheet will help you plan and code the phases portion of the IGYCOPT macro. Circle the value you plan to assign to each phase. For more information on the values that can be assigned to each phase, see "Compiler Phases and Their Defaults" in topic 1.2.3.2.

Note: All phase defaults are initially set to *IN*.

Table 13. IGYCOPT Macro Worksheet for Compiler Phases

Phase	Circle Selection	Syntax Description
ASM1=	*IN* / OUT	Topic 1.2.3.3
ASM2=	*IN* / OUT	Topic 1.2.3.4
DIAG=	*IN* / OUT	Topic 1.2.3.5
DMAP=	*IN* / OUT	Topic 1.2.3.6
FGEN=	*IN* / OUT	Topic 1.2.3.7
INIT=	*IN* / OUT	Topic 1.2.3.8
LIBO=	*IN* / OUT	Topic 1.2.3.9
LIBR=	*IN* / OUT	Topic 1.2.3.10
LSTR=	*IN* / OUT	Topic 1.2.3.11
MSGT=	*IN* / OUT	Topic 1.2.3.12
OPTM=	*IN* / OUT	Topic 1.2.3.13
OSCN=	*IN* / OUT	Topic 1.2.3.14
PGEN=	*IN* / OUT	Topic 1.2.3.15
RCTL=	*IN* / OUT	Topic 1.2.3.16
RWT=	*IN* / OUT	Topic 1.2.3.17
SAW=	*IN* / OUT	Topic 1.2.3.18
SCAN=	*IN* / OUT	Topic 1.2.3.19
SIMD=	*IN* / OUT	Topic 1.2.3.20
XREF=	*IN* / OUT	Topic 1.2.3.21

IBM Library Server

1.2.3.20 IGYCSIMD

IGYCSIMD is the system interface phase for the COBOL/VSE compiler. This phase is called by all other compiler phases to perform system-dependent functions.

Syntax

IBM Library Server

1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

The following worksheet will help you plan and code the phases portion of the IGYCOPT macro. Circle the value you plan to assign to each phase. For more information on the values that can be assigned to each phase, see "Compiler Phases and Their Defaults" in topic 1.2.3.2.

Note: All phase defaults are initially set to *IN*.

Table 13. IGYCOPT Macro Worksheet for Compiler Phases

Phase	Circle Selection	Syntax Description
ASM1=	*IN* / OUT	Topic 1.2.3.3
ASM2=	*IN* / OUT	Topic 1.2.3.4
DIAG=	*IN* / OUT	Topic 1.2.3.5
DMAP=	*IN* / OUT	Topic 1.2.3.6
FGEN=	*IN* / OUT	Topic 1.2.3.7
INIT=	*IN* / OUT	Topic 1.2.3.8
LIBO=	*IN* / OUT	Topic 1.2.3.9
LIBR=	*IN* / OUT	Topic 1.2.3.10
LSTR=	*IN* / OUT	Topic 1.2.3.11
MSGT=	*IN* / OUT	Topic 1.2.3.12
OPTM=	*IN* / OUT	Topic 1.2.3.13
OSCN=	*IN* / OUT	Topic 1.2.3.14
PGEN=	*IN* / OUT	Topic 1.2.3.15
RCTL=	*IN* / OUT	Topic 1.2.3.16
RWT=	*IN* / OUT	Topic 1.2.3.17
SAW=	*IN* / OUT	Topic 1.2.3.18
SCAN=	*IN* / OUT	Topic 1.2.3.19
SIMD=	*IN* / OUT	Topic 1.2.3.20
XREF=	*IN* / OUT	Topic 1.2.3.21

1.2.4.3.2 CICS Reserved Word Table (IGYCCICS)

COBOL/VSE provides an alternate reserved word table specifically for CICS application programs. It is set up so that COBOL words not supported under CICS are flagged by the compiler with an error message.

Contents of the Table: The CICS reserved word table is the same as the default reserved word table except that the following COBOL words are marked as restricted (RSTR):

ACCEPT	FILE-CONTROL	RERUN
CLOSE	INPUT-OUTPUT	REWRITE
DELETE	I-O-CONTROL	SD
DISPLAY	MERGE	SORT
FD	OPEN	START
FILE	READ	WRITE

SORT Users

If you intend to use the SORT statement under CICS (COBOL/VSE supports an interface for the SORT statement under CICS), you must modify the CICS reserved word table before using it. The words highlighted above must be removed from the list of words marked as restricted, because they are required for the SORT function.

Using the Table: In order to use the CICS reserved word table, the compiler option WORD(CICS) must be specified. If you want to use the CICS reserved word table as the default reserved word table, then you must set the default value of the WORD compiler option to WORD=CICS.

Location of the Table: The data used to create the CICS reserved word table is in member IGY8CICS.Z in the COBOL/VSE sublibrary.

1.2.4.1 Why Create Additional Reserved Word Tables?

You can create additional reserved word tables to do the following:

- Translate the reserved words into another language, such as French or German.
- Prevent application programmers from using a particular COBOL/VSE instruction, such as GO TO.
- Control the use of nested programs.
- Flag COBOL verbs without a run-time function in CICS, such as READ/WRITE.

IBM Library Server

1.2.4.2 Controlling Use of Nested Programs

You can allow source programs to access all VS COBOL II Release 3.0 and later NOCMPR2 features, except those related to nested programs, by modifying the reserved word table. This is accomplished through the INFO and RSTR control statements.

IBM Library Server

1.2.4.3 Reserved Word Tables Supplied with COBOL/VSE

Three reserved word tables come on the installation tape: the default reserved word table, the CICS reserved word table, and the SAA reserved word table.

Subtopics:

IBM Library Server

1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

The following worksheet will help you plan and code the phases portion of the IGYCOPT macro. Circle the value you plan to assign to each phase. For more information on the values that can be assigned to each phase, see "Compiler Phases and Their Defaults" in topic 1.2.3.2.

Note: All phase defaults are initially set to *IN*.

Table 13. IGYCOPT Macro Worksheet for Compiler Phases

Phase	Circle Selection	Syntax Description
ASM1=	*IN* / OUT	Topic 1.2.3.3
ASM2=	*IN* / OUT	Topic 1.2.3.4
DIAG=	*IN* / OUT	Topic 1.2.3.5
DMAP=	*IN* / OUT	Topic 1.2.3.6
FGEN=	*IN* / OUT	Topic 1.2.3.7
INIT=	*IN* / OUT	Topic 1.2.3.8
LIBO=	*IN* / OUT	Topic 1.2.3.9
LIBR=	*IN* / OUT	Topic 1.2.3.10
LSTR=	*IN* / OUT	Topic 1.2.3.11
MSGT=	*IN* / OUT	Topic 1.2.3.12
OPTM=	*IN* / OUT	Topic 1.2.3.13
OSCN=	*IN* / OUT	Topic 1.2.3.14
PGEN=	*IN* / OUT	Topic 1.2.3.15
RCTL=	*IN* / OUT	Topic 1.2.3.16
RWT=	*IN* / OUT	Topic 1.2.3.17
SAW=	*IN* / OUT	Topic 1.2.3.18
SCAN=	*IN* / OUT	Topic 1.2.3.19
SIMD=	*IN* / OUT	Topic 1.2.3.20
XREF=	*IN* / OUT	Topic 1.2.3.21

IBM Library Server

1.2.4 Planning to Create an Additional Reserved Word Table

You are provided with a default reserved word table when you install COBOL/VSE. A CICS-specific reserved word table is provided as an alternate reserved word table (See "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2). You can create additional reserved word tables after installation. (During compilation, the value of the WORD compiler option determines which reserved word table is used).

Subtopics:

IBM Library Server

1.2.4 Planning to Create an Additional Reserved Word Table

You are provided with a default reserved word table when you install COBOL/VSE. A CICS-specific reserved word table is provided as an alternate reserved word table (See "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2). You can create additional reserved word tables after installation. (During compilation, the value of the WORD compiler option determines which reserved word table is used).

Subtopics:

IBM Library Server

1.2.4.1 Why Create Additional Reserved Word Tables?

You can create additional reserved word tables to do the following:

- Translate the reserved words into another language, such as French or German.
- Prevent application programmers from using a particular COBOL/VSE instruction, such as GO TO.
- Control the use of nested programs.
- Flag COBOL verbs without a run-time function in CICS, such as READ/WRITE.

IBM Library Server

1.2.4.1 Why Create Additional Reserved Word Tables?

You can create additional reserved word tables to do the following:

- Translate the reserved words into another language, such as French or German.
- Prevent application programmers from using a particular COBOL/VSE instruction, such as GO TO.
- Control the use of nested programs.
- Flag COBOL verbs without a run-time function in CICS, such as READ/WRITE.

IBM Library Server

1.2.4.2 Controlling Use of Nested Programs

You can allow source programs to access all VS COBOL II Release 3.0 and later NOCMPR2 features, except those related to nested programs, by modifying the reserved word table. This is accomplished through the INFO and RSTR control statements.

1.2.4.3.1 Default Reserved Word Table (IGYCRWT)

The default reserved word table is described in an appendix in the *COBOL/VSE Language Reference*.

1.2.4.3.2 CICS Reserved Word Table (IGYCCICS)

COBOL/VSE provides an alternate reserved word table specifically for CICS application programs. It is set up so that COBOL words not supported under CICS are flagged by the compiler with an error message.

Contents of the Table: The CICS reserved word table is the same as the default reserved word table except that the following COBOL words are marked as restricted (RSTR):

ACCEPT	FILE-CONTROL	RERUN
CLOSE	INPUT-OUTPUT	REWRITE
DELETE	I-O-CONTROL	SD
DISPLAY	MERGE	SORT
FD	OPEN	START
FILE	READ	WRITE

SORT Users

If you intend to use the SORT statement under CICS (COBOL/VSE supports an interface for the SORT statement under CICS), you must modify the CICS reserved word table before using it. The words highlighted above must be removed from the list of words marked as restricted, because they are required for the SORT function.

Using the Table: In order to use the CICS reserved word table, the compiler option WORD(CICS) must be specified. If you want to use the CICS reserved word table as the default reserved word table, then you must set the default value of the WORD compiler option to WORD=CICS.

Location of the Table: The data used to create the CICS reserved word table is in member IGY8CICS.Z in the COBOL/VSE sublibrary.

IBM Library Server

| 1.2.4.3.3 SAA Reserved Word Table (IGY8SAAW)

```
| This table contains non-SAA reserved words.  Use of these words will be
| flagged with a warning message if the FLAGSAA compiler option is in
| effect.  This table is supplied to provide compatibility with COBOL
| compilations that comply with SAA standards.
```

IBM Library Server

1.2.4.2 Controlling Use of Nested Programs

You can allow source programs to access all VS COBOL II Release 3.0 and later NOCMPR2 features, except those related to nested programs, by modifying the reserved word table. This is accomplished through the INFO and RSTR control statements.

IBM Library Server

| 1.2.4.3.3 SAA Reserved Word Table (IGY8SAAW)

```
| This table contains non-SAA reserved words.  Use of these words will be
| flagged with a warning message if the FLAGSAA compiler option is in
| effect.  This table is supplied to provide compatibility with COBOL
| compilations that comply with SAA standards.
```

IBM Library Server

1.2.5.1 Specifying COBOL Compiler Options

When you specify compiler options in the IGYCOPT macro, both the option name and its value must be specified in uppercase. If the option name is not specified in uppercase, both the option name and its value will be ignored and the default value will be used. No error message will be issued. If only the option value is not in uppercase, an error message will be issued indicating that an invalid option value has been specified.

1.2.5.2 Options in Support of the COBOL 85 Standard

The following option values are required to conform to the COBOL 85 Standard:

ADV=YES	FLAGMIG=NO	NUMPROC=NOPFD or MIG
CMPR2=NO	FLAGSAA=NO	SEQ=NO
DATEPROC=NO	INTDATE=ANSI	TRUNC=STD
DBCS=NO	LIB=YES	WORD=NO
DYNAM=YES	LITCHAR=QUOTE	ZWB=YES
FASTSRT=NO	NUM=NO	

Note: The term "COBOL 85 Standard" is used in this book to refer to the combination of the following standards:

1. ISO 1989:1985, Programming Languages - COBOL.

   ISO 1989/Amendment 1, Programming Languages - COBOL - Amendment 1: Intrinsic Function Module.

2. X3.23-1985, American National Standard for Information Systems - Programming Language - COBOL.

   X3.23a-1989, American National Standard for Information Systems - Programming Language - Intrinsic Function Module for COBOL.

IBM Library Server

1.2.5.3 Conflicting Compiler Options

Specifying certain compiler option values can create conflicts with other compiler options. Table 14 will help you resolve possible conflicts between compiler options. If you do specify conflicting compiler option values, you will receive a nonzero return code when attempting to assemble the IGYCOPT macro.

Table 14. Conflicting Compiler Options

Compiler Option	Conflicts with:
CMPR2=NO	FLAGMIG=YES
CMPR2=YES	DATEPROC=YES DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
COMPILE=NOC	LIST=YES OFFSET=YES OPT=STD or OPT=FULL TEST=(other than NO) VBREF=YES
DATEPROC=YES	CMPR2=YES FLAGSAA=YES FLAGSTD=(other than NO)
DBCS=YES	CMPR2=YES FLAGMIG=YES FLAGSTD=(other than NO)
FLAGMIG=YES	CMPR2=NO DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
FLAGSAA=YES	ADV=NO CMPR2=YES DATEPROC=(other than NO) DYNAM=NO FLAGMIG=YES FLAGSTD=(other than NO) LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
FLAGSTD=(other than NO)	ADV=NO CMPR2=YES DATEPROC=(other than NO) DBCS=YES DYNAM=NO FLAGMIG=YES FLAGSAA=YES LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD

	SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
LIST=YES	OFFSET=YES
OFFSET=YES	LIST=YES
OPT=STD or OPT=FULL	TEST=(other than NONE for hook location)
TEST=(other than NONE for hook-location suboption)	OPT=STD or OPT=FULL
WORD=xxxx	FLAGSTD=(other than NO)

IBM Library Server

1.2.5.4 Compiler Options Syntax and Descriptions

The following syntax diagrams describe each modifiable compiler option. The text below each diagram describes the effect of selecting a specific parameter.

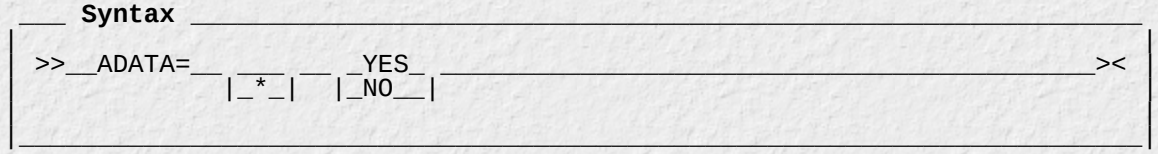
Notice the absence of the DUMP option. Unless changed at compile time, DUMP is always set to NODUMP. This option is not for general use; it is only used at the request of an IBM representative.

Note: The DECK and OBJECT compiler options may only be specified at compile time, and are therefore not described in the following sections. For more information, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.5 ADATA

Syntax



Default
ADATA=NO

YES
Specifies that COBOL/VSE associated data is to be collected and placed in the Associated Data file defined by the SYSADAT DLBL statement.

NO
Specifies that no associated data is to be collected.

Notes:

1. The ADATA option can only be specified at invocation via the option list on the PARM field of JCL.

2. Selection of the Japanese language option might result in DBCS characters written records in the Associated Data file.

3. Specification of NOCOMPILE(W|E|S) might stop compilation prematurely, resulting in a loss of specific Associated Data Records.

4. Specification of INEXIT will prohibit identification of the compilation source file for the Associated Data file.

IBM Library Server

1.2.5.6 ADEXIT

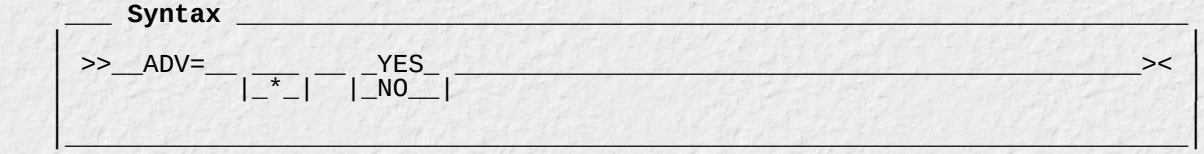
Default
No exit specified.

name
Identifies the name of a module, to be used with the EXIT option, that will be loaded and called by the compiler to monitor the associated data records written by the compiler to the Associated Data file. For more specific information, see the *COBOL/VSE Programming Guide*

Note: This is a read-only exit.

IBM Library Server

1.2.5.7 ADV



Default
ADV=YES

YES
Instructs the compiler to add one byte to the record length for the printer control character. This option may be useful to programmers who use WRITE...ADVANCING in their source files. The first character of the record does **not** have to be explicitly reserved by the programmer.

NO
Instructs the compiler not to adjust the record length for WRITE...ADVANCING. The compiler uses the first character of the specified record area to place the printer control character. The application programmer must ensure that the record description allows for this additional byte.

Notes:

1. ADV=YES conforms to the COBOL 85 Standard. With ADV=YES, the record length on the physical device is one byte larger than the record description length in the source program.

2. If the record length for the output file is not defined in the source code, COBOL ensures that it is appropriately set.

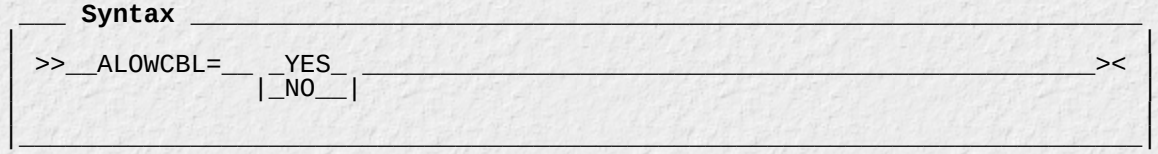
3. If ADV=YES is specified, and the record length for the output file has been defined in the source code, the programmer ***must*** specify the record description length as one byte larger than the source program record description. The programmer ***must*** also specify the block size in correct multiples of the larger record size.

4. If the LINAGE clause is specified in a file description (FD), the compiler treats that file as if ADV=YES had been specified.

IBM Library Server

1.2.5.8 ALOWCBL

Syntax



Default
ALOWCBL=YES

YES
Instructs the compiler to allow the use of the PROCESS (or CBL) statements in COBOL programs.

NO
Instructs the compiler to diagnose the use of PROCESS (or CBL) statements in a program as an error.

Notes:

1. ALOWCBL cannot be replaced at compile time because it cannot be included in the PROCESS (or CBL) statement.

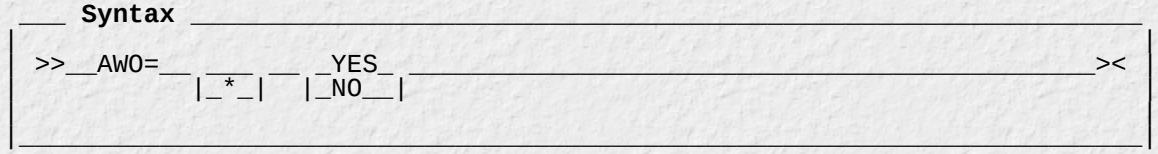
2. The PROCESS (or CBL) statement is used to specify compiler option parameters within source programs. If your installation requirements do not allow compiler options to be specified in a source program, specify ALOWCBL=NO.

3. You must specify ALOWCBL=YES for application programs that run under CICS.

IBM Library Server

1.2.5.9 AWO

Syntax



Default
AWO=NO

YES
Activates the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

Performance Consideration: Using AWO=YES generally results in fewer calls to Data Management Services for run-time files when handling I/O.

NO
Does not activate the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

IBM Library Server

1.2.5.10 BUF

Default
BUF=4K

integer
Specifies the amount of dynamic storage, in bytes, to be allocated to each compiler work file buffer. The minimum value is 256 bytes.

Performance Consideration: Using BUF=integer generally improves compile-time performance by reducing the number of I/Os to the work files.

integerK
Specifies the amount of dynamic storage to be allocated to buffers in increments of 1K (1024) bytes.

Notes:

1. BUF and SIZE values are used by the compiler in determining how much storage is to be used during compilation. The amount allocated to the buffers is included in the amount of main storage available to the compiler for the SIZE option.

2. BUF cannot exceed the track capacity for the device used, nor can it exceed the maximum allowed by data management services.

IBM Library Server

1.2.5.11 CMPR2

Syntax

Default
CMPR2=NO

YES
Causes the compiler to generate code that is run-time compatible with valid VS COBOL II Release 2 programs. See the *COBOL/VSE Migration Guide* for details of language elements that are sensitive to CMPR2.

NO
Causes the compiler to generate code that may not be run-time compatible with valid VS COBOL II Release 2 programs for some COBOL 85 standard language constructs.

Notes:

1. Existing applications which rely on such matters as the format of the listing file, mapping of message numbers to messages, and output of the TERM option may need to be converted to COBOL/VSE.

2. Functions are **NOT** allowed under CMPR2. The word 'function' may be a dataname under CMPR2, but under NOCMPR2 it is considered a reserved word.

3. New language extensions for defining and manipulating procedural pointers are only supported with NOCMPR2.

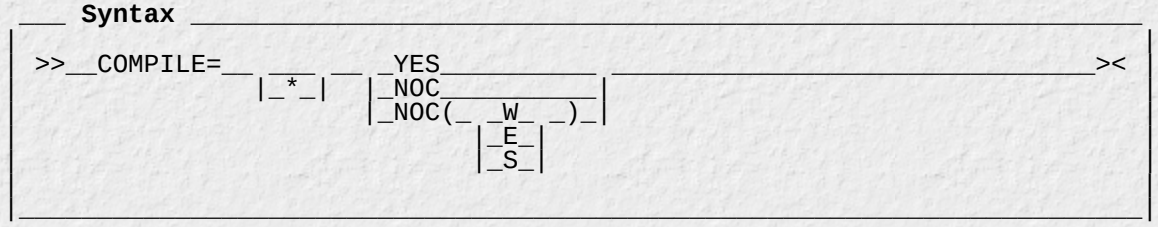
4. CMPR2=YES cannot be specified with certain other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflicting compiler options.

5. CMPR2=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.12 COMPILE

Syntax



Default
COMPILE=NOC(S)

YES
Indicates that you want full compilation, including diagnostics and object code.

NOC
Indicates that you want only a syntax check.

NOC(W)
NOC(E)
NOC(S)
Specifies an error message level:

W for warning

E for error

S for severe

When an error of the specified level or of a more severe level occurs, compilation stops, and only syntax checking is done for the remainder of the compilation.

Notes:

1. If the following compiler options are explicitly or implicitly specified in a program, COMPILE=NOC issues a warning message during compile time:

LIST=YES
OFFSET=YES
OPT=STD or OPT=FULL
TEST=(other than NO)
VBREF=YES

Specifying these options together with COMPILE=NOC, while attempting

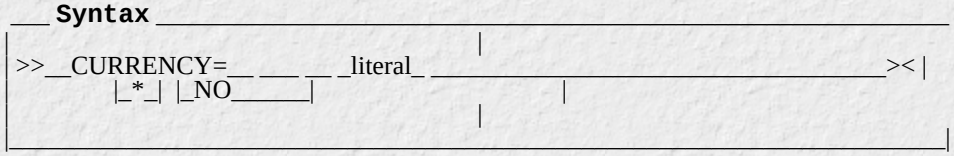
to assemble the customization macro, will result in a nonzero return code.

2. Specifying NOCOMPILE may affect the Associated Data file by stopping compilation prematurely, resulting in loss of specific messages.

IBM Library Server

1.2.5.13 CURRENCY

Syntax



The COBOL default currency symbol is the dollar sign ($). The CURRENCY option allows you to define an alternate default currency symbol.

Default
CURRENCY=NO

literal
Represents the default currency symbol you want to use in your program.

The literal must be a nonnumeric literal representing a one-byte EBCDIC character that **must not** be any of the following:

- Digits zero (0) through nine (9)
- Uppercase alphabetic characters: A B C D P R S V X Z
- Lowercase alphabetic characters a through z
- The space
- Special characters: * + - / , . ; ( ) = "
- Uppercase alphabetic character G, if the COBOL program defines a DBCS item with the PICTURE symbol G. The PICTURE clause will be invalid for that DBCS item because the symbol G is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character N, if the COBOL program defines a DBCS item with the PICTURE symbol N. The PICTURE clause will be invalid for that DBCS item because the symbol N is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character E, if the COBOL program defines an external floating point item. The PICTURE clause will be invalid for the external floating-point item because the symbol E is considered to be a currency symbol in the PICTURE clause.

The literal (including hex literal) syntax rules are as follows:

- The literal delimiters may be either quotes or apostrophes regardless of any option setting for literal delimiters.

- When an apostrophe (') is to be the currency sign, the embedded apostrophe must be "doubled". That is, two apostrophes must be coded to represent one apostrophe within the literal. For example:

  "" or """"

- The format for a hex literal specification is as follows:

  X'H1H2' or X"H1H2"

  where H1H2 is a valid hexadecimal value representing a one-byte EBCDIC character conforming to the rules for the currency sign literal as described above. Alphabetic characters in the hex literal must be in uppercase.

  Note: Hex values of X'20 or X'21' are not allowed.

NO
indicates that no alternate default currency sign is provided by way of the CURRENCY option, and the dollar sign will be used as the default currency sign for the program if the CURRENCY option is not specified at compile time.

The value NO provides the same results for the source program as omitting the CURRENCY SIGN clause in the COBOL source program.

Notes:

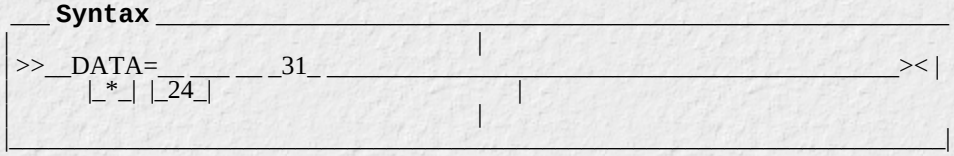
1. You can use the CURRENCY option as an alternative to the CURRENCY SIGN clause (which is specified in the COBOL source program) for selecting the currency symbol you will use in the PICTURE clause of your COBOL program.

2. When both the CURRENCY option and the CURRENCY SIGN clause are used in a program, the symbol specified in the CURRENCY SIGN clause will be considered the currency symbol in a PICTURE clause when that symbol is used (even if the CURRENCY option is fixed {*}).

IBM Library Server

1.2.5.14 DATA

Syntax



Default
DATA=31

31
Causes user data areas, such as working storage and FD record areas, to be allocated from unrestricted storage in storage acquired by a GETVIS with the LOC=ANY option. Specifying this option can result in storage being acquired in virtual addresses either above or below 16 megabytes. The operating system generally satisfies the request with space in virtual addresses above 16 megabytes, if it is available.

24
Causes user data areas to be allocated in virtual addresses below 16 megabytes in storage acquired by a GETVIS with the LOC=BELOW option.

Specify DATA=24 for programs running in 31-bit mode that are passing data parameters to programs in 24-bit mode. This includes the following cases:

- An AMODE(31) program is passing items in its working storage to an AMODE(24) program.
- An AMODE(31) program is passing, by reference, data items received from its caller to an AMODE(24) program. DATA=24 is required even when the data received is below the 16-megabyte line.

Otherwise, the data may not be addressable by the called program.

Notes:

1. When a program is compiled with the RENT option and run in 31-bit mode, the DATA option controls how dynamic storage is acquired.
2. This option has no effect on a program compiled with the NORENT option, or a program run in 24-bit mode.
3. This option is ignored at run time on a 24-bit mode system, but is always present in the object module. If the module is subsequently moved, without recompilation, to an extended architecture system and run in 31-bit mode, the DATA option will control how dynamic storage

is acquired.

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.15 DATEPROC

Syntax

The DATEPROC option determines whether the compiler will perform date processing using the DATE FORMAT clause and other language constructs. IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA must be installed to specify anything other than DATEPROC=NO.

Default
DATEPROC=NO

FLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. In addition, specifying DATEPROC=FLAG instructs the compiler to flag, either with an information-level message or a warning-level message as appropriate, each statement that uses or is affected by date processing.

NOFLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. Statements that use or are affected by date processing are not flagged with information-level or warning-level messages.

NO
Instructs the compiler to treat the DATE FORMAT clause as comments and to disable automatic date processing. In the case of the new intrinsic functions, specifying DATEPROC=NO instructs the compiler to generate object code that returns a default value whenever a new intrinsic function is used.

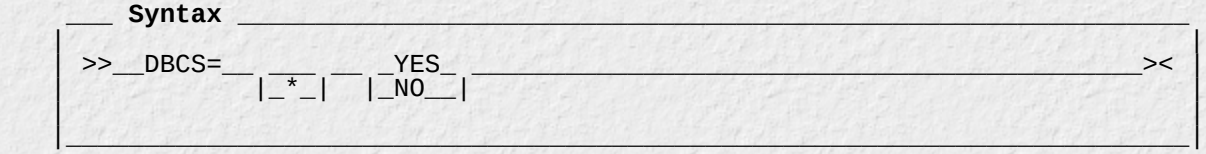
Notes:

1. Error-level and severe-level messages are issued regardless of whether DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. DATEPROC=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.16 DBCS



Default
DBCS=NO

YES
Instructs the compiler to recognize X'0E' and X'0F' in a nonnumeric literal and to treat them as Shift-Out and Shift-In control characters for delimiting DBCS data.

NO
Specifies the compiler is not to recognize X'0E' and X'0F' in a nonnumeric literal.

Notes:

1. The presence of DBCS data inside the nonnumeric literal may cause the compiler to disallow certain uses of that literal. For example, DBCS characters are not allowed as program names or file system-names.

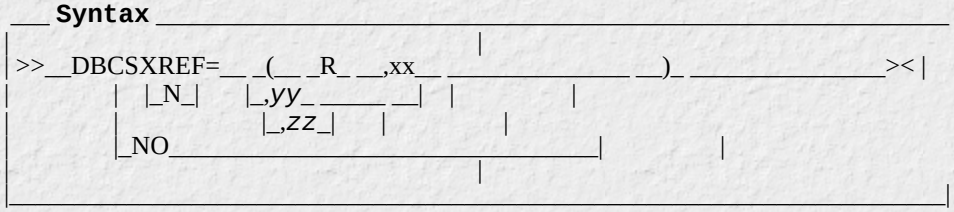
2. DBCS=NO supports the COBOL 85 standard.

3. DBCS=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.17 DBCSXREF

Syntax



Default
DBCSXREF=NO

NO
Specifies that no ordering program will be used for cross-reference of DBCS names. If the XREF phase is specified, a DBCS names cross-reference listing will be provided based on their physical order in the program.

R
Specifies that the DBCS Ordering Support Program (DBCSOS) will be loaded into the partition.

N
Specifies that the DBCSOS will be loaded into shared storage.

xx
Names a phase of the relevant ordering program to produce DBCS cross references. It must be 8 characters in length.

yy
Names an ordering type. It must be 2 characters in length. The default ordering type defined by the specified ordering program will occur if this parameter is omitted.

zz
Names the encode table used by the specified ordering type. It must be 8 characters in length. The default encode table associated with the particular ordering type will occur if this parameter is omitted.

Notes:

1. The DBCS Ordering Support Program (DBCSOS) must be installed to specify anything other than DBCSXREF=NO. DBCSOS is not available for use under VSE, therefore you should specify DBCSXREF=NO.

2. Specifying both XREFOPT=NO and DBCSXREF with an ordering program will result in a nonzero return code while attempting to assemble the customization macro.

3. The assembly process will terminate when validation diagnoses:

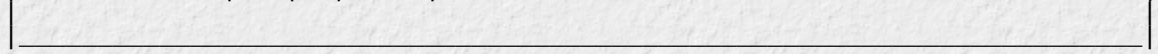
   - Characters other than 'R' and 'N'
   - An invalid parameter length
   - Missing parameters after a comma
   - Missing 'yy' when 'zz' is specified

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.18 DYNAM

Syntax



Default
DYNAM=NO

YES
Causes subprograms that are invoked through the CALL literal statement to be dynamically loaded.

Performance Consideration: Using DYNAM=YES eases subprogram maintenance since the application will not have to be relink-edited if the subprogram is changed. However, individual applications may experience some performance degradation due to a longer path length, but overall system performance may be slightly improved.

NO
Causes the text files of subprograms called with a CALL literal statement to be included with the calling program into a single phase.

Notes:

1. DYNAM=YES is used in support of the COBOL 85 Standard.

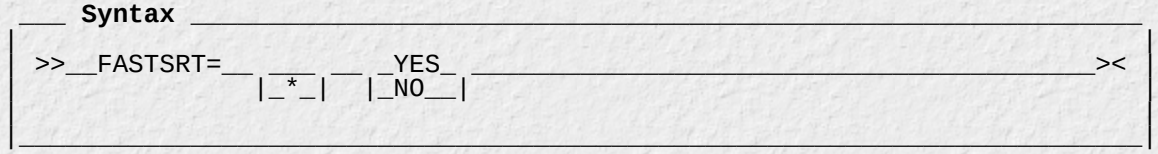
2. The DYNAM option has no effect on the CALL identifier statement at compile time. The CALL identifier statement always compiles to a dynamic call.

| 3. Do not specify DYNAM=YES for applications running under CICS.

IBM Library Server

1.2.5.19 FASTSRT

Syntax



Default
FASTSRT=NO

YES
Specifies that DFSORT/VSE or a comparable product (for example, Sort/Merge II), is to perform I/O when using either the USING or GIVING option.

Performance Consideration: Using FASTSRT=YES eliminates the overhead, in terms of CPU time, of returning to COBOL/VSE after each record is processed. However, there are restrictions you must follow if you choose to use this option. (For a detailed description of the restrictions, see *COBOL/VSE Programming Guide*.)

NO
Specifies that COBOL/VSE does the I/O for the sort/merge.

Notes:

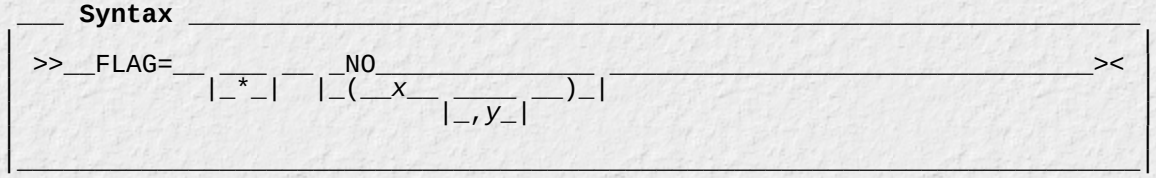
1. If FASTSRT is in effect at compile time, the compiler verifies that the FASTSRT interface can be used for all restrictions except: (1) those requiring use of a device other than a direct-access device for sort work files; and (2) if appropriate, the BLKSIZE parameter of the JCL DLBL statement for the input/output file must match the File Description (FD) of the file.

   If FASTSRT cannot be used, the compiler generates a diagnostic message and prevents the sort program from performing I/O when using either the USING or GIVING options. Therefore it may be to your advantage to specify YES as the default.

IBM Library Server

1.2.5.20 FLAG

Syntax



Default
FLAG=(I)

Note: The second severity level used in this syntax must be equal to or higher than the first.

x
I|W|E|S|U

Specifies that errors at or above the severity level specified are to be flagged and written at the end of the source listing.

ID	Type	Return Code
I	Information	0
W	Warning	4
E	Error	8
S	Severe error	12
U	Unrecoverable error	16

y
I|W|E|S|U

The optional second severity level specifies the level of syntax messages to be embedded in the source listing in addition to being at the end of the listing.

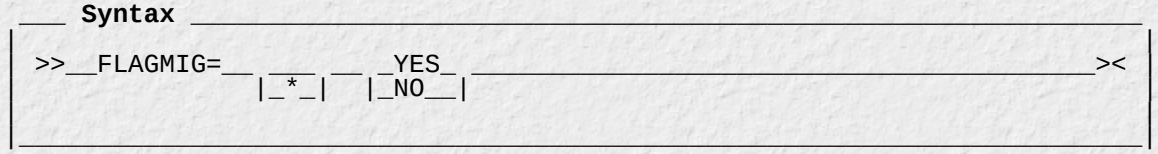
NO
Indicates that no error messages are flagged.

Note: If the messages are to be embedded, SOURCE must be specified at compile time. Embedded messages enhance productivity because they are placed after the referenced source statement.

IBM Library Server

1.2.5.21 FLAGMIG

Syntax



Default
FLAGMIG=NO

YES
Flags those VS COBOL II Release 2 language elements which might have different semantics in COBOL/VSE.

NO
Does not flag those VS COBOL II Release 2 language elements which have different semantics in COBOL/VSE.

Notes:

1. FLAGMIG=YES must be specified with CMPR2=YES to take advantage of this option.

2. FLAGMIG=YES has no effect when specified with CMPR2=NO.

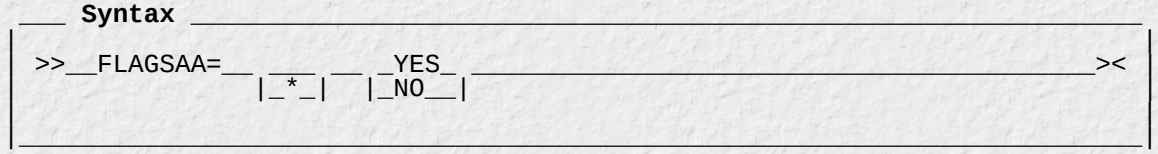
3. FLAGSAA=YES, FLAGSTD=YES, and DBCS=YES will all replace the FLAGMIG option.

4. FLAGMIG=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.22 FLAGSAA

Syntax



Default
FLAGSAA=NO

YES
Flags language elements which inhibit portability across Systems Application Architecture (SAA) COBOL systems.

NO
Does not flag language elements which inhibit portability across SAA COBOL systems.

Notes:

1. FLAGSAA=YES issues a warning (W) message to identify all SAA nonportable items.

   Specifying the following options together with FLAGSAA=YES, while attempting to assemble the customization macro, will result in a nonzero return code:

   CMPR2=YES
   FLAGSTD
   FLAGMIG

2. If the following compiler options are explicitly or implicitly specified in a program, FLAGSAA=YES issues a warning message during compile time:

   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DYNAM=NO
   FLAGMIG=YES
   FLAGSTD=(other than NO)
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT
   WORD=(other than NO or RWT)
   ZWB=NO

3. The FLAGSAA option identifies COBOL/VSE reserved words that are outside the current SAA specifications. Furthermore, the FLAGSAA option also identifies all COBOL words that are not in COBOL/VSE, but are reserved by other IBM SAA compilers.

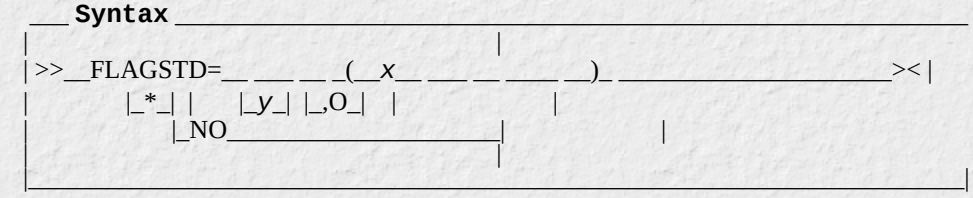
4. FLAGSAA=NO supports the COBOL 85 standard.

5. FLAGSAA can be in conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.23 FLAGSTD

Syntax



Default
FLAGSTD=NO

x
Can be M, I, or H to specify flagging for a FIPS (Federal Information Processing Standard) COBOL subset or standard.

M = ANS minimum subset of Standard COBOL.
I = ANS intermediate subset, containing those additional intermediate subset language elements which are not part of the ANS minimum subset.
H = ANS high subset, containing those additional high subset language elements which are not part of the ANS intermediate subset.

y
Can be any one or two combinations of D, N, or S to further define the level of flagging produced.

D Specifies ANS Debug module Level 1
N Specifies ANS Segmentation Module Level 1
S Specifies ANS Segmentation Module Level 2 (S is a superset of N.)

O
Specifies that obsolete language elements occurring in any of the above sets will be flagged.

NO
Specifies that no FIPS flagging will be accomplished.

Notes:

1. When FIPS flagging is specified, language syntax, such as the DATE FORMAT clause, used by the COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option) is flagged as nonconforming nonstandard IBM extensions to the COBOL 85 Standard.

2. When FIPS flagging is specified, informational messages in the source

program listing will identify:

- Whether the language element is obsolete, nonconforming standard, or nonconforming nonstandard (language elements that are both obsolete and nonconforming are flagged as obsolete only)
- The clause, statement, or header containing the nonconforming or obsolete syntax
- The source program line and an indication of the starting column within that line
- The level or optional module to which the language element belongs

3. FIPS flagging is suppressed when either:

   - Any error diagnosed as level E or higher occurs
   - The ABBR function of the WORD option is used, because standards conformance requires the standard set of reserved words

4. The FLAGSTD option can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

5. If the following compiler options are explicitly or implicitly specified in a program, FLAGSTD=(other than NO) issues a warning message during compile time:

   ADV=NO LITCHAR=APOST
   CMPR2=YES NUM=YES
   DATEPROC=(other than NO) NUMPROC=PFD
   DBCS=YES SEQ=YES
   DYNAM=NO TRUNC=OPT or BIN
   FASTSRT=YES WORD=(other than NO or RWT)
   FLAGMIG=YES ZWB=NO
   LIB=NO

6. Specifying the following options together with FLAGSTD=(other than NO), while attempting to assemble the customization macro, will result in a nonzero return code.

   ADV=NO LITCHAR=APOST
   CMPR2=YES NUM=YES
   DATEPROC=(other than NO) NUMPROC=PFD
   DBCS=YES SEQ=YES
   DYNAM=NO TRUNC=OPT or BIN
   FLAGMIG=YES WORD=(other than NO or RWT)
   LIB=NO ZWB=NO

7. FLAGSTD may affect the Associated Data file by generating error records for FIPS standard conformation messages. Error messages are not guaranteed to be sequential in regard to source record numbers.

IBM Library Server

1.2.5.24 INEXIT

Syntax

Default
No exit specified.

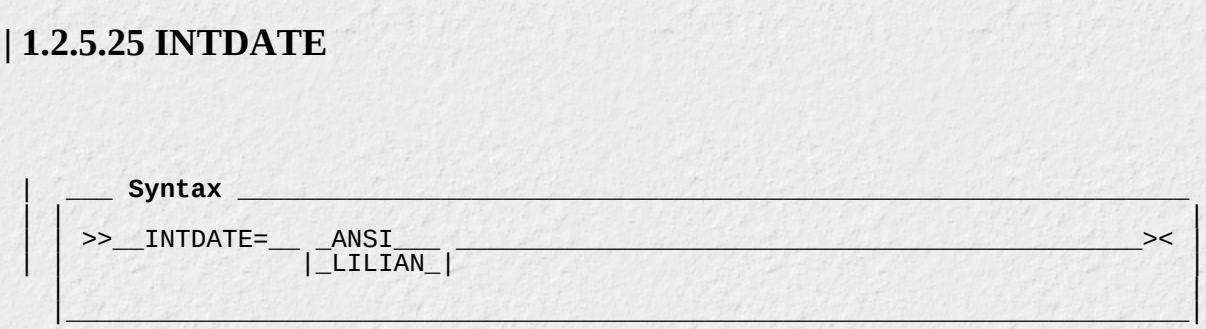
name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain source statements instead of using SYSIPT. When the option is supplied, SYSIPT is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

Note: Specification of INEXIT will prohibit identification of the compilation source file.

IBM Library Server

| 1.2.5.25 INTDATE

| **Syntax**



| Default
| INTDATE=ANSI

| ANSI
| Instructs the compiler to use the ANSI COBOL Standard starting date
| for Integer date format dates used with date intrinsic functions. Day
| 1 = Jan 1, 1601.

| With INTDATE(ANSI), the date intrinsic functions will return the same
| results as in COBOL/VSE without PTF UQ04360.

| LILIAN
| Instructs the compiler to use the Language Environment Lilian starting
| date for integer date format dates used with date intrinsic functions.
| Day 1 = Oct 15, 1582.

| With INTDATE(LILIAN), the date intrinsic functions will return results
| compatible with the LE/VSE date callable services. These results will
| be different than in COBOL/VSE without PTF UQ04360.

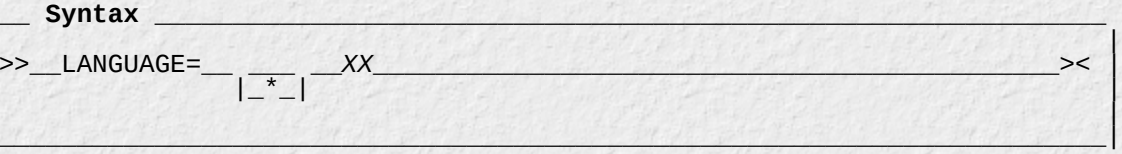
| **Notes:**

| 1. When INTDATE(LILIAN) is in effect, CEECBLDY will not be useable since
| you will have no way to turn an ANSI integer into a meaningful date
| using either intrinsic functions or callable services. If you code a
| CALL literal statement with CEECBLDY as the target of the call with
| INTDATE(LILIAN) in effect, the compiler will diagnose this and convert
| the call target to CEEDAYS.

| 2. If you set your installation option to INTDATE(LILIAN), you should
| recompile all of your COBOL/VSE programs that use Intrinsic Functions
| to ensure that all of your code will be using the LILIAN integer date
| standard. This method is the safest, because you can store integer
| dates, pass them between programs, and even pass them from PL/I to
| COBOL to C programs and have no problems.

1.2.5.26 LANGUAGE

Syntax



Default
LANGUAGE=UE

XX
Specifies the language for compiler output messages. Entries for this parameter may be selected from the following list:

Table 15. Entries for the LANGUAGE compiler option

Entry	Language
EN or **ENGLISH**	Mixed case US English
JA, **JP**, or **JAPANESE**	Japanese
UE or **UENGLISH**	Uppercase US English

Notes:

1. The LANGUAGE option name must consist of at least the first two identifying characters. Other characters following the first two identifiers may be used, however only the first two will be used to determine the language name.

2. This compiler option does not affect the language in which run-time messages are displayed. For more information on run-time options and messages, refer to the *LE/VSE Programming Guide*.

3. Some printers use only uppercase and may not accept output in mixed-case (LANGUAGE=ENGLISH).

4. To specify the Japanese language option, the Japanese National Language Feature must be installed.

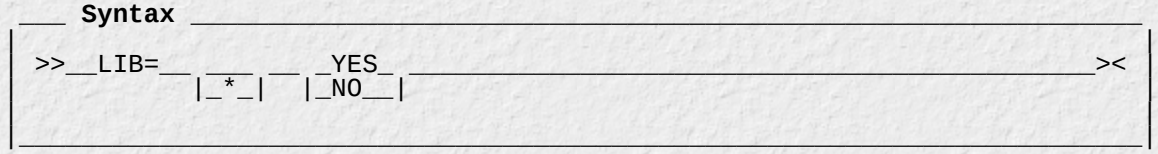
5. To specify the English language option (mixed-case English), the US English Language Feature must be installed.

6. If your installation provides a language other than those listed above, and you choose it to be your installation's default, you must specify at least the first two characters of the language name. The first two characters must be alphanumeric.

IBM Library Server

1.2.5.27 LIB

Syntax



Default
LIB=NO

YES
Indicates that the source program contains COPY or BASIS statements.

NO
Indicates that the source program does not contain COPY or BASIS statements.

Notes:

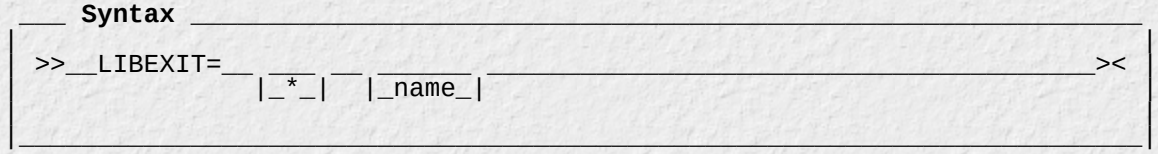
1. LIB=YES conforms to the COBOL 85 Standard.

2. LIB=YES is used when compiling a program containing COPY or BASIS statements. Specifying LIB=YES when there are no COPY or BASIS statements in a source program results in an unnecessary invocation of the COPY processing phase, unnecessary allocation of a buffer for the Librarian, and an unnecessary pass of the source text. However, compilation results are not affected.

IBM Library Server

1.2.5.28 LIBEXIT

Syntax



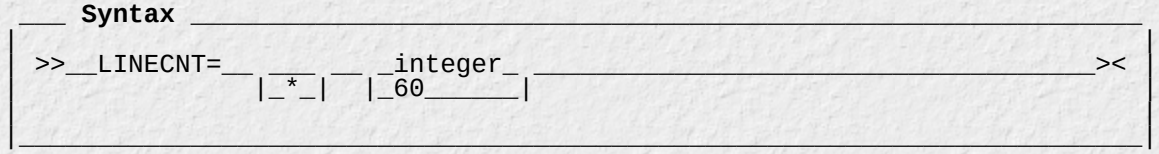
Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain copy statements instead of using the VSE Librarian to read the library search chain. When the option is supplied, the VSE Librarian is not called. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.29 LINECNT

Syntax



Default
LINECNT=60

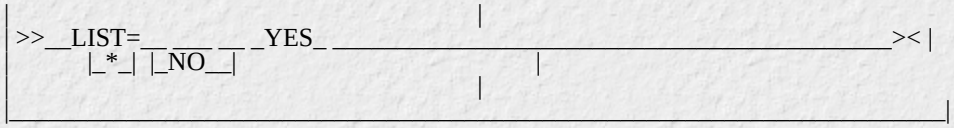
integer
Specifies the number of lines to be printed on each page of the compiler source code listing. Three of the lines are used to generate headings. For example, if you specify LINECNT=60, 57 lines of source code are printed on each page of the output listing, and 3 lines are used for headings.

| **Note:** The LINECNT installation option is equivalent to the LINECOUNT
| compile-time option.

IBM Library Server

1.2.5.30 LIST

Syntax



Default
NO

YES
Produces a listing that includes:

- The assembler-language expansion of source code
- Information about working storage
- Global tables
- Literal pools

NO
Suppresses this listing.

Notes:

1. Unless the installation default value of the LIST option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LISTX option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE LIST option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

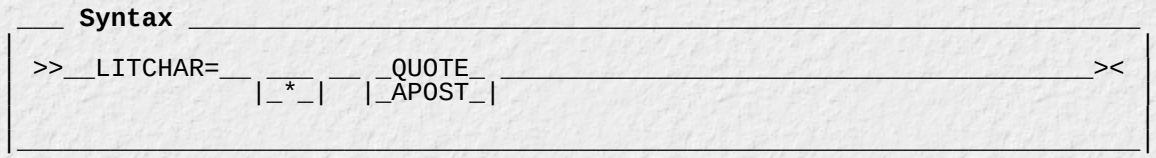
2. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code and an error message during assembly of the customization macro.

3. LIST=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.31 LITCHAR

Syntax



Default
LITCHAR=QUOTE

QUOTE
Specifies that double quotation marks (") are used to delineate literals and in the generation of figurative constants.

APOST
Specifies that an apostrophe (') is used to delineate literals and in the generation of figurative constants.

Notes:

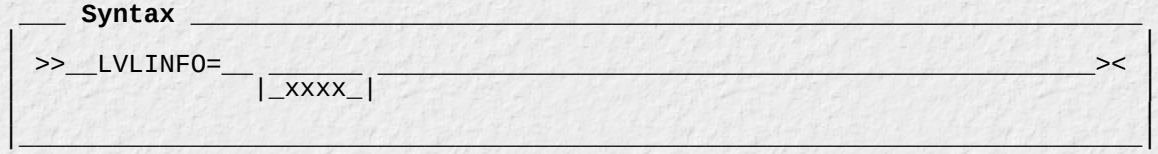
1. LITCHAR=QUOTE is used in support of the COBOL 85 Standard.

2. If you have an existing source library, determine whether QUOTE or APOST is used consistently.

IBM Library Server

1.2.5.32 LVLINFO

Syntax



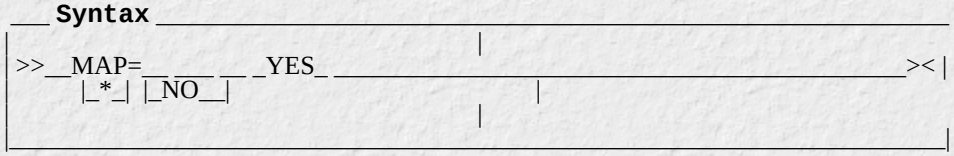
Default
No characters are specified.

xxxx
Identifies the one to four alphanumeric characters that will be inserted into the listing header following the Release number (the last four bytes of the signature area). This option may be used to identify "compiler level" information within the listing header.

IBM Library Server

1.2.5.33 MAP

Syntax



Default
MAP=NO

YES
Maps items declared in the Data Division. Map output includes:

- Data Division map
- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled without the RENT compiler option

NO
Mapping is not performed.

IBM Library Server

1.2.5.34 NAME

Syntax

Default
NAME=NO

NOALIAS
When used in conjunction with specific JCL options performs the following:

1. When used with LINK or CATAL, NOALIAS precedes each object deck written to SYSLNK with a linkage editor PHASE statement (PHASE phasename,*). The phase name (phasename) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

2. When used with DECK, NOALIAS precedes each object deck written to SYSPCH with a VSE Librarian CATALOG statement (CATALOG modname.OBJ,REPLACE=YES). The module name (modname) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

ALIAS
Specifying ALIAS has the same effect as NOALIAS.

NO
Does not produce linkage editor PHASE statements or VSE Librarian CATALOG statements.

Notes:

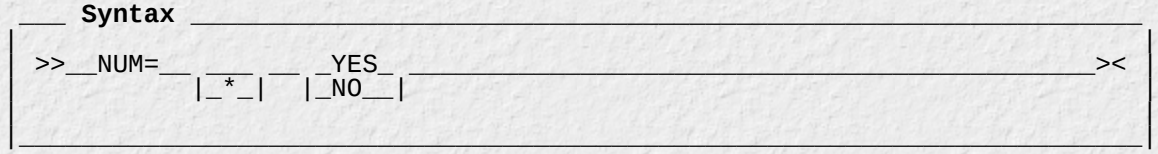
1. The NAME option allows you to create multiple modules in a program library with a single batch compilation. This can be useful for dynamic calls, for example.

2. NAME=NOALIAS or NAME=ALIAS supports the COBOL 85 Standard.

IBM Library Server

1.2.5.35 NUM

Syntax



Default
NUM=NO

YES
Uses the line numbers from the source program rather than compiler-generated line numbers for error messages and procedure maps.

NO
Uses the line numbers generated from the compiler for error messages and procedure maps.

Notes:

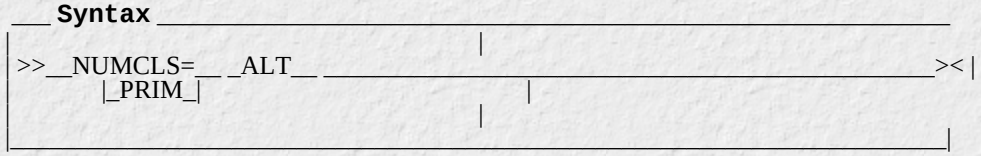
1. If COPY statements are used in a program and NUM=YES is in effect, the source program line numbers and the COPY member line numbers must be coordinated.

2. NUM=NO supports the COBOL 85 Standard.

IBM Library Server

1.2.5.36 NUMCLS

Syntax



Default
NUMCLS=PRIM

ALT
Used to specify the sign representations that are recognized as valid by the numeric class test for data items that are defined:

- As signed (with an "S" in the PICTURE clause)
- Using DISPLAY or COMPUTATIONAL-3 (packed-decimal)
- Without the SEPARATE phrase on any SIGN clause

Processing with ALT accepts hexadecimal A through F as valid.

PRIM
Processing with PRIM accepts only hexadecimal C, D, and F as valid.

Notes:

1. The numeric class test is affected by how both the NUMPROC and the NUMCLS options are specified. The NUMCLS option is effective only for NUMPROC=MIG or NUMPROC=NOPFD. NUMPROC=PFD specifies more strict rules for valid sign configuration.

IBM Library Server

1.2.5.37 NUMPROC

Syntax

Default
NUMPROC=NOPFD

MIG
Aids in migrating DOS/VS COBOL application programs to COBOL/VSE. Processing with MIG:

- Uses existing signs for comparison and arithmetic operations
- Generates preferred signs for the results of MOVE and arithmetic operations (These results meet the criteria for using NUMPROC=PFD.)
- Performs numeric rather than logical comparisons

NOPFD
Repairs signs on input. After repair is performed, the signs meet the criteria for NUMPROC=PFD.

PFD
Optimizes the generated code, especially when OPT=YES is specified. No explicit sign repair is performed. Note that NUMPROC=PFD has stringent criteria to produce correct results. To use NUMPROC=PFD:

- The sign position of unsigned numeric items must be X'F'.
- The sign position of signed numeric items must be either X'C' if positive or zero, or must be X'D' if negative.
- The sign position of separately signed numeric items must either be '+' if positive or zero, or '-' if negative.

Elementary MOVE and arithmetic statements in COBOL/VSE always generate results with these preferred signs, however group MOVEs and redefinitions may produce nonconforming results. The numeric class test may be used for verification. With NUMPROC=PFD, a numeric item will fail the numeric class test if the signs do not meet the preferred sign criteria.

Performance Consideration: Using NUMPROC=PFD will generate significantly more efficient code for numeric comparisons. For most references to COMP-3 and DISPLAY numeric data items, using NUMPROC=MIG and NUMPROC=NOPFD will generate extra code because of sign "fix-up" processing. This extra code may also inhibit some other types of optimization.

Before setting this option, please consult with your application programmers to determine the effect on the application program's output.

Notes:

1. NUMPROC=NOPFD or NUMPROC=MIG supports the COBOL 85 Standard.

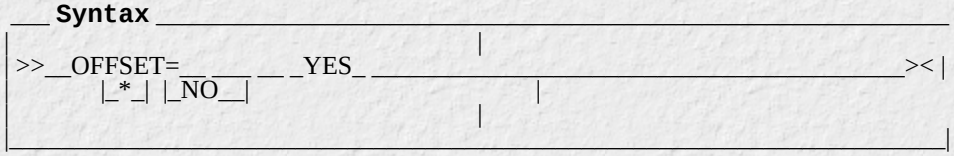
2. NUMPROC=NOPFD and NUMPROC=PFD are equivalent to the VS COBOL II Release 2 options PFDSGN=NO and PFDSGN=YES, respectively.

3. Both the NUMPROC and NUMCLS options affect the numeric class test. With NUMPROC=MIG or NUMPROC=NOPFD, the results of the numeric class test are controlled by how NUMCLS is set. When NUMPROC=PFD, a data item must meet the preferred sign criteria to be considered numeric.

IBM Library Server

1.2.5.38 OFFSET

Syntax



Default
OFFSET=NO

YES
Produces a condensed Procedure Division listing. The procedure portion of the listing will contain line numbers, verb references, and the location of the first instruction generated for each verb. In addition, the following are produced:

- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled with the NORENT compiler option

NO
Does not condense the listing and does not produce the items listed above.

Notes:

1. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code when attempting to assemble the customization macro. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflict resolution.

IBM Library Server

1.2.5.39 OPT

Syntax

Default
OPT=NO

STD
Causes the compiler to generate optimized object code.

FULL
Causes the compiler to generate optimized object code, and to delete unused WORKING-STORAGE data items as well as delete the VALUE clause code that initializes those items.

Performance Consideration: Using OPT=STD or OPT=FULL generally results in more efficient run-time code. This option requires more CPU time for compiles than OPT=NO.

NO
Does not cause the compiler to generate optimized object code.

Notes:

1. You can only specify TEST=(NONE,...) for the hook-location subparameter with OPT=STD or OPT=FULL. Any other combination results in a nonzero return code and an error during installation.

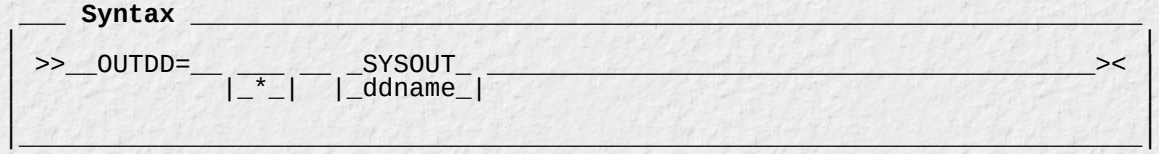
2. Do not use OPT=FULL if your programs depend on 'using' data items that are not referenced in the procedure division, such as adjacent tables, addresses passed to assembler programs, or 'eyecatchers' to identify the begin and end of the WORKING-STORAGE section. See *COBOL/VSE Programming Guide*.

3. Optimization is turned off if an S-level error or higher is flagged by the compiler.

IBM Library Server

1.2.5.40 OUTDD

Syntax



Default
OUTDD=SYSOUT

filename
Specifies the file name of the file to be used for run-time DISPLAY output.

Notes:

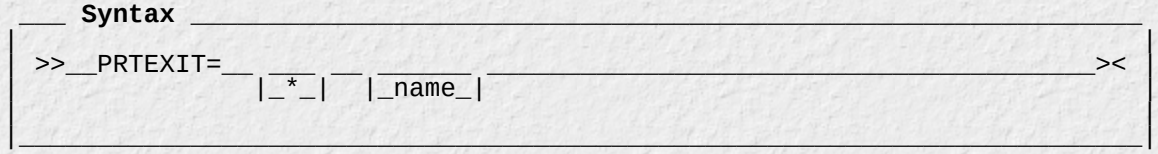
1. See the *LE/VSE Programming Reference* description of the MSGFILE run-time option to see how OUTDD interacts with MSGFILE.

2. Change the default for this option if, at run time, you expect to have a conflict with another product that requires SYSOUT as a file name. Both SYSOUT and SYSLST direct the output to SYSLST.

IBM Library Server

1.2.5.41 PRTEXIT

Syntax



Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it instead of writing to SYSLST. When the option is supplied, SYSLST is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.42 RENT

Syntax

```
>>__RENT=__ ___ __ _YES_ ____________________________________________><
          |_*_|  |_NO__|
```

Default
RENT=NO

YES
Indicates that the object code produced for a COBOL program is to be reentrant.

Using RENT=YES enables the program to be placed in the SVA for running above the 16-megabyte line. However, this option causes the compiler to generate additional code to ensure that the application program is reentrant.

NO
Indicates that the object code produced for a COBOL program is not to be reentrant.

Notes:

1. Programs must be compiled with RENT=YES or RMODE=ANY if they will be run with extended addressing in virtual storage addresses above 16 megabytes.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RENT/NORENT and RMODE compiler options. Valid combinations include:

Table 16. Effect of RENT and RMODE on Residency Mode

RENT/NORENT Setting	RMODE Setting	Residency Mode Assigned
RENT	AUTO	RMODE(ANY)
RENT	ANY	RMODE(ANY)
RENT	24	RMODE(24)
NORENT	AUTO	RMODE(24)
NORENT	ANY	RMODE(ANY)

	NORENT	24	RMODE(24)

IBM Library Server

1.2.5.43 RMODE

Syntax

Default
RMODE=AUTO

AUTO
A program compiled with the RMODE=AUTO option will have residency mode 24 if NORENT is specified, and residency mode ANY if RENT is specified.

24
A program compiled with the RMODE=24 option will have residency mode 24 whether NORENT or RENT is specified.

ANY
A program compiled with the RMODE=ANY option will have residency mode ANY whether NORENT or RENT is specified.

Notes:

1. COBOL/VSE NORENT programs that are required to pass data to programs running in AMODE(24) must either be compiled with the RMODE(24) option, or link-edited with RMODE(24). The data areas for NORENT programs will be above the line or below the line depending on the RMODE of the program, even if DATA(24) has been specified. DATA(24) applies to programs compiled with the RENT option only.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RMODE and RENT/NORENT compiler options. Valid combinations include:

Table 17. Effect of RMODE and RENT/NORENT on Residency Mode

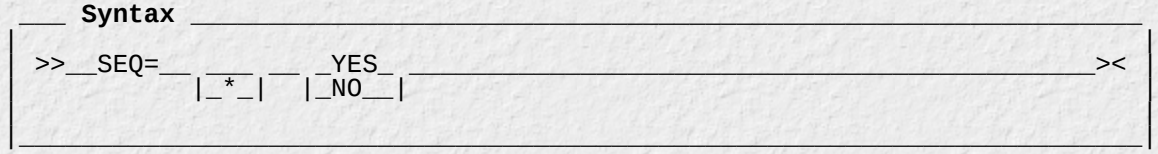
RMODE Setting	RENT/NORENT Setting	Residency Mode Assigned
AUTO	RENT	RMODE(ANY)
AUTO	NORENT	RMODE(24)
ANY	RENT	RMODE(ANY)

ANY	NORENT	RMODE(ANY)
24	RENT	RMODE(24)
24	NORENT	RMODE(24)

IBM Library Server

1.2.5.44 SEQ

Syntax



Default
SEQ=YES

YES
Instructs the compiler to check that the source statements are in ascending alphanumeric order by line number.

NO
Instructs the compiler not to perform sequence checking.

Notes:

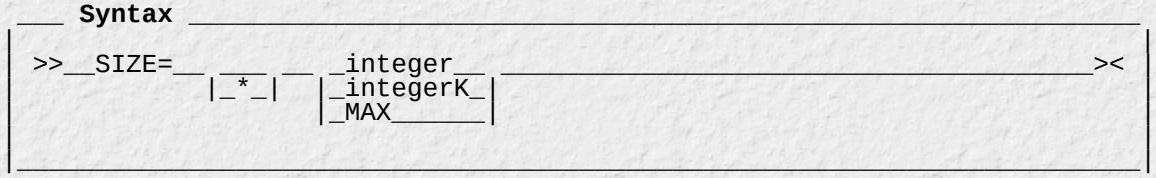
1. If both SEQ and NUM are in effect at compile time, the sequence is checked according to numeric, rather than alphanumeric, collating sequence.

2. SEQ=NO supports the COBOL 85 Standard.

IBM Library Server

1.2.5.45 SIZE

Syntax



Default
SIZE=MAX

integer
Specifies the amount, in bytes, of virtual storage available to the compiler.

The minimum acceptable value is 778240.

integerK
Specifies the amount of virtual storage available to the compiler in 1024-byte (K) increments.

The minimum acceptable value is 760K.

MAX
The compiler will request all available space in the partition GETVIS for use during the compilation. For Extended Architecture, the compiler obtains the largest contiguous block of free storage above the 16-megabyte line.

Notes:

1. Using SIZE=MAX simplifies compiler invocation by eliminating the need to determine a specific value for the SIZE option.

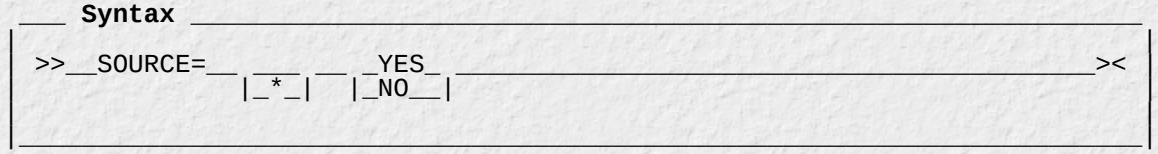
2. Do not use SIZE=MAX if, when you invoke the compiler, you require it to leave a specific amount of unused storage available in the partition.

3. SIZE=MAX in Extended Architecture allows the compiler to obtain all available above-the-line storage in the partition and below-the-line storage for work file buffers and compiler modules that must be below the 16-megabyte line. This occurs unless tuning is done for the Extended Architecture environment. Therefore, you may not want to fix this option as SIZE=MAX at installation.

IBM Library Server

1.2.5.46 SOURCE

Syntax



Default
SOURCE=YES

YES
Indicates that you want a listing of the source statements in the compiler-generated output. This listing also includes any statements embedded by COPY.

NO
Source statements will not appear in the output.

Notes:

1. Unless the installation default value of the SOURCE option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LIST option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE SOURCE option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

2. The SOURCE compiler option must be in effect at compile time if you want embedded messages in the source listing.

IBM Library Server

1.2.5.47 SPACE

Syntax

Default
SPACE=1

1
Indicates that you want single spacing for the source statement listing.

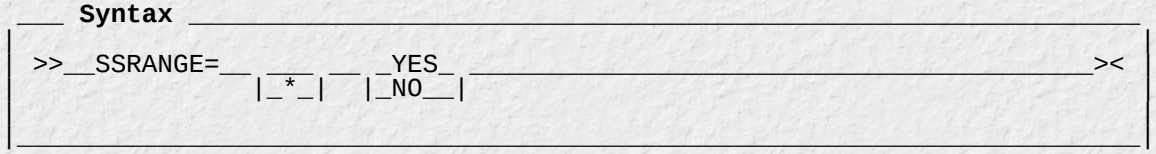
2
Indicates that you want double spacing for the source statement listing.

3
Indicates that you want triple spacing for the source statement listing.

IBM Library Server

1.2.5.48 SSRANGE

Syntax



Default
SSRANGE=NO

YES
At compile time, generates code that checks subscripts, reference modifications, variable-length group ranges, and indexes in the source program at run time to ensure that they do not refer to storage outside the area assigned. It also verifies that a table with ALL subscripting, specified as a function argument, will contain at least one occurrence in the table.

The generated code also checks that a variable-length item does not exceed its defined maximum length as a result of incorrect setting of the OCCURS DEPENDING ON object.

Performance Consideration: If SSRANGE=YES at compile time, object code size will be increased and there will be an increase in run-time overhead to accomplish the range checking.

NO
No code is generated to perform subscript or index checking at run time.

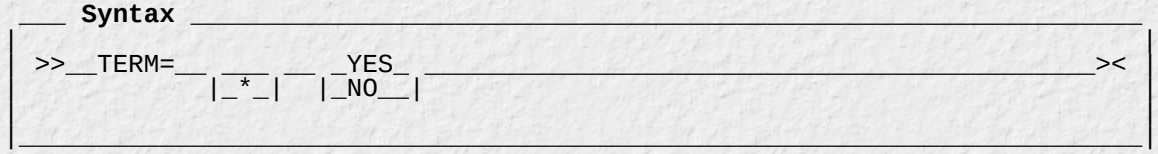
Notes:

1. If the SSRANGE option is in effect at compile time, the range-checking code is generated. Range-checking can be inhibited at run time by specifying the LE/VSE run-time option CHECK(OFF). However, the range-checking code still requires overhead and is dormant within the object code. The range-checking code can then be used optionally to aid in resolving any unexpected errors without recompilation.

IBM Library Server

1.2.5.49 TERM

Syntax



Default
TERM=NO

YES
Specifies that the progress and diagnostic messages are to be sent to the SYSLOG file.

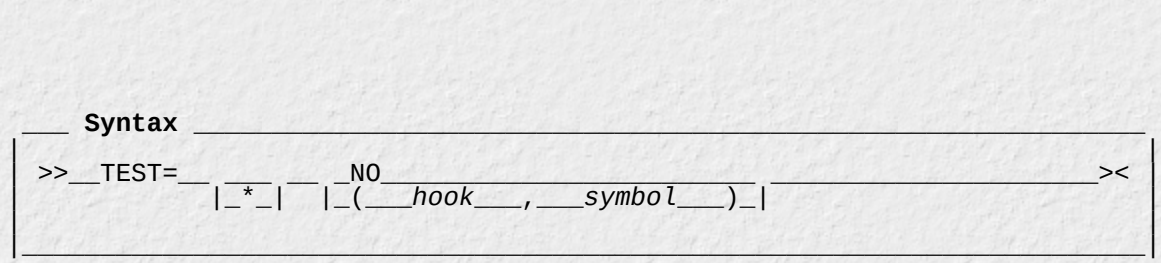
NO
Specifies that no messages are to be sent to SYSLOG.

| **Note:** Unless the installation default value of the TERM option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the TERM option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE TERM option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.50 TEST

Syntax



Default
TEST=NO

Other than NO
Produces object code containing symbol table information that is used by LE/VSE to produce a formatted dump, and hooks for Debug Tool/VSE.

You must specify options for both the hook-location (*hook*) and the symbol table (*symbol*) values.

hook values:

ALL Activates the generation of all compiled-hooks. Hooks will be generated at all statements, all path points, and at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

NONE Suppresses the generation of all compiled-hooks. TEST(NONE) is compatible with the OPT compiler option.

STMT Hooks will be compiled at every statement and label, as well as at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

PATH Hooks will be compiled at all path points, including program entry and exit points.

BLOCK Hooks will be compiled at all program entry and exit points.

symbol values:

SYM Activates inclusion of symbolic dictionary information in your object program.

NOSYM Deactivates inclusion of symbolic dictionary information in your object program.

Performance Consideration: Because TEST=(a hook-location suboption other than NONE) generates additional code, it can cause significant performance degradation at run time when used in a production environment.

NO
Produces object code that does not contain the symbol table information that is used by LE/VSE to produce a formatted dump, and the hooks for Debug Tool/VSE.

Notes:

1. If you specify TEST= (other than NONE for the hook-location suboption), the following option is put into effect at compilation time:

   OPT=NO

2. Only TEST(NONE) is compatible with the OPT compiler option. See "Conflicting Compiler Options" in topic 1.2.5.3 for more conflict resolution information.

3. Programmers might notice an increase in run-time for programs compiled with any hook-location suboption other than NONE in effect.

4. A date processing statement is any statement that references a date field, or any EVALUATE or SEARCH statement WHEN phrase that references a date field.

5. To get the full capability of Debug Tool/VSE, compile your program with TEST(ALL,SYM).

6. For production programs compile your programs with TEST(NONE,NOSYM) if you do not want to increase the size of your production modules, but still get minimum debugging capability.

IBM Library Server

1.2.5.51 TRUNC

Syntax

Default
TRUNC=STD

STD
Controls the way arithmetic fields are shortened during MOVE and arithmetic operations. The TRUNC option applies only to binary (COMP) receiving fields in MOVE statements and in arithmetic expressions. When TRUNC=STD is in effect, the final intermediate result of an arithmetic expression, or of the sending field in the MOVE statement, is shortened to the number of digits in the PICTURE clause of the binary receiving field.

OPT
The compiler assumes that the data conforms to PICTURE and USAGE specifications. The compiler manipulates the result based on the size of the field in storage (halfword or fullword).

TRUNC=OPT should only be specified when data being moved into binary areas will not have a value with larger precision than that defined by the binary item PICTURE clause. Otherwise, truncation of high-order digits may occur. The truncation results are dependent on the particular code sequence generated and may not necessarily be the same in DOS/VS COBOL and VS COBOL II.

BIN
Specifies that:

1. Output binary fields will be shortened only at the S/370* half/full/double word boundaries, rather than at COBOL base 10 picture limits.

2. Input binary fields will be treated as S/370 half/full/double words, and no assumption will be made that the values are limited to those implied by the base 10 PICTURE clause.

3. DISPLAY will convert and output the full content of binary fields with no truncation to the PICTURE description.

Performance Consideration: Using TRUNC=OPT does not generate extra code and generally improves performance. However, both TRUNC=BIN and TRUNC=STD generate extra code whenever a BINARY data item is changed. TRUNC=BIN is usually the slowest of these options.

Notes:

1. TRUNC=STD supports the COBOL 85 Standard.

2. TRUNC=STD and TRUNC=OPT are equivalent to TRUNC=YES and TRUNC=NO (respectively) in VS COBOL II Release 2.

3. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether the COBOL/VSE source programs assume a particular setting for correct running.

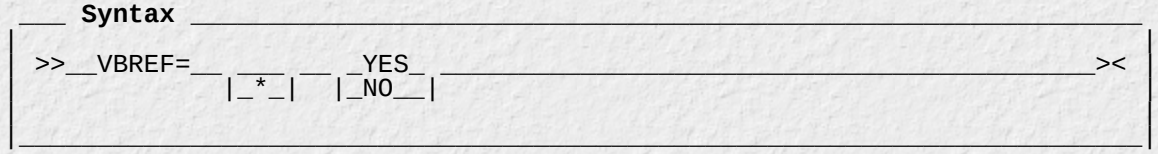
4. TRUNC=BIN is the recommended option when interfacing with other products that have S/370-format binary data (such as CICS, SQL/DS*, FORTRAN, and PL/I). This is especially true if there is a possibility of having more than 9 digits in a fullword or more than 4 digits in a halfword.

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.52 VBREF

Syntax



YES
Produces a cross-reference of all verb types in a source program to the line numbers at which they were found. VBREF=YES also produces a summary of how many times each verb was used in the program.

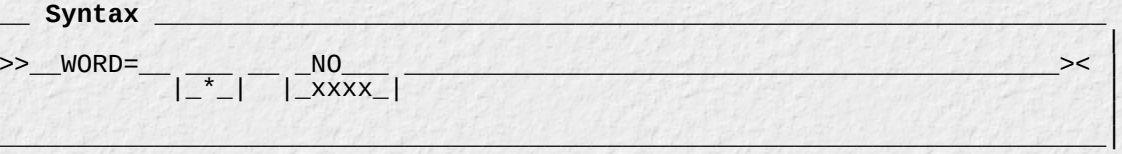
NO
Does not produce a cross-reference or verb summary listing.

Default
VBREF=NO

IBM Library Server

1.2.5.53 WORD

Syntax



Default
WORD=*NO

NO
Indicates that no alternative reserved word table is to be used as the default.

xxxx
Specifies an alternative default reserved word table to be used during compilation. **xxxx** represents the ending characters (may be 1 to 4 characters in length) of the name of the reserved word table to be used. The first 4 characters are IGYC. The last 4 characters may not be any one of the character strings listed below, nor may any of them contain the dollar sign character ($).

ASM1	LIBO	LVL8	RDSC
ASM2	LIBR	OPTM	RWT
DIAG	LSTR	OSCN	SAW
DMAP	LVL0	PGEN	SCAN
DOPT	LVL1	RCTL	SIMD
FGEN	LVL2	RDPR	XREF
INIT	LVL3		

Notes:

1. The default for the WORD option is specified with an asterisk. When the option is installed with the default (WORD=*NO), the application programmer cannot replace the option at compile time to specify an alternative reserved word table.

2. WORD=NO supports the COBOL 85 Standard.

3. Specification of WORD affects the interpretation of input reserved words. System names (such as UPSI and SYSPCH) and the intrinsic function names should not be used as aliases for reserved words. If a function name is specified as an alias via the reserved word table ABBR control-statement, that function name will be recognized and diagnosed by the compiler as a reserved word and the intrinsic function will not be performed.

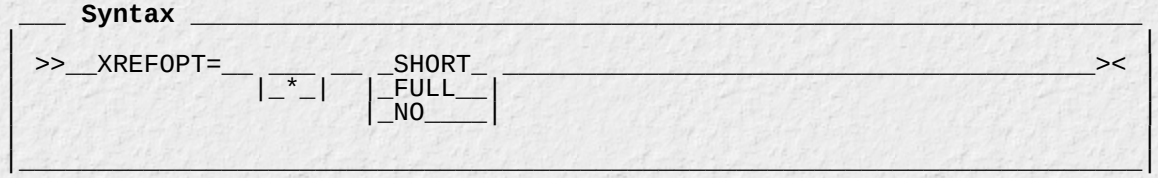
4. Changing the default value of the WORD option could cause compiler option conflicts. See "Conflicting Compiler Options" in topic 1.2.5.3 for information on conflict resolution.

5. A CICS-specific reserved word table is provided as an alternate reserved word table. For a description, see "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2.

IBM Library Server

1.2.5.54 XREFOPT

Syntax



Default
XREFOPT=NO

SHORT
Produces only the explicitly referenced variables in the cross-reference listing.

FULL
Produces both a sorted and embedded cross-reference listing. If SOURCE=YES is also specified, the listing line numbers cross-reference recurrences of particular data-names.

NO
Suppresses the cross-reference listing.

Notes:

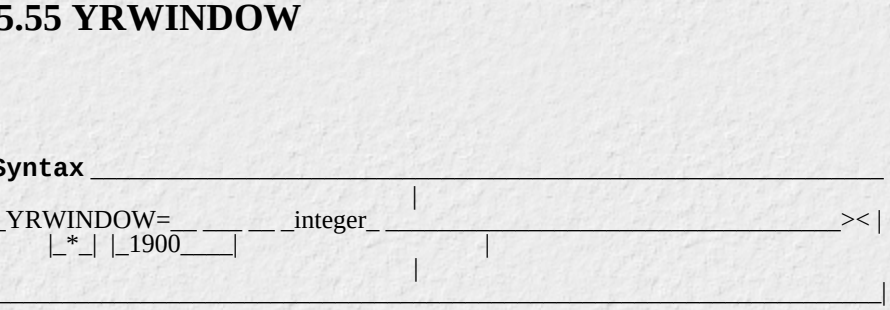
1. The XREFOPT option sets the default value for the compiler option XREF.

| **Note:** Unless the installation default value of the XREFOPT option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the XREF or SXREF option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE XREF option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.55 YRWINDOW

Syntax



Default
YRWINDOW=1900

integer
Specifies the first year of the 100-year window used by COBOL windowed date fields, and may be one of the following:

- An unsigned decimal integer from 1900 through 1999.
- A negative decimal integer from -1 through -99, representing an offset from the current year at run time. The current year is determined for each compilation unit when it is first initialized, or re-initialized after execution of a CANCEL statement that refers to the compilation unit.

Notes:

1. YRWINDOW has no effect unless DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. At run-time, two conditions must be true:

   a. The 100-year window must have its beginning year in the 1900s.

   b. The current year must lie within the 100-year window for the compilation unit.

3. All windowed dates have a year relative to the base year. For example, if the base year were specified as 1965, all windowed year values would be interpreted as years within the 100-year window of 1965 to 2064, inclusive. So, a windowed year value of 67 would represent the year 1967, whereas a windowed year value of 05 would represent the year 2005.

   If the base year were specified as -30, then the window would depend on when the application were run. Running it during 1998 would give a 100-year window of 1968 through 2067. So, a windowed year value of 68 would represent the year 1968, whereas a windowed year value of 67

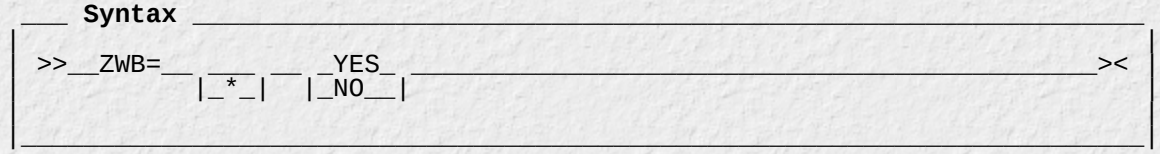
would represent the year 2067.

4. The YRWINDOW installation option is equivalent to the YEARWINDOW compile-time option.

IBM Library Server

1.2.5.56 ZWB

Syntax



Default
ZWB=YES

YES
Instructs the compiler to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

NO
Instructs the compiler not to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

Notes:

1. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether your COBOL/VSE source programs assume a particular setting to run correctly.

2. ZWB=YES supports the COBOL 85 Standard.

3. Application programmers use ZWB=NO to test input numeric fields for SPACES.

IBM Library Server

| 1.2.4.3.3 SAA Reserved Word Table (IGY8SAAW)

```
| This table contains non-SAA reserved words.  Use of these words will be
| flagged with a warning message if the FLAGSAA compiler option is in
| effect.  This table is supplied to provide compatibility with COBOL
| compilations that comply with SAA standards.
```

IBM Library Server

1.2.5 COBOL/VSE Compiler Options

This section describes those compiler options whose default values you can change. The notes that accompany some of the descriptions provide additional information about these options, such as how they interact with other options during compilation. This information may assist you in making decisions regarding the default values appropriate for your installation. For more information on using the compiler options, see the *COBOL/VSE Programming Guide*.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

1.2.5 COBOL/VSE Compiler Options

This section describes those compiler options whose default values you can change. The notes that accompany some of the descriptions provide additional information about these options, such as how they interact with other options during compilation. This information may assist you in making decisions regarding the default values appropriate for your installation. For more information on using the compiler options, see the *COBOL/VSE Programming Guide*.

Important

Make sure that COBOL/VSE serves the needs of the application programmers at your site. Please confer with them while you plan the customization of COBOL/VSE. Doing so will ensure that the modifications you make at install time best support the application programs being developed at your site.

Subtopics:

IBM Library Server

1.2.5.1 Specifying COBOL Compiler Options

When you specify compiler options in the IGYCOPT macro, both the option name and its value must be specified in uppercase. If the option name is not specified in uppercase, both the option name and its value will be ignored and the default value will be used. No error message will be issued. If only the option value is not in uppercase, an error message will be issued indicating that an invalid option value has been specified.

IBM Library Server

1.2.5.1 Specifying COBOL Compiler Options

When you specify compiler options in the IGYCOPT macro, both the option name and its value must be specified in uppercase. If the option name is not specified in uppercase, both the option name and its value will be ignored and the default value will be used. No error message will be issued. If only the option value is not in uppercase, an error message will be issued indicating that an invalid option value has been specified.

IBM Library Server

1.2.5.2 Options in Support of the COBOL 85 Standard

The following option values are required to conform to the COBOL 85 Standard:

ADV=YES	FLAGMIG=NO	NUMPROC=NOPFD or MIG
CMPR2=NO	FLAGSAA=NO	SEQ=NO
DATEPROC=NO	INTDATE=ANSI	TRUNC=STD
DBCS=NO	LIB=YES	WORD=NO
DYNAM=YES	LITCHAR=QUOTE	ZWB=YES
FASTSRT=NO	NUM=NO	

Note: The term "COBOL 85 Standard" is used in this book to refer to the combination of the following standards:

1. ISO 1989:1985, Programming Languages - COBOL.

   ISO 1989/Amendment 1, Programming Languages - COBOL - Amendment 1: Intrinsic Function Module.

2. X3.23-1985, American National Standard for Information Systems - Programming Language - COBOL.

   X3.23a-1989, American National Standard for Information Systems - Programming Language - Intrinsic Function Module for COBOL.

IBM Library Server

1.2.5.2 Options in Support of the COBOL 85 Standard

The following option values are required to conform to the COBOL 85 Standard:

ADV=YES	FLAGMIG=NO	NUMPROC=NOPFD or MIG
CMPR2=NO	FLAGSAA=NO	SEQ=NO
DATEPROC=NO	INTDATE=ANSI	TRUNC=STD
DBCS=NO	LIB=YES	WORD=NO
DYNAM=YES	LITCHAR=QUOTE	ZWB=YES
FASTSRT=NO	NUM=NO	

Note: The term "COBOL 85 Standard" is used in this book to refer to the combination of the following standards:

1. ISO 1989:1985, Programming Languages - COBOL.

   ISO 1989/Amendment 1, Programming Languages - COBOL - Amendment 1: Intrinsic Function Module.

2. X3.23-1985, American National Standard for Information Systems - Programming Language - COBOL.

   X3.23a-1989, American National Standard for Information Systems - Programming Language - Intrinsic Function Module for COBOL.

IBM Library Server

1.2.5.3 Conflicting Compiler Options

Specifying certain compiler option values can create conflicts with other compiler options. Table 14 will help you resolve possible conflicts between compiler options. If you do specify conflicting compiler option values, you will receive a nonzero return code when attempting to assemble the IGYCOPT macro.

Table 14. Conflicting Compiler Options

Compiler Option	Conflicts with:
CMPR2=NO	FLAGMIG=YES
CMPR2=YES	DATEPROC=YES DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
COMPILE=NOC	LIST=YES OFFSET=YES OPT=STD or OPT=FULL TEST=(other than NO) VBREF=YES
DATEPROC=YES	CMPR2=YES FLAGSAA=YES FLAGSTD=(other than NO)
DBCS=YES	CMPR2=YES FLAGMIG=YES FLAGSTD=(other than NO)
FLAGMIG=YES	CMPR2=NO DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
FLAGSAA=YES	ADV=NO CMPR2=YES DATEPROC=(other than NO) DYNAM=NO FLAGMIG=YES FLAGSTD=(other than NO) LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
FLAGSTD=(other than NO)	ADV=NO CMPR2=YES DATEPROC=(other than NO) DBCS=YES DYNAM=NO FLAGMIG=YES FLAGSAA=YES LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD

	SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
LIST=YES	OFFSET=YES
OFFSET=YES	LIST=YES
OPT=STD or OPT=FULL	TEST=(other than NONE for hook location)
TEST=(other than NONE for hook-location suboption)	OPT=STD or OPT=FULL
WORD=xxxx	FLAGSTD=(other than NO)

IBM Library Server

1.2.5.3 Conflicting Compiler Options

Specifying certain compiler option values can create conflicts with other compiler options. Table 14 will help you resolve possible conflicts between compiler options. If you do specify conflicting compiler option values, you will receive a nonzero return code when attempting to assemble the IGYCOPT macro.

Table 14. Conflicting Compiler Options

Compiler Option	Conflicts with:
CMPR2=NO	FLAGMIG=YES
CMPR2=YES	DATEPROC=YES DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
COMPILE=NOC	LIST=YES OFFSET=YES OPT=STD or OPT=FULL TEST=(other than NO) VBREF=YES
DATEPROC=YES	CMPR2=YES FLAGSAA=YES FLAGSTD=(other than NO)
DBCS=YES	CMPR2=YES FLAGMIG=YES FLAGSTD=(other than NO)
FLAGMIG=YES	CMPR2=NO DBCS=YES FLAGSAA=YES FLAGSTD=(other than NO)
FLAGSAA=YES	ADV=NO CMPR2=YES DATEPROC=(other than NO) DYNAM=NO FLAGMIG=YES FLAGSTD=(other than NO) LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
FLAGSTD=(other than NO)	ADV=NO CMPR2=YES DATEPROC=(other than NO) DBCS=YES DYNAM=NO FLAGMIG=YES FLAGSAA=YES LIB=NO LITCHAR=APOST NUM=YES NUMPROC=PFD

	SEQ=YES TRUNC=OPT or BIN WORD=(other than NO or RWT) ZWB=NO
LIST=YES	OFFSET=YES
OFFSET=YES	LIST=YES
OPT=STD or OPT=FULL	TEST=(other than NONE for hook location)
TEST=(other than NONE for hook-location suboption)	OPT=STD or OPT=FULL
WORD=xxxx	FLAGSTD=(other than NO)

IBM Library Server

1.2.5.4 Compiler Options Syntax and Descriptions

The following syntax diagrams describe each modifiable compiler option. The text below each diagram describes the effect of selecting a specific parameter.

Notice the absence of the DUMP option. Unless changed at compile time, DUMP is always set to NODUMP. This option is not for general use; it is only used at the request of an IBM representative.

Note: The DECK and OBJECT compiler options may only be specified at compile time, and are therefore not described in the following sections. For more information, see *COBOL/VSE Programming Guide*.

1.2.5.4 Compiler Options Syntax and Descriptions

The following syntax diagrams describe each modifiable compiler option. The text below each diagram describes the effect of selecting a specific parameter.

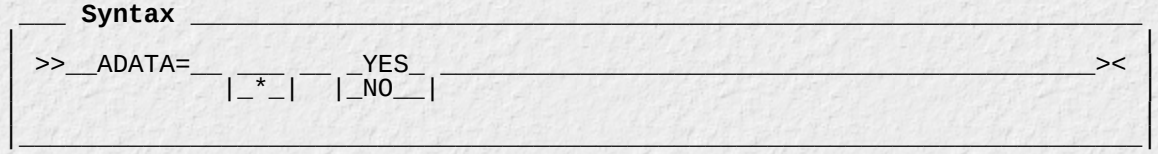
Notice the absence of the DUMP option. Unless changed at compile time, DUMP is always set to NODUMP. This option is not for general use; it is only used at the request of an IBM representative.

Note: The DECK and OBJECT compiler options may only be specified at compile time, and are therefore not described in the following sections. For more information, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.5 ADATA

Syntax



Default
ADATA=NO

YES
Specifies that COBOL/VSE associated data is to be collected and placed in the Associated Data file defined by the SYSADAT DLBL statement.

NO
Specifies that no associated data is to be collected.

Notes:

1. The ADATA option can only be specified at invocation via the option list on the PARM field of JCL.

2. Selection of the Japanese language option might result in DBCS characters written records in the Associated Data file.

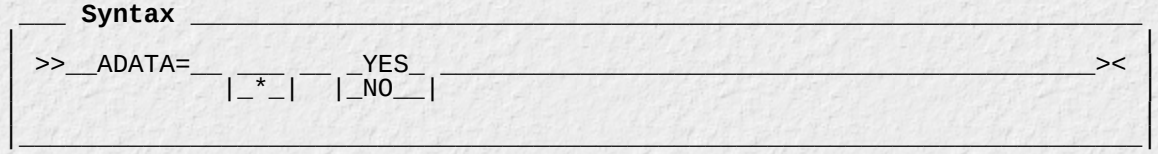
3. Specification of NOCOMPILE(W|E|S) might stop compilation prematurely, resulting in a loss of specific Associated Data Records.

4. Specification of INEXIT will prohibit identification of the compilation source file for the Associated Data file.

IBM Library Server

1.2.5.5 ADATA

Syntax



Default
ADATA=NO

YES
Specifies that COBOL/VSE associated data is to be collected and placed in the Associated Data file defined by the SYSADAT DLBL statement.

NO
Specifies that no associated data is to be collected.

Notes:

1. The ADATA option can only be specified at invocation via the option list on the PARM field of JCL.

2. Selection of the Japanese language option might result in DBCS characters written records in the Associated Data file.

3. Specification of NOCOMPILE(W|E|S) might stop compilation prematurely, resulting in a loss of specific Associated Data Records.

4. Specification of INEXIT will prohibit identification of the compilation source file for the Associated Data file.

IBM Library Server

1.2.5.6 ADEXIT

Syntax

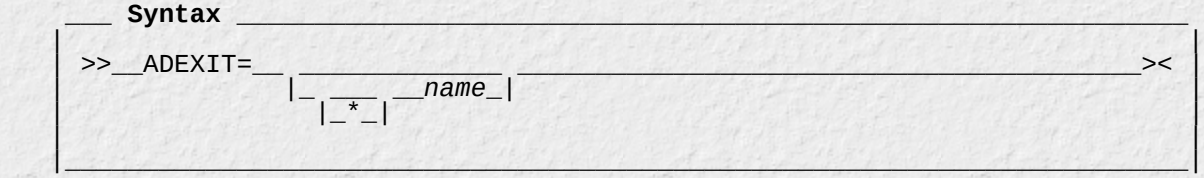
Default
No exit specified.

name
Identifies the name of a module, to be used with the EXIT option, that will be loaded and called by the compiler to monitor the associated data records written by the compiler to the Associated Data file. For more specific information, see the *COBOL/VSE Programming Guide*

Note: This is a read-only exit.

IBM Library Server

1.2.5.6 ADEXIT



Default
No exit specified.

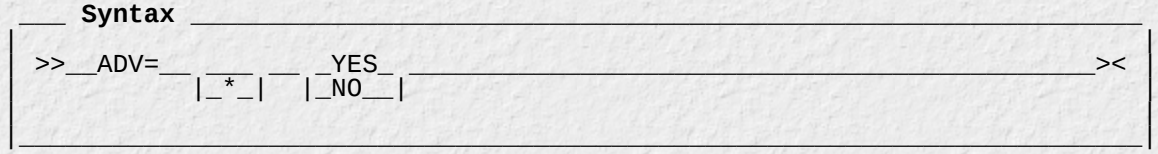
name
Identifies the name of a module, to be used with the EXIT option, that will be loaded and called by the compiler to monitor the associated data records written by the compiler to the Associated Data file. For more specific information, see the *COBOL/VSE Programming Guide*

Note: This is a read-only exit.

IBM Library Server

1.2.5.7 ADV

Syntax



Default
ADV=YES

YES
Instructs the compiler to add one byte to the record length for the printer control character. This option may be useful to programmers who use WRITE...ADVANCING in their source files. The first character of the record does **not** have to be explicitly reserved by the programmer.

NO
Instructs the compiler not to adjust the record length for WRITE...ADVANCING. The compiler uses the first character of the specified record area to place the printer control character. The application programmer must ensure that the record description allows for this additional byte.

Notes:

1. ADV=YES conforms to the COBOL 85 Standard. With ADV=YES, the record length on the physical device is one byte larger than the record description length in the source program.

2. If the record length for the output file is not defined in the source code, COBOL ensures that it is appropriately set.

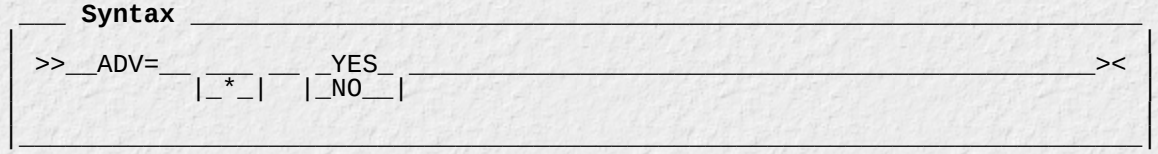
3. If ADV=YES is specified, and the record length for the output file has been defined in the source code, the programmer ***must*** specify the record description length as one byte larger than the source program record description. The programmer ***must*** also specify the block size in correct multiples of the larger record size.

4. If the LINAGE clause is specified in a file description (FD), the compiler treats that file as if ADV=YES had been specified.

IBM Library Server

1.2.5.7 ADV

Syntax



Default
ADV=YES

YES
Instructs the compiler to add one byte to the record length for the printer control character. This option may be useful to programmers who use WRITE...ADVANCING in their source files. The first character of the record does **not** have to be explicitly reserved by the programmer.

NO
Instructs the compiler not to adjust the record length for WRITE...ADVANCING. The compiler uses the first character of the specified record area to place the printer control character. The application programmer must ensure that the record description allows for this additional byte.

Notes:

1. ADV=YES conforms to the COBOL 85 Standard. With ADV=YES, the record length on the physical device is one byte larger than the record description length in the source program.

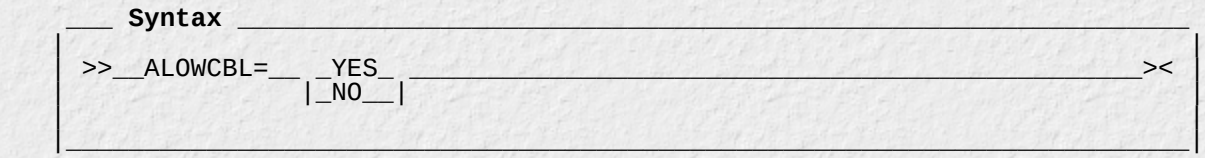
2. If the record length for the output file is not defined in the source code, COBOL ensures that it is appropriately set.

3. If ADV=YES is specified, and the record length for the output file has been defined in the source code, the programmer ***must*** specify the record description length as one byte larger than the source program record description. The programmer ***must*** also specify the block size in correct multiples of the larger record size.

4. If the LINAGE clause is specified in a file description (FD), the compiler treats that file as if ADV=YES had been specified.

IBM Library Server

1.2.5.8 ALOWCBL



Default
ALOWCBL=YES

YES
Instructs the compiler to allow the use of the PROCESS (or CBL) statements in COBOL programs.

NO
Instructs the compiler to diagnose the use of PROCESS (or CBL) statements in a program as an error.

Notes:

1. ALOWCBL cannot be replaced at compile time because it cannot be included in the PROCESS (or CBL) statement.

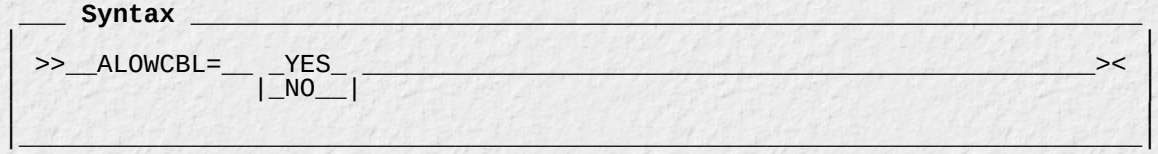
2. The PROCESS (or CBL) statement is used to specify compiler option parameters within source programs. If your installation requirements do not allow compiler options to be specified in a source program, specify ALOWCBL=NO.

3. You must specify ALOWCBL=YES for application programs that run under CICS.

IBM Library Server

1.2.5.8 ALOWCBL

Syntax



Default
ALOWCBL=YES

YES
Instructs the compiler to allow the use of the PROCESS (or CBL) statements in COBOL programs.

NO
Instructs the compiler to diagnose the use of PROCESS (or CBL) statements in a program as an error.

Notes:

1. ALOWCBL cannot be replaced at compile time because it cannot be included in the PROCESS (or CBL) statement.

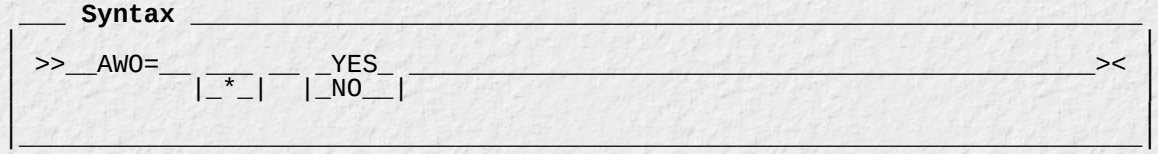
2. The PROCESS (or CBL) statement is used to specify compiler option parameters within source programs. If your installation requirements do not allow compiler options to be specified in a source program, specify ALOWCBL=NO.

3. You must specify ALOWCBL=YES for application programs that run under CICS.

IBM Library Server

1.2.5.9 AWO

Syntax



Default
AWO=NO

YES
Activates the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

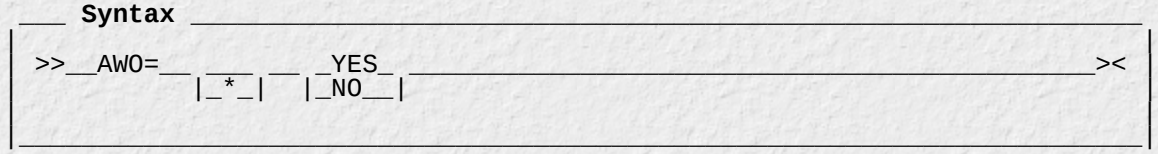
Performance Consideration: Using AWO=YES generally results in fewer calls to Data Management Services for run-time files when handling I/O.

NO
Does not activate the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

IBM Library Server

1.2.5.9 AWO

Syntax



Default
AWO=NO

YES
Activates the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

Performance Consideration: Using AWO=YES generally results in fewer calls to Data Management Services for run-time files when handling I/O.

NO
Does not activate the APPLY-WRITE-ONLY clause for any file within the program that is physical sequential with Variable Block format unless the APPLY-WRITE-ONLY clause is specified in the program.

IBM Library Server

1.2.5.10 BUF

Syntax

Default
BUF=4K

integer
Specifies the amount of dynamic storage, in bytes, to be allocated to each compiler work file buffer. The minimum value is 256 bytes.

Performance Consideration: Using BUF=integer generally improves compile-time performance by reducing the number of I/Os to the work files.

integerK
Specifies the amount of dynamic storage to be allocated to buffers in increments of 1K (1024) bytes.

Notes:

1. BUF and SIZE values are used by the compiler in determining how much storage is to be used during compilation. The amount allocated to the buffers is included in the amount of main storage available to the compiler for the SIZE option.

2. BUF cannot exceed the track capacity for the device used, nor can it exceed the maximum allowed by data management services.

IBM Library Server

1.2.5.10 BUF

Syntax

Default
BUF=4K

integer
Specifies the amount of dynamic storage, in bytes, to be allocated to each compiler work file buffer. The minimum value is 256 bytes.

Performance Consideration: Using BUF=integer generally improves compile-time performance by reducing the number of I/Os to the work files.

integerK
Specifies the amount of dynamic storage to be allocated to buffers in increments of 1K (1024) bytes.

Notes:

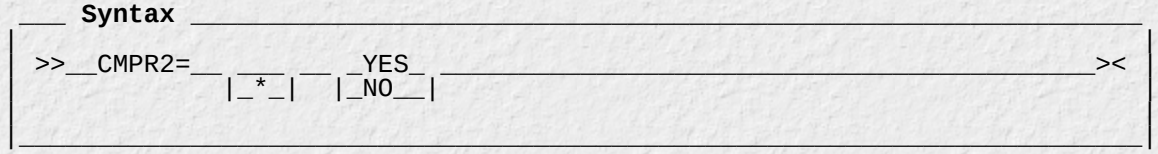
1. BUF and SIZE values are used by the compiler in determining how much storage is to be used during compilation. The amount allocated to the buffers is included in the amount of main storage available to the compiler for the SIZE option.

2. BUF cannot exceed the track capacity for the device used, nor can it exceed the maximum allowed by data management services.

IBM Library Server

1.2.5.11 CMPR2

Syntax



Default
CMPR2=NO

YES
Causes the compiler to generate code that is run-time compatible with valid VS COBOL II Release 2 programs. See the *COBOL/VSE Migration Guide* for details of language elements that are sensitive to CMPR2.

NO
Causes the compiler to generate code that may not be run-time compatible with valid VS COBOL II Release 2 programs for some COBOL 85 standard language constructs.

Notes:

1. Existing applications which rely on such matters as the format of the listing file, mapping of message numbers to messages, and output of the TERM option may need to be converted to COBOL/VSE.

2. Functions are **NOT** allowed under CMPR2. The word 'function' may be a dataname under CMPR2, but under NOCMPR2 it is considered a reserved word.

3. New language extensions for defining and manipulating procedural pointers are only supported with NOCMPR2.

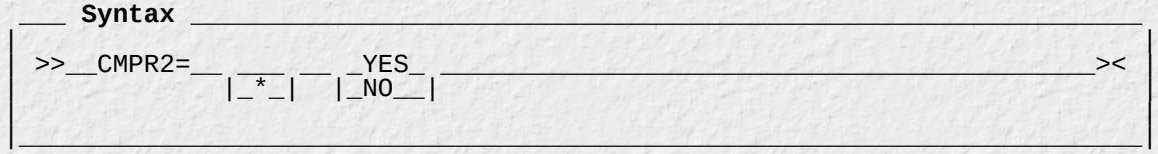
4. CMPR2=YES cannot be specified with certain other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflicting compiler options.

5. CMPR2=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.11 CMPR2

Syntax



Default
CMPR2=NO

YES
Causes the compiler to generate code that is run-time compatible with valid VS COBOL II Release 2 programs. See the *COBOL/VSE Migration Guide* for details of language elements that are sensitive to CMPR2.

NO
Causes the compiler to generate code that may not be run-time compatible with valid VS COBOL II Release 2 programs for some COBOL 85 standard language constructs.

Notes:

1. Existing applications which rely on such matters as the format of the listing file, mapping of message numbers to messages, and output of the TERM option may need to be converted to COBOL/VSE.

2. Functions are **NOT** allowed under CMPR2. The word 'function' may be a dataname under CMPR2, but under NOCMPR2 it is considered a reserved word.

3. New language extensions for defining and manipulating procedural pointers are only supported with NOCMPR2.

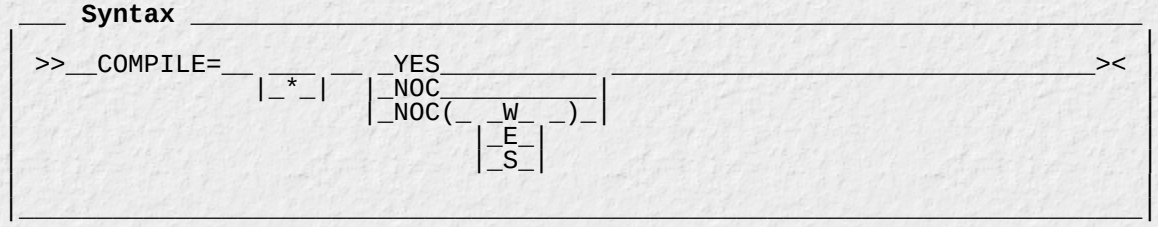
4. CMPR2=YES cannot be specified with certain other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflicting compiler options.

5. CMPR2=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.12 COMPILE

Syntax



Default
COMPILE=NOC(S)

YES
Indicates that you want full compilation, including diagnostics and object code.

NOC
Indicates that you want only a syntax check.

NOC(W)
NOC(E)
NOC(S)
Specifies an error message level:

W for warning

E for error

S for severe

When an error of the specified level or of a more severe level occurs, compilation stops, and only syntax checking is done for the remainder of the compilation.

Notes:

1. If the following compiler options are explicitly or implicitly specified in a program, COMPILE=NOC issues a warning message during compile time:

LIST=YES	TEST=(other than NO)
OFFSET=YES	VBREF=YES
OPT=STD or OPT=FULL	

Specifying these options together with COMPILE=NOC, while attempting

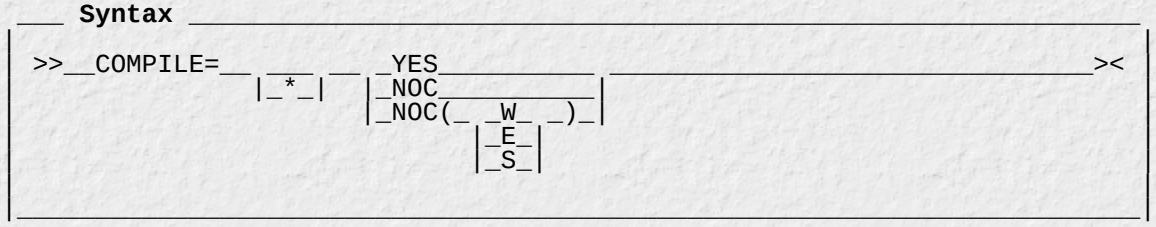
to assemble the customization macro, will result in a nonzero return code.

2. Specifying NOCOMPILE may affect the Associated Data file by stopping compilation prematurely, resulting in loss of specific messages.

IBM Library Server

1.2.5.12 COMPILE

Syntax



Default
COMPILE=NOC(S)

YES
Indicates that you want full compilation, including diagnostics and object code.

NOC
Indicates that you want only a syntax check.

NOC(W)
NOC(E)
NOC(S)
Specifies an error message level:

W for warning

E for error

S for severe

When an error of the specified level or of a more severe level occurs, compilation stops, and only syntax checking is done for the remainder of the compilation.

Notes:

1. If the following compiler options are explicitly or implicitly specified in a program, COMPILE=NOC issues a warning message during compile time:

LIST=YES	TEST=(other than NO)
OFFSET=YES	VBREF=YES
OPT=STD or OPT=FULL	

Specifying these options together with COMPILE=NOC, while attempting

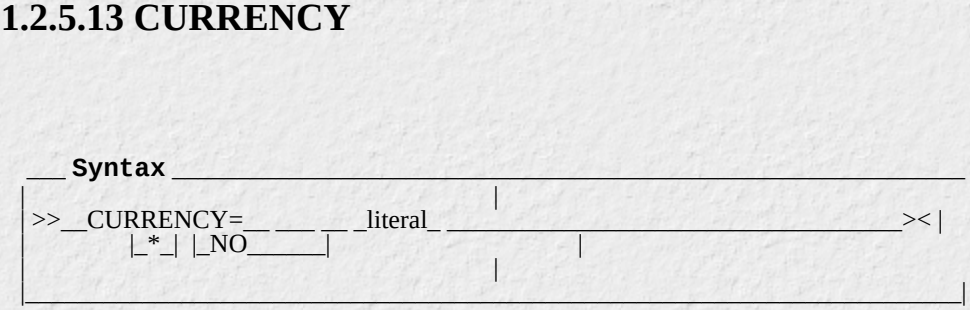
to assemble the customization macro, will result in a nonzero return code.

2. Specifying NOCOMPILE may affect the Associated Data file by stopping compilation prematurely, resulting in loss of specific messages.

 *Copyright IBM Corp. 1983,1998*

1.2.5.13 CURRENCY

Syntax



The COBOL default currency symbol is the dollar sign ($). The CURRENCY option allows you to define an alternate default currency symbol.

Default
CURRENCY=NO

literal
Represents the default currency symbol you want to use in your program.

The literal must be a nonnumeric literal representing a one-byte EBCDIC character that **must not** be any of the following:

- Digits zero (0) through nine (9)
- Uppercase alphabetic characters: A B C D P R S V X Z
- Lowercase alphabetic characters a through z
- The space
- Special characters: * + - / , . ; ( ) = "
- Uppercase alphabetic character G, if the COBOL program defines a DBCS item with the PICTURE symbol G. The PICTURE clause will be invalid for that DBCS item because the symbol G is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character N, if the COBOL program defines a DBCS item with the PICTURE symbol N. The PICTURE clause will be invalid for that DBCS item because the symbol N is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character E, if the COBOL program defines an external floating point item. The PICTURE clause will be invalid for the external floating-point item because the symbol E is considered to be a currency symbol in the PICTURE clause.

The literal (including hex literal) syntax rules are as follows:

- The literal delimiters may be either quotes or apostrophes regardless of any option setting for literal delimiters.

- When an apostrophe (') is to be the currency sign, the embedded apostrophe must be "doubled". That is, two apostrophes must be coded to represent one apostrophe within the literal. For example:

  "" or """"

- The format for a hex literal specification is as follows:

  X'H1H2' or X"H1H2"

  where H1H2 is a valid hexadecimal value representing a one-byte EBCDIC character conforming to the rules for the currency sign literal as described above. Alphabetic characters in the hex literal must be in uppercase.

  Note: Hex values of X'20 or X'21' are not allowed.

NO
indicates that no alternate default currency sign is provided by way of the CURRENCY option, and the dollar sign will be used as the default currency sign for the program if the CURRENCY option is not specified at compile time.

The value NO provides the same results for the source program as omitting the CURRENCY SIGN clause in the COBOL source program.

Notes:

1. You can use the CURRENCY option as an alternative to the CURRENCY SIGN clause (which is specified in the COBOL source program) for selecting the currency symbol you will use in the PICTURE clause of your COBOL program.

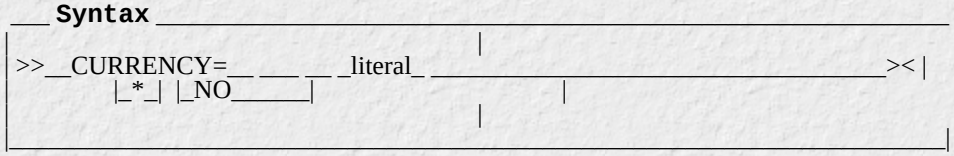
2. When both the CURRENCY option and the CURRENCY SIGN clause are used in a program, the symbol specified in the CURRENCY SIGN clause will be considered the currency symbol in a PICTURE clause when that symbol is used (even if the CURRENCY option is fixed {*}).

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.13 CURRENCY

Syntax



The COBOL default currency symbol is the dollar sign ($). The CURRENCY option allows you to define an alternate default currency symbol.

Default
CURRENCY=NO

literal
Represents the default currency symbol you want to use in your program.

The literal must be a nonnumeric literal representing a one-byte EBCDIC character that **must not** be any of the following:

- Digits zero (0) through nine (9)
- Uppercase alphabetic characters: A B C D P R S V X Z
- Lowercase alphabetic characters a through z
- The space
- Special characters: * + - / , . ; ( ) = "
- Uppercase alphabetic character G, if the COBOL program defines a DBCS item with the PICTURE symbol G. The PICTURE clause will be invalid for that DBCS item because the symbol G is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character N, if the COBOL program defines a DBCS item with the PICTURE symbol N. The PICTURE clause will be invalid for that DBCS item because the symbol N is considered to be a currency symbol in the PICTURE clause.
- Uppercase alphabetic character E, if the COBOL program defines an external floating point item. The PICTURE clause will be invalid for the external floating-point item because the symbol E is considered to be a currency symbol in the PICTURE clause.

The literal (including hex literal) syntax rules are as follows:

- The literal delimiters may be either quotes or apostrophes regardless of any option setting for literal delimiters.

- When an apostrophe (') is to be the currency sign, the embedded apostrophe must be "doubled". That is, two apostrophes must be coded to represent one apostrophe within the literal. For example:

  "" or """"

- The format for a hex literal specification is as follows:

  X'H1H2' or X"H1H2"

  where H1H2 is a valid hexadecimal value representing a one-byte EBCDIC character conforming to the rules for the currency sign literal as described above. Alphabetic characters in the hex literal must be in uppercase.

  Note: Hex values of X'20 or X'21' are not allowed.

NO
indicates that no alternate default currency sign is provided by way of the CURRENCY option, and the dollar sign will be used as the default currency sign for the program if the CURRENCY option is not specified at compile time.

The value NO provides the same results for the source program as omitting the CURRENCY SIGN clause in the COBOL source program.

Notes:

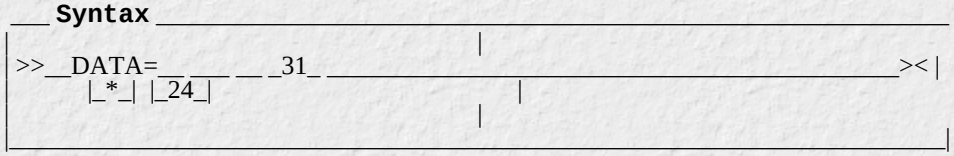
1. You can use the CURRENCY option as an alternative to the CURRENCY SIGN clause (which is specified in the COBOL source program) for selecting the currency symbol you will use in the PICTURE clause of your COBOL program.

2. When both the CURRENCY option and the CURRENCY SIGN clause are used in a program, the symbol specified in the CURRENCY SIGN clause will be considered the currency symbol in a PICTURE clause when that symbol is used (even if the CURRENCY option is fixed {*}).

IBM Library Server

1.2.5.14 DATA

Syntax



Default
DATA=31

31
Causes user data areas, such as working storage and FD record areas, to be allocated from unrestricted storage in storage acquired by a GETVIS with the LOC=ANY option. Specifying this option can result in storage being acquired in virtual addresses either above or below 16 megabytes. The operating system generally satisfies the request with space in virtual addresses above 16 megabytes, if it is available.

24
Causes user data areas to be allocated in virtual addresses below 16 megabytes in storage acquired by a GETVIS with the LOC=BELOW option.

Specify DATA=24 for programs running in 31-bit mode that are passing data parameters to programs in 24-bit mode. This includes the following cases:

- An AMODE(31) program is passing items in its working storage to an AMODE(24) program.
- An AMODE(31) program is passing, by reference, data items received from its caller to an AMODE(24) program. DATA=24 is required even when the data received is below the 16-megabyte line.

Otherwise, the data may not be addressable by the called program.

Notes:

1. When a program is compiled with the RENT option and run in 31-bit mode, the DATA option controls how dynamic storage is acquired.
2. This option has no effect on a program compiled with the NORENT option, or a program run in 24-bit mode.
3. This option is ignored at run time on a 24-bit mode system, but is always present in the object module. If the module is subsequently moved, without recompilation, to an extended architecture system and run in 31-bit mode, the DATA option will control how dynamic storage

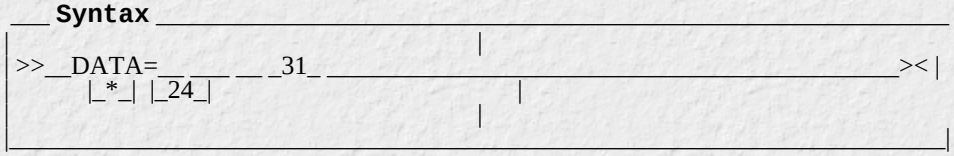
is acquired.

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.14 DATA

Syntax



Default
DATA=31

31
Causes user data areas, such as working storage and FD record areas, to be allocated from unrestricted storage in storage acquired by a GETVIS with the LOC=ANY option. Specifying this option can result in storage being acquired in virtual addresses either above or below 16 megabytes. The operating system generally satisfies the request with space in virtual addresses above 16 megabytes, if it is available.

24
Causes user data areas to be allocated in virtual addresses below 16 megabytes in storage acquired by a GETVIS with the LOC=BELOW option.

Specify DATA=24 for programs running in 31-bit mode that are passing data parameters to programs in 24-bit mode. This includes the following cases:

- An AMODE(31) program is passing items in its working storage to an AMODE(24) program.
- An AMODE(31) program is passing, by reference, data items received from its caller to an AMODE(24) program. DATA=24 is required even when the data received is below the 16-megabyte line.

Otherwise, the data may not be addressable by the called program.

Notes:

1. When a program is compiled with the RENT option and run in 31-bit mode, the DATA option controls how dynamic storage is acquired.

2. This option has no effect on a program compiled with the NORENT option, or a program run in 24-bit mode.

3. This option is ignored at run time on a 24-bit mode system, but is always present in the object module. If the module is subsequently moved, without recompilation, to an extended architecture system and run in 31-bit mode, the DATA option will control how dynamic storage

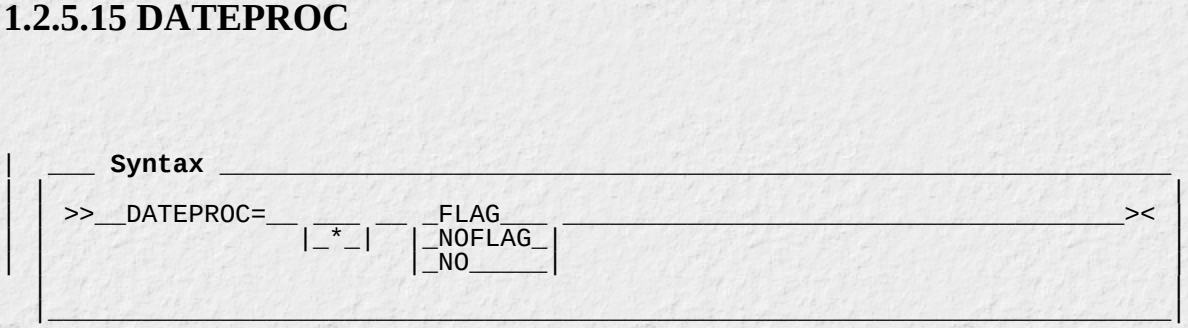
is acquired.

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.2.5.15 DATEPROC

Syntax



The DATEPROC option determines whether the compiler will perform date processing using the DATE FORMAT clause and other language constructs. IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA must be installed to specify anything other than DATEPROC=NO.

Default
DATEPROC=NO

FLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. In addition, specifying DATEPROC=FLAG instructs the compiler to flag, either with an information-level message or a warning-level message as appropriate, each statement that uses or is affected by date processing.

NOFLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. Statements that use or are affected by date processing are not flagged with information-level or warning-level messages.

NO
Instructs the compiler to treat the DATE FORMAT clause as comments and to disable automatic date processing. In the case of the new intrinsic functions, specifying DATEPROC=NO instructs the compiler to generate object code that returns a default value whenever a new intrinsic function is used.

Notes:

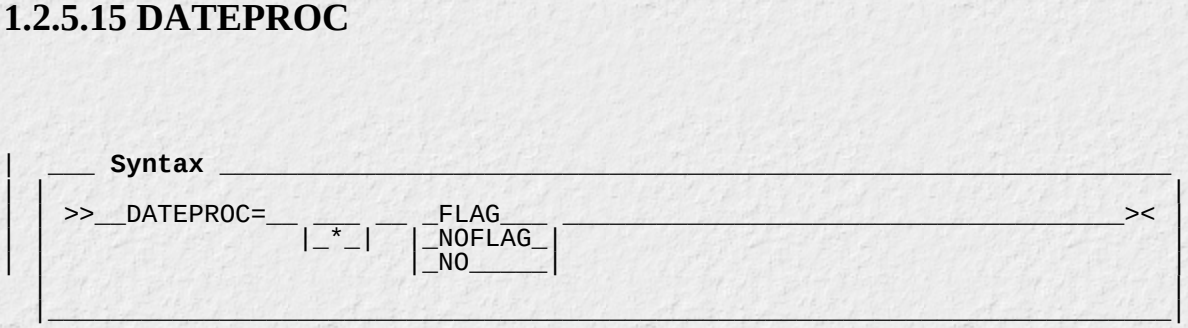
1. Error-level and severe-level messages are issued regardless of whether DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. DATEPROC=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.15 DATEPROC

Syntax



The DATEPROC option determines whether the compiler will perform date processing using the DATE FORMAT clause and other language constructs. IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA must be installed to specify anything other than DATEPROC=NO.

Default
DATEPROC=NO

FLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. In addition, specifying DATEPROC=FLAG instructs the compiler to flag, either with an information-level message or a warning-level message as appropriate, each statement that uses or is affected by date processing.

NOFLAG
Instructs the compiler to recognize the DATE FORMAT clause and to perform automatic date processing. Statements that use or are affected by date processing are not flagged with information-level or warning-level messages.

NO
Instructs the compiler to treat the DATE FORMAT clause as comments and to disable automatic date processing. In the case of the new intrinsic functions, specifying DATEPROC=NO instructs the compiler to generate object code that returns a default value whenever a new intrinsic function is used.

Notes:

1. Error-level and severe-level messages are issued regardless of whether DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. DATEPROC=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.16 DBCS

Syntax

Default
DBCS=NO

YES
Instructs the compiler to recognize X'0E' and X'0F' in a nonnumeric literal and to treat them as Shift-Out and Shift-In control characters for delimiting DBCS data.

NO
Specifies the compiler is not to recognize X'0E' and X'0F' in a nonnumeric literal.

Notes:

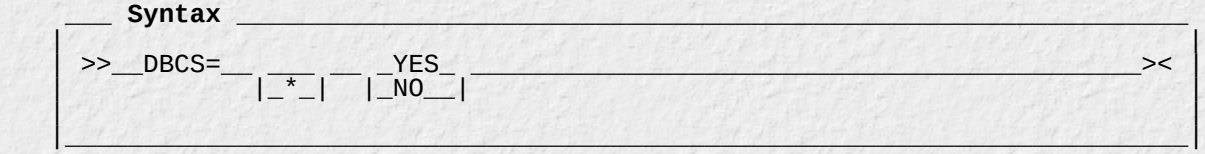
1. The presence of DBCS data inside the nonnumeric literal may cause the compiler to disallow certain uses of that literal. For example, DBCS characters are not allowed as program names or file system-names.

2. DBCS=NO supports the COBOL 85 standard.

3. DBCS=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.16 DBCS



Default
DBCS=NO

YES
Instructs the compiler to recognize X'0E' and X'0F' in a nonnumeric literal and to treat them as Shift-Out and Shift-In control characters for delimiting DBCS data.

NO
Specifies the compiler is not to recognize X'0E' and X'0F' in a nonnumeric literal.

Notes:

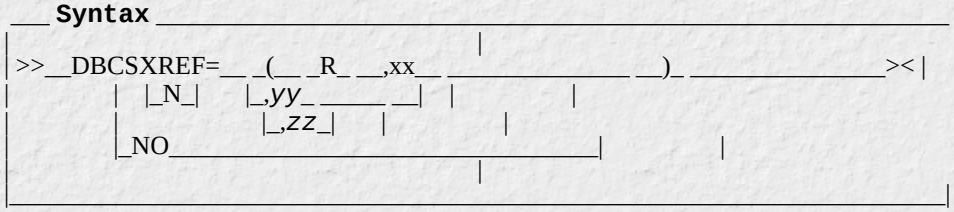
1. The presence of DBCS data inside the nonnumeric literal may cause the compiler to disallow certain uses of that literal. For example, DBCS characters are not allowed as program names or file system-names.

2. DBCS=NO supports the COBOL 85 standard.

3. DBCS=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

1.2.5.17 DBCSXREF

Syntax



Default
DBCSXREF=NO

NO
Specifies that no ordering program will be used for cross-reference of DBCS names. If the XREF phase is specified, a DBCS names cross-reference listing will be provided based on their physical order in the program.

R
Specifies that the DBCS Ordering Support Program (DBCSOS) will be loaded into the partition.

N
Specifies that the DBCSOS will be loaded into shared storage.

xx
Names a phase of the relevant ordering program to produce DBCS cross references. It must be 8 characters in length.

yy
Names an ordering type. It must be 2 characters in length. The default ordering type defined by the specified ordering program will occur if this parameter is omitted.

zz
Names the encode table used by the specified ordering type. It must be 8 characters in length. The default encode table associated with the particular ordering type will occur if this parameter is omitted.

Notes:

1. The DBCS Ordering Support Program (DBCSOS) must be installed to specify anything other than DBCSXREF=NO. DBCSOS is not available for use under VSE, therefore you should specify DBCSXREF=NO.

2. Specifying both XREFOPT=NO and DBCSXREF with an ordering program will result in a nonzero return code while attempting to assemble the customization macro.

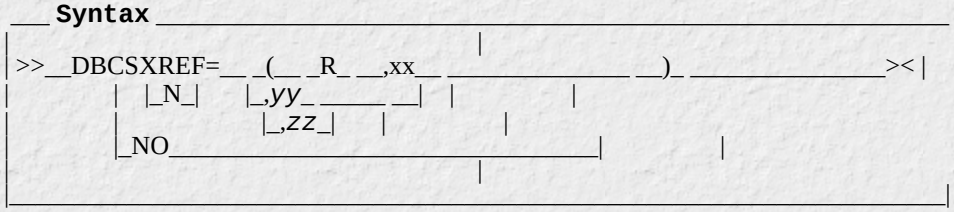
3. The assembly process will terminate when validation diagnoses:

   - Characters other than 'R' and 'N'
   - An invalid parameter length
   - Missing parameters after a comma
   - Missing 'yy' when 'zz' is specified

IBM Library Server

1.2.5.17 DBCSXREF

Syntax



Default
DBCSXREF=NO

NO
Specifies that no ordering program will be used for cross-reference of DBCS names. If the XREF phase is specified, a DBCS names cross-reference listing will be provided based on their physical order in the program.

R
Specifies that the DBCS Ordering Support Program (DBCSOS) will be loaded into the partition.

N
Specifies that the DBCSOS will be loaded into shared storage.

xx
Names a phase of the relevant ordering program to produce DBCS cross references. It must be 8 characters in length.

yy
Names an ordering type. It must be 2 characters in length. The default ordering type defined by the specified ordering program will occur if this parameter is omitted.

zz
Names the encode table used by the specified ordering type. It must be 8 characters in length. The default encode table associated with the particular ordering type will occur if this parameter is omitted.

Notes:

1. The DBCS Ordering Support Program (DBCSOS) must be installed to specify anything other than DBCSXREF=NO. DBCSOS is not available for use under VSE, therefore you should specify DBCSXREF=NO.

2. Specifying both XREFOPT=NO and DBCSXREF with an ordering program will result in a nonzero return code while attempting to assemble the customization macro.

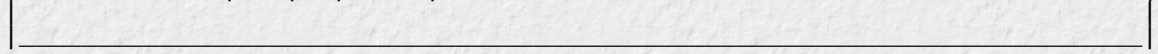
3. The assembly process will terminate when validation diagnoses:

   - Characters other than 'R' and 'N'
   - An invalid parameter length
   - Missing parameters after a comma
   - Missing 'yy' when 'zz' is specified

IBM Library Server

1.2.5.18 DYNAM

Syntax



Default
DYNAM=NO

YES
Causes subprograms that are invoked through the CALL literal statement to be dynamically loaded.

Performance Consideration: Using DYNAM=YES eases subprogram maintenance since the application will not have to be relink-edited if the subprogram is changed. However, individual applications may experience some performance degradation due to a longer path length, but overall system performance may be slightly improved.

NO
Causes the text files of subprograms called with a CALL literal statement to be included with the calling program into a single phase.

Notes:

1. DYNAM=YES is used in support of the COBOL 85 Standard.

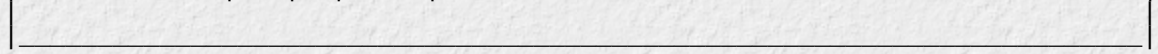
2. The DYNAM option has no effect on the CALL identifier statement at compile time. The CALL identifier statement always compiles to a dynamic call.

| 3. Do not specify DYNAM=YES for applications running under CICS.

IBM Library Server

1.2.5.18 DYNAM

Syntax



Default
DYNAM=NO

YES
Causes subprograms that are invoked through the CALL literal statement to be dynamically loaded.

Performance Consideration: Using DYNAM=YES eases subprogram maintenance since the application will not have to be relink-edited if the subprogram is changed. However, individual applications may experience some performance degradation due to a longer path length, but overall system performance may be slightly improved.

NO
Causes the text files of subprograms called with a CALL literal statement to be included with the calling program into a single phase.

Notes:

1. DYNAM=YES is used in support of the COBOL 85 Standard.

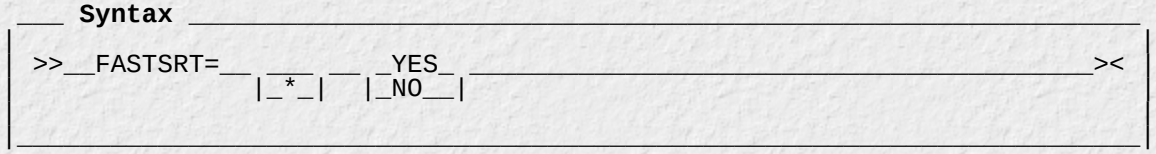
2. The DYNAM option has no effect on the CALL identifier statement at compile time. The CALL identifier statement always compiles to a dynamic call.

| 3. Do not specify DYNAM=YES for applications running under CICS.

IBM Library Server

1.2.5.19 FASTSRT

Syntax



Default
FASTSRT=NO

YES
Specifies that DFSORT/VSE or a comparable product (for example, Sort/Merge II), is to perform I/O when using either the USING or GIVING option.

Performance Consideration: Using FASTSRT=YES eliminates the overhead, in terms of CPU time, of returning to COBOL/VSE after each record is processed. However, there are restrictions you must follow if you choose to use this option. (For a detailed description of the restrictions, see *COBOL/VSE Programming Guide*.)

NO
Specifies that COBOL/VSE does the I/O for the sort/merge.

Notes:

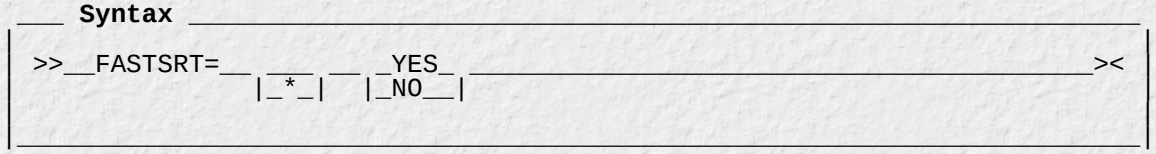
1. If FASTSRT is in effect at compile time, the compiler verifies that the FASTSRT interface can be used for all restrictions except: (1) those requiring use of a device other than a direct-access device for sort work files; and (2) if appropriate, the BLKSIZE parameter of the JCL DLBL statement for the input/output file must match the File Description (FD) of the file.

   If FASTSRT cannot be used, the compiler generates a diagnostic message and prevents the sort program from performing I/O when using either the USING or GIVING options. Therefore it may be to your advantage to specify YES as the default.

IBM Library Server

1.2.5.19 FASTSRT

Syntax



Default
FASTSRT=NO

YES
Specifies that DFSORT/VSE or a comparable product (for example, Sort/Merge II), is to perform I/O when using either the USING or GIVING option.

Performance Consideration: Using FASTSRT=YES eliminates the overhead, in terms of CPU time, of returning to COBOL/VSE after each record is processed. However, there are restrictions you must follow if you choose to use this option. (For a detailed description of the restrictions, see *COBOL/VSE Programming Guide*.)

NO
Specifies that COBOL/VSE does the I/O for the sort/merge.

Notes:

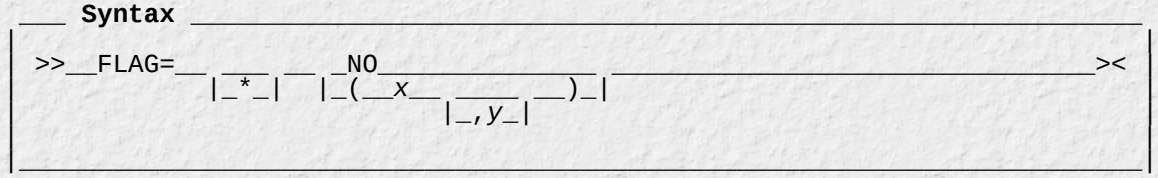
1. If FASTSRT is in effect at compile time, the compiler verifies that the FASTSRT interface can be used for all restrictions except: (1) those requiring use of a device other than a direct-access device for sort work files; and (2) if appropriate, the BLKSIZE parameter of the JCL DLBL statement for the input/output file must match the File Description (FD) of the file.

   If FASTSRT cannot be used, the compiler generates a diagnostic message and prevents the sort program from performing I/O when using either the USING or GIVING options. Therefore it may be to your advantage to specify YES as the default.

IBM Library Server

1.2.5.20 FLAG

Syntax



Default
FLAG=(I)

Note: The second severity level used in this syntax must be equal to or higher than the first.

x
I|W|E|S|U

Specifies that errors at or above the severity level specified are to be flagged and written at the end of the source listing.

ID	Type	Return Code
I	Information	0
W	Warning	4
E	Error	8
S	Severe error	12
U	Unrecoverable error	16

y
I|W|E|S|U

The optional second severity level specifies the level of syntax messages to be embedded in the source listing in addition to being at the end of the listing.

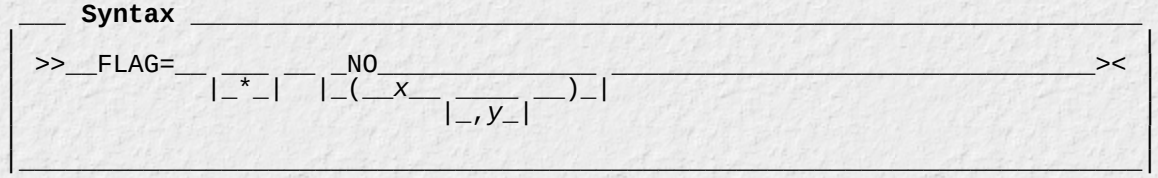
NO
Indicates that no error messages are flagged.

Note: If the messages are to be embedded, SOURCE must be specified at compile time. Embedded messages enhance productivity because they are placed after the referenced source statement.

IBM Library Server

1.2.5.20 FLAG

Syntax



Default
FLAG=(I)

Note: The second severity level used in this syntax must be equal to or higher than the first.

x
I|W|E|S|U

Specifies that errors at or above the severity level specified are to be flagged and written at the end of the source listing.

ID	Type	Return Code
I	Information	0
W	Warning	4
E	Error	8
S	Severe error	12
U	Unrecoverable error	16

y
I|W|E|S|U

The optional second severity level specifies the level of syntax messages to be embedded in the source listing in addition to being at the end of the listing.

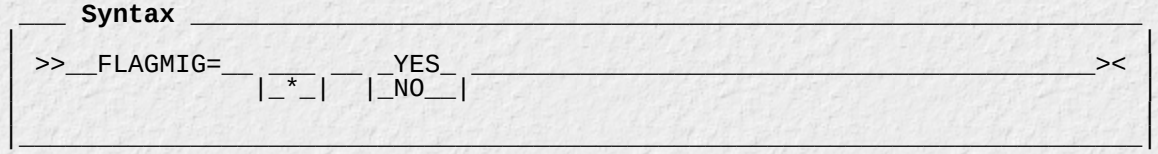
NO
Indicates that no error messages are flagged.

Note: If the messages are to be embedded, SOURCE must be specified at compile time. Embedded messages enhance productivity because they are placed after the referenced source statement.

IBM Library Server

1.2.5.21 FLAGMIG

Syntax



Default
FLAGMIG=NO

YES
Flags those VS COBOL II Release 2 language elements which might have different semantics in COBOL/VSE.

NO
Does not flag those VS COBOL II Release 2 language elements which have different semantics in COBOL/VSE.

Notes:

1. FLAGMIG=YES must be specified with CMPR2=YES to take advantage of this option.

2. FLAGMIG=YES has no effect when specified with CMPR2=NO.

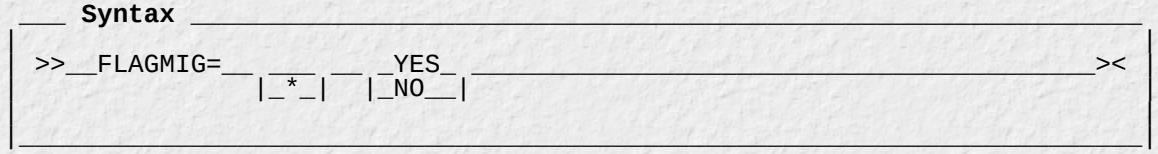
3. FLAGSAA=YES, FLAGSTD=YES, and DBCS=YES will all replace the FLAGMIG option.

4. FLAGMIG=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.21 FLAGMIG

Syntax



Default
FLAGMIG=NO

YES
Flags those VS COBOL II Release 2 language elements which might have different semantics in COBOL/VSE.

NO
Does not flag those VS COBOL II Release 2 language elements which have different semantics in COBOL/VSE.

Notes:

1. FLAGMIG=YES must be specified with CMPR2=YES to take advantage of this option.

2. FLAGMIG=YES has no effect when specified with CMPR2=NO.

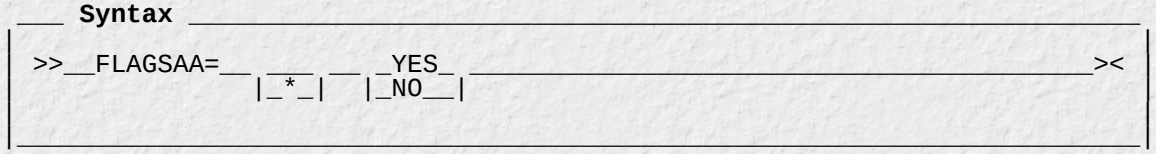
3. FLAGSAA=YES, FLAGSTD=YES, and DBCS=YES will all replace the FLAGMIG option.

4. FLAGMIG=NO supports the COBOL 85 standard.

IBM Library Server

1.2.5.22 FLAGSAA

Syntax



Default
FLAGSAA=NO

YES
Flags language elements which inhibit portability across Systems Application Architecture (SAA) COBOL systems.

NO
Does not flag language elements which inhibit portability across SAA COBOL systems.

Notes:

1. FLAGSAA=YES issues a warning (W) message to identify all SAA nonportable items.

   Specifying the following options together with FLAGSAA=YES, while attempting to assemble the customization macro, will result in a nonzero return code:

   CMPR2=YES
   FLAGSTD
   FLAGMIG

2. If the following compiler options are explicitly or implicitly specified in a program, FLAGSAA=YES issues a warning message during compile time:

   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DYNAM=NO
   FLAGMIG=YES
   FLAGSTD=(other than NO)
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT
   WORD=(other than NO or RWT)
   ZWB=NO

3. The FLAGSAA option identifies COBOL/VSE reserved words that are outside the current SAA specifications. Furthermore, the FLAGSAA option also identifies all COBOL words that are not in COBOL/VSE, but are reserved by other IBM SAA compilers.

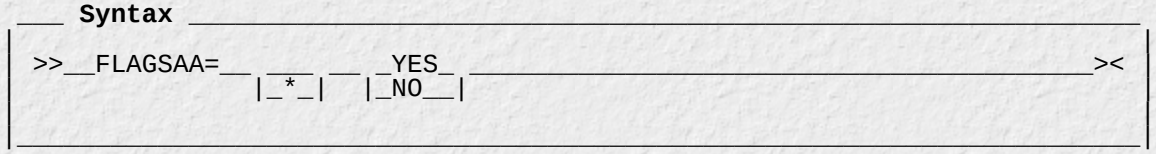
4. FLAGSAA=NO supports the COBOL 85 standard.

5. FLAGSAA can be in conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.22 FLAGSAA

Syntax



Default
FLAGSAA=NO

YES
Flags language elements which inhibit portability across Systems Application Architecture (SAA) COBOL systems.

NO
Does not flag language elements which inhibit portability across SAA COBOL systems.

Notes:

1. FLAGSAA=YES issues a warning (W) message to identify all SAA nonportable items.

   Specifying the following options together with FLAGSAA=YES, while attempting to assemble the customization macro, will result in a nonzero return code:

   CMPR2=YES
   FLAGSTD
   FLAGMIG

2. If the following compiler options are explicitly or implicitly specified in a program, FLAGSAA=YES issues a warning message during compile time:

   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DYNAM=NO
   FLAGMIG=YES
   FLAGSTD=(other than NO)
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT
   WORD=(other than NO or RWT)
   ZWB=NO

3. The FLAGSAA option identifies COBOL/VSE reserved words that are outside the current SAA specifications. Furthermore, the FLAGSAA option also identifies all COBOL words that are not in COBOL/VSE, but are reserved by other IBM SAA compilers.

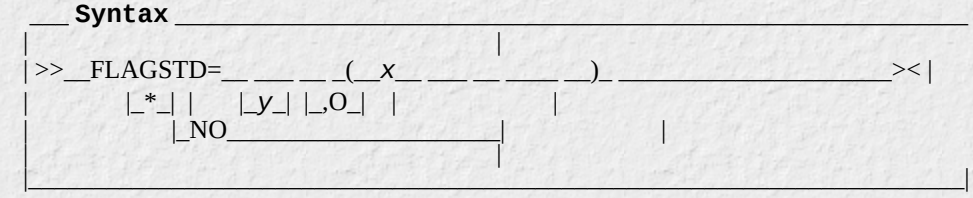
4. FLAGSAA=NO supports the COBOL 85 standard.

5. FLAGSAA can be in conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.23 FLAGSTD

Syntax



Default
FLAGSTD=NO

x
Can be M, I, or H to specify flagging for a FIPS (Federal Information Processing Standard) COBOL subset or standard.

M = ANS minimum subset of Standard COBOL.
I = ANS intermediate subset, containing those additional intermediate subset language elements which are not part of the ANS minimum subset.
H = ANS high subset, containing those additional high subset language elements which are not part of the ANS intermediate subset.

y
Can be any one or two combinations of D, N, or S to further define the level of flagging produced.

D Specifies ANS Debug module Level 1
N Specifies ANS Segmentation Module Level 1
S Specifies ANS Segmentation Module Level 2 (S is a superset of N.)

O
Specifies that obsolete language elements occurring in any of the above sets will be flagged.

NO
Specifies that no FIPS flagging will be accomplished.

Notes:

1. When FIPS flagging is specified, language syntax, such as the DATE FORMAT clause, used by the COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option) is flagged as nonconforming nonstandard IBM extensions to the COBOL 85 Standard.

2. When FIPS flagging is specified, informational messages in the source

program listing will identify:

- Whether the language element is obsolete, nonconforming standard, or nonconforming nonstandard (language elements that are both obsolete and nonconforming are flagged as obsolete only)
- The clause, statement, or header containing the nonconforming or obsolete syntax
- The source program line and an indication of the starting column within that line
- The level or optional module to which the language element belongs

3. FIPS flagging is suppressed when either:

   - Any error diagnosed as level E or higher occurs
   - The ABBR function of the WORD option is used, because standards conformance requires the standard set of reserved words

4. The FLAGSTD option can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

5. If the following compiler options are explicitly or implicitly specified in a program, FLAGSTD=(other than NO) issues a warning message during compile time:

   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DBCS=YES
   DYNAM=NO
   FASTSRT=YES
   FLAGMIG=YES
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT or BIN
   WORD=(other than NO or RWT)
   ZWB=NO

6. Specifying the following options together with FLAGSTD=(other than NO), while attempting to assemble the customization macro, will result in a nonzero return code.

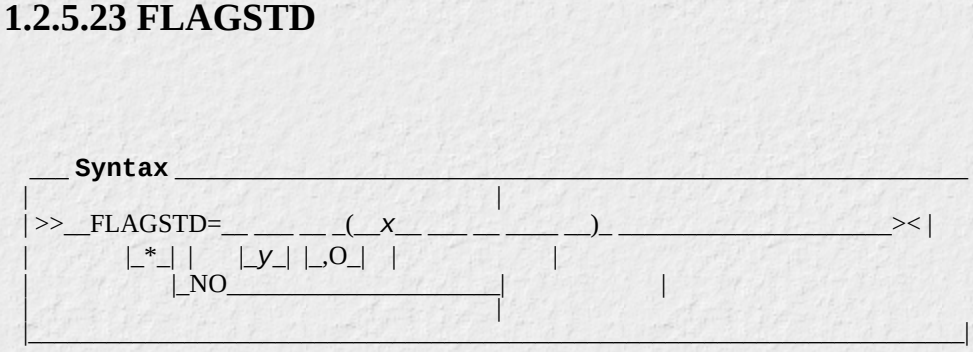
   ADV=NO
   CMPR2=YES
   DATEPROC=(other than NO)
   DBCS=YES
   DYNAM=NO
   FLAGMIG=YES
   LIB=NO
   LITCHAR=APOST
   NUM=YES
   NUMPROC=PFD
   SEQ=YES
   TRUNC=OPT or BIN
   WORD=(other than NO or RWT)
   ZWB=NO

7. FLAGSTD may affect the Associated Data file by generating error records for FIPS standard conformation messages. Error messages are not guaranteed to be sequential in regard to source record numbers.

IBM Library Server

1.2.5.23 FLAGSTD

Syntax



Default
FLAGSTD=NO

x
Can be M, I, or H to specify flagging for a FIPS (Federal Information Processing Standard) COBOL subset or standard.

M = ANS minimum subset of Standard COBOL.
I = ANS intermediate subset, containing those additional intermediate subset language elements which are not part of the ANS minimum subset.
H = ANS high subset, containing those additional high subset language elements which are not part of the ANS intermediate subset.

y
Can be any one or two combinations of D, N, or S to further define the level of flagging produced.

D Specifies ANS Debug module Level 1
N Specifies ANS Segmentation Module Level 1
S Specifies ANS Segmentation Module Level 2 (S is a superset of N.)

O
Specifies that obsolete language elements occurring in any of the above sets will be flagged.

NO
Specifies that no FIPS flagging will be accomplished.

Notes:

1. When FIPS flagging is specified, language syntax, such as the DATE FORMAT clause, used by the COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option) is flagged as nonconforming nonstandard IBM extensions to the COBOL 85 Standard.

2. When FIPS flagging is specified, informational messages in the source

program listing will identify:

- Whether the language element is obsolete, nonconforming standard, or nonconforming nonstandard (language elements that are both obsolete and nonconforming are flagged as obsolete only)
- The clause, statement, or header containing the nonconforming or obsolete syntax
- The source program line and an indication of the starting column within that line
- The level or optional module to which the language element belongs

3. FIPS flagging is suppressed when either:

   - Any error diagnosed as level E or higher occurs
   - The ABBR function of the WORD option is used, because standards conformance requires the standard set of reserved words

4. The FLAGSTD option can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

5. If the following compiler options are explicitly or implicitly specified in a program, FLAGSTD=(other than NO) issues a warning message during compile time:

   ADV=NO LITCHAR=APOST
   CMPR2=YES NUM=YES
   DATEPROC=(other than NO) NUMPROC=PFD
   DBCS=YES SEQ=YES
   DYNAM=NO TRUNC=OPT or BIN
   FASTSRT=YES WORD=(other than NO or RWT)
   FLAGMIG=YES ZWB=NO
   LIB=NO

6. Specifying the following options together with FLAGSTD=(other than NO), while attempting to assemble the customization macro, will result in a nonzero return code.

   ADV=NO LITCHAR=APOST
   CMPR2=YES NUM=YES
   DATEPROC=(other than NO) NUMPROC=PFD
   DBCS=YES SEQ=YES
   DYNAM=NO TRUNC=OPT or BIN
   FLAGMIG=YES WORD=(other than NO or RWT)
   LIB=NO ZWB=NO

7. FLAGSTD may affect the Associated Data file by generating error records for FIPS standard conformation messages. Error messages are not guaranteed to be sequential in regard to source record numbers.

IBM Library Server

1.2.5.24 INEXIT

Syntax

Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain source statements instead of using SYSIPT. When the option is supplied, SYSIPT is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

Note: Specification of INEXIT will prohibit identification of the compilation source file.

IBM Library Server

1.2.5.24 INEXIT

Syntax

Default
No exit specified.

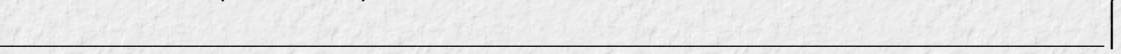
name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain source statements instead of using SYSIPT. When the option is supplied, SYSIPT is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

Note: Specification of INEXIT will prohibit identification of the compilation source file.

IBM Library Server

1.2.5.25 INTDATE

Syntax



Default
INTDATE=ANSI

ANSI
Instructs the compiler to use the ANSI COBOL Standard starting date for Integer date format dates used with date intrinsic functions. Day 1 = Jan 1, 1601.

With INTDATE(ANSI), the date intrinsic functions will return the same results as in COBOL/VSE without PTF UQ04360.

LILIAN
Instructs the compiler to use the Language Environment Lilian starting date for integer date format dates used with date intrinsic functions. Day 1 = Oct 15, 1582.

With INTDATE(LILIAN), the date intrinsic functions will return results compatible with the LE/VSE date callable services. These results will be different than in COBOL/VSE without PTF UQ04360.

Notes:

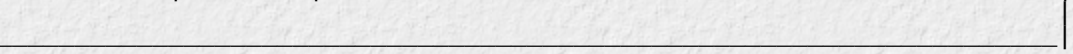
1. When INTDATE(LILIAN) is in effect, CEECBLDY will not be useable since you will have no way to turn an ANSI integer into a meaningful date using either intrinsic functions or callable services. If you code a CALL literal statement with CEECBLDY as the target of the call with INTDATE(LILIAN) in effect, the compiler will diagnose this and convert the call target to CEEDAYS.

2. If you set your installation option to INTDATE(LILIAN), you should recompile all of your COBOL/VSE programs that use Intrinsic Functions to ensure that all of your code will be using the LILIAN integer date standard. This method is the safest, because you can store integer dates, pass them between programs, and even pass them from PL/I to COBOL to C programs and have no problems.

IBM Library Server

1.2.5.25 INTDATE

Syntax



Default
INTDATE=ANSI

ANSI
Instructs the compiler to use the ANSI COBOL Standard starting date for Integer date format dates used with date intrinsic functions. Day 1 = Jan 1, 1601.

With INTDATE(ANSI), the date intrinsic functions will return the same results as in COBOL/VSE without PTF UQ04360.

LILIAN
Instructs the compiler to use the Language Environment Lilian starting date for integer date format dates used with date intrinsic functions. Day 1 = Oct 15, 1582.

With INTDATE(LILIAN), the date intrinsic functions will return results compatible with the LE/VSE date callable services. These results will be different than in COBOL/VSE without PTF UQ04360.

Notes:

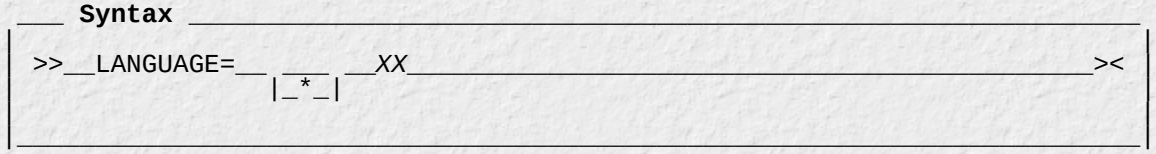
1. When INTDATE(LILIAN) is in effect, CEECBLDY will not be useable since you will have no way to turn an ANSI integer into a meaningful date using either intrinsic functions or callable services. If you code a CALL literal statement with CEECBLDY as the target of the call with INTDATE(LILIAN) in effect, the compiler will diagnose this and convert the call target to CEEDAYS.

2. If you set your installation option to INTDATE(LILIAN), you should recompile all of your COBOL/VSE programs that use Intrinsic Functions to ensure that all of your code will be using the LILIAN integer date standard. This method is the safest, because you can store integer dates, pass them between programs, and even pass them from PL/I to COBOL to C programs and have no problems.

IBM Library Server

1.2.5.26 LANGUAGE

Syntax



Default
LANGUAGE=UE

XX
Specifies the language for compiler output messages. Entries for this parameter may be selected from the following list:

Table 15. Entries for the LANGUAGE compiler option

Entry	Language
EN or **ENGLISH**	Mixed case US English
JA, **JP**, or **JAPANESE**	Japanese
UE or **UENGLISH**	Uppercase US English

Notes:

1. The LANGUAGE option name must consist of at least the first two identifying characters. Other characters following the first two identifiers may be used, however only the first two will be used to determine the language name.

2. This compiler option does not affect the language in which run-time messages are displayed. For more information on run-time options and messages, refer to the *LE/VSE Programming Guide*.

3. Some printers use only uppercase and may not accept output in mixed-case (LANGUAGE=ENGLISH).

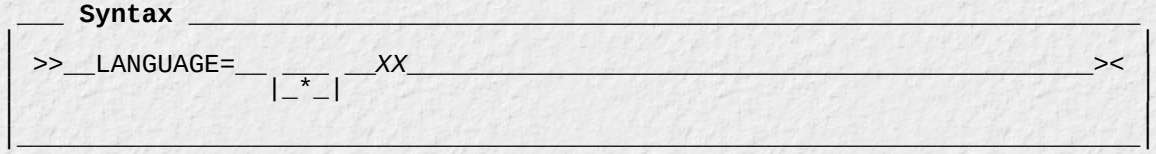
4. To specify the Japanese language option, the Japanese National Language Feature must be installed.

5. To specify the English language option (mixed-case English), the US English Language Feature must be installed.

6. If your installation provides a language other than those listed above, and you choose it to be your installation's default, you must specify at least the first two characters of the language name. The first two characters must be alphanumeric.

1.2.5.26 LANGUAGE

Syntax



Default
LANGUAGE=UE

XX
Specifies the language for compiler output messages. Entries for this parameter may be selected from the following list:

Table 15. Entries for the LANGUAGE compiler option

Entry	Language
EN or **ENGLISH**	Mixed case US English
JA, **JP**, or **JAPANESE**	Japanese
UE or **UENGLISH**	Uppercase US English

Notes:

1. The LANGUAGE option name must consist of at least the first two identifying characters. Other characters following the first two identifiers may be used, however only the first two will be used to determine the language name.

2. This compiler option does not affect the language in which run-time messages are displayed. For more information on run-time options and messages, refer to the *LE/VSE Programming Guide*.

3. Some printers use only uppercase and may not accept output in mixed-case (LANGUAGE=ENGLISH).

4. To specify the Japanese language option, the Japanese National Language Feature must be installed.

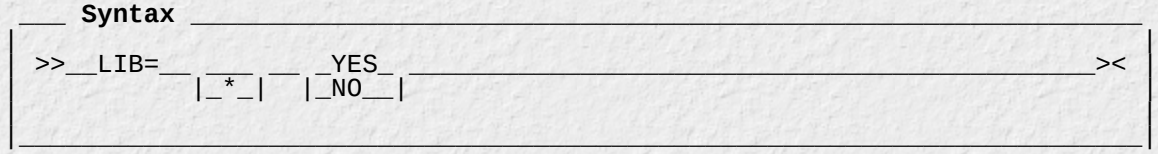
5. To specify the English language option (mixed-case English), the US English Language Feature must be installed.

6. If your installation provides a language other than those listed above, and you choose it to be your installation's default, you must specify at least the first two characters of the language name. The first two characters must be alphanumeric.

IBM Library Server

1.2.5.27 LIB

Syntax



Default
LIB=NO

YES
Indicates that the source program contains COPY or BASIS statements.

NO
Indicates that the source program does not contain COPY or BASIS statements.

Notes:

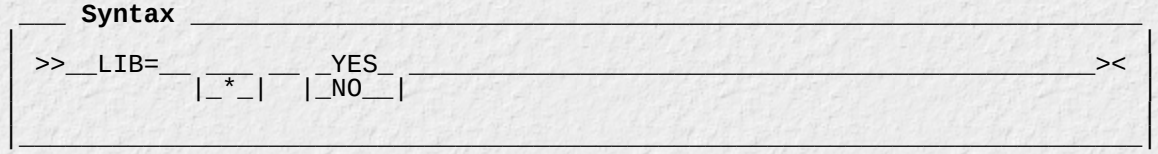
1. LIB=YES conforms to the COBOL 85 Standard.

2. LIB=YES is used when compiling a program containing COPY or BASIS statements. Specifying LIB=YES when there are no COPY or BASIS statements in a source program results in an unnecessary invocation of the COPY processing phase, unnecessary allocation of a buffer for the Librarian, and an unnecessary pass of the source text. However, compilation results are not affected.

IBM Library Server

1.2.5.27 LIB

Syntax



Default
LIB=NO

YES
Indicates that the source program contains COPY or BASIS statements.

NO
Indicates that the source program does not contain COPY or BASIS statements.

Notes:

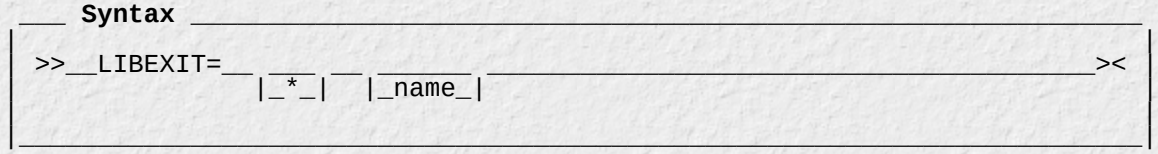
1. LIB=YES conforms to the COBOL 85 Standard.

2. LIB=YES is used when compiling a program containing COPY or BASIS statements. Specifying LIB=YES when there are no COPY or BASIS statements in a source program results in an unnecessary invocation of the COPY processing phase, unnecessary allocation of a buffer for the Librarian, and an unnecessary pass of the source text. However, compilation results are not affected.

IBM Library Server

1.2.5.28 LIBEXIT

Syntax



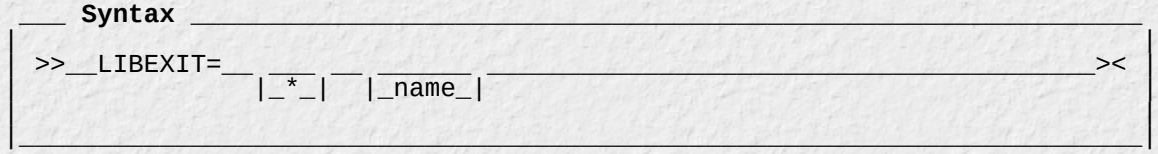
Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain copy statements instead of using the VSE Librarian to read the library search chain. When the option is supplied, the VSE Librarian is not called. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.28 LIBEXIT

Syntax



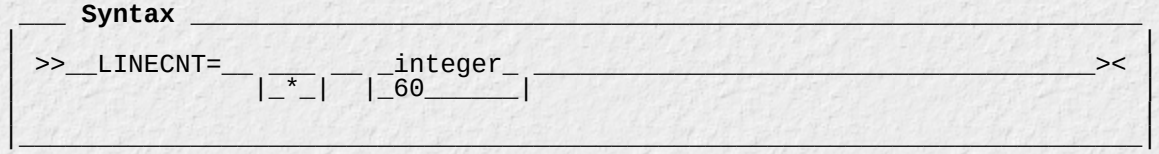
Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it to obtain copy statements instead of using the VSE Librarian to read the library search chain. When the option is supplied, the VSE Librarian is not called. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.29 LINECNT

Syntax



Default
LINECNT=60

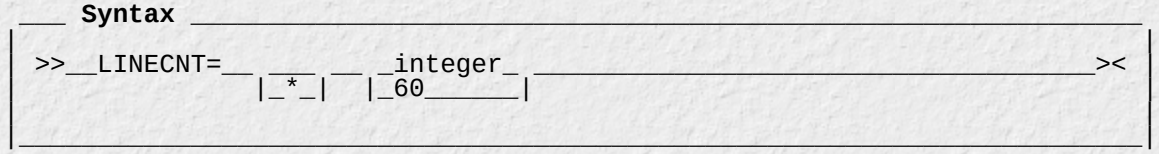
integer
Specifies the number of lines to be printed on each page of the compiler source code listing. Three of the lines are used to generate headings. For example, if you specify LINECNT=60, 57 lines of source code are printed on each page of the output listing, and 3 lines are used for headings.

| **Note:** The LINECNT installation option is equivalent to the LINECOUNT
| compile-time option.

IBM Library Server

1.2.5.29 LINECNT

Syntax



Default
LINECNT=60

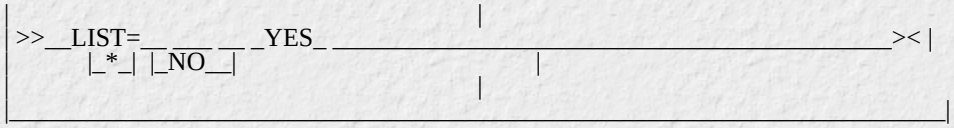
integer
Specifies the number of lines to be printed on each page of the compiler source code listing. Three of the lines are used to generate headings. For example, if you specify LINECNT=60, 57 lines of source code are printed on each page of the output listing, and 3 lines are used for headings.

| **Note:** The LINECNT installation option is equivalent to the LINECOUNT
| compile-time option.

IBM Library Server

1.2.5.30 LIST

Syntax



Default
NO

YES
Produces a listing that includes:

- The assembler-language expansion of source code
- Information about working storage
- Global tables
- Literal pools

NO
Suppresses this listing.

Notes:

1. Unless the installation default value of the LIST option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LISTX option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE LIST option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

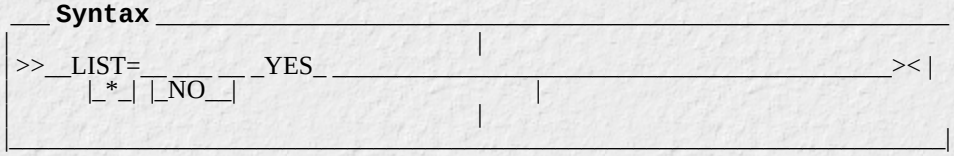
2. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code and an error message during assembly of the customization macro.

3. LIST=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.30 LIST

Syntax



Default
NO

YES
Produces a listing that includes:

- The assembler-language expansion of source code
- Information about working storage
- Global tables
- Literal pools

NO
Suppresses this listing.

Notes:

1. Unless the installation default value of the LIST option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LISTX option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE LIST option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

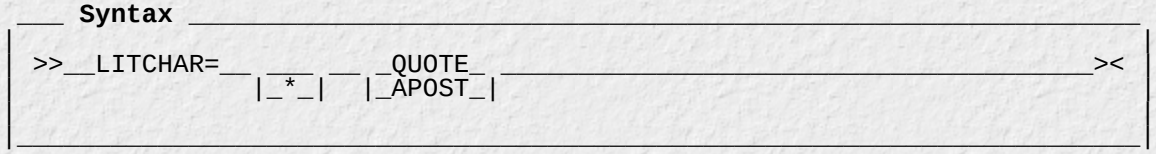
2. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code and an error message during assembly of the customization macro.

3. LIST=YES can conflict with other compiler options. See "Conflicting Compiler Options" in topic 1.2.5.3 for conflict resolution information.

IBM Library Server

1.2.5.31 LITCHAR

Syntax



Default
LITCHAR=QUOTE

QUOTE
Specifies that double quotation marks (") are used to delineate literals and in the generation of figurative constants.

APOST
Specifies that an apostrophe (') is used to delineate literals and in the generation of figurative constants.

Notes:

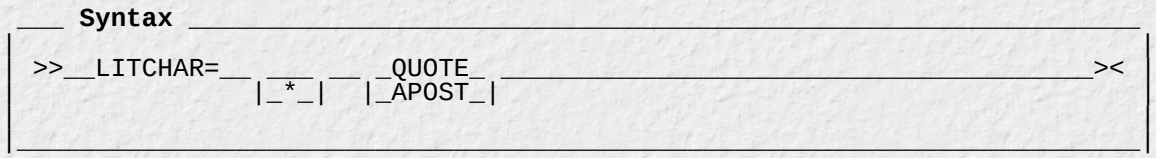
1. LITCHAR=QUOTE is used in support of the COBOL 85 Standard.

2. If you have an existing source library, determine whether QUOTE or APOST is used consistently.

IBM Library Server

1.2.5.31 LITCHAR

Syntax



Default
LITCHAR=QUOTE

QUOTE
Specifies that double quotation marks (") are used to delineate literals and in the generation of figurative constants.

APOST
Specifies that an apostrophe (') is used to delineate literals and in the generation of figurative constants.

Notes:

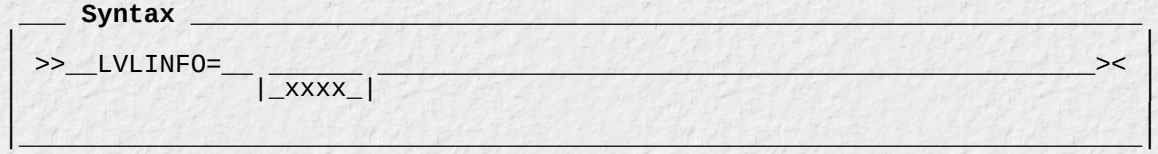
1. LITCHAR=QUOTE is used in support of the COBOL 85 Standard.

2. If you have an existing source library, determine whether QUOTE or APOST is used consistently.

IBM Library Server

1.2.5.32 LVLINFO

Syntax



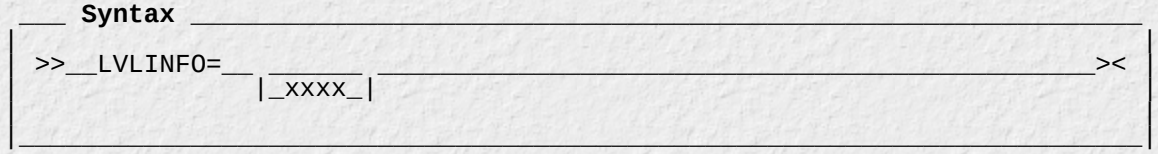
Default
No characters are specified.

xxxx
Identifies the one to four alphanumeric characters that will be inserted into the listing header following the Release number (the last four bytes of the signature area). This option may be used to identify "compiler level" information within the listing header.

IBM Library Server

1.2.5.32 LVLINFO

Syntax



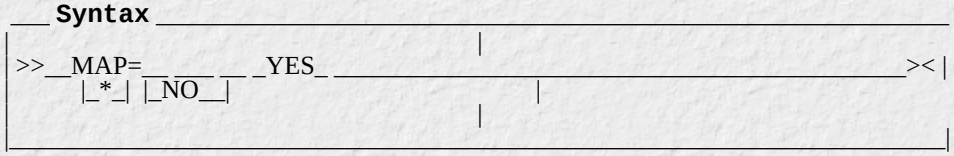
Default
No characters are specified.

xxxx
Identifies the one to four alphanumeric characters that will be inserted into the listing header following the Release number (the last four bytes of the signature area). This option may be used to identify "compiler level" information within the listing header.

IBM Library Server

1.2.5.33 MAP

Syntax



Default
MAP=NO

YES
Maps items declared in the Data Division. Map output includes:

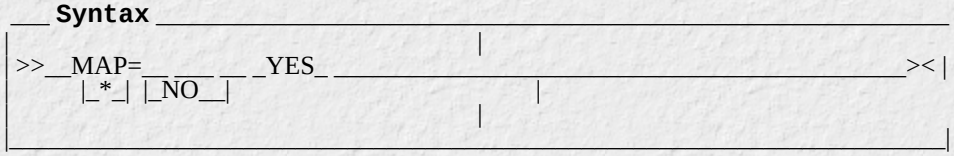
- Data Division map
- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled without the RENT compiler option

NO
Mapping is not performed.

IBM Library Server

1.2.5.33 MAP

Syntax



Default
MAP=NO

YES
Maps items declared in the Data Division. Map output includes:

- Data Division map
- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled without the RENT compiler option

NO
Mapping is not performed.

IBM Library Server

1.2.5.34 NAME

Syntax

Default
NAME=NO

NOALIAS
When used in conjunction with specific JCL options performs the following:

1. When used with LINK or CATAL, NOALIAS precedes each object deck written to SYSLNK with a linkage editor PHASE statement (PHASE phasename,*). The phase name (phasename) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

2. When used with DECK, NOALIAS precedes each object deck written to SYSPCH with a VSE Librarian CATALOG statement (CATALOG modname.OBJ,REPLACE=YES). The module name (modname) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

ALIAS
Specifying ALIAS has the same effect as NOALIAS.

NO
Does not produce linkage editor PHASE statements or VSE Librarian CATALOG statements.

Notes:

1. The NAME option allows you to create multiple modules in a program library with a single batch compilation. This can be useful for dynamic calls, for example.

2. NAME=NOALIAS or NAME=ALIAS supports the COBOL 85 Standard.

IBM Library Server

1.2.5.34 NAME

Syntax

Default
NAME=NO

NOALIAS
When used in conjunction with specific JCL options performs the following:

1. When used with LINK or CATAL, NOALIAS precedes each object deck written to SYSLNK with a linkage editor PHASE statement (PHASE phasename,*). The phase name (phasename) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

2. When used with DECK, NOALIAS precedes each object deck written to SYSPCH with a VSE Librarian CATALOG statement (CATALOG modname.OBJ,REPLACE=YES). The module name (modname) is derived from the PROGRAM-ID statement according to the rules for forming external module names.

ALIAS
Specifying ALIAS has the same effect as NOALIAS.

NO
Does not produce linkage editor PHASE statements or VSE Librarian CATALOG statements.

Notes:

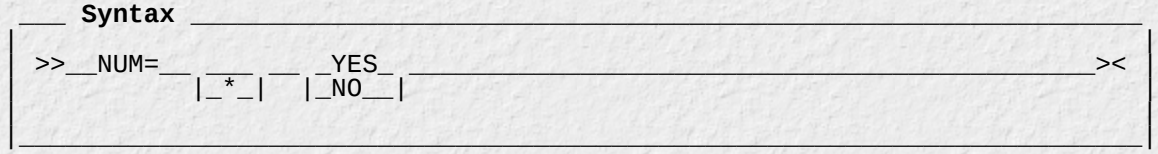
1. The NAME option allows you to create multiple modules in a program library with a single batch compilation. This can be useful for dynamic calls, for example.

2. NAME=NOALIAS or NAME=ALIAS supports the COBOL 85 Standard.

IBM Library Server

1.2.5.35 NUM

Syntax



Default
NUM=NO

YES
Uses the line numbers from the source program rather than compiler-generated line numbers for error messages and procedure maps.

NO
Uses the line numbers generated from the compiler for error messages and procedure maps.

Notes:

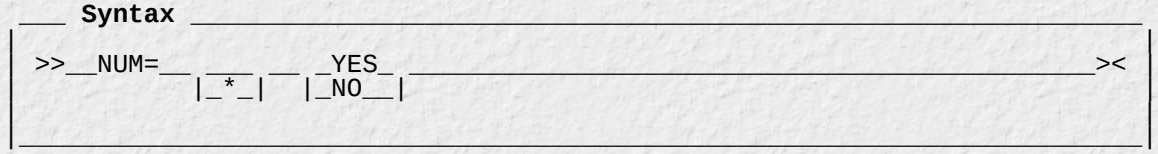
1. If COPY statements are used in a program and NUM=YES is in effect, the source program line numbers and the COPY member line numbers must be coordinated.

2. NUM=NO supports the COBOL 85 Standard.

IBM Library Server

1.2.5.35 NUM

Syntax



Default
NUM=NO

YES
Uses the line numbers from the source program rather than compiler-generated line numbers for error messages and procedure maps.

NO
Uses the line numbers generated from the compiler for error messages and procedure maps.

Notes:

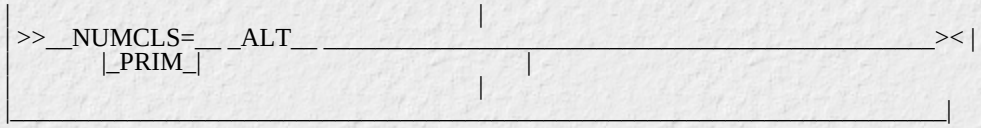
1. If COPY statements are used in a program and NUM=YES is in effect, the source program line numbers and the COPY member line numbers must be coordinated.

2. NUM=NO supports the COBOL 85 Standard.

IBM Library Server

1.2.5.36 NUMCLS

Syntax



Default
NUMCLS=PRIM

ALT
Used to specify the sign representations that are recognized as valid by the numeric class test for data items that are defined:

- As signed (with an "S" in the PICTURE clause)
- Using DISPLAY or COMPUTATIONAL-3 (packed-decimal)
- Without the SEPARATE phrase on any SIGN clause

Processing with ALT accepts hexadecimal A through F as valid.

PRIM
Processing with PRIM accepts only hexadecimal C, D, and F as valid.

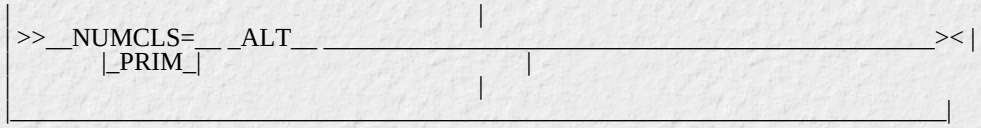
Notes:

1. The numeric class test is affected by how both the NUMPROC and the NUMCLS options are specified. The NUMCLS option is effective only for NUMPROC=MIG or NUMPROC=NOPFD. NUMPROC=PFD specifies more strict rules for valid sign configuration.

IBM Library Server

1.2.5.36 NUMCLS

Syntax



Default
NUMCLS=PRIM

ALT
Used to specify the sign representations that are recognized as valid by the numeric class test for data items that are defined:

- As signed (with an "S" in the PICTURE clause)
- Using DISPLAY or COMPUTATIONAL-3 (packed-decimal)
- Without the SEPARATE phrase on any SIGN clause

Processing with ALT accepts hexadecimal A through F as valid.

PRIM
Processing with PRIM accepts only hexadecimal C, D, and F as valid.

Notes:

1. The numeric class test is affected by how both the NUMPROC and the NUMCLS options are specified. The NUMCLS option is effective only for NUMPROC=MIG or NUMPROC=NOPFD. NUMPROC=PFD specifies more strict rules for valid sign configuration.

IBM Library Server

1.2.5.37 NUMPROC

Syntax

Default
NUMPROC=NOPFD

MIG
Aids in migrating DOS/VS COBOL application programs to COBOL/VSE. Processing with MIG:

- Uses existing signs for comparison and arithmetic operations
- Generates preferred signs for the results of MOVE and arithmetic operations (These results meet the criteria for using NUMPROC=PFD.)
- Performs numeric rather than logical comparisons

NOPFD
Repairs signs on input. After repair is performed, the signs meet the criteria for NUMPROC=PFD.

PFD
Optimizes the generated code, especially when OPT=YES is specified. No explicit sign repair is performed. Note that NUMPROC=PFD has stringent criteria to produce correct results. To use NUMPROC=PFD:

- The sign position of unsigned numeric items must be X'F'.
- The sign position of signed numeric items must be either X'C' if positive or zero, or must be X'D' if negative.
- The sign position of separately signed numeric items must either be '+' if positive or zero, or '-' if negative.

Elementary MOVE and arithmetic statements in COBOL/VSE always generate results with these preferred signs, however group MOVEs and redefinitions may produce nonconforming results. The numeric class test may be used for verification. With NUMPROC=PFD, a numeric item will fail the numeric class test if the signs do not meet the preferred sign criteria.

Performance Consideration: Using NUMPROC=PFD will generate significantly more efficient code for numeric comparisons. For most references to COMP-3 and DISPLAY numeric data items, using NUMPROC=MIG and NUMPROC=NOPFD will generate extra code because of sign "fix-up" processing. This extra code may also inhibit some other types of optimization.

Before setting this option, please consult with your application programmers to determine the effect on the application program's output.

Notes:

1. NUMPROC=NOPFD or NUMPROC=MIG supports the COBOL 85 Standard.

2. NUMPROC=NOPFD and NUMPROC=PFD are equivalent to the VS COBOL II Release 2 options PFDSGN=NO and PFDSGN=YES, respectively.

3. Both the NUMPROC and NUMCLS options affect the numeric class test. With NUMPROC=MIG or NUMPROC=NOPFD, the results of the numeric class test are controlled by how NUMCLS is set. When NUMPROC=PFD, a data item must meet the preferred sign criteria to be considered numeric.

IBM Library Server

1.2.5.37 NUMPROC

Syntax

Default
NUMPROC=NOPFD

MIG
Aids in migrating DOS/VS COBOL application programs to COBOL/VSE. Processing with MIG:

- Uses existing signs for comparison and arithmetic operations
- Generates preferred signs for the results of MOVE and arithmetic operations (These results meet the criteria for using NUMPROC=PFD.)
- Performs numeric rather than logical comparisons

NOPFD
Repairs signs on input. After repair is performed, the signs meet the criteria for NUMPROC=PFD.

PFD
Optimizes the generated code, especially when OPT=YES is specified. No explicit sign repair is performed. Note that NUMPROC=PFD has stringent criteria to produce correct results. To use NUMPROC=PFD:

- The sign position of unsigned numeric items must be X'F'.
- The sign position of signed numeric items must be either X'C' if positive or zero, or must be X'D' if negative.
- The sign position of separately signed numeric items must either be '+' if positive or zero, or '-' if negative.

Elementary MOVE and arithmetic statements in COBOL/VSE always generate results with these preferred signs, however group MOVEs and redefinitions may produce nonconforming results. The numeric class test may be used for verification. With NUMPROC=PFD, a numeric item will fail the numeric class test if the signs do not meet the preferred sign criteria.

Performance Consideration: Using NUMPROC=PFD will generate significantly more efficient code for numeric comparisons. For most references to COMP-3 and DISPLAY numeric data items, using NUMPROC=MIG and NUMPROC=NOPFD will generate extra code because of sign "fix-up" processing. This extra code may also inhibit some other types of optimization.

Before setting this option, please consult with your application programmers to determine the effect on the application program's output.

Notes:

1. NUMPROC=NOPFD or NUMPROC=MIG supports the COBOL 85 Standard.

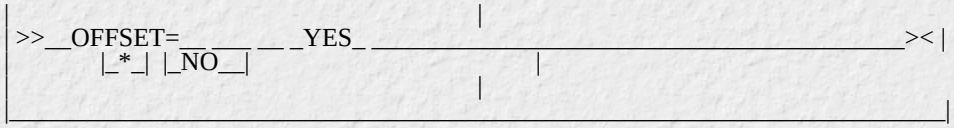
2. NUMPROC=NOPFD and NUMPROC=PFD are equivalent to the VS COBOL II Release 2 options PFDSGN=NO and PFDSGN=YES, respectively.

3. Both the NUMPROC and NUMCLS options affect the numeric class test. With NUMPROC=MIG or NUMPROC=NOPFD, the results of the numeric class test are controlled by how NUMCLS is set. When NUMPROC=PFD, a data item must meet the preferred sign criteria to be considered numeric.

IBM Library Server

1.2.5.38 OFFSET

Syntax



Default
OFFSET=NO

YES
Produces a condensed Procedure Division listing. The procedure portion of the listing will contain line numbers, verb references, and the location of the first instruction generated for each verb. In addition, the following are produced:

- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled with the NORENT compiler option

NO
Does not condense the listing and does not produce the items listed above.

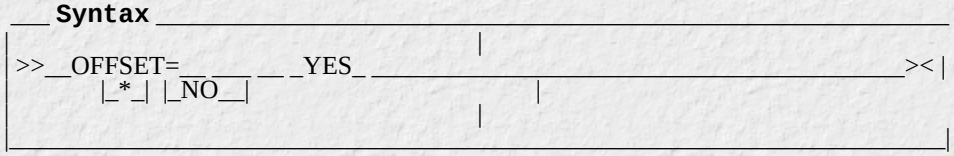
Notes:

1. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code when attempting to assemble the customization macro. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflict resolution.

IBM Library Server

1.2.5.38 OFFSET

Syntax



Default
OFFSET=NO

YES
Produces a condensed Procedure Division listing. The procedure portion of the listing will contain line numbers, verb references, and the location of the first instruction generated for each verb. In addition, the following are produced:

- Global tables
- Literal pools
- Program statistics
- Size of the program's working storage, and its location in the object code if the program is compiled with the NORENT compiler option

NO
Does not condense the listing and does not produce the items listed above.

Notes:

1. The LIST and OFFSET compiler options are mutually exclusive. Setting OFFSET=YES and LIST=YES results in a nonzero return code when attempting to assemble the customization macro. See "Conflicting Compiler Options" in topic 1.2.5.3 for more information on conflict resolution.

1.2.5.39 OPT

Syntax

Default
OPT=NO

STD
Causes the compiler to generate optimized object code.

FULL
Causes the compiler to generate optimized object code, and to delete unused WORKING-STORAGE data items as well as delete the VALUE clause code that initializes those items.

Performance Consideration: Using OPT=STD or OPT=FULL generally results in more efficient run-time code. This option requires more CPU time for compiles than OPT=NO.

NO
Does not cause the compiler to generate optimized object code.

Notes:

1. You can only specify TEST=(NONE,...) for the hook-location subparameter with OPT=STD or OPT=FULL. Any other combination results in a nonzero return code and an error during installation.

2. Do not use OPT=FULL if your programs depend on 'using' data items that are not referenced in the procedure division, such as adjacent tables, addresses passed to assembler programs, or 'eyecatchers' to identify the begin and end of the WORKING-STORAGE section. See *COBOL/VSE Programming Guide*.

3. Optimization is turned off if an S-level error or higher is flagged by the compiler.

IBM Library Server

1.2.5.39 OPT

Syntax

Default
OPT=NO

STD
Causes the compiler to generate optimized object code.

FULL
Causes the compiler to generate optimized object code, and to delete unused WORKING-STORAGE data items as well as delete the VALUE clause code that initializes those items.

Performance Consideration: Using OPT=STD or OPT=FULL generally results in more efficient run-time code. This option requires more CPU time for compiles than OPT=NO.

NO
Does not cause the compiler to generate optimized object code.

Notes:

1. You can only specify TEST=(NONE,...) for the hook-location subparameter with OPT=STD or OPT=FULL. Any other combination results in a nonzero return code and an error during installation.

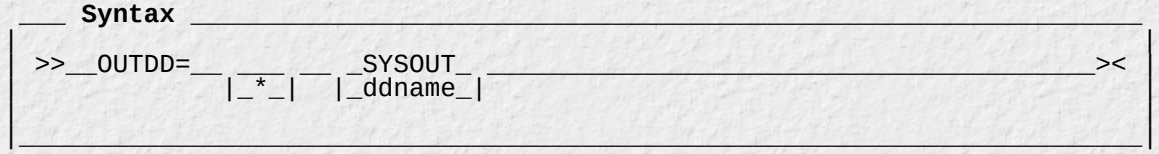
2. Do not use OPT=FULL if your programs depend on 'using' data items that are not referenced in the procedure division, such as adjacent tables, addresses passed to assembler programs, or 'eyecatchers' to identify the begin and end of the WORKING-STORAGE section. See *COBOL/VSE Programming Guide*.

3. Optimization is turned off if an S-level error or higher is flagged by the compiler.

IBM Library Server

1.2.5.40 OUTDD

Syntax



Default
OUTDD=SYSOUT

filename
Specifies the file name of the file to be used for run-time DISPLAY output.

Notes:

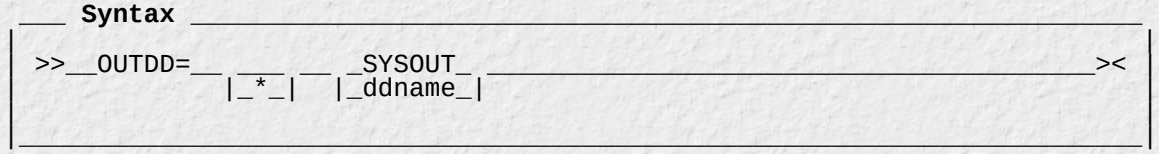
1. See the *LE/VSE Programming Reference* description of the MSGFILE run-time option to see how OUTDD interacts with MSGFILE.

2. Change the default for this option if, at run time, you expect to have a conflict with another product that requires SYSOUT as a file name. Both SYSOUT and SYSLST direct the output to SYSLST.

IBM Library Server

1.2.5.40 OUTDD

Syntax



Default
OUTDD=SYSOUT

filename
Specifies the file name of the file to be used for run-time DISPLAY output.

Notes:

1. See the *LE/VSE Programming Reference* description of the MSGFILE run-time option to see how OUTDD interacts with MSGFILE.

2. Change the default for this option if, at run time, you expect to have a conflict with another product that requires SYSOUT as a file name. Both SYSOUT and SYSLST direct the output to SYSLST.

IBM Library Server

1.2.5.41 PRTEXIT

Syntax

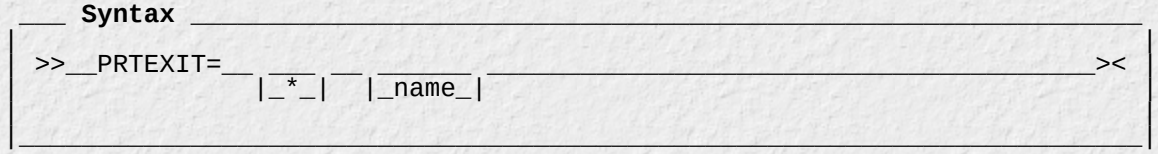
Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it instead of writing to SYSLST. When the option is supplied, SYSLST is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.41 PRTEXIT

Syntax



Default
No exit specified.

name
Identifies a module to be used with the EXIT compiler option. When the suboption for this user exit is specified, the compiler loads that module and calls it instead of writing to SYSLST. When the option is supplied, SYSLST is not opened. For more specific information, see the *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.42 RENT

Syntax

```
>>__RENT=__ ___ __ _YES_ ____________________________________________><
          |_*_|  |_NO__|
```

Default
RENT=NO

YES
Indicates that the object code produced for a COBOL program is to be reentrant.

Using RENT=YES enables the program to be placed in the SVA for running above the 16-megabyte line. However, this option causes the compiler to generate additional code to ensure that the application program is reentrant.

NO
Indicates that the object code produced for a COBOL program is not to be reentrant.

Notes:

1. Programs must be compiled with RENT=YES or RMODE=ANY if they will be run with extended addressing in virtual storage addresses above 16 megabytes.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RENT/NORENT and RMODE compiler options. Valid combinations include:

Table 16. Effect of RENT and RMODE on Residency Mode

RENT/NORENT Setting	RMODE Setting	Residency Mode Assigned
RENT	AUTO	RMODE(ANY)
RENT	ANY	RMODE(ANY)
RENT	24	RMODE(24)
NORENT	AUTO	RMODE(24)
NORENT	ANY	RMODE(ANY)

NORENT	24	RMODE(24)

IBM Library Server

1.2.5.42 RENT

Syntax

```
>>__RENT=__ ___ __ _YES_ ____________________________________________><
          |_*_|  |_NO__|
```

Default
RENT=NO

YES
Indicates that the object code produced for a COBOL program is to be reentrant.

Using RENT=YES enables the program to be placed in the SVA for running above the 16-megabyte line. However, this option causes the compiler to generate additional code to ensure that the application program is reentrant.

NO
Indicates that the object code produced for a COBOL program is not to be reentrant.

Notes:

1. Programs must be compiled with RENT=YES or RMODE=ANY if they will be run with extended addressing in virtual storage addresses above 16 megabytes.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RENT/NORENT and RMODE compiler options. Valid combinations include:

Table 16. Effect of RENT and RMODE on Residency Mode

RENT/NORENT Setting	RMODE Setting	Residency Mode Assigned
RENT	AUTO	RMODE(ANY)
RENT	ANY	RMODE(ANY)
RENT	24	RMODE(24)
NORENT	AUTO	RMODE(24)
NORENT	ANY	RMODE(ANY)

	NORENT	24	RMODE(24)

IBM Library Server

1.2.5.43 RMODE

Syntax

```
>>__RMODE=__ ___ __ _AUTO_ _____________________________________________><
            |_*_|  |_24___|
                   |_ANY__|
```

Default
RMODE=AUTO

AUTO
A program compiled with the RMODE=AUTO option will have residency mode 24 if NORENT is specified, and residency mode ANY if RENT is specified.

24
A program compiled with the RMODE=24 option will have residency mode 24 whether NORENT or RENT is specified.

ANY
A program compiled with the RMODE=ANY option will have residency mode ANY whether NORENT or RENT is specified.

Notes:

1. COBOL/VSE NORENT programs that are required to pass data to programs running in AMODE(24) must either be compiled with the RMODE(24) option, or link-edited with RMODE(24). The data areas for NORENT programs will be above the line or below the line depending on the RMODE of the program, even if DATA(24) has been specified. DATA(24) applies to programs compiled with the RENT option only.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RMODE and RENT/NORENT compiler options. Valid combinations include:

Table 17. Effect of RMODE and RENT/NORENT on Residency Mode

RMODE Setting	RENT/NORENT Setting	Residency Mode Assigned
AUTO	RENT	RMODE(ANY)
AUTO	NORENT	RMODE(24)
ANY	RENT	RMODE(ANY)

ANY	NORENT	RMODE(ANY)
24	RENT	RMODE(24)
24	NORENT	RMODE(24)

IBM Library Server

1.2.5.43 RMODE

Syntax

Default
RMODE=AUTO

AUTO
A program compiled with the RMODE=AUTO option will have residency mode 24 if NORENT is specified, and residency mode ANY if RENT is specified.

24
A program compiled with the RMODE=24 option will have residency mode 24 whether NORENT or RENT is specified.

ANY
A program compiled with the RMODE=ANY option will have residency mode ANY whether NORENT or RENT is specified.

Notes:

1. COBOL/VSE NORENT programs that are required to pass data to programs running in AMODE(24) must either be compiled with the RMODE(24) option, or link-edited with RMODE(24). The data areas for NORENT programs will be above the line or below the line depending on the RMODE of the program, even if DATA(24) has been specified. DATA(24) applies to programs compiled with the RENT option only.

2. Programs compiled with COBOL/VSE always have AMODE(ANY). The RMODE assigned to a program depends on the RMODE and RENT/NORENT compiler options. Valid combinations include:

Table 17. Effect of RMODE and RENT/NORENT on Residency Mode

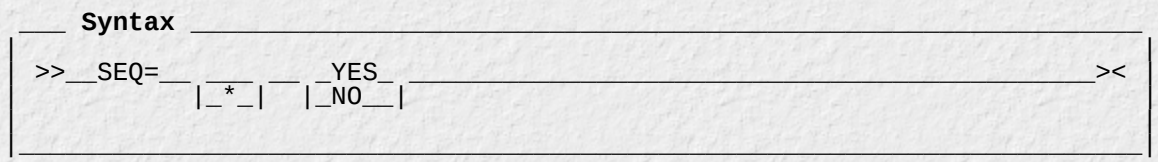
RMODE Setting	RENT/NORENT Setting	Residency Mode Assigned
AUTO	RENT	RMODE(ANY)
AUTO	NORENT	RMODE(24)
ANY	RENT	RMODE(ANY)

ANY	NORENT	RMODE(ANY)
24	RENT	RMODE(24)
24	NORENT	RMODE(24)

IBM Library Server

1.2.5.44 SEQ

Syntax



Default
SEQ=YES

YES
Instructs the compiler to check that the source statements are in ascending alphanumeric order by line number.

NO
Instructs the compiler not to perform sequence checking.

Notes:

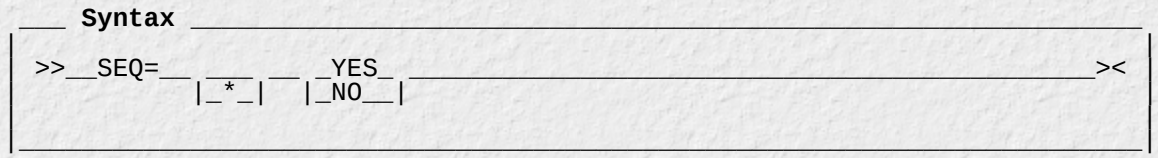
1. If both SEQ and NUM are in effect at compile time, the sequence is checked according to numeric, rather than alphanumeric, collating sequence.

2. SEQ=NO supports the COBOL 85 Standard.

IBM Library Server

1.2.5.44 SEQ

Syntax



Default
SEQ=YES

YES
Instructs the compiler to check that the source statements are in ascending alphanumeric order by line number.

NO
Instructs the compiler not to perform sequence checking.

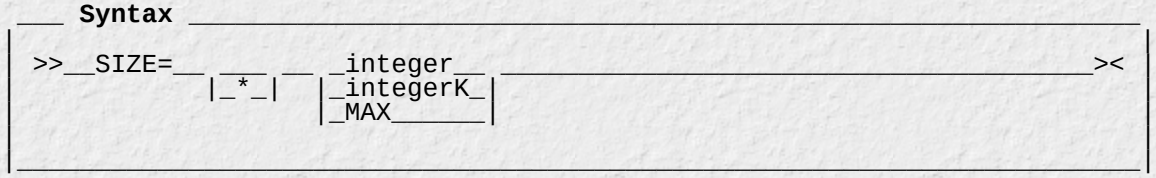
Notes:

1. If both SEQ and NUM are in effect at compile time, the sequence is checked according to numeric, rather than alphanumeric, collating sequence.

2. SEQ=NO supports the COBOL 85 Standard.

1.2.5.45 SIZE

Syntax



Default
SIZE=MAX

integer
Specifies the amount, in bytes, of virtual storage available to the compiler.

The minimum acceptable value is 778240.

integerK
Specifies the amount of virtual storage available to the compiler in 1024-byte (K) increments.

The minimum acceptable value is 760K.

MAX
The compiler will request all available space in the partition GETVIS for use during the compilation. For Extended Architecture, the compiler obtains the largest contiguous block of free storage above the 16-megabyte line.

Notes:

1. Using SIZE=MAX simplifies compiler invocation by eliminating the need to determine a specific value for the SIZE option.

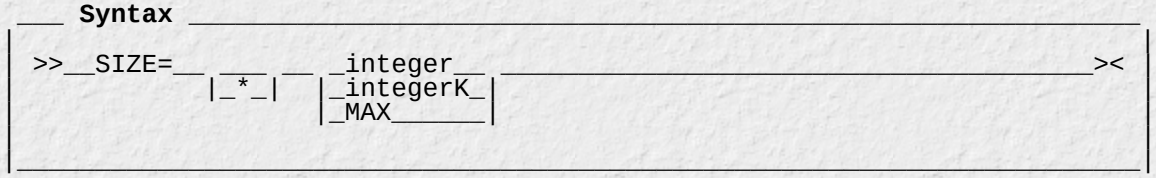
2. Do not use SIZE=MAX if, when you invoke the compiler, you require it to leave a specific amount of unused storage available in the partition.

3. SIZE=MAX in Extended Architecture allows the compiler to obtain all available above-the-line storage in the partition and below-the-line storage for work file buffers and compiler modules that must be below the 16-megabyte line. This occurs unless tuning is done for the Extended Architecture environment. Therefore, you may not want to fix this option as SIZE=MAX at installation.

IBM Library Server

1.2.5.45 SIZE

Syntax



Default
SIZE=MAX

integer
Specifies the amount, in bytes, of virtual storage available to the compiler.

The minimum acceptable value is 778240.

integerK
Specifies the amount of virtual storage available to the compiler in 1024-byte (K) increments.

The minimum acceptable value is 760K.

MAX
The compiler will request all available space in the partition GETVIS for use during the compilation. For Extended Architecture, the compiler obtains the largest contiguous block of free storage above the 16-megabyte line.

Notes:

1. Using SIZE=MAX simplifies compiler invocation by eliminating the need to determine a specific value for the SIZE option.

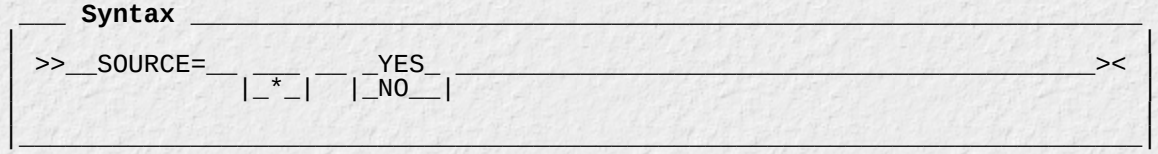
2. Do not use SIZE=MAX if, when you invoke the compiler, you require it to leave a specific amount of unused storage available in the partition.

3. SIZE=MAX in Extended Architecture allows the compiler to obtain all available above-the-line storage in the partition and below-the-line storage for work file buffers and compiler modules that must be below the 16-megabyte line. This occurs unless tuning is done for the Extended Architecture environment. Therefore, you may not want to fix this option as SIZE=MAX at installation.

IBM Library Server

1.2.5.46 SOURCE

Syntax



Default
SOURCE=YES

YES
Indicates that you want a listing of the source statements in the compiler-generated output. This listing also includes any statements embedded by COPY.

NO
Source statements will not appear in the output.

Notes:

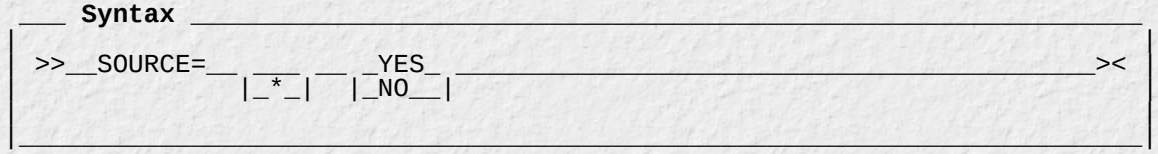
1. Unless the installation default value of the SOURCE option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LIST option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE SOURCE option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

2. The SOURCE compiler option must be in effect at compile time if you want embedded messages in the source listing.

IBM Library Server

1.2.5.46 SOURCE

Syntax



Default
SOURCE=YES

YES
Indicates that you want a listing of the source statements in the compiler-generated output. This listing also includes any statements embedded by COPY.

NO
Source statements will not appear in the output.

Notes:

1. Unless the installation default value of the SOURCE option is fixed by prefixing the value with an asterisk (*), the value will be overridden at compile time by the setting of the LIST option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE SOURCE option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

2. The SOURCE compiler option must be in effect at compile time if you want embedded messages in the source listing.

IBM Library Server

1.2.5.47 SPACE

Syntax

Default
SPACE=1

1
Indicates that you want single spacing for the source statement listing.

2
Indicates that you want double spacing for the source statement listing.

3
Indicates that you want triple spacing for the source statement listing.

IBM Library Server

1.2.5.47 SPACE

Syntax

Default
SPACE=1

1
Indicates that you want single spacing for the source statement listing.

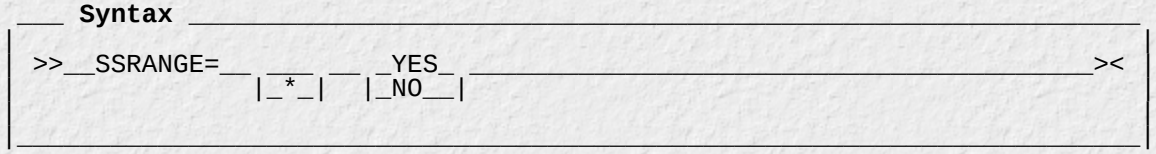
2
Indicates that you want double spacing for the source statement listing.

3
Indicates that you want triple spacing for the source statement listing.

IBM Library Server

1.2.5.48 SSRANGE

Syntax



Default
SSRANGE=NO

YES
At compile time, generates code that checks subscripts, reference modifications, variable-length group ranges, and indexes in the source program at run time to ensure that they do not refer to storage outside the area assigned. It also verifies that a table with ALL subscripting, specified as a function argument, will contain at least one occurrence in the table.

The generated code also checks that a variable-length item does not exceed its defined maximum length as a result of incorrect setting of the OCCURS DEPENDING ON object.

Performance Consideration: If SSRANGE=YES at compile time, object code size will be increased and there will be an increase in run-time overhead to accomplish the range checking.

NO
No code is generated to perform subscript or index checking at run time.

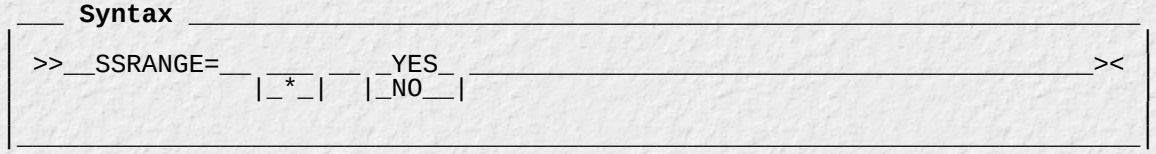
Notes:

1. If the SSRANGE option is in effect at compile time, the range-checking code is generated. Range-checking can be inhibited at run time by specifying the LE/VSE run-time option CHECK(OFF). However, the range-checking code still requires overhead and is dormant within the object code. The range-checking code can then be used optionally to aid in resolving any unexpected errors without recompilation.

IBM Library Server

1.2.5.48 SSRANGE

Syntax



Default
SSRANGE=NO

YES
At compile time, generates code that checks subscripts, reference modifications, variable-length group ranges, and indexes in the source program at run time to ensure that they do not refer to storage outside the area assigned. It also verifies that a table with ALL subscripting, specified as a function argument, will contain at least one occurrence in the table.

The generated code also checks that a variable-length item does not exceed its defined maximum length as a result of incorrect setting of the OCCURS DEPENDING ON object.

Performance Consideration: If SSRANGE=YES at compile time, object code size will be increased and there will be an increase in run-time overhead to accomplish the range checking.

NO
No code is generated to perform subscript or index checking at run time.

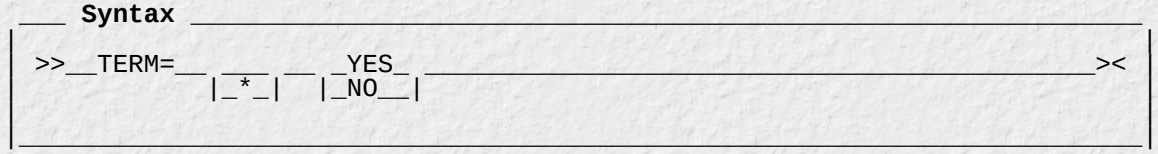
Notes:

1. If the SSRANGE option is in effect at compile time, the range-checking code is generated. Range-checking can be inhibited at run time by specifying the LE/VSE run-time option CHECK(OFF). However, the range-checking code still requires overhead and is dormant within the object code. The range-checking code can then be used optionally to aid in resolving any unexpected errors without recompilation.

IBM Library Server

1.2.5.49 TERM

Syntax



Default
TERM=NO

YES
Specifies that the progress and diagnostic messages are to be sent to the SYSLOG file.

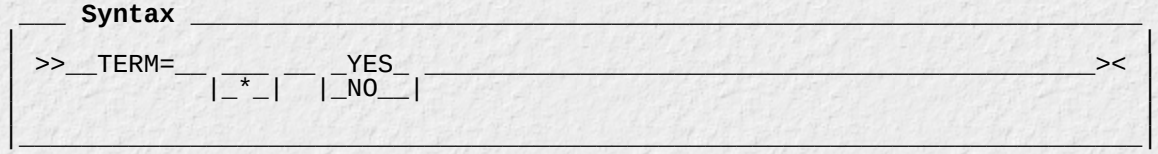
NO
Specifies that no messages are to be sent to SYSLOG.

Note: Unless the installation default value of the TERM option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the TERM option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE TERM option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.49 TERM

Syntax



Default
TERM=NO

YES
Specifies that the progress and diagnostic messages are to be sent to the SYSLOG file.

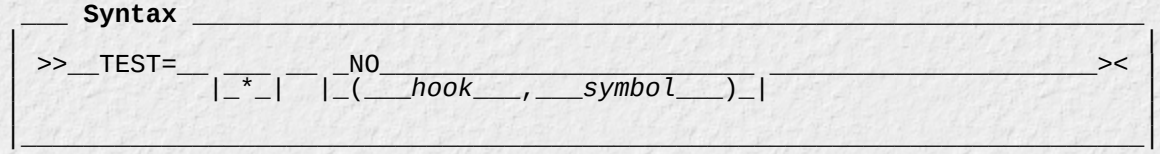
NO
Specifies that no messages are to be sent to SYSLOG.

| **Note:** Unless the installation default value of the TERM option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the TERM option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE TERM option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.50 TEST

Syntax



Default
TEST=NO

Other than NO
Produces object code containing symbol table information that is used by LE/VSE to produce a formatted dump, and hooks for Debug Tool/VSE.

You must specify options for both the hook-location (*hook*) and the symbol table (*symbol*) values.

hook values:

ALL Activates the generation of all compiled-hooks. Hooks will be generated at all statements, all path points, and at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

NONE Suppresses the generation of all compiled-hooks. TEST(NONE) is compatible with the OPT compiler option.

STMT Hooks will be compiled at every statement and label, as well as at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

PATH Hooks will be compiled at all path points, including program entry and exit points.

BLOCK Hooks will be compiled at all program entry and exit points.

symbol values:

SYM Activates inclusion of symbolic dictionary information in your object program.

NOSYM Deactivates inclusion of symbolic dictionary information in your object program.

Performance Consideration: Because TEST=(a hook-location suboption other than NONE) generates additional code, it can cause significant performance degradation at run time when used in a production environment.

NO
Produces object code that does not contain the symbol table information that is used by LE/VSE to produce a formatted dump, and the hooks for Debug Tool/VSE.

Notes:

1. If you specify TEST= (other than NONE for the hook-location suboption), the following option is put into effect at compilation time:

   OPT=NO

2. Only TEST(NONE) is compatible with the OPT compiler option. See "Conflicting Compiler Options" in topic 1.2.5.3 for more conflict resolution information.

3. Programmers might notice an increase in run-time for programs compiled with any hook-location suboption other than NONE in effect.

4. A date processing statement is any statement that references a date field, or any EVALUATE or SEARCH statement WHEN phrase that references a date field.

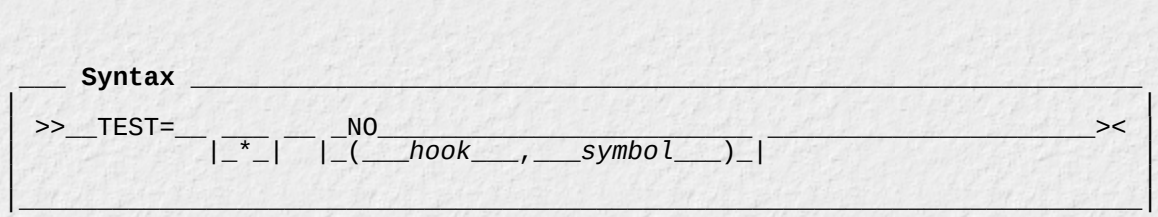
5. To get the full capability of Debug Tool/VSE, compile your program with TEST(ALL,SYM).

6. For production programs compile your programs with TEST(NONE,NOSYM) if you do not want to increase the size of your production modules, but still get minimum debugging capability.

IBM Library Server

1.2.5.50 TEST

Syntax



Default
TEST=NO

Other than NO
Produces object code containing symbol table information that is used by LE/VSE to produce a formatted dump, and hooks for Debug Tool/VSE.

You must specify options for both the hook-location (*hook*) and the symbol table (*symbol*) values.

hook values:

ALL Activates the generation of all compiled-hooks. Hooks will be generated at all statements, all path points, and at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

NONE Suppresses the generation of all compiled-hooks. TEST(NONE) is compatible with the OPT compiler option.

STMT Hooks will be compiled at every statement and label, as well as at all program entry and exit points. In addition, if either the DATEPROC=FLAG option or DATEPROC=NOFLAG option is in effect, hooks will be generated for all date processing statements.

PATH Hooks will be compiled at all path points, including program entry and exit points.

BLOCK Hooks will be compiled at all program entry and exit points.

symbol values:

SYM Activates inclusion of symbolic dictionary information in your object program.

NOSYM Deactivates inclusion of symbolic dictionary information in your object program.

Performance Consideration: Because TEST=(a hook-location suboption other than NONE) generates additional code, it can cause significant performance degradation at run time when used in a production environment.

NO
Produces object code that does not contain the symbol table information that is used by LE/VSE to produce a formatted dump, and the hooks for Debug Tool/VSE.

Notes:

1. If you specify TEST= (other than NONE for the hook-location suboption), the following option is put into effect at compilation time:

   OPT=NO

2. Only TEST(NONE) is compatible with the OPT compiler option. See "Conflicting Compiler Options" in topic 1.2.5.3 for more conflict resolution information.

3. Programmers might notice an increase in run-time for programs compiled with any hook-location suboption other than NONE in effect.

4. A date processing statement is any statement that references a date field, or any EVALUATE or SEARCH statement WHEN phrase that references a date field.

5. To get the full capability of Debug Tool/VSE, compile your program with TEST(ALL,SYM).

6. For production programs compile your programs with TEST(NONE,NOSYM) if you do not want to increase the size of your production modules, but still get minimum debugging capability.

IBM Library Server

1.2.5.51 TRUNC

Syntax

Default
TRUNC=STD

STD
Controls the way arithmetic fields are shortened during MOVE and arithmetic operations. The TRUNC option applies only to binary (COMP) receiving fields in MOVE statements and in arithmetic expressions. When TRUNC=STD is in effect, the final intermediate result of an arithmetic expression, or of the sending field in the MOVE statement, is shortened to the number of digits in the PICTURE clause of the binary receiving field.

OPT
The compiler assumes that the data conforms to PICTURE and USAGE specifications. The compiler manipulates the result based on the size of the field in storage (halfword or fullword).

TRUNC=OPT should only be specified when data being moved into binary areas will not have a value with larger precision than that defined by the binary item PICTURE clause. Otherwise, truncation of high-order digits may occur. The truncation results are dependent on the particular code sequence generated and may not necessarily be the same in DOS/VS COBOL and VS COBOL II.

BIN
Specifies that:

1. Output binary fields will be shortened only at the S/370* half/full/double word boundaries, rather than at COBOL base 10 picture limits.

2. Input binary fields will be treated as S/370 half/full/double words, and no assumption will be made that the values are limited to those implied by the base 10 PICTURE clause.

3. DISPLAY will convert and output the full content of binary fields with no truncation to the PICTURE description.

Performance Consideration: Using TRUNC=OPT does not generate extra code and generally improves performance. However, both TRUNC=BIN and TRUNC=STD generate extra code whenever a BINARY data item is changed. TRUNC=BIN is usually the slowest of these options.

Notes:

1. TRUNC=STD supports the COBOL 85 Standard.

2. TRUNC=STD and TRUNC=OPT are equivalent to TRUNC=YES and TRUNC=NO (respectively) in VS COBOL II Release 2.

3. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether the COBOL/VSE source programs assume a particular setting for correct running.

4. TRUNC=BIN is the recommended option when interfacing with other products that have S/370-format binary data (such as CICS, SQL/DS*, FORTRAN, and PL/I). This is especially true if there is a possibility of having more than 9 digits in a fullword or more than 4 digits in a halfword.

IBM Library Server

1.2.5.51 TRUNC

Syntax

Default
TRUNC=STD

STD
Controls the way arithmetic fields are shortened during MOVE and arithmetic operations. The TRUNC option applies only to binary (COMP) receiving fields in MOVE statements and in arithmetic expressions. When TRUNC=STD is in effect, the final intermediate result of an arithmetic expression, or of the sending field in the MOVE statement, is shortened to the number of digits in the PICTURE clause of the binary receiving field.

OPT
The compiler assumes that the data conforms to PICTURE and USAGE specifications. The compiler manipulates the result based on the size of the field in storage (halfword or fullword).

TRUNC=OPT should only be specified when data being moved into binary areas will not have a value with larger precision than that defined by the binary item PICTURE clause. Otherwise, truncation of high-order digits may occur. The truncation results are dependent on the particular code sequence generated and may not necessarily be the same in DOS/VS COBOL and VS COBOL II.

BIN
Specifies that:

1. Output binary fields will be shortened only at the S/370* half/full/double word boundaries, rather than at COBOL base 10 picture limits.

2. Input binary fields will be treated as S/370 half/full/double words, and no assumption will be made that the values are limited to those implied by the base 10 PICTURE clause.

3. DISPLAY will convert and output the full content of binary fields with no truncation to the PICTURE description.

Performance Consideration: Using TRUNC=OPT does not generate extra code and generally improves performance. However, both TRUNC=BIN and TRUNC=STD generate extra code whenever a BINARY data item is changed. TRUNC=BIN is usually the slowest of these options.

Notes:

1. TRUNC=STD supports the COBOL 85 Standard.

2. TRUNC=STD and TRUNC=OPT are equivalent to TRUNC=YES and TRUNC=NO (respectively) in VS COBOL II Release 2.

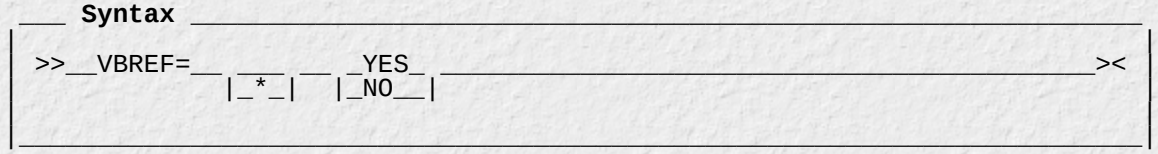
3. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether the COBOL/VSE source programs assume a particular setting for correct running.

4. TRUNC=BIN is the recommended option when interfacing with other products that have S/370-format binary data (such as CICS, SQL/DS*, FORTRAN, and PL/I). This is especially true if there is a possibility of having more than 9 digits in a fullword or more than 4 digits in a halfword.

IBM Library Server

1.2.5.52 VBREF

Syntax



YES
Produces a cross-reference of all verb types in a source program to the line numbers at which they were found. VBREF=YES also produces a summary of how many times each verb was used in the program.

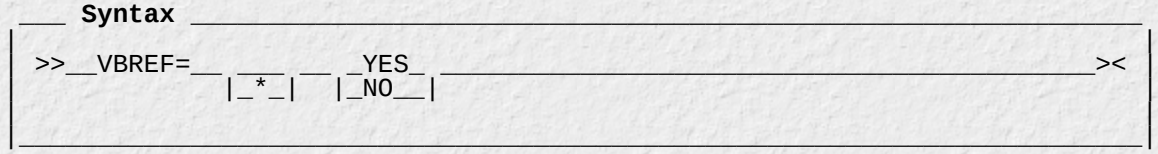
NO
Does not produce a cross-reference or verb summary listing.

Default
VBREF=NO

IBM Library Server

1.2.5.52 VBREF

Syntax



YES
Produces a cross-reference of all verb types in a source program to the line numbers at which they were found. VBREF=YES also produces a summary of how many times each verb was used in the program.

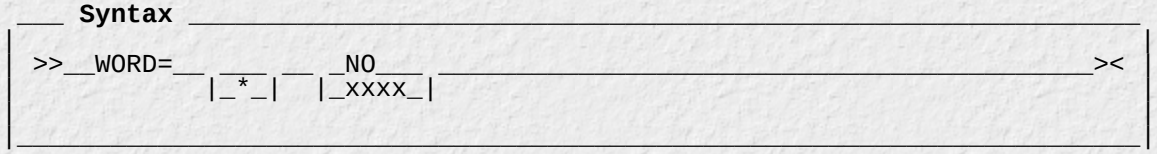
NO
Does not produce a cross-reference or verb summary listing.

Default
VBREF=NO

IBM Library Server

1.2.5.53 WORD

Syntax



Default
WORD=*NO

NO
Indicates that no alternative reserved word table is to be used as the default.

xxxx
Specifies an alternative default reserved word table to be used during compilation. **xxxx** represents the ending characters (may be 1 to 4 characters in length) of the name of the reserved word table to be used. The first 4 characters are IGYC. The last 4 characters may not be any one of the character strings listed below, nor may any of them contain the dollar sign character ($).

ASM1	LIBO	LVL8	RDSC
ASM2	LIBR	OPTM	RWT
DIAG	LSTR	OSCN	SAW
DMAP	LVL0	PGEN	SCAN
DOPT	LVL1	RCTL	SIMD
FGEN	LVL2	RDPR	XREF
INIT	LVL3		

Notes:

1. The default for the WORD option is specified with an asterisk. When the option is installed with the default (WORD=*NO), the application programmer cannot replace the option at compile time to specify an alternative reserved word table.

2. WORD=NO supports the COBOL 85 Standard.

3. Specification of WORD affects the interpretation of input reserved words. System names (such as UPSI and SYSPCH) and the intrinsic function names should not be used as aliases for reserved words. If a function name is specified as an alias via the reserved word table ABBR control-statement, that function name will be recognized and diagnosed by the compiler as a reserved word and the intrinsic function will not be performed.

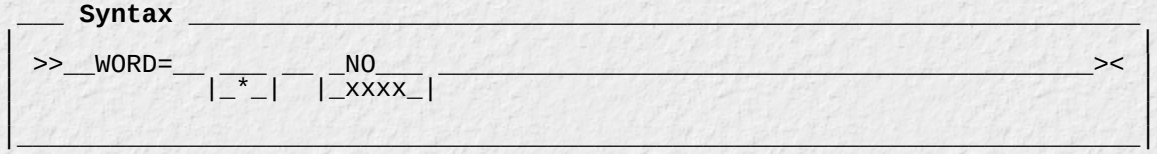
4. Changing the default value of the WORD option could cause compiler option conflicts. See "Conflicting Compiler Options" in topic 1.2.5.3 for information on conflict resolution.

5. A CICS-specific reserved word table is provided as an alternate reserved word table. For a description, see "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2.

IBM Library Server

1.2.5.53 WORD

Syntax



Default
WORD=*NO

NO
Indicates that no alternative reserved word table is to be used as the default.

xxxx
Specifies an alternative default reserved word table to be used during compilation. **xxxx** represents the ending characters (may be 1 to 4 characters in length) of the name of the reserved word table to be used. The first 4 characters are IGYC. The last 4 characters may not be any one of the character strings listed below, nor may any of them contain the dollar sign character ($).

ASM1	LIBO	LVL8	RDSC
ASM2	LIBR	OPTM	RWT
DIAG	LSTR	OSCN	SAW
DMAP	LVL0	PGEN	SCAN
DOPT	LVL1	RCTL	SIMD
FGEN	LVL2	RDPR	XREF
INIT	LVL3		

Notes:

1. The default for the WORD option is specified with an asterisk. When the option is installed with the default (WORD=*NO), the application programmer cannot replace the option at compile time to specify an alternative reserved word table.

2. WORD=NO supports the COBOL 85 Standard.

3. Specification of WORD affects the interpretation of input reserved words. System names (such as UPSI and SYSPCH) and the intrinsic function names should not be used as aliases for reserved words. If a function name is specified as an alias via the reserved word table ABBR control-statement, that function name will be recognized and diagnosed by the compiler as a reserved word and the intrinsic function will not be performed.

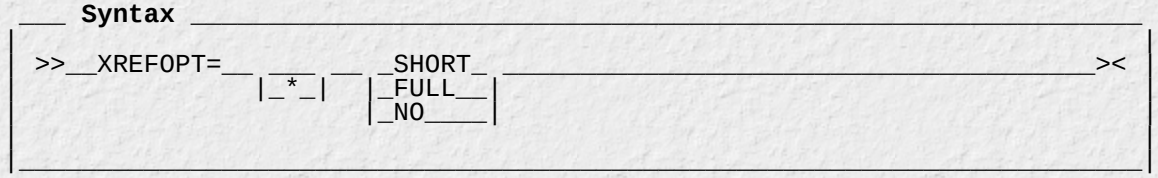
4. Changing the default value of the WORD option could cause compiler option conflicts. See "Conflicting Compiler Options" in topic 1.2.5.3 for information on conflict resolution.

5. A CICS-specific reserved word table is provided as an alternate reserved word table. For a description, see "CICS Reserved Word Table (IGYCCICS)" in topic 1.2.4.3.2.

IBM Library Server

1.2.5.54 XREFOPT

Syntax



Default
XREFOPT=NO

SHORT
Produces only the explicitly referenced variables in the cross-reference listing.

FULL
Produces both a sorted and embedded cross-reference listing. If SOURCE=YES is also specified, the listing line numbers cross-reference recurrences of particular data-names.

NO
Suppresses the cross-reference listing.

Notes:

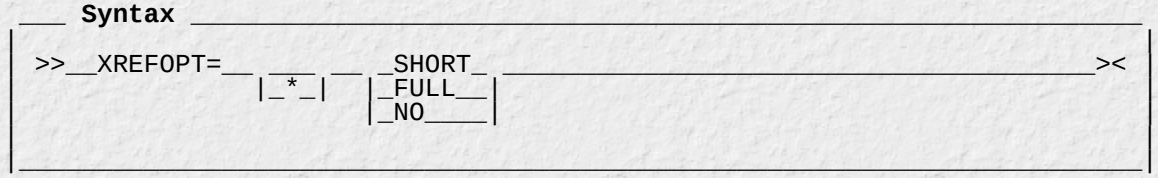
1. The XREFOPT option sets the default value for the compiler option XREF.

Note: Unless the installation default value of the XREFOPT option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the XREF or SXREF option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE XREF option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.54 XREFOPT

Syntax



Default
XREFOPT=NO

SHORT
Produces only the explicitly referenced variables in the cross-reference listing.

FULL
Produces both a sorted and embedded cross-reference listing. If SOURCE=YES is also specified, the listing line numbers cross-reference recurrences of particular data-names.

NO
Suppresses the cross-reference listing.

Notes:

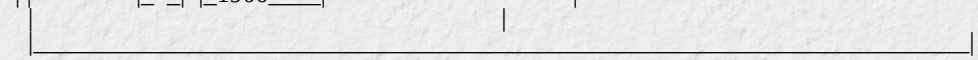
1. The XREFOPT option sets the default value for the compiler option XREF.

Note: Unless the installation default value of the XREFOPT option is fixed by prefixing the value with *, the value will be overridden at compile time by the setting of the XREF or SXREF option of the VSE STDOPT command. To override the STDOPT settings, the required COBOL/VSE XREF option must be specified at compile time in a PROCESS or CBL compiler statement, or in the PARM parameter of the JCL EXEC statement. For information about the precedence of compiler options, see *COBOL/VSE Programming Guide*.

IBM Library Server

1.2.5.55 YRWINDOW

Syntax



Default
YRWINDOW=1900

integer
Specifies the first year of the 100-year window used by COBOL windowed date fields, and may be one of the following:

- An unsigned decimal integer from 1900 through 1999.
- A negative decimal integer from -1 through -99, representing an offset from the current year at run time. The current year is determined for each compilation unit when it is first initialized, or re-initialized after execution of a CANCEL statement that refers to the compilation unit.

Notes:

1. YRWINDOW has no effect unless DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. At run-time, two conditions must be true:

   a. The 100-year window must have its beginning year in the 1900s.

   b. The current year must lie within the 100-year window for the compilation unit.

3. All windowed dates have a year relative to the base year. For example, if the base year were specified as 1965, all windowed year values would be interpreted as years within the 100-year window of 1965 to 2064, inclusive. So, a windowed year value of 67 would represent the year 1967, whereas a windowed year value of 05 would represent the year 2005.

   If the base year were specified as -30, then the window would depend on when the application were run. Running it during 1998 would give a 100-year window of 1968 through 2067. So, a windowed year value of 68 would represent the year 1968, whereas a windowed year value of 67

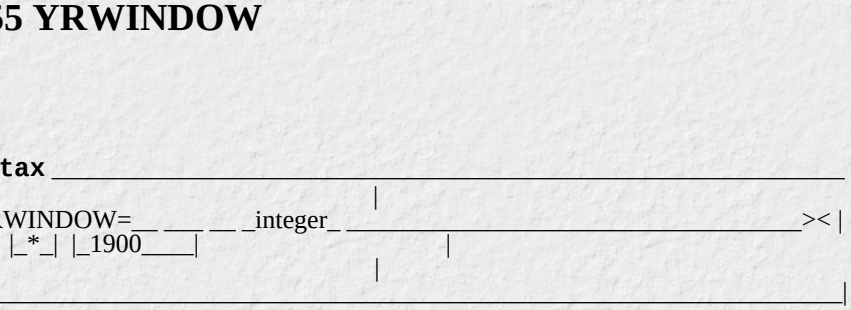
would represent the year 2067.

4. The YRWINDOW installation option is equivalent to the YEARWINDOW compile-time option.

IBM Library Server

1.2.5.55 YRWINDOW

Syntax



Default
YRWINDOW=1900

integer
Specifies the first year of the 100-year window used by COBOL windowed date fields, and may be one of the following:

- An unsigned decimal integer from 1900 through 1999.
- A negative decimal integer from -1 through -99, representing an offset from the current year at run time. The current year is determined for each compilation unit when it is first initialized, or re-initialized after execution of a CANCEL statement that refers to the compilation unit.

Notes:

1. YRWINDOW has no effect unless DATEPROC=FLAG or DATEPROC=NOFLAG is specified.

2. At run-time, two conditions must be true:

   a. The 100-year window must have its beginning year in the 1900s.

   b. The current year must lie within the 100-year window for the compilation unit.

3. All windowed dates have a year relative to the base year. For example, if the base year were specified as 1965, all windowed year values would be interpreted as years within the 100-year window of 1965 to 2064, inclusive. So, a windowed year value of 67 would represent the year 1967, whereas a windowed year value of 05 would represent the year 2005.

   If the base year were specified as -30, then the window would depend on when the application were run. Running it during 1998 would give a 100-year window of 1968 through 2067. So, a windowed year value of 68 would represent the year 1968, whereas a windowed year value of 67

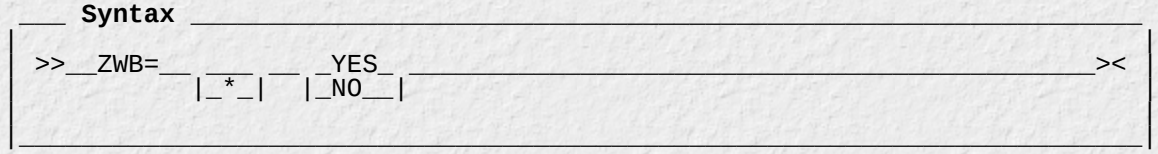
would represent the year 2067.

4. The YRWINDOW installation option is equivalent to the YEARWINDOW compile-time option.

IBM Library Server

1.2.5.56 ZWB

Syntax



Default
ZWB=YES

YES
Instructs the compiler to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

NO
Instructs the compiler not to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

Notes:

1. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether your COBOL/VSE source programs assume a particular setting to run correctly.

2. ZWB=YES supports the COBOL 85 Standard.

3. Application programmers use ZWB=NO to test input numeric fields for SPACES.

IBM Library Server

1.3.1 Installation Overview

The following table provides an overview of the installation steps.

Step	Description	Topic
1	Backup the Original System	1.3.2.1
2	Allocate Space for the Library	1.3.2.2
3a	Install COBOL/VSE using the Interactive Interface	1.3.2.3.1
3b	Install COBOL/VSE using a batch job	1.3.2.3.2
4	Verify the COBOL/VSE Installation	1.3.2.4

IBM Library Server

1.3.2 Procedure for Installing COBOL/VSE

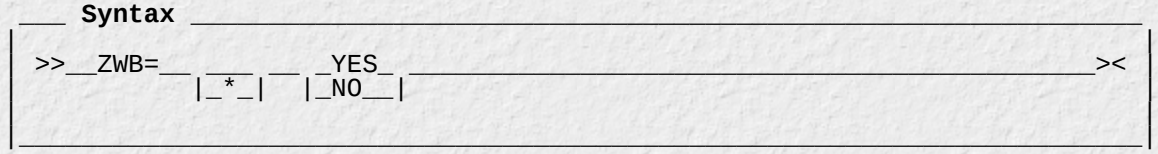
The following sections detail the installation steps outlined in "Installation Overview."

Subtopics:

IBM Library Server

1.2.5.56 ZWB

Syntax



Default
ZWB=YES

YES
Instructs the compiler to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

NO
Instructs the compiler not to remove the sign from a signed external decimal (DISPLAY) field when comparing this field to an alphanumeric field during run time.

Notes:

1. Setting this option affects program run-time logic; that is, the same COBOL source program can give different results, depending on the option setting. Verify whether your COBOL/VSE source programs assume a particular setting to run correctly.

2. ZWB=YES supports the COBOL 85 Standard.

3. Application programmers use ZWB=NO to test input numeric fields for SPACES.

IBM Library Server

1.3 Chapter 3. Installing COBOL/VSE

This chapter provides the following information:

- Overview of the installation procedure
- Procedure for installing COBOL/VSE

Note: Installing COBOL/VSE requires the use of the Maintain System History Program (MSHP).

Subtopics:

1.3.2.3.1 Step 3a: Install COBOL/VSE using the Interactive Interface

VSE/ESA provides dialogs (Interactive Interface) to install IBM licensed programs.

To install COBOL/VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (Additional information about the functions of the Interactive Interface are in *VSE/ESA Administration*, and for further information about installing licensed programs using the Interactive Interface see *VSE/ESA Installation and Service*.)

Mount the COBOL/VSE tape on an available tape drive.

In the following menus specify the **highlighted** items that appear after the ===> symbol.

1. **VSE/ESA FUNCTION SELECTION** menu:
   ===> ***1*** (Installation)

2. **INSTALLATION** menu:

   ===> ***1*** (Install Programs - Stacked V2 Format)

INSTALL PROGRAMS - STACKED V2 FORMAT menu:
===> ***1*** (Prepare for Installation (Stacked Tapes Only))

Note: Despite the comment which says you can only use option 1 for stacked tapes, you can use option 1 for both a stacked distribution tape containing one or more optional licensed programs, and a nonstacked tape containing just COBOL/VSE.

PREPARE FOR INSTALLATION (STACKED TAPES ONLY) menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the distribution tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job.

The output listing from this job will give a list of the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifiers for COBOL/VSE are COB.BASE...1.1.0, COB.ENU....1.1.0, and COB.JPN....1.1.0.

The program identifiers of the optional programs on the distribution

tape are also automatically entered on the **INSTALL PRODUCT(S) FROM TAPE** menu.

Return to the **VSE/ESA FUNCTION SELECTION** menu:
===> ***1*** (Installation)

INSTALLATION menu:
===> ***1*** (Install Programs - Stacked V2 Format)

INSTALL PROGRAMS - STACKED V2 FORMAT menu:
===> ***2*** (Install Product(s) from Tape)

INSTALL PRODUCT(S) FROM TAPE menu:

Enter **1** (install) in the OPT field against each of the identifiers below that you intend to install:

COB.BASE...1.1.0 (COBOL/VSE)

COB.ENU....1.1.0 (COBOL/VSE US English Language Feature)

COB.JPN....1.1.0 (COBOL/VSE Japanese Language Feature)

and **2** (skip installation) against any other optional products you do not intend to install at this time.

If you did not use the default library PRD2.PROD, enter the name of your library and sublibrary on this screen.

Press PF5 to generate the installation job.

VSE/ESA INSTALL PRODUCT(S) TAPE SPECIFICATION menu:

===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the COBOL/VSE tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job and install COBOL/VSE.

Check the output from the batch job created by the previous steps to ensure that the installation was successful. If you are installing COBOL/VSE from a nonstacked tape, you will receive the following message:
IESI0083I TAPE IS NOT PID-V2-STACKED

This message is for information only and can be ignored. The Librarian RESTORE job ends with a return code of 4. This is not an error: Installation was successful.

IBM Library Server

1.3 Chapter 3. Installing COBOL/VSE

This chapter provides the following information:

- Overview of the installation procedure
- Procedure for installing COBOL/VSE

Note: Installing COBOL/VSE requires the use of the Maintain System History Program (MSHP).

Subtopics:

IBM Library Server

1.3.1 Installation Overview

The following table provides an overview of the installation steps.

Step	Description	Topic
1	Backup the Original System	1.3.2.1
2	Allocate Space for the Library	1.3.2.2
3a	Install COBOL/VSE using the Interactive Interface	1.3.2.3.1
3b	Install COBOL/VSE using a batch job	1.3.2.3.2
4	Verify the COBOL/VSE Installation	1.3.2.4

IBM Library Server

1.3.2.1 Step 1: Backup the Original System

Make a backup copy of your original system, including the system history file. For more information about backing up your VSE system, see *VSE/ESA System Control Statements*.

1.3.2.2 Step 2: Allocate Space for the Library

By default COBOL/VSE is installed into the PRD2.PROD sublibrary. If you decide to install COBOL/VSE into a different sublibrary proceed with this step. If you decide to install COBOL/VSE in the default sublibrary, proceed to "Step 3a: Install COBOL/VSE using the Interactive Interface" in topic 1.3.2.3.1 or "Step 3b: Install COBOL/VSE using a Batch Job" in topic 1.3.2.3.2. Once you have determined how much space you require, decide where to allocate space for the library. You can allocate space for the library in non-VSAM space or in VSAM space.

- Non-VSAM

  Identify, on the disk volume (or volumes) to be used for the library, suitable areas of free space. To do this, list the volume table of contents (VTOC) of the disk or disks to be used. The sample job shown in Figure 2 illustrates the JCL needed to list the VTOC for the volume with serial number COBVOL.

```
// JOB COBVSE  LIST VTOC for COBVOL
// ASSGN SYS004,DISK,TEMP,VOL=COBVOL,SHR
// ASSGN SYS005,SYSLST
// EXEC LVTOC
/*
/&
```

Figure 2. Listing the Contents of a DASD Volume

- VSAM

  Examine the user catalog to check for enough VSAM space. To do this, use the access method services (IDCAMS) LISTCAT command to provide a report showing the space available. The sample job shown in Figure 3 illustrates the JCL needed to display the space use in the VSAM user catalog supplied with VSE/ESA.

```
// JOB IDCAMS  SHOW USER CATALOG SPACE
// EXEC IDCAMS,SIZE=AUTO
 LISTCAT SPACE ALL CATALOG(VSESP.USER.CATALOG)
/*
/&
```

Figure 3. Listing the Space in a VSAM Catalog

If you intend to use VSAM space for the COBOL/VSE library, see *VSE/ESA Guide to System Functions* for information about how to define the required VSAM cluster. VSE/ESA also provides dialogs (Interactive Interface) for these tasks. Refer to *VSE/ESA Administration* for details.

Once you have determined where to allocate space for the library, you can define the library using the JCL shown in Figure 4.

```
// JOB LIBRDEF CREATE A LIBRARY
// OPTION LOG
*  Label for the Product Library as a Non-VSAM file  1
// DLBL COBVSE,'COBOL.VSE.LIBRARY',99/365,SD
// EXTENT SYS002,volid,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volid,SHR
*  Label for the Product Library as a VSAM file      2
// DLBL COBVSE,'COBOL.VSE.LIBRARY',,VSAM,CAT=catname
* --------------------------------------
*  This step defines the Product Library             3
* --------------------------------------
// EXEC LIBR
 DELETE LIB=COBVSE
 DEFINE LIB=COBVSE,REPLACE=NO
/*
/&
```

Figure 4. Job to Allocate the COBOL/VSE Library Space

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. If you intend to allocate space for the COBOL/VSE library in non-VSAM space, delete the DLBL statement in area **2** of the sample job, and make the following changes in area **1** :

   a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

   b. Change the variable *volid* in the EXTENT statement to the volume serial number of the volume that will hold your library.

   c. Change the variable *rtrk* in the EXTENT ASSGN statements to the start location (track or block number) of the extent.

d. Change the variable *ntrk* in the EXTENT statement to the number of tracks or blocks required for the library.

4. If you intend to allocate space for the COBOL/VSE library in VSAM space, delete the DLBL, EXTENT and ASSGN statements in area **1** of the sample job, and make the following changes in area **2**:

a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

b. Change the variable *catname* to the file name of the VSAM catalog that will own the library. You can omit *catname* (and the CAT keyword and preceding comma) if the library is to be cataloged in the IJSYSUC user catalog.

The Librarian job step in area **3** includes a DELETE statement before the DEFINE statement, so the job can be rerun. This means the following messages will be issued when the job runs for the first time. These may be ignored and the job will continue to allocate the library.

```
L101I  LIBRARY COBVSE DOES NOT EXIST    <--------------for Non-VSAM
L255I  LIBRARY COBVSE - OPEN FAILURE    <--------------for VSAM
L027I  ABNORMAL END DURING DELETE COMMAND PROCESSING
L113I  RETURN CODE OF DELETE IS 8
```

IBM Library Server

1.3.2.3 Step 3: Install COBOL/VSE

You can use the Interactive Interface (Step 3a) or a batch job (Step 3b) to install COBOL/VSE.

Depending on how you ordered the COBOL/VSE product, you will receive one of the following types of distribution tape:

- A distribution tape containing only the COBOL/VSE product
- A VSE stacked tape containing one or more optional products

The installation methods shown will handle both types of distribution tape.

Continue with either Step 3a or Step 3b.

Subtopics:

1.3.2.4 Step 4: Verify the COBOL/VSE Installation

The installation verification program IGYWEIVP.C is a sample program provided on the distribution tape. It allows you to check that your installation is successful by exercising representative features of COBOL/VSE.

1. Job **CALLIVP1**: The JCL (IGYWEIN1.Z) and the source (IGYWEIVP.C) to run the verification program CALLIVP1 are supplied. They test the compiler and library, and verify the product installation.

   Before you run the JCL from member IGYWEIN1.Z, do the following:

   a. Modify the JOB statement to meet the requirements of your site.

   b. Add POWER JECL statements if wanted.

   c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

   d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

      Six compiler work files are always required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

   e. The JCL provided in IGYWEIN1.Z, without modification to the EXEC statement, compiles, link-edits, and runs the program. If you have not installed LE/VSE you must remove the two steps at the end of the job that link-edit and run the program.

   f. Execute the modified JCL to run the installation verification program.

   After you run job CALLIVP1 you should receive the following messages, printed on SYSLST:

   ```
   ***** START OF CALLIVP1 *****
   ***** CALLIVP1 SUCCESSFUL *****
   ```

2. Job **ERRMSG**: The JCL (IGYWEIN2.Z) and source (IGYWEERM.C) are used to run the verification program ERRMSG. This job will verify the operation of the compiler, and produce a complete list of compiler messages.

   Before you run the JCL from member IGYWEIN2.Z, do the following:

a. Modify the JOB statement to meet the requirements of your site.

b. Add POWER JECL statements if wanted.

c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

   Six compiler work files are required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

e. Execute the modified JCL to run the ERRMSG program.

After you run job ERRMSG, you will receive a listing of COBOL/VSE compiler messages.

IBM Library Server

1.3.1 Installation Overview

The following table provides an overview of the installation steps.

Step	Description	Topic
1	Backup the Original System	1.3.2.1
2	Allocate Space for the Library	1.3.2.2
3a	Install COBOL/VSE using the Interactive Interface	1.3.2.3.1
3b	Install COBOL/VSE using a batch job	1.3.2.3.2
4	Verify the COBOL/VSE Installation	1.3.2.4

IBM Library Server

1.3.2 Procedure for Installing COBOL/VSE

The following sections detail the installation steps outlined in "Installation Overview."

Subtopics:

IBM Library Server

1.3.2 Procedure for Installing COBOL/VSE

The following sections detail the installation steps outlined in "Installation Overview."

Subtopics:

IBM Library Server

1.3.2.1 Step 1: Backup the Original System

Make a backup copy of your original system, including the system history file. For more information about backing up your VSE system, see *VSE/ESA System Control Statements*.

IBM Library Server

1.3.2.1 Step 1: Backup the Original System

Make a backup copy of your original system, including the system history file. For more information about backing up your VSE system, see *VSE/ESA System Control Statements*.

IBM Library Server

1.3.2.2 Step 2: Allocate Space for the Library

By default COBOL/VSE is installed into the PRD2.PROD sublibrary. If you decide to install COBOL/VSE into a different sublibrary proceed with this step. If you decide to install COBOL/VSE in the default sublibrary, proceed to "Step 3a: Install COBOL/VSE using the Interactive Interface" in topic 1.3.2.3.1 or "Step 3b: Install COBOL/VSE using a Batch Job" in topic 1.3.2.3.2. Once you have determined how much space you require, decide where to allocate space for the library. You can allocate space for the library in non-VSAM space or in VSAM space.

- Non-VSAM

  Identify, on the disk volume (or volumes) to be used for the library, suitable areas of free space. To do this, list the volume table of contents (VTOC) of the disk or disks to be used. The sample job shown in Figure 2 illustrates the JCL needed to list the VTOC for the volume with serial number COBVOL.

```
// JOB COBVSE  LIST VTOC for COBVOL
// ASSGN SYS004,DISK,TEMP,VOL=COBVOL,SHR
// ASSGN SYS005,SYSLST
// EXEC LVTOC
/*
/&
```

Figure 2. Listing the Contents of a DASD Volume

- VSAM

  Examine the user catalog to check for enough VSAM space. To do this, use the access method services (IDCAMS) LISTCAT command to provide a report showing the space available. The sample job shown in Figure 3 illustrates the JCL needed to display the space use in the VSAM user catalog supplied with VSE/ESA.

```
// JOB IDCAMS  SHOW USER CATALOG SPACE
// EXEC IDCAMS,SIZE=AUTO
 LISTCAT SPACE ALL CATALOG(VSESP.USER.CATALOG)
/*
/&
```

Figure 3. Listing the Space in a VSAM Catalog

If you intend to use VSAM space for the COBOL/VSE library, see *VSE/ESA Guide to System Functions* for information about how to define the required VSAM cluster. VSE/ESA also provides dialogs (Interactive Interface) for these tasks. Refer to *VSE/ESA Administration* for details.

Once you have determined where to allocate space for the library, you can define the library using the JCL shown in Figure 4.

```
// JOB LIBRDEF CREATE A LIBRARY
// OPTION LOG
*  Label for the Product Library as a Non-VSAM file  1
// DLBL COBVSE,'COBOL.VSE.LIBRARY',99/365,SD
// EXTENT SYS002,volid,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volid,SHR
*  Label for the Product Library as a VSAM file      2
// DLBL COBVSE,'COBOL.VSE.LIBRARY',,VSAM,CAT=catname
* --------------------------------------
*  This step defines the Product Library             3
* --------------------------------------
// EXEC LIBR
 DELETE LIB=COBVSE
 DEFINE LIB=COBVSE,REPLACE=NO
/*
/&
```

Figure 4. Job to Allocate the COBOL/VSE Library Space

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. If you intend to allocate space for the COBOL/VSE library in non-VSAM space, delete the DLBL statement in area **2** of the sample job, and make the following changes in area **1** :

   a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

   b. Change the variable *volid* in the EXTENT statement to the volume serial number of the volume that will hold your library.

   c. Change the variable *rtrk* in the EXTENT ASSGN statements to the start location (track or block number) of the extent.

d. Change the variable *ntrk* in the EXTENT statement to the number of tracks or blocks required for the library.

4. If you intend to allocate space for the COBOL/VSE library in VSAM space, delete the DLBL, EXTENT and ASSGN statements in area **1** of the sample job, and make the following changes in area **2**:

a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

b. Change the variable *catname* to the file name of the VSAM catalog that will own the library. You can omit *catname* (and the CAT keyword and preceding comma) if the library is to be cataloged in the IJSYSUC user catalog.

The Librarian job step in area **3** includes a DELETE statement before the DEFINE statement, so the job can be rerun. This means the following messages will be issued when the job runs for the first time. These may be ignored and the job will continue to allocate the library.

```
L101I  LIBRARY COBVSE DOES NOT EXIST    <----------for Non-VSAM
L255I  LIBRARY COBVSE - OPEN FAILURE    <----------for VSAM
L027I  ABNORMAL END DURING DELETE COMMAND PROCESSING
L113I  RETURN CODE OF DELETE IS 8
```

IBM Library Server

1.3.2.3.1 Step 3a: Install COBOL/VSE using the Interactive Interface

VSE/ESA provides dialogs (Interactive Interface) to install IBM licensed programs.

To install COBOL/VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (Additional information about the functions of the Interactive Interface are in *VSE/ESA Administration*, and for further information about installing licensed programs using the Interactive Interface see *VSE/ESA Installation and Service*.)

Mount the COBOL/VSE tape on an available tape drive.

In the following menus specify the **highlighted** items that appear after the ===> symbol.

1. **VSE/ESA FUNCTION SELECTION** menu:
   ===> ***1*** (Installation)

2. **INSTALLATION** menu:

   ===> ***1*** (Install Programs - Stacked V2 Format)

   INSTALL PROGRAMS - STACKED V2 FORMAT menu:
   ===> ***1*** (Prepare for Installation (Stacked Tapes Only))

   Note: Despite the comment which says you can only use option 1 for stacked tapes, you can use option 1 for both a stacked distribution tape containing one or more optional licensed programs, and a nonstacked tape containing just COBOL/VSE.

   PREPARE FOR INSTALLATION (STACKED TAPES ONLY) menu:
   ===> ***cuu***
   (*cuu* is the address of the tape drive where you mounted the distribution tape)

   JOB DISPOSITION menu:

   Make any changes required and press ENTER to submit the job.

   The output listing from this job will give a list of the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifiers for COBOL/VSE are COB.BASE...1.1.0, COB.ENU....1.1.0, and COB.JPN....1.1.0.

   The program identifiers of the optional programs on the distribution

tape are also automatically entered on the **INSTALL PRODUCT(S) FROM TAPE** menu.

Return to the **VSE/ESA FUNCTION SELECTION** menu:
===> ***1*** (Installation)

INSTALLATION menu:
===> ***1*** (Install Programs - Stacked V2 Format)

INSTALL PROGRAMS - STACKED V2 FORMAT menu:
===> ***2*** (Install Product(s) from Tape)

INSTALL PRODUCT(S) FROM TAPE menu:

Enter **1** (install) in the OPT field against each of the identifiers below that you intend to install:

COB.BASE...1.1.0 (COBOL/VSE)

COB.ENU....1.1.0 (COBOL/VSE US English Language Feature)

COB.JPN....1.1.0 (COBOL/VSE Japanese Language Feature)

and **2** (skip installation) against any other optional products you do not intend to install at this time.

If you did not use the default library PRD2.PROD, enter the name of your library and sublibrary on this screen.

Press PF5 to generate the installation job.

VSE/ESA INSTALL PRODUCT(S) TAPE SPECIFICATION menu:

===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the COBOL/VSE tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job and install COBOL/VSE.

Check the output from the batch job created by the previous steps to ensure that the installation was successful. If you are installing COBOL/VSE from a nonstacked tape, you will receive the following message:
IESI0083I TAPE IS NOT PID-V2-STACKED

This message is for information only and can be ignored. The Librarian RESTORE job ends with a return code of 4. This is not an error: Installation was successful.

IBM Library Server

1.3.2.3.2 Step 3b: Install COBOL/VSE using a Batch Job

The batch installation job for installing COBOL/VSE uses the MSHP system history file that already exists in the VSE system. This file may already be defined in the system standard labels. If it is not defined, make sure that appropriate DLBL and EXTENT statements are included in the job stream.

Figure 5 shows sample JCL for installing COBOL/VSE which will handle both types of distribution tape.

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. Tailor the areas in the chosen sample job that are flagged with reference keys, such as **1** . Information on how to tailor these areas is provided following Figure 5.

Mount the distribution tape and run the installation job.

```
// JOB INSTALL 5686-068 COBOL/VSE COMPILER
*  Label for the Product Library                    1
*  Assign the install tape as SYS006                2
// ASSGN SYS006,cuu
// MTC REW,SYS006
* ----------------------------------------
*  This step installs the distribution tape
*  using the VSE system history file                3
* ----------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'
INSTALL PRODUCT FROMTAPE ID='COB.BASE...1.1.0' -
          PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.ENU....1.1.0' -    4
          PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.JPN....1.1.0' -    4
          PRODUCTION  INTO=PRD2.PROD
/*
* ------------------------------------
*  This step lists the Product Library              5
* ------------------------------------
// EXEC LIBR
 LISTDIR SUBLIB=PRD2.PROD -
     OUTPUT=NORMAL -
     UNIT=SYSLST
/*
* ------------------------------------
*  Retrace the COBOL/VSE product                    6
```

```
* ------------------------------------
// EXEC MSHP,SIZE=900K
RETRACE COMPONENT IDENTIFIER=5686-068-00
RETRACE COMPONENT IDENTIFIER=5686-068-01
RETRACE COMPONENT IDENTIFIER=5686-068-02
/*
// MTC  RUN,SYS006
/*
/&
```

Figure 5. Job to Install COBOL/VSE Compiler (5686-068)

Specify the Label Information: In area **1** , if you are installing COBOL/VSE in a sublibrary other than the default, insert the DLBL, EXTENT and ASSGN information as specified in Figure 4 in topic 1.3.2.2. The library name must match the name used in the allocation job in Figure 4 in topic 1.3.2.2.

Assign the Distribution Tape: Assign the distribution tape in area **2** to logical unit SYS006. Replace *cuu* with the address of the tape drive on which to mount the distribution tape.

Install COBOL/VSE: Area **3** of the job executes MSHP to install COBOL/VSE into the VSE Librarian sublibraries identified by the INTO operands of the INSTALL statements. If you are installing COBOL/VSE into sublibraries other than the installation default, change the names of the sublibraries specified on the INTO operands of the INSTALL statements.

For more information about the install options, see *VSE/ESA System Control Statements*.

National Language Feature: Choose the national language feature or features you want. Do this by examining the areas indicated by **4** . Remove the language or languages that you do not want.

COB.ENU....1.1.0 US English Language Feature (mixed-case)

COB.JPN....1.1.0 Japanese Language Feature

List the Directory Entries: The step in area **5** lists the directory entries of the VSE Librarian sublibraries where COBOL/VSE was installed. Remove this step if a directory list is not required.

If you are installing COBOL/VSE into sublibraries other than the installation default, change the name of the sublibrary specified on the SUBLIB operand of the LISTDIR statement.

The COBOL/VSE entries have a three character prefix of *IGY*, except for $$$CO18M.Z, $$$CO18N.Z, and $$$CO18O.Z.

Retrace the COBOL/VSE product in the system history file: The final step in area **6** of the job prints the component records from the system history file for COBOL/VSE. Remove this step if you do not want a retrace listing.

Note: If you use this step, you may remove the RETRACE statements that are not required.
Here is what the three RETRACE COMPONENT statements represent:

5686-068-00 Retracing the COBOL/VSE base component

5686-068-01 Retracing the COBOL/VSE US English Language Feature

5686-068-02 Retracing the COBOL/VSE Japanese Language Feature

If this job has to be rerun, remember first to restore the system history file, which should have been backed up before running this installation job, and to rerun the library allocation step, if applicable.

IBM Library Server

1.3.2.2 Step 2: Allocate Space for the Library

By default COBOL/VSE is installed into the PRD2.PROD sublibrary. If you decide to install COBOL/VSE into a different sublibrary proceed with this step. If you decide to install COBOL/VSE in the default sublibrary, proceed to "Step 3a: Install COBOL/VSE using the Interactive Interface" in topic 1.3.2.3.1 or "Step 3b: Install COBOL/VSE using a Batch Job" in topic 1.3.2.3.2. Once you have determined how much space you require, decide where to allocate space for the library. You can allocate space for the library in non-VSAM space or in VSAM space.

- Non-VSAM

  Identify, on the disk volume (or volumes) to be used for the library, suitable areas of free space. To do this, list the volume table of contents (VTOC) of the disk or disks to be used. The sample job shown in Figure 2 illustrates the JCL needed to list the VTOC for the volume with serial number COBVOL.

```
// JOB COBVSE  LIST VTOC for COBVOL
// ASSGN SYS004,DISK,TEMP,VOL=COBVOL,SHR
// ASSGN SYS005,SYSLST
// EXEC LVTOC
/*
/&
```

Figure 2. Listing the Contents of a DASD Volume

- VSAM

  Examine the user catalog to check for enough VSAM space. To do this, use the access method services (IDCAMS) LISTCAT command to provide a report showing the space available. The sample job shown in Figure 3 illustrates the JCL needed to display the space use in the VSAM user catalog supplied with VSE/ESA.

```
// JOB IDCAMS  SHOW USER CATALOG SPACE
// EXEC IDCAMS,SIZE=AUTO
 LISTCAT SPACE ALL CATALOG(VSESP.USER.CATALOG)
/*
/&
```

Figure 3. Listing the Space in a VSAM Catalog

If you intend to use VSAM space for the COBOL/VSE library, see *VSE/ESA Guide to System Functions* for information about how to define the required VSAM cluster. VSE/ESA also provides dialogs (Interactive Interface) for these tasks. Refer to *VSE/ESA Administration* for details.

Once you have determined where to allocate space for the library, you can define the library using the JCL shown in Figure 4.

```
// JOB LIBRDEF CREATE A LIBRARY
// OPTION LOG
*  Label for the Product Library as a Non-VSAM file  1
// DLBL COBVSE,'COBOL.VSE.LIBRARY',99/365,SD
// EXTENT SYS002,volid,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volid,SHR
*  Label for the Product Library as a VSAM file      2
// DLBL COBVSE,'COBOL.VSE.LIBRARY',,VSAM,CAT=catname
* --------------------------------------
*  This step defines the Product Library             3
* --------------------------------------
// EXEC LIBR
 DELETE LIB=COBVSE
 DEFINE LIB=COBVSE,REPLACE=NO
/*
/&
```

Figure 4. Job to Allocate the COBOL/VSE Library Space

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. If you intend to allocate space for the COBOL/VSE library in non-VSAM space, delete the DLBL statement in area **2** of the sample job, and make the following changes in area **1** :

   a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

   b. Change the variable *volid* in the EXTENT statement to the volume serial number of the volume that will hold your library.

   c. Change the variable *rtrk* in the EXTENT ASSGN statements to the start location (track or block number) of the extent.

d. Change the variable *ntrk* in the EXTENT statement to the number of tracks or blocks required for the library.

4. If you intend to allocate space for the COBOL/VSE library in VSAM space, delete the DLBL, EXTENT and ASSGN statements in area **1** of the sample job, and make the following changes in area **2**:

a. Change the DLBL *filename* (COBVSE) and *file-ID* (COBOL.VSE.LIBRARY) of the COBOL/VSE library to suit your installation.

b. Change the variable *catname* to the file name of the VSAM catalog that will own the library. You can omit *catname* (and the CAT keyword and preceding comma) if the library is to be cataloged in the IJSYSUC user catalog.

The Librarian job step in area **3** includes a DELETE statement before the DEFINE statement, so the job can be rerun. This means the following messages will be issued when the job runs for the first time. These may be ignored and the job will continue to allocate the library.

```
L101I  LIBRARY COBVSE DOES NOT EXIST    <--------------for Non-VSAM
L255I  LIBRARY COBVSE - OPEN FAILURE    <--------------for VSAM
L027I  ABNORMAL END DURING DELETE COMMAND PROCESSING
L113I  RETURN CODE OF DELETE IS 8
```

IBM Library Server

1.3.2.3.2 Step 3b: Install COBOL/VSE using a Batch Job

The batch installation job for installing COBOL/VSE uses the MSHP system history file that already exists in the VSE system. This file may already be defined in the system standard labels. If it is not defined, make sure that appropriate DLBL and EXTENT statements are included in the job stream.

Figure 5 shows sample JCL for installing COBOL/VSE which will handle both types of distribution tape.

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. Tailor the areas in the chosen sample job that are flagged with reference keys, such as **1** . Information on how to tailor these areas is provided following Figure 5.

Mount the distribution tape and run the installation job.

```
// JOB INSTALL 5686-068 COBOL/VSE COMPILER
*  Label for the Product Library                    1
*  Assign the install tape as SYS006                2
// ASSGN SYS006,cuu
// MTC REW,SYS006
* ----------------------------------------
*  This step installs the distribution tape
*  using the VSE system history file                3
* ----------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'
INSTALL PRODUCT FROMTAPE ID='COB.BASE...1.1.0' -
          PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.ENU....1.1.0' -    4
          PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.JPN....1.1.0' -    4
          PRODUCTION  INTO=PRD2.PROD
/*
* ------------------------------------
*  This step lists the Product Library              5
* ------------------------------------
// EXEC LIBR
 LISTDIR SUBLIB=PRD2.PROD -
     OUTPUT=NORMAL -
     UNIT=SYSLST
/*
* ------------------------------------
*  Retrace the COBOL/VSE product                    6
```

```
* ------------------------------------
// EXEC MSHP,SIZE=900K
RETRACE COMPONENT IDENTIFIER=5686-068-00
RETRACE COMPONENT IDENTIFIER=5686-068-01
RETRACE COMPONENT IDENTIFIER=5686-068-02
/*
// MTC  RUN,SYS006
/*
/&
```

Figure 5. Job to Install COBOL/VSE Compiler (5686-068)

Specify the Label Information: In area **1** , if you are installing COBOL/VSE in a sublibrary other than the default, insert the DLBL, EXTENT and ASSGN information as specified in Figure 4 in topic 1.3.2.2. The library name must match the name used in the allocation job in Figure 4 in topic 1.3.2.2.

Assign the Distribution Tape: Assign the distribution tape in area **2** to logical unit SYS006. Replace *cuu* with the address of the tape drive on which to mount the distribution tape.

Install COBOL/VSE: Area **3** of the job executes MSHP to install COBOL/VSE into the VSE Librarian sublibraries identified by the INTO operands of the INSTALL statements. If you are installing COBOL/VSE into sublibraries other than the installation default, change the names of the sublibraries specified on the INTO operands of the INSTALL statements.

For more information about the install options, see *VSE/ESA System Control Statements*.

National Language Feature: Choose the national language feature or features you want. Do this by examining the areas indicated by **4** . Remove the language or languages that you do not want.

COB.ENU....1.1.0 US English Language Feature (mixed-case)

COB.JPN....1.1.0 Japanese Language Feature

List the Directory Entries: The step in area **5** lists the directory entries of the VSE Librarian sublibraries where COBOL/VSE was installed. Remove this step if a directory list is not required.

If you are installing COBOL/VSE into sublibraries other than the installation default, change the name of the sublibrary specified on the SUBLIB operand of the LISTDIR statement.

The COBOL/VSE entries have a three character prefix of *IGY*, except for $$$CO18M.Z, $$$CO18N.Z, and $$$CO18O.Z.

Retrace the COBOL/VSE product in the system history file: The final step in area **6** of the job prints the component records from the system history file for COBOL/VSE. Remove this step if you do not want a retrace listing.

Note: If you use this step, you may remove the RETRACE statements that are not required.
Here is what the three RETRACE COMPONENT statements represent:

5686-068-00 Retracing the COBOL/VSE base component

5686-068-01 Retracing the COBOL/VSE US English Language Feature

5686-068-02 Retracing the COBOL/VSE Japanese Language Feature

If this job has to be rerun, remember first to restore the system history file, which should have been backed up before running this installation job, and to rerun the library allocation step, if applicable.

IBM Library Server

1.3.2.3.2 Step 3b: Install COBOL/VSE using a Batch Job

The batch installation job for installing COBOL/VSE uses the MSHP system history file that already exists in the VSE system. This file may already be defined in the system standard labels. If it is not defined, make sure that appropriate DLBL and EXTENT statements are included in the job stream.

Figure 5 shows sample JCL for installing COBOL/VSE which will handle both types of distribution tape.

Before you run the job, do the following:

1. Modify the JOB statement to suit your site.

2. Add POWER JECL statements if wanted.

3. Tailor the areas in the chosen sample job that are flagged with reference keys, such as **1** . Information on how to tailor these areas is provided following Figure 5.

Mount the distribution tape and run the installation job.

```
// JOB INSTALL 5686-068 COBOL/VSE COMPILER
*  Label for the Product Library                    1
*  Assign the install tape as SYS006                2
// ASSGN SYS006,cuu
// MTC REW,SYS006
* ----------------------------------------
*  This step installs the distribution tape
*  using the VSE system history file                3
* ----------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'
INSTALL PRODUCT FROMTAPE ID='COB.BASE...1.1.0' -
         PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.ENU....1.1.0' -    4
         PRODUCTION  INTO=PRD2.PROD
INSTALL PRODUCT FROMTAPE ID='COB.JPN....1.1.0' -    4
         PRODUCTION  INTO=PRD2.PROD
/*
* ------------------------------------
*  This step lists the Product Library              5
* ------------------------------------
// EXEC LIBR
 LISTDIR SUBLIB=PRD2.PROD -
     OUTPUT=NORMAL -
     UNIT=SYSLST
/*
* ------------------------------------
*  Retrace the COBOL/VSE product                    6
```

```
  * ----------------------------------
  // EXEC MSHP,SIZE=900K
  RETRACE COMPONENT IDENTIFIER=5686-068-00
  RETRACE COMPONENT IDENTIFIER=5686-068-01
  RETRACE COMPONENT IDENTIFIER=5686-068-02
  /*
  // MTC  RUN,SYS006
  /*
  /&
```

Figure 5. Job to Install COBOL/VSE Compiler (5686-068)

Specify the Label Information: In area **1** , if you are installing COBOL/VSE in a sublibrary other than the default, insert the DLBL, EXTENT and ASSGN information as specified in Figure 4 in topic 1.3.2.2. The library name must match the name used in the allocation job in Figure 4 in topic 1.3.2.2.

Assign the Distribution Tape: Assign the distribution tape in area **2** to logical unit SYS006. Replace *cuu* with the address of the tape drive on which to mount the distribution tape.

Install COBOL/VSE: Area **3** of the job executes MSHP to install COBOL/VSE into the VSE Librarian sublibraries identified by the INTO operands of the INSTALL statements. If you are installing COBOL/VSE into sublibraries other than the installation default, change the names of the sublibraries specified on the INTO operands of the INSTALL statements.

For more information about the install options, see *VSE/ESA System Control Statements*.

National Language Feature: Choose the national language feature or features you want. Do this by examining the areas indicated by **4** . Remove the language or languages that you do not want.

COB.ENU....1.1.0 US English Language Feature (mixed-case)

COB.JPN....1.1.0 Japanese Language Feature

List the Directory Entries: The step in area **5** lists the directory entries of the VSE Librarian sublibraries where COBOL/VSE was installed. Remove this step if a directory list is not required.

If you are installing COBOL/VSE into sublibraries other than the installation default, change the name of the sublibrary specified on the SUBLIB operand of the LISTDIR statement.

The COBOL/VSE entries have a three character prefix of *IGY*, except for $$$CO18M.Z, $$$CO18N.Z, and $$$CO18O.Z.

Retrace the COBOL/VSE product in the system history file: The final step in area **6** of the job prints the component records from the system history file for COBOL/VSE. Remove this step if you do not want a retrace listing.

Note: If you use this step, you may remove the RETRACE statements that are not required.
Here is what the three RETRACE COMPONENT statements represent:

5686-068-00 Retracing the COBOL/VSE base component

5686-068-01 Retracing the COBOL/VSE US English Language Feature

5686-068-02 Retracing the COBOL/VSE Japanese Language Feature

If this job has to be rerun, remember first to restore the system history file, which should have been backed up before running this installation job, and to rerun the library allocation step, if applicable.

1.3.2.4 Step 4: Verify the COBOL/VSE Installation

The installation verification program IGYWEIVP.C is a sample program provided on the distribution tape. It allows you to check that your installation is successful by exercising representative features of COBOL/VSE.

1. Job **CALLIVP1**: The JCL (IGYWEIN1.Z) and the source (IGYWEIVP.C) to run the verification program CALLIVP1 are supplied. They test the compiler and library, and verify the product installation.

   Before you run the JCL from member IGYWEIN1.Z, do the following:

   a. Modify the JOB statement to meet the requirements of your site.

   b. Add POWER JECL statements if wanted.

   c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

   d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

      Six compiler work files are always required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

   e. The JCL provided in IGYWEIN1.Z, without modification to the EXEC statement, compiles, link-edits, and runs the program. If you have not installed LE/VSE you must remove the two steps at the end of the job that link-edit and run the program.

   f. Execute the modified JCL to run the installation verification program.

   After you run job CALLIVP1 you should receive the following messages, printed on SYSLST:

   ```
   ***** START OF CALLIVP1 *****
   ***** CALLIVP1 SUCCESSFUL *****
   ```

2. Job **ERRMSG**: The JCL (IGYWEIN2.Z) and source (IGYWEERM.C) are used to run the verification program ERRMSG. This job will verify the operation of the compiler, and produce a complete list of compiler messages.

   Before you run the JCL from member IGYWEIN2.Z, do the following:

a. Modify the JOB statement to meet the requirements of your site.

b. Add POWER JECL statements if wanted.

c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

   Six compiler work files are required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

e. Execute the modified JCL to run the ERRMSG program.

After you run job ERRMSG, you will receive a listing of COBOL/VSE compiler messages.

1.4.1 General Rules for Changing Default Values

If you choose to modify the default values for options, follow these assembler coding instructions:

- Code in columns 2 through 71.
- Do not put a comma in front of the first option in your macro.
- Begin any continuation lines in column 16 and have a nonblank character in column 72 of the previous line. You can break the coding after any comma.
- Place an END statement after the macro instruction line or lines.

If you choose to modify only some of the options in the macro, COBOL/VSE will use the defaults supplied by IBM for those not changed. A syntax format and a detailed description of each of these options begin under "Making Changes after Installation--Why Customize?" in topic 1.2.1. For a description of the syntax notation that applies to the macro formats, see "How to Read the Syntax Diagrams" in topic FRONT_2.1.

IBM Library Server

1.4.2 Modifying Compiler Options and Phases

During installation, default compiler option and phase values were stored in the options module IGYCDOPT. To change the compiler-option defaults or tell the compiler where its phases are expected to reside, you should code your IGYCOPT macro calls in the sample JCL supplied in member IGYWEOP1.Z.

Note: If you specify that the compiler phases are to be resident in the SVA, you must ensure that you have placed them there. Failure to do so could result in an abnormal termination of the compiler because enough space was not available for the phase.

Subtopics:

1.4.3 Placing COBOL/VSE in the Shared Virtual Area

After planning for the SVA requirements, complete the following procedure according to the planning that has been performed.

To place the COBOL/VSE compiler phases in the SVA, the following steps must be performed:

1. Customize IGYCDOPT to indicate which phases of the compiler are resident in the SVA.

2. Modify the SVA statement of the VSE IPL ASI (Automated System Initialization) procedure to allow space for the compiler phases.

   - Increase the SDL parameter by the number of new phases being added to the SVA for the compiler.

   - Increase the PSIZE parameters for the 24-bit SVA and the 31-bit SVA by the number of K bytes required to contain the new phases being added to the SVA for the compiler.

3. Modify the VSE background ASI (Automated System Initialization) procedure to automatically load the selected compiler phases into the SVA.

   - Modify the ALLOC statements for the partitions to ensure that the remaining SVA space is large enough to contain the selected compiler phases.

   - Modify the LIBDEF PHASE,SEARCH=PRD2.PROD statement preceding the SET SDL statement to include the name of the VSE Librarian sublibrary containing COBOL/VSE.

   - After the SET SDL command add the statement:

     phasename,SVA

     for each compiler phase that is to be loaded into the SVA.

     The sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary contains SVA statements for all the SVA-eligible compiler phase names.

4. Shut down and re-IPL your VSE system

Note: If you do not want to place the compiler phases in the SVA at

IPL-time, you can do this at any time after IPL using the JCL statements supplied in the sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary. You should first customize IGYCDOPT as described in step 1 above. If you do not already have enough space available in the SVA, you should also perform steps 2 and 4 before you attempt to place the compiler phases in the SVA.

Using other languages

Member IGYWESVA.Z contains the uppercase language phases for English as shown:

- IGYCUE$R
- IGYCUE$D
- IGYCUE$0
- IGYCUE$1
- IGYCUE$2
- IGYCUE$3
- IGYCUE$4
- IGYCUE$5
- IGYCUE$8

If you wish to use other languages you should include the following phases:

For mixed-case English:

- IGYCEN$R
- IGYCEN$D
- IGYCEN$0
- IGYCEN$1
- IGYCEN$2
- IGYCEN$3
- IGYCEN$4
- IGYCEN$5
- IGYCEN$8

For Japanese:

- IGYCJA$R
- IGYCJA$D
- IGYCJA$0
- IGYCJA$1
- IGYCJA$2
- IGYCJA$3
- IGYCJA$4
- IGYCJA$5
- IGYCJA$8

For more information on loading phases in the SVA refer to *VSE/ESA System Control Statements* and *VSE/ESA Installation and Service*.

IBM Library Server

1.4.4 Modifying or Creating Additional Reserved Word Tables

The reserved words used by the COBOL/VSE compiler are maintained in a table (IGYCRWT) provided with the product. A CICS-specific reserved word table (IGYCCICS) is provided as an alternate reserved word table. You can change the reserved words by using the reserved word table utility (IGY8RWTU) to create additional reserved word tables.

The reserved word table utility accepts control statements you can use to create a new reserved word table. This new table then contains the reserved words from the table, as supplied by IBM, with all the changes you have applied.

You can make the following changes to the reserved word table:

- Add an alternative form of an existing reserved word.
- Add words to be flagged with an informational message whenever they are used in a program.
- Add words to be flagged with an error message whenever they are used in a program.
- Indicate that words currently flagged with an informational or error message should no longer be flagged.

Each reserved word table you create must have a unique 1- to 4-character identifier. For a list of 1- to 4-character strings that **cannot** be used, see the WORD compiler option in topic 1.2.5.53.

At compile time, the value of the compiler option **WORD(xxxx)** identifies the reserved word table to be used, where **xxxx** is the unique 1- to 4-character identification you specified in the member name IGYCxxxx. You can create multiple reserved word tables, but only one can be specified at compile time.

Note: The total amount of storage used for all entries in a reserved word table should not exceed 1536 bytes or 1.5K bytes.

Subtopics:

1.3.2.4 Step 4: Verify the COBOL/VSE Installation

The installation verification program IGYWEIVP.C is a sample program provided on the distribution tape. It allows you to check that your installation is successful by exercising representative features of COBOL/VSE.

1. Job **CALLIVP1**: The JCL (IGYWEIN1.Z) and the source (IGYWEIVP.C) to run the verification program CALLIVP1 are supplied. They test the compiler and library, and verify the product installation.

   Before you run the JCL from member IGYWEIN1.Z, do the following:

   a. Modify the JOB statement to meet the requirements of your site.

   b. Add POWER JECL statements if wanted.

   c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

   d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

      Six compiler work files are always required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

   e. The JCL provided in IGYWEIN1.Z, without modification to the EXEC statement, compiles, link-edits, and runs the program. If you have not installed LE/VSE you must remove the two steps at the end of the job that link-edit and run the program.

   f. Execute the modified JCL to run the installation verification program.

   After you run job CALLIVP1 you should receive the following messages, printed on SYSLST:

   ```
   ***** START OF CALLIVP1 *****
   ***** CALLIVP1 SUCCESSFUL *****
   ```

2. Job **ERRMSG**: The JCL (IGYWEIN2.Z) and source (IGYWEERM.C) are used to run the verification program ERRMSG. This job will verify the operation of the compiler, and produce a complete list of compiler messages.

   Before you run the JCL from member IGYWEIN2.Z, do the following:

a. Modify the JOB statement to meet the requirements of your site.

b. Add POWER JECL statements if wanted.

c. Change the LIBDEF statements to match the VSE sublibraries where COBOL/VSE and LE/VSE have been installed at your site.

d. Change the work files to match the standards used at your site or, if your site has the work files in system standard labels, remove the JCL for the work files.

   Six compiler work files are required: IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS06, and IJSYS07. The IJSYS05 work file is also required here because the LIB option is specified. The compiler work files must reside on direct access storage devices (DASD).

e. Execute the modified JCL to run the ERRMSG program.

After you run job ERRMSG, you will receive a listing of COBOL/VSE compiler messages.

1.4 Chapter 4. Customizing COBOL/VSE

You can make modifications to COBOL/VSE only after installation of the product is complete. "Making Changes after Installation--Why Customize?" in topic 1.2.1 provides information on what you can modify, and why you might want to customize COBOL/VSE. This chapter tells how to *make* the modifications or where to find the necessary coding information to tailor COBOL/VSE to the needs of your site.

The issues discussed in this chapter are:

- General rules for changing default values
- Modifying compiler options and phases
- Placing COBOL/VSE in the shared virtual area
- Creating additional reserved word tables

To make the modifications to the option macros, you will need to modify and run the sample JCL supplied. Sample JCL for customization is available in the sublibrary for the compiler.

Subtopics:

IBM Library Server

1.4 Chapter 4. Customizing COBOL/VSE

You can make modifications to COBOL/VSE only after installation of the product is complete. "Making Changes after Installation--Why Customize?" in topic 1.2.1 provides information on what you can modify, and why you might want to customize COBOL/VSE. This chapter tells how to *make* the modifications or where to find the necessary coding information to tailor COBOL/VSE to the needs of your site.

The issues discussed in this chapter are:

- General rules for changing default values
- Modifying compiler options and phases
- Placing COBOL/VSE in the shared virtual area
- Creating additional reserved word tables

To make the modifications to the option macros, you will need to modify and run the sample JCL supplied. Sample JCL for customization is available in the sublibrary for the compiler.

Subtopics:

IBM Library Server

1.4.1 General Rules for Changing Default Values

If you choose to modify the default values for options, follow these assembler coding instructions:

- Code in columns 2 through 71.
- Do not put a comma in front of the first option in your macro.
- Begin any continuation lines in column 16 and have a nonblank character in column 72 of the previous line. You can break the coding after any comma.
- Place an END statement after the macro instruction line or lines.

If you choose to modify only some of the options in the macro, COBOL/VSE will use the defaults supplied by IBM for those not changed. A syntax format and a detailed description of each of these options begin under "Making Changes after Installation--Why Customize?" in topic 1.2.1. For a description of the syntax notation that applies to the macro formats, see "How to Read the Syntax Diagrams" in topic FRONT_2.1.

IBM Library Server

1.4.2.1 Procedure for Modifying Compiler Options

Code your required IGYCOPT macro calls with the sample JCL supplied in member IGYWEOP1.Z. If you choose to modify only some of the compiler default values or phase residency selections, then COBOL/VSE will use the supplied defaults for any unchanged options or phase residency selections. You must observe the macro coding rules described under "General Rules for Changing Default Values" in topic 1.4.1 when coding the macro operands in your job. If you require more than one line for coding your option and phase changes, be sure to observe the line continuation rules. For a detailed description of each of the compiler options and phases in IGYCOPT, see "COBOL/VSE Compiler Options" in topic 1.2.5.

Member IGYWEOP1.Z contains a sample VSE JCL job stream you can use to assemble and link-edit the compiler options module.

IBM Library Server

1.4.1 General Rules for Changing Default Values

If you choose to modify the default values for options, follow these assembler coding instructions:

- Code in columns 2 through 71.
- Do not put a comma in front of the first option in your macro.
- Begin any continuation lines in column 16 and have a nonblank character in column 72 of the previous line. You can break the coding after any comma.
- Place an END statement after the macro instruction line or lines.

If you choose to modify only some of the options in the macro, COBOL/VSE will use the defaults supplied by IBM for those not changed. A syntax format and a detailed description of each of these options begin under "Making Changes after Installation--Why Customize?" in topic 1.2.1. For a description of the syntax notation that applies to the macro formats, see "How to Read the Syntax Diagrams" in topic FRONT_2.1.

IBM Library Server

1.4.2 Modifying Compiler Options and Phases

During installation, default compiler option and phase values were stored in the options module IGYCDOPT. To change the compiler-option defaults or tell the compiler where its phases are expected to reside, you should code your IGYCOPT macro calls in the sample JCL supplied in member IGYWEOP1.Z.

Note: If you specify that the compiler phases are to be resident in the SVA, you must ensure that you have placed them there. Failure to do so could result in an abnormal termination of the compiler because enough space was not available for the phase.

Subtopics:

IBM Library Server

1.4.2 Modifying Compiler Options and Phases

During installation, default compiler option and phase values were stored in the options module IGYCDOPT. To change the compiler-option defaults or tell the compiler where its phases are expected to reside, you should code your IGYCOPT macro calls in the sample JCL supplied in member IGYWEOP1.Z.

Note: If you specify that the compiler phases are to be resident in the SVA, you must ensure that you have placed them there. Failure to do so could result in an abnormal termination of the compiler because enough space was not available for the phase.

Subtopics:

1.4.2.1 Procedure for Modifying Compiler Options

Code your required IGYCOPT macro calls with the sample JCL supplied in member IGYWEOP1.Z. If you choose to modify only some of the compiler default values or phase residency selections, then COBOL/VSE will use the supplied defaults for any unchanged options or phase residency selections. You must observe the macro coding rules described under "General Rules for Changing Default Values" in topic 1.4.1 when coding the macro operands in your job. If you require more than one line for coding your option and phase changes, be sure to observe the line continuation rules. For a detailed description of each of the compiler options and phases in IGYCOPT, see "COBOL/VSE Compiler Options" in topic 1.2.5.

Member IGYWEOP1.Z contains a sample VSE JCL job stream you can use to assemble and link-edit the compiler options module.

IBM Library Server

1.4.2.1 Procedure for Modifying Compiler Options

Code your required IGYCOPT macro calls with the sample JCL supplied in member IGYWEOP1.Z. If you choose to modify only some of the compiler default values or phase residency selections, then COBOL/VSE will use the supplied defaults for any unchanged options or phase residency selections. You must observe the macro coding rules described under "General Rules for Changing Default Values" in topic 1.4.1 when coding the macro operands in your job. If you require more than one line for coding your option and phase changes, be sure to observe the line continuation rules. For a detailed description of each of the compiler options and phases in IGYCOPT, see "COBOL/VSE Compiler Options" in topic 1.2.5.

Member IGYWEOP1.Z contains a sample VSE JCL job stream you can use to assemble and link-edit the compiler options module.

1.4.3 Placing COBOL/VSE in the Shared Virtual Area

After planning for the SVA requirements, complete the following procedure according to the planning that has been performed.

To place the COBOL/VSE compiler phases in the SVA, the following steps must be performed:

1. Customize IGYCDOPT to indicate which phases of the compiler are resident in the SVA.

2. Modify the SVA statement of the VSE IPL ASI (Automated System Initialization) procedure to allow space for the compiler phases.

   - Increase the SDL parameter by the number of new phases being added to the SVA for the compiler.

   - Increase the PSIZE parameters for the 24-bit SVA and the 31-bit SVA by the number of K bytes required to contain the new phases being added to the SVA for the compiler.

3. Modify the VSE background ASI (Automated System Initialization) procedure to automatically load the selected compiler phases into the SVA.

   - Modify the ALLOC statements for the partitions to ensure that the remaining SVA space is large enough to contain the selected compiler phases.

   - Modify the LIBDEF PHASE,SEARCH=PRD2.PROD statement preceding the SET SDL statement to include the name of the VSE Librarian sublibrary containing COBOL/VSE.

   - After the SET SDL command add the statement:

     phasename,SVA

     for each compiler phase that is to be loaded into the SVA.

     The sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary contains SVA statements for all the SVA-eligible compiler phase names.

4. Shut down and re-IPL your VSE system

Note: If you do not want to place the compiler phases in the SVA at

IPL-time, you can do this at any time after IPL using the JCL statements supplied in the sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary. You should first customize IGYCDOPT as described in step 1 above. If you do not already have enough space available in the SVA, you should also perform steps 2 and 4 before you attempt to place the compiler phases in the SVA.

Using other languages

Member IGYWESVA.Z contains the uppercase language phases for English as shown:

- IGYCUE$R
- IGYCUE$D
- IGYCUE$0
- IGYCUE$1
- IGYCUE$2
- IGYCUE$3
- IGYCUE$4
- IGYCUE$5
- IGYCUE$8

If you wish to use other languages you should include the following phases:

For mixed-case English:

- IGYCEN$R
- IGYCEN$D
- IGYCEN$0
- IGYCEN$1
- IGYCEN$2
- IGYCEN$3
- IGYCEN$4
- IGYCEN$5
- IGYCEN$8

For Japanese:

- IGYCJA$R
- IGYCJA$D
- IGYCJA$0
- IGYCJA$1
- IGYCJA$2
- IGYCJA$3
- IGYCJA$4
- IGYCJA$5
- IGYCJA$8

For more information on loading phases in the SVA refer to *VSE/ESA System Control Statements* and *VSE/ESA Installation and Service*.

IBM Library Server

1.4.4.1 Procedure for Creating or Modifying a Reserved Word Table

To create or modify a reserved word table you must edit the appropriate source member from the compiler sublibrary:

- Member IGY8RWRD.Z (default reserved word table supplied by IBM)
- Member IGY8CICS.Z (CICS reserved word table supplied by IBM)
- A user member (user-supplied reserved word table)

You must also modify and invoke the appropriate JCL.

The JCL for the default reserved word table is IGYWERWD.Z

The JCL for the CICS reserved word table is IGYWERWC.Z

Your source member should have four parts: Parts I, II, III, and IV. Modify the member and JCL as follows:

1. Make a private copy of the member.
2. Do not edit Part I (all lines up to and including the line with the keyword MOD).
3. Edit Part II by placing asterisks in column 1 of the lines that contain CODASYL reserved words for which you do not want informational messages issued.
4. Edit Part III by placing asterisks in column 1 of the lines that contain obsolete reserved words for which you do not want severe messages issued.
5. Edit Part IV by coding additional reserved word control statements that create the modifications you want, as described under "Coding Control Statements" in topic 1.4.4.1.1.
6. Modify and run the JCL, as discussed under "Modifying and Running JCL to Create a New Reserved Word Table" in topic 1.4.4.5. You will also have to create a unique 1- to 4-character identification for the new reserved word table and supply it in the JCL.

Subtopics:

- 1.4.4.1.3 Coding Operands in Control Statements
- 1.4.4.1.4 Rules for Coding Control Statement Operands

IBM Library Server

1.4.4.2 ABBR Statement

```
ABBR  reserved-word:  user-word [ comments ]
    Defines an alternative symbol for the reserved word specified.  The
    symbol can be used when coding a program.
```

Notes:

1. *User-word* becomes a reserved word and can be used in place of the reserved word specified in this statement.

2. *Reserved-word* remains a reserved word with its original definition.

3. The source listing will show the original source--the symbol as you coded it.

4. The reserved word will be used in compiler output--other listings, some messages, and so forth.

Subtopics:

- 1.4.4.2.1 Example of an ABBR Statement

1.4.4.3 INFO Statement

```
INFO  COBOL-word  [ comments ]
    Specifies a COBOL word that is to be flagged by the compiler.


The messages will be handled as information (I) messages.  The compilation
condition is not changed.
```

Subtopics:

- 1.4.4.3.1 Example of an INFO Statement

IBM Library Server

1.4.4.4 RSTR Statement

RSTR COBOL-word [comments]
Specifies a COBOL word that cannot be used in a program.

Note: The following reserved words may not be restricted:

IDENTIFICATION	FD
ENVIRONMENT	DATA
WORKING-STORAGE	PROCEDURE
DIVISION	SECTION
PROGRAM-ID	

Subtopics:

- 1.4.4.4.1 Examples of RSTR Statements

IBM Library Server

1.4.4.5 Modifying and Running JCL to Create a New Reserved Word Table

The JCL you use to create a new reserved word table contains three steps:

STEP 1 Runs the reserved word table utility with your modified table.

STEP 2 Assembles your modified reserved word table.

STEP 3 Produces a run-time loadable phase from the object module.

After you run the job, a new reserved word table will be created, the sublibrary you specified will contain the new table, and the name of the table will be IGYC plus the 1- to 4-character identification you specified.

Subtopics:

1.4.3 Placing COBOL/VSE in the Shared Virtual Area

After planning for the SVA requirements, complete the following procedure according to the planning that has been performed.

To place the COBOL/VSE compiler phases in the SVA, the following steps must be performed:

1. Customize IGYCDOPT to indicate which phases of the compiler are resident in the SVA.

2. Modify the SVA statement of the VSE IPL ASI (Automated System Initialization) procedure to allow space for the compiler phases.

   - Increase the SDL parameter by the number of new phases being added to the SVA for the compiler.

   - Increase the PSIZE parameters for the 24-bit SVA and the 31-bit SVA by the number of K bytes required to contain the new phases being added to the SVA for the compiler.

3. Modify the VSE background ASI (Automated System Initialization) procedure to automatically load the selected compiler phases into the SVA.

   - Modify the ALLOC statements for the partitions to ensure that the remaining SVA space is large enough to contain the selected compiler phases.

   - Modify the LIBDEF PHASE,SEARCH=PRD2.PROD statement preceding the SET SDL statement to include the name of the VSE Librarian sublibrary containing COBOL/VSE.

   - After the SET SDL command add the statement:

     phasename,SVA

     for each compiler phase that is to be loaded into the SVA.

     The sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary contains SVA statements for all the SVA-eligible compiler phase names.

4. Shut down and re-IPL your VSE system

Note: If you do not want to place the compiler phases in the SVA at

IPL-time, you can do this at any time after IPL using the JCL statements supplied in the sample member IGYWESVA.Z in the COBOL/VSE compiler sublibrary. You should first customize IGYCDOPT as described in step 1 above. If you do not already have enough space available in the SVA, you should also perform steps 2 and 4 before you attempt to place the compiler phases in the SVA.

Using other languages

Member IGYWESVA.Z contains the uppercase language phases for English as shown:

- IGYCUE$R
- IGYCUE$D
- IGYCUE$0
- IGYCUE$1
- IGYCUE$2
- IGYCUE$3
- IGYCUE$4
- IGYCUE$5
- IGYCUE$8

If you wish to use other languages you should include the following phases:

For mixed-case English:

- IGYCEN$R
- IGYCEN$D
- IGYCEN$0
- IGYCEN$1
- IGYCEN$2
- IGYCEN$3
- IGYCEN$4
- IGYCEN$5
- IGYCEN$8

For Japanese:

- IGYCJA$R
- IGYCJA$D
- IGYCJA$0
- IGYCJA$1
- IGYCJA$2
- IGYCJA$3
- IGYCJA$4
- IGYCJA$5
- IGYCJA$8

For more information on loading phases in the SVA refer to *VSE/ESA System Control Statements* and *VSE/ESA Installation and Service*.

1.4.4 Modifying or Creating Additional Reserved Word Tables

The reserved words used by the COBOL/VSE compiler are maintained in a table (IGYCRWT) provided with the product. A CICS-specific reserved word table (IGYCCICS) is provided as an alternate reserved word table. You can change the reserved words by using the reserved word table utility (IGY8RWTU) to create additional reserved word tables.

The reserved word table utility accepts control statements you can use to create a new reserved word table. This new table then contains the reserved words from the table, as supplied by IBM, with all the changes you have applied.

You can make the following changes to the reserved word table:

- Add an alternative form of an existing reserved word.
- Add words to be flagged with an informational message whenever they are used in a program.
- Add words to be flagged with an error message whenever they are used in a program.
- Indicate that words currently flagged with an informational or error message should no longer be flagged.

Each reserved word table you create must have a unique 1- to 4-character identifier. For a list of 1- to 4-character strings that **cannot** be used, see the WORD compiler option in topic 1.2.5.53.

At compile time, the value of the compiler option **WORD(xxxx)** identifies the reserved word table to be used, where **xxxx** is the unique 1- to 4-character identification you specified in the member name IGYCxxxx. You can create multiple reserved word tables, but only one can be specified at compile time.

Note: The total amount of storage used for all entries in a reserved word table should not exceed 1536 bytes or 1.5K bytes.

Subtopics:

1.4.4.1.1 Coding Control Statements

To create a reserved word table, you must understand the syntax rules for the control statements and for the operands within the control statements.

Figure 6 illustrates the format for coding reserved word processor control statements.

```
ABBR   reserved-word: user-word [comments]
     [reserved-word: user-word [comments]]

.
.
.
INFO   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
RSTR   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
```

Figure 6. Syntax Format for Reserved Word Processor Control Statements

As shown in Figure 6, keywords you can use are:

ABBR to specify an alternative form of an existing reserved word

INFO to specify words that are to be flagged with an informational message whenever they are used in a program

RSTR to specify words that are to be flagged with an error message whenever a program employs them

Note: All words you identify with the control statement keywords INFO and RSTR will be flagged with a message in the source listing of the COBOL/VSE program that uses them. Words that are abbreviated will not be flagged in the source listing unless you have also specified them on the INFO or RSTR control statements.

1.4.4.1.2 Rules for Coding Control Statements

When you code your control statements, do the following:

- Begin the control statement in column 1.
- Place one or more spaces between the keyword and the first operand.
- When specifying a second operand, include a colon (:) and one or more spaces after the first operand.
- Continue a control statement by putting blanks in columns 1 through 5, followed by the operand or operands, to make additional specifications.
- Specify comments by putting an asterisk (*) in column 1 of the control statements. You can also place comments on the same line as the control statement. In that case, however, there must be at least one space following the operand(s) before a comment begins.
- To specify more than one change within a single control statement, put each additional specification on a separate line.
- Do not add any blank lines.

1.4.4.1.3 Coding Operands in Control Statements

The following list shows the types of operands you will be coding in the control statements:

reserved-word is an existing reserved word.

user-word is a user-defined COBOL word that is not a reserved word.

comments are any comments you want to put on the same line with the control statement, or on a separate line that has an asterisk in column 1.

COBOL-word is a word of up to 30 characters that may be a system name, a reserved word, or a user-defined word.

1.4.4.1.4 Rules for Coding Control Statement Operands

When you are coding the control statement operands, follow these rules:

- A *user-word* can be used in only one ABBR statement in any particular reserved word table.
- A *reserved-word* specified in an ABBR statement may also be specified in either a RSTR or an INFO statement.
- A particular *reserved-word* can be specified only once in an ABBR statement.
- A particular *COBOL-word* can be specified only once in either a RSTR or an INFO statement.

The remaining sections provide examples for coding each type of control statement.

1.4.4 Modifying or Creating Additional Reserved Word Tables

The reserved words used by the COBOL/VSE compiler are maintained in a table (IGYCRWT) provided with the product. A CICS-specific reserved word table (IGYCCICS) is provided as an alternate reserved word table. You can change the reserved words by using the reserved word table utility (IGY8RWTU) to create additional reserved word tables.

The reserved word table utility accepts control statements you can use to create a new reserved word table. This new table then contains the reserved words from the table, as supplied by IBM, with all the changes you have applied.

You can make the following changes to the reserved word table:

- Add an alternative form of an existing reserved word.
- Add words to be flagged with an informational message whenever they are used in a program.
- Add words to be flagged with an error message whenever they are used in a program.
- Indicate that words currently flagged with an informational or error message should no longer be flagged.

Each reserved word table you create must have a unique 1- to 4-character identifier. For a list of 1- to 4-character strings that **cannot** be used, see the WORD compiler option in topic 1.2.5.53.

At compile time, the value of the compiler option **WORD(xxxx)** identifies the reserved word table to be used, where **xxxx** is the unique 1- to 4-character identification you specified in the member name IGYCxxxx. You can create multiple reserved word tables, but only one can be specified at compile time.

Note: The total amount of storage used for all entries in a reserved word table should not exceed 1536 bytes or 1.5K bytes.

Subtopics:

1.4.4.1.4 Rules for Coding Control Statement Operands

When you are coding the control statement operands, follow these rules:

- A *user-word* can be used in only one ABBR statement in any particular reserved word table.
- A *reserved-word* specified in an ABBR statement may also be specified in either a RSTR or an INFO statement.
- A particular *reserved-word* can be specified only once in an ABBR statement.
- A particular *COBOL-word* can be specified only once in either a RSTR or an INFO statement.

The remaining sections provide examples for coding each type of control statement.

1.4.4.2.1 Example of an ABBR Statement

```
  ABBR  REDEFINES:  REDEF
        SEPARATE:   SEP
```

REDEF and SEP become abbreviations that can be used in source programs. Action appropriate to the use of REDEFINES and SEPARATE will then be taken when the source program is compiled.

1.4.4.1.4 Rules for Coding Control Statement Operands

When you are coding the control statement operands, follow these rules:

- A *user-word* can be used in only one ABBR statement in any particular reserved word table.
- A *reserved-word* specified in an ABBR statement may also be specified in either a RSTR or an INFO statement.
- A particular *reserved-word* can be specified only once in an ABBR statement.
- A particular *COBOL-word* can be specified only once in either a RSTR or an INFO statement.

The remaining sections provide examples for coding each type of control statement.

1.4.4.2.1 Example of an ABBR Statement

```
  ABBR  REDEFINES:  REDEF
        SEPARATE:   SEP
```

REDEF and SEP become abbreviations that can be used in source programs. Action appropriate to the use of REDEFINES and SEPARATE will then be taken when the source program is compiled.

IBM Library Server

1.4.4.3.1 Example of an INFO Statement

```
  INFO  ENTRY
        EVALUATE
```

When the IBM extension reserved words, ENTRY and EVALUATE, are used in a program, they will be flagged in the messages section of the source listing.

1.4.4.2.1 Example of an ABBR Statement

```
  ABBR  REDEFINES:  REDEF
        SEPARATE:   SEP

REDEF and SEP become abbreviations that can be used in source programs.
Action appropriate to the use of REDEFINES and SEPARATE will then be taken
when the source program is compiled.
```

1.4.4.3.1 Example of an INFO Statement

```
  INFO  ENTRY
        EVALUATE
```

When the IBM extension reserved words, ENTRY and EVALUATE, are used in a program, they will be flagged in the messages section of the source listing.

1.4.4.4.1 Examples of RSTR Statements

Example 1

```
  RSTR  BOOLEAN
        XD
        PARENT
```

The use of BOOLEAN, XD, and PARENT will cause error messages.

Example 2

```
  RSTR  GO
        ALTER
```

The use of GO TO and ALTER will cause error messages.

IBM Library Server

1.4.4.3.1 Example of an INFO Statement

```
  INFO  ENTRY
        EVALUATE
```

When the IBM extension reserved words, ENTRY and EVALUATE, are used in a program, they will be flagged in the messages section of the source listing.

1.4.4.4.1 Examples of RSTR Statements

Example 1

```
  RSTR  BOOLEAN
        XD
        PARENT
```

The use of BOOLEAN, XD, and PARENT will cause error messages.

Example 2

```
  RSTR  GO
        ALTER
```

The use of GO TO and ALTER will cause error messages.

1.4.4.5.1 Procedure for Modifying and Running JCL

Use the JCL in members IGYWERWD.Z or IGYWERWC.Z, in the COBOL/VSE sublibrary, to create your new reserved word table. Before you run the job, do the following:

1. Modify the job statement for your site.

2. Add POWER JECL statements if required.

3. Change the LIBDEF PHASE,CATALOG statement to point to the VSE sublibrary into which you wish to catalog the new reserved word table.

4. Insert the LIBR update statements, as indicated in the sample job, to modify your private copy of member IGY8RWRD.Z or IGY8CICS.Z.

5. Specify the name of your modified reserved word table in the PHASE linkage editor control statement.

6. Change the LIBDEF PHASE,SEARCH statement to match the VSE sublibrary where COBOL/VSE has been installed at your site.

Note: For specific instructions, see the comments in member IGYWERWD.Z or IGYWERWC.Z.

If, when you run IGYWERWD.Z or IGYWERWC.Z, you receive a nonzero return code from the table utility, use the error messages in the output listing to identify any errors. Rerun the job when you have corrected the errors.

1.4.4.4.1 Examples of RSTR Statements

Example 1

```
  RSTR  BOOLEAN
        XD
        PARENT
```

The use of BOOLEAN, XD, and PARENT will cause error messages.

Example 2

```
  RSTR  GO
        ALTER
```

The use of GO TO and ALTER will cause error messages.

1.4.4.5.1 Procedure for Modifying and Running JCL

Use the JCL in members IGYWERWD.Z or IGYWERWC.Z, in the COBOL/VSE sublibrary, to create your new reserved word table. Before you run the job, do the following:

1. Modify the job statement for your site.

2. Add POWER JECL statements if required.

3. Change the LIBDEF PHASE,CATALOG statement to point to the VSE sublibrary into which you wish to catalog the new reserved word table.

4. Insert the LIBR update statements, as indicated in the sample job, to modify your private copy of member IGY8RWRD.Z or IGY8CICS.Z.

5. Specify the name of your modified reserved word table in the PHASE linkage editor control statement.

6. Change the LIBDEF PHASE,SEARCH statement to match the VSE sublibrary where COBOL/VSE has been installed at your site.

Note: For specific instructions, see the comments in member IGYWERWD.Z or IGYWERWC.Z.

If, when you run IGYWERWD.Z or IGYWERWC.Z, you receive a nonzero return code from the table utility, use the error messages in the output listing to identify any errors. Rerun the job when you have corrected the errors.

IBM Library Server

1.5.1 Reinstalling COBOL/VSE

You do not need to perform all the planning and installation procedures to re-install COBOL/VSE. For example, you might not need to reconsider your storage needs if COBOL/VSE replaces the existing COBOL/VSE sublibrary.

You do not need to remove COBOL/VSE from your system before re-installing it unless you intend to re-install the product in a different sublibrary. In this case you must remove COBOL/VSE from the system history file before re-installation can take place. Figure 10 in topic 1.5.3 shows a job to remove COBOL/VSE from the system history file.

To re-install COBOL/VSE you simply follow the same steps as for installing it. See Chapter 3, "Installing COBOL/VSE" in topic 1.3.

IBM Library Server

1.5.2 Applying Service Updates

You might need to apply maintenance or service updates to COBOL/VSE periodically.

Subtopics:

IBM Library Server

1.5.3 Removing COBOL/VSE

You do not have to remove COBOL/VSE from your system before installing a new version or release.

If you do have to remove COBOL/VSE from your system for any reason at all, you must delete all the COBOL/VSE entries from your sublibrary and remove COBOL/VSE from the system history file. Figure 9 shows the JCL needed to remove COBOL/VSE from the sublibrary.

```
// JOB IGYDELV  Remove COBOL/VSE
* Label for the COBOL/VSE library     1
// EXEC LIBR,SIZE=200K
ACCESS S=PRD2.PROD                   2
DELETE IGY*.*
DELETE $$$CO18M.Z
DELETE $$$CO18N.Z
DELETE $$$CO18O.Z
/*
/&
```

Figure 9. Job to Remove COBOL/VSE from a Sublibrary

If you have installed COBOL/VSE into a sublibrary other than the default (PRD2.PROD):

- Insert the required DLBL, EXTENT and ASSGN information for the COBOL/VSE library in area **1** .
- Change the statement in area **2** .

To remove COBOL/VSE from the system history file, use the REMOVE command of MSHP. The sample job shown in Figure 10 shows the JCL needed to remove COBOL/VSE from the system history file.

```
// JOB IGYDELH  Remove Product
```

```
// EXEC MSHP,SIZE=900K
REMOVE 5686-068-00-18M      1
REMOVE 5686-068-01-18N      2
REMOVE 5686-068-02-18O      2
/*
/&
```

Figure 10. Job to Remove COBOL/VSE from the System History File

Area **1** shows the component ID for the COBOL/VSE base component. This REMOVE statement should be the first because it removes the COBOL compiler itself. See below for the other REMOVE statements.

Areas **2** show the National Language components of COBOL/VSE. Use REMOVE only for those which have been installed. For example, if you have installed the Japanese Language Feature but not the US English Language Feature, you should include only the REMOVE statement for the Japanese National Language Feature.

IBM Library Server

1.5.4 How to Report a Problem with COBOL/VSE

For information on how to report a problem with COBOL/VSE, see the *COBOL/VSE Diagnosis Guide*.

1.4.4.5.1 Procedure for Modifying and Running JCL

Use the JCL in members IGYWERWD.Z or IGYWERWC.Z, in the COBOL/VSE sublibrary, to create your new reserved word table. Before you run the job, do the following:

1. Modify the job statement for your site.

2. Add POWER JECL statements if required.

3. Change the LIBDEF PHASE,CATALOG statement to point to the VSE sublibrary into which you wish to catalog the new reserved word table.

4. Insert the LIBR update statements, as indicated in the sample job, to modify your private copy of member IGY8RWRD.Z or IGY8CICS.Z.

5. Specify the name of your modified reserved word table in the PHASE linkage editor control statement.

6. Change the LIBDEF PHASE,SEARCH statement to match the VSE sublibrary where COBOL/VSE has been installed at your site.

Note: For specific instructions, see the comments in member IGYWERWD.Z or IGYWERWC.Z.

If, when you run IGYWERWD.Z or IGYWERWC.Z, you receive a nonzero return code from the table utility, use the error messages in the output listing to identify any errors. Rerun the job when you have corrected the errors.

IBM Library Server

1.5 Chapter 5. Maintaining COBOL/VSE

This chapter describes how to replace or re-install COBOL/VSE, and how to apply service updates to COBOL/VSE. To use the maintenance procedures effectively, you must have already installed COBOL/VSE and any required products.

In addition, this chapter describes how to remove COBOL/VSE.

Subtopics:

IBM Library Server

1.5 Chapter 5. Maintaining COBOL/VSE

This chapter describes how to replace or re-install COBOL/VSE, and how to apply service updates to COBOL/VSE. To use the maintenance procedures effectively, you must have already installed COBOL/VSE and any required products.

In addition, this chapter describes how to remove COBOL/VSE.

Subtopics:

IBM Library Server

1.5.1 Reinstalling COBOL/VSE

You do not need to perform all the planning and installation procedures to re-install COBOL/VSE. For example, you might not need to reconsider your storage needs if COBOL/VSE replaces the existing COBOL/VSE sublibrary.

You do not need to remove COBOL/VSE from your system before re-installing it unless you intend to re-install the product in a different sublibrary. In this case you must remove COBOL/VSE from the system history file before re-installation can take place. Figure 10 in topic 1.5.3 shows a job to remove COBOL/VSE from the system history file.

To re-install COBOL/VSE you simply follow the same steps as for installing it. See Chapter 3, "Installing COBOL/VSE" in topic 1.3.

IBM Library Server

1.5.2.1 What You Receive

If you report a problem with COBOL/VSE to your IBM Support Center, you will receive a tape containing one or more PTFs (Program Temporary Fix) which have been created to solve your problem.

You may also receive a list of pre-requisite APARs (Authorized Program Analysis Report) or PTFs which should have been applied to your system before applying the current service. These pre-requisite APARs and PTFs may relate to COBOL/VSE or any other licensed product you have installed, including VSE/ESA.

You apply service to COBOL/VSE using either the VSE/ESA Interactive Interface or a batch job.

The following checklist provides a summary of the steps you should use to apply service to COBOL/VSE.

IBM Library Server

1.5.2.2 Checklist for Applying Service

Table 18 lists the steps for installing corrective service on COBOL/VSE. You can use this table as a checklist.

Table 18. Summary of Steps for Installing Service on COBOL/VSE

Step	Description	MSHP Command or Jobname	Topic
1	Check pre-requisite APARs or PTFs	RETRACE	1.5.2.2.1
2	Backup the Original System	----	1.5.2.2.2
3a	Apply Service using the Interactive Interface	INSTALL	1.5.2.2.3
3b	Apply Service using a batch job	INSTALL	1.5.2.2.3
4	Run the Installation Verification Program	IGYWEIVP	1.5.2.2.6

Subtopics:

IBM Library Server

1.5.1 Reinstalling COBOL/VSE

You do not need to perform all the planning and installation procedures to re-install COBOL/VSE. For example, you might not need to reconsider your storage needs if COBOL/VSE replaces the existing COBOL/VSE sublibrary.

You do not need to remove COBOL/VSE from your system before re-installing it unless you intend to re-install the product in a different sublibrary. In this case you must remove COBOL/VSE from the system history file before re-installation can take place. Figure 10 in topic 1.5.3 shows a job to remove COBOL/VSE from the system history file.

To re-install COBOL/VSE you simply follow the same steps as for installing it. See Chapter 3, "Installing COBOL/VSE" in topic 1.3.

IBM Library Server

1.5.2 Applying Service Updates

You might need to apply maintenance or service updates to COBOL/VSE periodically.

Subtopics:

1.5.2 Applying Service Updates

You might need to apply maintenance or service updates to COBOL/VSE periodically.

Subtopics:

1.5.2.1 What You Receive

If you report a problem with COBOL/VSE to your IBM Support Center, you will receive a tape containing one or more PTFs (Program Temporary Fix) which have been created to solve your problem.

You may also receive a list of pre-requisite APARs (Authorized Program Analysis Report) or PTFs which should have been applied to your system before applying the current service. These pre-requisite APARs and PTFs may relate to COBOL/VSE or any other licensed product you have installed, including VSE/ESA.

You apply service to COBOL/VSE using either the VSE/ESA Interactive Interface or a batch job.

The following checklist provides a summary of the steps you should use to apply service to COBOL/VSE.

1.5.2.2.2 Step 2. Backup the Original System

Make a backup copy of your current COBOL/VSE sublibrary and the system history file. For information about backing up, see *VSE/ESA System Control Statements*.

1.5.2.2.3 Step 3. Apply Service

You can apply service to COBOL/VSE from the provided service tape using either the Interactive Interface or a batch job.

You will receive instructions for applying service with the service tape.

Continue either with Step 3a or Step 3b.

IBM Library Server

1.5.2.2.6 Step 4. Run the Installation Verification Program (IVP)

After you have applied all the files on the service tape, run the installation verification program IGYWEIVP to ensure that COBOL/VSE functions properly. See the description of IGYWEIVP in "Step 4: Verify the COBOL/VSE Installation" in topic 1.3.2.4.

1.5.2.2.1 Step 1. Check Pre-requisite APARs or PTFs

Pre-requisite APARs or PTFs are those that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to COBOL/VSE or any licensed program you have installed.

Your IBM Support Center will have given you a list of any relevant pre-requisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 7 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYRETR  Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 7. Job to Retrace APARs and PTFs

Use the resulting listing to check that you have already applied any pre-requisite APARs or PTFs. If you have not, your IBM Support Center will send them to you, and you should apply them before applying any other service.

 *Copyright IBM Corp. 1983,1998*

1.5.2.2.2 Step 2. Backup the Original System

Make a backup copy of your current COBOL/VSE sublibrary and the system history file. For information about backing up, see *VSE/ESA System Control Statements*.

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

1.5.2.2.3 Step 3. Apply Service

You can apply service to COBOL/VSE from the provided service tape using either the Interactive Interface or a batch job.

You will receive instructions for applying service with the service tape.

Continue either with Step 3a or Step 3b.

IBM Library Server

1.5.2.2.4 Step 3a: Apply Service Using the Interactive Interface

To apply service to COBOL/VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (Additional information about the functions of the Interactive Interface are in *VSE/ESA Administration*.)

Mount the COBOL/VSE service tape on an available tape drive.

In the following menus specify the **highlighted** items that appear after the ===> symbol.

1. **VSE/ESA FUNCTION SELECTION** menu:
   ===> ***1*** (Installation)

2. **INSTALLATION** menu:
   ===> ***4*** (IBM Service)

PTF HANDLING menu

- If you want to print the documentation about the supplied PTFs before applying the service, select:
  ===> ***1*** (Print Service Document)

PRINT SERVICE DOCUMENT menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the service tape)

- If you want to apply the service directly, select:
  ===> ***3*** (Apply PTFs from Service Tape)

APPLY PTF menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the service tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job to apply the service.

IBM Library Server

1.5.2.2.5 Step 3b: Apply Service Using a Batch Job

The batch job to apply service to COBOL/VSE uses the MSHP system history file where COBOL/VSE was installed.

A sample job to apply service using the Maintain System History Program (MSHP) is shown in Figure 8. For more information on MSHP see *VSE/ESA System Control Statements*.

```
// JOB IGYAPP   Apply Service
// ASSGN SYS006,cuu              1
// EXEC MSHP,SIZE=900K
INSTALL SERVICE FROMTAPE         2
/*
/&
```

Figure 8. Job to Apply Service

In area **1** change *cuu* to the address of the tape drive where you have mounted the service tape.

Area **2** shows the MSHP statement to install service from a tape. The information in the system history file will direct MSHP to apply the service to the sublibrary in which COBOL/VSE is installed.

IBM Library Server

1.5.2.2.6 Step 4. Run the Installation Verification Program (IVP)

After you have applied all the files on the service tape, run the installation verification program IGYWEIVP to ensure that COBOL/VSE functions properly. See the description of IGYWEIVP in "Step 4: Verify the COBOL/VSE Installation" in topic 1.3.2.4.

IBM Library Server

1.5.2.1 What You Receive

If you report a problem with COBOL/VSE to your IBM Support Center, you will receive a tape containing one or more PTFs (Program Temporary Fix) which have been created to solve your problem.

You may also receive a list of pre-requisite APARs (Authorized Program Analysis Report) or PTFs which should have been applied to your system before applying the current service. These pre-requisite APARs and PTFs may relate to COBOL/VSE or any other licensed product you have installed, including VSE/ESA.

You apply service to COBOL/VSE using either the VSE/ESA Interactive Interface or a batch job.

The following checklist provides a summary of the steps you should use to apply service to COBOL/VSE.

IBM Library Server

1.5.2.2.6 Step 4. Run the Installation Verification Program (IVP)

After you have applied all the files on the service tape, run the installation verification program IGYWEIVP to ensure that COBOL/VSE functions properly. See the description of IGYWEIVP in "Step 4: Verify the COBOL/VSE Installation" in topic 1.3.2.4.

IBM Library Server

1.5.2.2.6 Step 4. Run the Installation Verification Program (IVP)

After you have applied all the files on the service tape, run the installation verification program IGYWEIVP to ensure that COBOL/VSE functions properly. See the description of IGYWEIVP in "Step 4: Verify the COBOL/VSE Installation" in topic 1.3.2.4.

IBM Library Server

1.5.3 Removing COBOL/VSE

You do not have to remove COBOL/VSE from your system before installing a new version or release.

If you do have to remove COBOL/VSE from your system for any reason at all, you must delete all the COBOL/VSE entries from your sublibrary and remove COBOL/VSE from the system history file. Figure 9 shows the JCL needed to remove COBOL/VSE from the sublibrary.

```
// JOB IGYDELV  Remove COBOL/VSE
* Label for the COBOL/VSE library     1
// EXEC LIBR,SIZE=200K
ACCESS S=PRD2.PROD                   2
DELETE IGY*.*
DELETE $$$CO18M.Z
DELETE $$$CO18N.Z
DELETE $$$CO18O.Z
/*
/&
```

Figure 9. Job to Remove COBOL/VSE from a Sublibrary

If you have installed COBOL/VSE into a sublibrary other than the default (PRD2.PROD):

- Insert the required DLBL, EXTENT and ASSGN information for the COBOL/VSE library in area **1** .
- Change the statement in area **2** .

To remove COBOL/VSE from the system history file, use the REMOVE command of MSHP. The sample job shown in Figure 10 shows the JCL needed to remove COBOL/VSE from the system history file.

```
// JOB IGYDELH  Remove Product
```

```
// EXEC MSHP,SIZE=900K
REMOVE 5686-068-00-18M      1
REMOVE 5686-068-01-18N      2
REMOVE 5686-068-02-18O      2
/*
/&
```

Figure 10. Job to Remove COBOL/VSE from the System History File

Area **1** shows the component ID for the COBOL/VSE base component. This REMOVE statement should be the first because it removes the COBOL compiler itself. See below for the other REMOVE statements.

Areas **2** show the National Language components of COBOL/VSE. Use REMOVE only for those which have been installed. For example, if you have installed the Japanese Language Feature but not the US English Language Feature, you should include only the REMOVE statement for the Japanese National Language Feature.

IBM Library Server

1.5.3 Removing COBOL/VSE

You do not have to remove COBOL/VSE from your system before installing a new version or release.

If you do have to remove COBOL/VSE from your system for any reason at all, you must delete all the COBOL/VSE entries from your sublibrary and remove COBOL/VSE from the system history file. Figure 9 shows the JCL needed to remove COBOL/VSE from the sublibrary.

```
// JOB IGYDELV  Remove COBOL/VSE
* Label for the COBOL/VSE library     1
// EXEC LIBR,SIZE=200K
ACCESS S=PRD2.PROD                   2
DELETE IGY*.*
DELETE $$$CO18M.Z
DELETE $$$CO18N.Z
DELETE $$$CO18O.Z
/*
/&
```

Figure 9. Job to Remove COBOL/VSE from a Sublibrary

If you have installed COBOL/VSE into a sublibrary other than the default (PRD2.PROD):

- Insert the required DLBL, EXTENT and ASSGN information for the COBOL/VSE library in area **1** .
- Change the statement in area **2** .

To remove COBOL/VSE from the system history file, use the REMOVE command of MSHP. The sample job shown in Figure 10 shows the JCL needed to remove COBOL/VSE from the system history file.

```
// JOB IGYDELH  Remove Product
```

```
// EXEC MSHP,SIZE=900K
REMOVE 5686-068-00-18M      1
REMOVE 5686-068-01-18N      2
REMOVE 5686-068-02-18O      2
/*
/&
```

Figure 10. Job to Remove COBOL/VSE from the System History File

Area **1** shows the component ID for the COBOL/VSE base component. This REMOVE statement should be the first because it removes the COBOL compiler itself. See below for the other REMOVE statements.

Areas **2** show the National Language components of COBOL/VSE. Use REMOVE only for those which have been installed. For example, if you have installed the Japanese Language Feature but not the US English Language Feature, you should include only the REMOVE statement for the Japanese National Language Feature.

1.5.4 How to Report a Problem with COBOL/VSE

For information on how to report a problem with COBOL/VSE, see the *COBOL/VSE Diagnosis Guide*.

IBM Library Server

| 2.1 Chapter 6. Planning to Install VisualAge COBOL MLE for VSE

| This chapter provides information for planning the installation of
| VisualAge COBOL MLE for VSE. It includes information on:

- What you receive with VisualAge COBOL MLE for VSE
- What you need to install VisualAge COBOL MLE for VSE
- Checking service updates

Subtopics:

IBM Library Server

| 2.2 Chapter 7. Installing VisualAge COBOL MLE for VSE

| This chapter describes the installation method and the step-by-step
| procedures you use to install VisualAge COBOL MLE for VSE from an optional
| program tape or a separate distribution tape.

Subtopics:

IBM Library Server

| 2.3 Chapter 8. Maintaining VisualAge COBOL MLE for VSE

| This chapter describes how to replace or reinstall VisualAge COBOL MLE for VSE, and how to apply service updates to VisualAge COBOL MLE for VSE. To effectively use the maintenance procedures, you must have already installed VisualAge COBOL MLE for VSE and any required products.

| In addition, this chapter describes how to remove VisualAge COBOL MLE for VSE from your system.

Subtopics:

IBM Library Server

1.5.4 How to Report a Problem with COBOL/VSE

For information on how to report a problem with COBOL/VSE, see the *COBOL/VSE Diagnosis Guide*.

IBM Library Server

| 2.0 Planning for, Installing, and Maintaining VisualAge COBOL MLE for VSE

Subtopics:

IBM Library Server

| 2.1.1 What You Receive with VisualAge COBOL MLE for VSE

| You receive VisualAge COBOL MLE for VSE in one of two forms:

| • As an optional program on the VSE/ESA optional program tape
| • As a separately-ordered licensed program on a separate distribution tape

| When you receive VisualAge COBOL MLE for VSE, either as an optional program, or as a separately-ordered licensed program, you receive:

| • Basic machine-readable material

| • Basic unlicensed publications

| • Optional unlicensed publications (if you specify the required feature numbers when ordering VisualAge COBOL MLE for VSE)

| You receive publications in the national language (U.S. English or Japanese) you specify when you order.

| Table 19 describes the component supplied by Release 1 of VisualAge COBOL MLE for VSE.

| Table 19. VisualAge COBOL MLE for VSE Component and Component Level Code (CLC)

Component ID	CLC	Component Name
5686-MLE-00	1JQ	VA COBOL MLE for VSE

Subtopics:

- 2.1.1.1 Distribution Media
- 2.1.1.2 Program Documentation

IBM Library Server

| 2.1.2 What You Need to Install VisualAge COBOL MLE for VSE

| The following sections identify the system requirements for installing and
| activating VisualAge COBOL MLE for VSE.

Subtopics:

IBM Library Server

| 2.0 Planning for, Installing, and Maintaining VisualAge COBOL MLE for VSE

Subtopics:

IBM Library Server

| 2.1 Chapter 6. Planning to Install VisualAge COBOL MLE for VSE

| This chapter provides information for planning the installation of
| VisualAge COBOL MLE for VSE. It includes information on:

- What you receive with VisualAge COBOL MLE for VSE
- What you need to install VisualAge COBOL MLE for VSE
- Checking service updates

Subtopics:

IBM Library Server

| 2.1.1.1 Distribution Media

| You will receive VisualAge COBOL MLE for VSE on one of the following:

| - 9-track magnetic tape written at 6250 BPI
| - IBM 3480 or IBM 3490 cartridge
| - 4mm DAT cartridge

| The tape or cartridge contains all the programs and data needed for
| installation.

Subtopics:

IBM Library Server

| 2.1.1.2 Program Documentation

| This section identifies the basic and optional VisualAge COBOL MLE for VSE
| documentation you receive.

Subtopics:

IBM Library Server

| 2.1 Chapter 6. Planning to Install VisualAge COBOL MLE for VSE

| This chapter provides information for planning the installation of
| VisualAge COBOL MLE for VSE. It includes information on:

| - What you receive with VisualAge COBOL MLE for VSE
| - What you need to install VisualAge COBOL MLE for VSE
| - Checking service updates

Subtopics:

IBM Library Server

2.1.1 What You Receive with VisualAge COBOL MLE for VSE

You receive VisualAge COBOL MLE for VSE in one of two forms:

- As an optional program on the VSE/ESA optional program tape
- As a separately-ordered licensed program on a separate distribution tape

When you receive VisualAge COBOL MLE for VSE, either as an optional program, or as a separately-ordered licensed program, you receive:

- Basic machine-readable material
- Basic unlicensed publications
- Optional unlicensed publications (if you specify the required feature numbers when ordering VisualAge COBOL MLE for VSE)

You receive publications in the national language (U.S. English or Japanese) you specify when you order.

Table 19 describes the component supplied by Release 1 of VisualAge COBOL MLE for VSE.

Table 19. VisualAge COBOL MLE for VSE Component and Component Level Code (CLC)

Component ID	CLC	Component Name
5686-MLE-00	1JQ	VA COBOL MLE for VSE

Subtopics:

IBM Library Server

2.1.1.1.1 Basic Machine-Readable Material

Table 20 describes the separately-ordered licensed program distribution tape.

Table 20. Basic Material: Separately-Ordered Distribution Tape

National Language	Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
US English(1)	6250 tape	5801	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5802	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6300	1 of 1	VA CBL MLE V1R1	unlabeled
Japanese(1)	6250 tape	5811	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5812	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6301	1 of 1	VA CBL MLE V1R1	unlabeled

Note:

1. There is no difference between the program tapes for the US English Language Feature and the Japanese Language Feature. The different feature numbers are used to specify the national language of the program documentation.

The file content of the distribution tape you receive depends upon the

form in which you receive VisualAge COBOL MLE for VSE. If you receive VisualAge COBOL MLE for VSE on a VSE/ESA optional program tape, there might be other licensed programs on the tape. If you ordered VisualAge COBOL MLE for VSE separately, the distribution tape will contain only VisualAge COBOL MLE for VSE. Table 21 describes the file content of the separately-ordered licensed program distribution tape.

Table 21. File Content: Separately-Ordered Licensed Program Distribution Tape

File	Description
1	Header file containing the VisualAge COBOL MLE for VSE copyright statement
2	Backup file ID COBOLMLE...1.1.0. followed by an MSHP System History File
3	VisualAge COBOL MLE for VSE product
4	Tape Mark
5	End of backup (EOB) record
6	Tape Mark

IBM Library Server

| 2.1.1.1.2 Optional Machine-Readable Material

| There are no optional machine-readable materials for VisualAge COBOL MLE
| for VSE.

IBM Library Server

2.1.1 What You Receive with VisualAge COBOL MLE for VSE

You receive VisualAge COBOL MLE for VSE in one of two forms:

- As an optional program on the VSE/ESA optional program tape
- As a separately-ordered licensed program on a separate distribution tape

When you receive VisualAge COBOL MLE for VSE, either as an optional program, or as a separately-ordered licensed program, you receive:

- Basic machine-readable material
- Basic unlicensed publications
- Optional unlicensed publications (if you specify the required feature numbers when ordering VisualAge COBOL MLE for VSE)

You receive publications in the national language (U.S. English or Japanese) you specify when you order.

Table 19 describes the component supplied by Release 1 of VisualAge COBOL MLE for VSE.

Table 19. VisualAge COBOL MLE for VSE Component and Component Level Code (CLC)

Component ID	CLC	Component Name
5686-MLE-00	1JQ	VA COBOL MLE for VSE

Subtopics:

- 2.1.1.1 Distribution Media
- 2.1.1.2 Program Documentation

IBM Library Server

| 2.1.1.1.2 Optional Machine-Readable Material

| There are no optional machine-readable materials for VisualAge COBOL MLE
| for VSE.

IBM Library Server

2.1.1.2.1 Basic Unlicensed Program Publications

Table 22 identifies the basic program publications for VisualAge COBOL MLE for VSE. Unless otherwise noted, you receive one copy of each of these publications when you receive the basic materials for VisualAge COBOL MLE for VSE. For additional copies, contact your IBM representative.

Table 22. Basic VisualAge COBOL MLE for VSE Documentation

Title	Order Number
VisualAge COBOL MLE for VSE Licensed Program Specifications	GC26-9417
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

| 2.1.1.2.2 Optional Unlicensed Program Publications and Softcopy

| Table 23 in topic 2.1.1.2.3 identifies the optional unlicensed program
| publications for VisualAge COBOL MLE for VSE. You receive one free
| printed copy of each publication listed if you specify the feature number
| 7014 (or 7035 for Japanese publications) when you order VisualAge COBOL
| MLE for VSE.

IBM Library Server

| 2.1.1.2.3 Optional Documentation

Table 23. Optional VisualAge COBOL MLE for VSE Documentation

Title	Order Number
COBOL/VSE Language Reference	SC26-8073
COBOL/VSE Programming Guide	SC26-8072
COBOL Millennium Language Extensions Guide	GC26-9266

IBM Library Server

| 2.1.1.1.2 Optional Machine-Readable Material

| There are no optional machine-readable materials for VisualAge COBOL MLE
| for VSE.

IBM Library Server

| 2.1.1.2.3 Optional Documentation

Table 23. Optional VisualAge COBOL MLE for VSE Documentation

Title	Order Number
COBOL/VSE Language Reference	SC26-8073
COBOL/VSE Programming Guide	SC26-8072
COBOL Millennium Language Extensions Guide	GC26-9266

IBM Library Server

| 2.1.2.1 Machine Requirements

```
| VisualAge COBOL MLE for VSE runs on any hardware configuration that
| supports COBOL/VSE.
```

IBM Library Server

| 2.1.2.2 Operating System Requirements

| VisualAge COBOL MLE for VSE operates under the same operating systems as
| COBOL/VSE. See "Choosing Required and Optional Licensed Programs" in
| topic 1.1.7 for more information.

IBM Library Server

2.1.2.3 Programming Requirements

VisualAge COBOL MLE for VSE has the following programming requirements:

1. COBOL/VSE Version 1 Release 1 must be installed before VisualAge COBOL MLE for VSE is installed. In addition, the PTFs that resolve the APARs shown in Table 24 must be applied to the components of COBOL/VSE you have installed before VisualAge COBOL MLE for VSE is installed.

Table 24. COBOL/VSE Service Requirements

Component Name	APAR
COBOL/VSE Base	PQ13830 PQ13831
COBOL/VSE US English Language Feature	PQ13832
COBOL/VSE Japanese Language Feature	PQ13834

For information about installing COBOL/VSE, see "Planning for, Installing, Customizing, and Maintaining COBOL/VSE" in topic 1.0.

2. LE/VSE Version 1 Release 4 is required to compile and run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option). In addition, the PTFs that resolve the APARs shown in Table 25 must be applied to the components of LE/VSE you have installed.

Table 25. LE/VSE Service Requirements

Component Name	APAR
LE/VSE COBOL-Specific Base	PQ15106
LE/VSE COBOL-Specific Japanese Language Feature	PQ15112
LE/VSE COBOL-Specific CICS	PQ15114

3. If you have Debug Tool/VSE installed on your system, the PTFs that resolve the APARs shown in Table 26 must be applied to the components of Debug Tool/VSE you have installed before you can use Debug Tool/VSE with programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option).

Table 26. Debug Tool/VSE Service Requirements

Component Name	APAR
Debug Tool/VSE Base	PQ15548 PQ15549
Debug Tool/VSE Japanese Language Feature	PQ15550

|_____|_____|

| 4. High Level Assembler for MVS & VM & VSE might be required to apply service.

| In addition to the above requirements, VisualAge COBOL MLE for VSE has the same run-time programming requirements as COBOL/VSE. See "Choosing Required and Optional Licensed Programs" in topic 1.1.7 for more information.

IBM Library Server

| 2.1.2.4 DASD Storage Requirements

| VisualAge COBOL MLE for VSE should be installed in the same sublibrary as
| COBOL/VSE and has only minor storage requirements for installation. For
| more information, see "DASD Storage Required for Installation" in
| topic 1.1.6.1.

IBM Library Server

2.1.2.5 Checking Service Updates

Before installing VisualAge COBOL MLE for VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 27. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLMLE	MLE1JQ

IBM Library Server

| 2.1.1.2.3 Optional Documentation

Table 23. Optional VisualAge COBOL MLE for VSE Documentation

Title	Order Number
COBOL/VSE Language Reference	SC26-8073
COBOL/VSE Programming Guide	SC26-8072
COBOL Millennium Language Extensions Guide	GC26-9266

IBM Library Server

| 2.1.2 What You Need to Install VisualAge COBOL MLE for VSE

| The following sections identify the system requirements for installing and
| activating VisualAge COBOL MLE for VSE.

Subtopics:

IBM Library Server

| 2.1.2 What You Need to Install VisualAge COBOL MLE for VSE

| The following sections identify the system requirements for installing and
| activating VisualAge COBOL MLE for VSE.

Subtopics:

IBM Library Server

| 2.1.2.1 Machine Requirements

```
| VisualAge COBOL MLE for VSE runs on any hardware configuration that
| supports COBOL/VSE.
```

| 2.1.2.1 Machine Requirements

```
| VisualAge COBOL MLE for VSE runs on any hardware configuration that
| supports COBOL/VSE.
```

IBM Library Server

| 2.1.2.2 Operating System Requirements

| VisualAge COBOL MLE for VSE operates under the same operating systems as COBOL/VSE. See "Choosing Required and Optional Licensed Programs" in topic 1.1.7 for more information.

IBM Library Server

| 2.1.2.2 Operating System Requirements

| VisualAge COBOL MLE for VSE operates under the same operating systems as COBOL/VSE. See "Choosing Required and Optional Licensed Programs" in topic 1.1.7 for more information.

IBM Library Server

2.1.2.3 Programming Requirements

VisualAge COBOL MLE for VSE has the following programming requirements:

1. COBOL/VSE Version 1 Release 1 must be installed before VisualAge COBOL MLE for VSE is installed. In addition, the PTFs that resolve the APARs shown in Table 24 must be applied to the components of COBOL/VSE you have installed before VisualAge COBOL MLE for VSE is installed.

Table 24. COBOL/VSE Service Requirements

Component Name	APAR
COBOL/VSE Base	PQ13830 PQ13831
COBOL/VSE US English Language Feature	PQ13832
COBOL/VSE Japanese Language Feature	PQ13834

For information about installing COBOL/VSE, see "Planning for, Installing, Customizing, and Maintaining COBOL/VSE" in topic 1.0.

2. LE/VSE Version 1 Release 4 is required to compile and run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option). In addition, the PTFs that resolve the APARs shown in Table 25 must be applied to the components of LE/VSE you have installed.

Table 25. LE/VSE Service Requirements

Component Name	APAR
LE/VSE COBOL-Specific Base	PQ15106
LE/VSE COBOL-Specific Japanese Language Feature	PQ15112
LE/VSE COBOL-Specific CICS	PQ15114

3. If you have Debug Tool/VSE installed on your system, the PTFs that resolve the APARs shown in Table 26 must be applied to the components of Debug Tool/VSE you have installed before you can use Debug Tool/VSE with programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option).

Table 26. Debug Tool/VSE Service Requirements

Component Name	APAR
Debug Tool/VSE Base	PQ15548 PQ15549
Debug Tool/VSE Japanese Language Feature	PQ15550

|_____|_____|

| 4. High Level Assembler for MVS & VM & VSE might be required to apply service.

| In addition to the above requirements, VisualAge COBOL MLE for VSE has the same run-time programming requirements as COBOL/VSE. See "Choosing Required and Optional Licensed Programs" in topic 1.1.7 for more information.

IBM Library Server

2.1.2.3 Programming Requirements

VisualAge COBOL MLE for VSE has the following programming requirements:

1. COBOL/VSE Version 1 Release 1 must be installed before VisualAge COBOL MLE for VSE is installed. In addition, the PTFs that resolve the APARs shown in Table 24 must be applied to the components of COBOL/VSE you have installed before VisualAge COBOL MLE for VSE is installed.

Table 24. COBOL/VSE Service Requirements

Component Name	APAR
COBOL/VSE Base	PQ13830 PQ13831
COBOL/VSE US English Language Feature	PQ13832
COBOL/VSE Japanese Language Feature	PQ13834

For information about installing COBOL/VSE, see "Planning for, Installing, Customizing, and Maintaining COBOL/VSE" in topic 1.0.

2. LE/VSE Version 1 Release 4 is required to compile and run COBOL/VSE programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option). In addition, the PTFs that resolve the APARs shown in Table 25 must be applied to the components of LE/VSE you have installed.

Table 25. LE/VSE Service Requirements

Component Name	APAR
LE/VSE COBOL-Specific Base	PQ15106
LE/VSE COBOL-Specific Japanese Language Feature	PQ15112
LE/VSE COBOL-Specific CICS	PQ15114

3. If you have Debug Tool/VSE installed on your system, the PTFs that resolve the APARs shown in Table 26 must be applied to the components of Debug Tool/VSE you have installed before you can use Debug Tool/VSE with programs that use COBOL automatic date processing facilities (as enabled by the DATEPROC compiler option).

Table 26. Debug Tool/VSE Service Requirements

Component Name	APAR
Debug Tool/VSE Base	PQ15548 PQ15549
Debug Tool/VSE Japanese Language Feature	PQ15550

```
      |_____________________________________________________|_________________|

  | 4.  High Level Assembler for MVS & VM & VSE might be required to apply
  |     service.



  | In addition to the above requirements, VisualAge COBOL MLE for VSE has the
  | same run-time programming requirements as COBOL/VSE.  See "Choosing
  | Required and Optional Licensed Programs" in topic 1.1.7 for more
  | information.
```

IBM Library Server

| 2.1.2.4 DASD Storage Requirements

| VisualAge COBOL MLE for VSE should be installed in the same sublibrary as
| COBOL/VSE and has only minor storage requirements for installation. For
| more information, see "DASD Storage Required for Installation" in
| topic 1.1.6.1.

IBM Library Server

| 2.1.2.4 DASD Storage Requirements

| VisualAge COBOL MLE for VSE should be installed in the same sublibrary as
| COBOL/VSE and has only minor storage requirements for installation. For
| more information, see "DASD Storage Required for Installation" in
| topic 1.1.6.1.

IBM Library Server

2.1.2.5 Checking Service Updates

Before installing VisualAge COBOL MLE for VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 27. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLMLE	MLE1JQ

IBM Library Server

| 2.2.1 Overview of Installation

| You install this release of VisualAge COBOL MLE for VSE by using the
| Maintain System History Program (MSHP), or by using the VSE/ESA
| Interactive Interface.

Subtopics:

- 2.2.1.1 List of Installation Steps

IBM Library Server

| 2.2.2 Step 1: Read this Book and Plan the Installation

| Read this book and familiarize yourself with all the installation material
| and procedures.

Subtopics:

- 2.2.2.1 Plan the Installation

IBM Library Server

| 2.2.3 Step 2: Back Up the Original System

| Make a backup copy of the library you intend to install VisualAge COBOL
| MLE for VSE into, and the system history file.

| For information about backing up libraries and the system history file,
| see *VSE/ESA System Control Statements*.

IBM Library Server

| 2.2.4 Step 3: Install VisualAge COBOL MLE for VSE

| You can install VisualAge COBOL MLE for VSE using either the VSE/ESA
| Interactive Interface or a batch installation job.

Subtopics:

IBM Library Server

| 2.2.5 Step 4: Verify the Installation of VisualAge COBOL MLE for VSE

| After you install VisualAge COBOL MLE for VSE, run sample job IGYMLIV1 to
| verify automatic date processing in COBOL/VSE.

Subtopics:

- 2.2.5.1 Verify Date Processing in COBOL/VSE

IBM Library Server

2.2.6 Step 5: Place VisualAge COBOL MLE for VSE in the SVA

After you have verified the successful installation of VisualAge COBOL MLE for VSE, you might want to place VisualAge COBOL MLE for VSE in the SVA.

Note: Placing VisualAge COBOL MLE for VSE in the SVA must be done by a system administrator.

To assist you in placing the VisualAge COBOL MLE for VSE phase in the SVA, a sample job is provided in member IGYMLSV1.Z. This sample job is shown in Figure 12. See the notes following Figure 12 for information about tailoring the sample job.

```
// JOB IGYMLSV1 LOAD SVA-ELIGIBLE PHASE
*
*  Load SVA
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
IGYCMLE,SVA
/*
/&
```

Figure 12. Job to Place VisualAge COBOL MLE for VSE in the SVA

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

Alternatively, as shown by the sample job in Figure 13, you can place the VisualAge COBOL MLE for VSE phase in the SVA using a load list. To help you do this, the load list $SVAIGYM is provided. You can also specify this load list in the BG startup procedure. If you choose to specify the load list in the BG startup procedure, the sublibrary where you installed VisualAge COBOL MLE for VSE must be in the LIBDEF search chain within the LIBSDL procedure.

```
// JOB IGYMLSVA LOAD SVA-ELIGIBLE PHASES
*
```

```
*  Load SVA using a load list
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
LIST=$SVAIGYM
/*
/&
```

Figure 13. Job to Place VisualAge COBOL MLE for VSE in the SVA Using a Load List

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

For more information about placing phases in the SVA, see *VSE/ESA System Control Statements*.

IBM Library Server

2.1.2.5 Checking Service Updates

Before installing VisualAge COBOL MLE for VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional preventive service planning (PSP) information you need. To obtain this information, specify the following UPGRADE and SUBSET values:

Table 27. PSP UPGRADE and SUBSET IDs

UPGRADE	SUBSET
COBOLMLE	MLE1JQ

IBM Library Server

| 2.2 Chapter 7. Installing VisualAge COBOL MLE for VSE

| This chapter describes the installation method and the step-by-step
| procedures you use to install VisualAge COBOL MLE for VSE from an optional
| program tape or a separate distribution tape.

Subtopics:

IBM Library Server

2.2.1.1 List of Installation Steps

Table 28 lists the steps and associated jobs to install VisualAge COBOL MLE for VSE. The remaining sections in this chapter describe what each job does and how to modify it. You can use Table 28 as a checklist.

Table 28. Summary of Steps for Installing VisualAge COBOL MLE for VSE

Step	Description	Installation Job	Topic
1	Read this book to plan the installation.	-	2.2.2
2	Back up the original system.	-	2.2.3
3	Install VisualAge COBOL MLE for VSE.		2.2.4
	Method 1: Install using the VSE/Interactive Interface.	-	2.2.4.1
	Method 2: Install using a batch job.	IGYMLINS	2.2.4.2
4	Verify VisualAge COBOL MLE for VSE installation.	IGYMLIV1	2.2.5
5	Place VisualAge COBOL MLE for VSE in the SVA.	IGYMLSV1 IGYMLSVA	2.2.6

IBM Library Server

| 2.2 Chapter 7. Installing VisualAge COBOL MLE for VSE

| This chapter describes the installation method and the step-by-step
| procedures you use to install VisualAge COBOL MLE for VSE from an optional
| program tape or a separate distribution tape.

Subtopics:

IBM Library Server

| 2.2.1 Overview of Installation

| You install this release of VisualAge COBOL MLE for VSE by using the
| Maintain System History Program (MSHP), or by using the VSE/ESA
| Interactive Interface.

Subtopics:

- 2.2.1.1 List of Installation Steps

IBM Library Server

| 2.2.1 Overview of Installation

| You install this release of VisualAge COBOL MLE for VSE by using the
| Maintain System History Program (MSHP), or by using the VSE/ESA
| Interactive Interface.

Subtopics:

- 2.2.1.1 List of Installation Steps

IBM Library Server

2.2.1.1 List of Installation Steps

Table 28 lists the steps and associated jobs to install VisualAge COBOL MLE for VSE. The remaining sections in this chapter describe what each job does and how to modify it. You can use Table 28 as a checklist.

Table 28. Summary of Steps for Installing VisualAge COBOL MLE for VSE

Step	Description	Installation Job	Topic
1	Read this book to plan the installation.	-	2.2.2
2	Back up the original system.	-	2.2.3
3	Install VisualAge COBOL MLE for VSE.		2.2.4
	Method 1: Install using the VSE/Interactive Interface.	-	2.2.4.1
	Method 2: Install using a batch job.	IGYMLINS	2.2.4.2
4	Verify VisualAge COBOL MLE for VSE installation.	IGYMLIV1	2.2.5
5	Place VisualAge COBOL MLE for VSE in the SVA.	IGYMLSV1 IGYMLSVA	2.2.6

IBM Library Server

2.2.2.1 Plan the Installation

Before you install VisualAge COBOL MLE for VSE, you should do the following:

1. Verify that you have the required machine and software configuration to run VisualAge COBOL MLE for VSE. See "What You Need to Install VisualAge COBOL MLE for VSE" in topic 2.1.2 for more information.

2. Determine which tape volumes you need for the installation.

   - If you order VisualAge COBOL MLE for VSE as a VSE/ESA optional licensed program, you receive one or more tapes that contain the optional licensed programs you ordered. These tapes are called stacked tapes. The external label on each stacked tape is:

     VSE/ESA x.x.x OPTIONAL PROGRAMS n OF x

   - If you order VisualAge COBOL MLE for VSE separately, you receive one non-stacked tape. The external label on this tape is:

     VA CBL MLE V1R1

3. Determine the name of the sublibrary where COBOL/VSE is installed. VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE.

4. Check on the latest service updates needed. For more information, see "Checking Service Updates" in topic 2.1.2.5.

5. Decide whether you are going to install VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface or an MSHP batch job.

6. Plan any necessary changes to the installation jobs described in this chapter.

IBM Library Server

2.2.1.1 List of Installation Steps

Table 28 lists the steps and associated jobs to install VisualAge COBOL MLE for VSE. The remaining sections in this chapter describe what each job does and how to modify it. You can use Table 28 as a checklist.

Table 28. Summary of Steps for Installing VisualAge COBOL MLE for VSE

Step	Description	Installation Job	Topic
1	Read this book to plan the installation.	-	2.2.2
2	Back up the original system.	-	2.2.3
3	Install VisualAge COBOL MLE for VSE.		2.2.4
	Method 1: Install using the VSE/Interactive Interface.	-	2.2.4.1
	Method 2: Install using a batch job.	IGYMLINS	2.2.4.2
4	Verify VisualAge COBOL MLE for VSE installation.	IGYMLIV1	2.2.5
5	Place VisualAge COBOL MLE for VSE in the SVA.	IGYMLSV1 IGYMLSVA	2.2.6

IBM Library Server

| 2.2.2 Step 1: Read this Book and Plan the Installation

| Read this book and familiarize yourself with all the installation material
| and procedures.

Subtopics:

- 2.2.2.1 Plan the Installation

IBM Library Server

| 2.2.2 Step 1: Read this Book and Plan the Installation

| Read this book and familiarize yourself with all the installation material
| and procedures.

Subtopics:

- 2.2.2.1 Plan the Installation

IBM Library Server

2.2.2.1 Plan the Installation

Before you install VisualAge COBOL MLE for VSE, you should do the following:

1. Verify that you have the required machine and software configuration to run VisualAge COBOL MLE for VSE. See "What You Need to Install VisualAge COBOL MLE for VSE" in topic 2.1.2 for more information.

2. Determine which tape volumes you need for the installation.

   - If you order VisualAge COBOL MLE for VSE as a VSE/ESA optional licensed program, you receive one or more tapes that contain the optional licensed programs you ordered. These tapes are called stacked tapes. The external label on each stacked tape is:

     VSE/ESA x.x.x OPTIONAL PROGRAMS n OF x

   - If you order VisualAge COBOL MLE for VSE separately, you receive one non-stacked tape. The external label on this tape is:

     VA CBL MLE V1R1

3. Determine the name of the sublibrary where COBOL/VSE is installed. VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE.

4. Check on the latest service updates needed. For more information, see "Checking Service Updates" in topic 2.1.2.5.

5. Decide whether you are going to install VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface or an MSHP batch job.

6. Plan any necessary changes to the installation jobs described in this chapter.

IBM Library Server

2.2.2.1 Plan the Installation

Before you install VisualAge COBOL MLE for VSE, you should do the following:

1. Verify that you have the required machine and software configuration to run VisualAge COBOL MLE for VSE. See "What You Need to Install VisualAge COBOL MLE for VSE" in topic 2.1.2 for more information.

2. Determine which tape volumes you need for the installation.

   - If you order VisualAge COBOL MLE for VSE as a VSE/ESA optional licensed program, you receive one or more tapes that contain the optional licensed programs you ordered. These tapes are called stacked tapes. The external label on each stacked tape is:

     VSE/ESA x.x.x OPTIONAL PROGRAMS n OF x

   - If you order VisualAge COBOL MLE for VSE separately, you receive one non-stacked tape. The external label on this tape is:

     VA CBL MLE V1R1

3. Determine the name of the sublibrary where COBOL/VSE is installed. VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE.

4. Check on the latest service updates needed. For more information, see "Checking Service Updates" in topic 2.1.2.5.

5. Decide whether you are going to install VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface or an MSHP batch job.

6. Plan any necessary changes to the installation jobs described in this chapter.

IBM Library Server

| 2.2.3 Step 2: Back Up the Original System

| Make a backup copy of the library you intend to install VisualAge COBOL
| MLE for VSE into, and the system history file.

| For information about backing up libraries and the system history file,
| see *VSE/ESA System Control Statements*.

IBM Library Server

| 2.2.4.1 Method 1: Install VisualAge COBOL MLE for VSE Using the Interactive Interface

| The VSE/ESA Interactive Interface enables you to use dialog requests to
| install VisualAge COBOL MLE for VSE. For more information about
| installing licensed programs using the Interactive Interface, see *VSE/ESA Installation*.

| To install VisualAge COBOL MLE for VSE using the Interactive Interface, do
| the following:

| 1. Logon to the Interactive Interface as the system administrator.

| 2. Mount the VisualAge COBOL MLE for VSE distribution tape on an
| available tape drive.

| 3. In the following menus, specify the items that appear following the
| ===> symbol.

| a. **VSE/ESA FUNCTION SELECTION** menu:

| ===> 1 (Installation)

| b. **INSTALLATION** menu:

| ===> 1 (Install Programs - V2 Format)

| **Note:** You can use option 1 for both a stacked distribution tape
| containing one or more optional licensed programs, and a
| non-stacked tape containing just VisualAge COBOL MLE for VSE.
| Both are in Librarian V2 format.

| c. **INSTALL PROGRAMS - V2 FORMAT** menu:

| ===> 1 (Prepare for Installation)

| d. **PREPARE FOR INSTALLATION** menu:

| ===> cuu (address of drive with distribution tape)

| e. **JOB DISPOSITION** menu:

Make any changes required and press ENTER to submit the job. While the job is running, reply to any system console messages as necessary.

The output listing from this job lists the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifier for the VisualAge COBOL MLE for VSE component has the format COBOLMLE...1.1.0.

f. Check the output from the batch job created by the previous steps to ensure that the install was successful. Once the batch job created by this step has successfully run, the program identifiers of the optional programs on the distribution tape are also automatically entered on the **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu.

g. After the batch job created by the previous step has successfully run, return to the **VSE/ESA FUNCTION SELECTION** menu:

===> 1 (Installation)

h. **INSTALLATION** menu:

===> 1 (Install Programs - V2 Format)

i. **INSTALL PROGRAMS - V2 FORMAT** menu:

===> 2 (Install Program(s) from Tape)

j. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

Enter 1 (install) in the OPT field against the required identifier of the format COBOLMLE...1.1.0, and 2 (skip installation) against any other optional licensed programs you do not intend to install at this time.

VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE. If you did not install COBOL/VSE in the default sublibrary PRD2.PROD, enter the name of the library and sublibrary where COBOL/VSE is installed on this screen.

k. Press PF5 to generate the installation job.

l. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> 1 (Save the list)

m. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> cuu (address of drive with distribution tape)

n. **JOB DISPOSITION** menu:

```
|          Make any changes required and press ENTER to submit the job and
|          install VisualAge COBOL MLE for VSE.  While the job is running,
|          reply to any system console messages as necessary.


|      o.  Check the output from the batch job created by the previous steps
|          to ensure that the install was successful.  If you are installing
|          VisualAge COBOL MLE for VSE from a non-stacked tape, you received
|          the following message:


|            IESI0083I  TAPE IS NOT V2-STACKED


|          This message is for information only and can be ignored.  The
|          Librarian RESTORE job ends with a return code of 4.  This is not
|          an error.  Installation was successful.
```

Subtopics:

- 2.2.4.1.1 Condition Code and Messages

IBM Library Server

2.2.4.2 Method 2: Install VisualAge COBOL MLE for VSE Using a Batch Job

A sample batch job to install VisualAge COBOL MLE for VSE using MSHP is shown in Figure 11. This sample job uses the MSHP system history file that already exists as part of the VSE system. This system history file may already be defined in the system standard labels; if not, make sure that DLBL and EXTENT job control statements with the necessary information for the system history file, are included in the job stream.

Depending on how you ordered the VisualAge COBOL MLE for VSE licensed program, you will have received a stacked tape containing one or more optional licensed programs, or a non-stacked tape containing only the VisualAge COBOL MLE for VSE licensed program. The JCL shown in Figure 11 will handle both stacked and non-stacked tapes.

To install VisualAge COBOL MLE for VSE using a batch job, create and tailor the job stream shown in Figure 11, mount the distribution tape, and run the installation job.

There are four main sections in the sample job that you will need to tailor to suit the requirements of your installation. The tailoring requirements for each section are discussed in the notes that follow Figure 11.

```
// JOB IGYMLINS
* ------------------------------------------------------------ *
*   LICENSED MATERIALS - PROPERTY OF IBM                       *
*                                                              *
*   5686-MLE                                                   *
*                                                              *
*   (C) COPYRIGHT IBM CORP. 1998. ALL RIGHTS RESERVED.         *
*                                                              *
*   US GOVERNMENT USERS RESTRICTED RIGHTS - USE,               *
*   DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP            *
*   SCHEDULE CONTRACT WITH IBM CORP.                           *
*                                                              *
* ------------------------------------------------------------ *
* --------------------- Description -------------------------- *
*  Name       : IGYMLINS                                      *
*  Description : Sample job to install VISUALAGE COBOL/VSE MLE  *
*                See the Installation & Customization Guide    *
*                for the tailoring requirements.               *
*                                                              *
* ------------------------------------------------------------ *
// OPTION LOG
*
*  Label information for the COBOL/VSE library if required
*
*  ____________________________________________ 1
*
// DLBL COBVSE,'COBVSE.LIBRARY',99/365,SD
// EXTENT SYS002,volser,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volser,SHR
*
*  Assign for the distribution tape.
*
```

```
// ASSGN SYS006,cuu  _____________________________  2
// MTC REW,SYS006
*  ---------------------------------------------------------------
*  Install VisualAge COBOL MLE for VSE from the distribution tape
*  ---------------------------------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'  _________  3
 INSTALL PROD FROMTAPE ID='COBOLMLE...1.1.0' -
   PROD INTO=PRD2.PROD
/*
*
*  Retrace VisualAge COBOL MLE
*
// EXEC MSHP,SIZE=900K  _________________________  4
RETRACE COMPONENT IDENTIFIER=5686-MLE-00
/*
// MTC RUN,SYS006
/*
/&
```

Figure 11. Job to Install VisualAge COBOL MLE for VSE

1 Specify the label information.

You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a library other than the default (PRD2), code the DLBL job control statement (and EXTENT and ASSGN statements if necessary) of the library in which you installed COBOL/VSE.

Usually, you do not need to specify label information for the system history file. Your installation should have a permanent system standard label for this, with IJSYSHF as the filename. (IJSYSHF is the default filename that MSHP looks for in a label statement.)

2 Assign the distribution tape.

Replace *cuu* with the address of the tape drive on which you have mounted the distribution tape.

3 Install VisualAge COBOL MLE for VSE.

This job step invokes MSHP to install VisualAge COBOL MLE for VSE into the sublibrary identified on the INTO operand of the INSTALL statement. You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a sublibrary other than the default PRD2.PROD, change the name of the sublibrary on the INTO operand of each INSTALL statement to the name of the sublibrary into which you installed COBOL/VSE. For example, if you installed COBOL/VSE into sublibrary COBVSE.BASE, change references to PRD2.PROD to COBVSE.BASE.

For more information about the MSHP install options, see *VSE/ESA System Control Statements*.

4 Retrace the VisualAge COBOL MLE for VSE product in the system history file.

This step prints the component records from the system history file for VisualAge COBOL MLE for VSE. Remove this step if an MSHP retrace listing is not required.

If the install job runs successfully, it should finish with a return code of zero.

If you need to rerun this install job, make sure you first restore the system history file, which you should have backed up before you started the install process.

IBM Library Server

| 2.2.3 Step 2: Back Up the Original System

| Make a backup copy of the library you intend to install VisualAge COBOL
| MLE for VSE into, and the system history file.

| For information about backing up libraries and the system history file,
| see *VSE/ESA System Control Statements*.

IBM Library Server

| 2.2.4 Step 3: Install VisualAge COBOL MLE for VSE

| You can install VisualAge COBOL MLE for VSE using either the VSE/ESA
| Interactive Interface or a batch installation job.

Subtopics:

IBM Library Server

| 2.2.4.1.1 Condition Code and Messages

| If you receive a condition code greater than 4:

- Check the list output for error conditions.
- See *VSE/ESA Messages and Codes* for corrective action.
- Correct the error.
- Rerun the job.
- Recheck the condition code.

IBM Library Server

| 2.2.4 Step 3: Install VisualAge COBOL MLE for VSE

| You can install VisualAge COBOL MLE for VSE using either the VSE/ESA
| Interactive Interface or a batch installation job.

Subtopics:

IBM Library Server

| 2.2.4.1.1 Condition Code and Messages

| If you receive a condition code greater than 4:

- Check the list output for error conditions.
- See *VSE/ESA Messages and Codes* for corrective action.
- Correct the error.
- Rerun the job.
- Recheck the condition code.

IBM Library Server

| 2.2.4.1.1 Condition Code and Messages

| If you receive a condition code greater than 4:

- Check the list output for error conditions.
- See *VSE/ESA Messages and Codes* for corrective action.
- Correct the error.
- Rerun the job.
- Recheck the condition code.

IBM Library Server

2.2.4.2 Method 2: Install VisualAge COBOL MLE for VSE Using a Batch Job

A sample batch job to install VisualAge COBOL MLE for VSE using MSHP is shown in Figure 11. This sample job uses the MSHP system history file that already exists as part of the VSE system. This system history file may already be defined in the system standard labels; if not, make sure that DLBL and EXTENT job control statements with the necessary information for the system history file, are included in the job stream.

Depending on how you ordered the VisualAge COBOL MLE for VSE licensed program, you will have received a stacked tape containing one or more optional licensed programs, or a non-stacked tape containing only the VisualAge COBOL MLE for VSE licensed program. The JCL shown in Figure 11 will handle both stacked and non-stacked tapes.

To install VisualAge COBOL MLE for VSE using a batch job, create and tailor the job stream shown in Figure 11, mount the distribution tape, and run the installation job.

There are four main sections in the sample job that you will need to tailor to suit the requirements of your installation. The tailoring requirements for each section are discussed in the notes that follow Figure 11.

```
// JOB IGYMLINS
* ------------------------------------------------------------ *
*   LICENSED MATERIALS - PROPERTY OF IBM                       *
*                                                              *
*   5686-MLE                                                   *
*                                                              *
*   (C) COPYRIGHT IBM CORP. 1998. ALL RIGHTS RESERVED.         *
*                                                              *
*   US GOVERNMENT USERS RESTRICTED RIGHTS - USE,               *
*   DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP            *
*   SCHEDULE CONTRACT WITH IBM CORP.                           *
*                                                              *
* ------------------------------------------------------------ *
* -------------------- Description --------------------------- *
*  Name       : IGYMLINS                                      *
*  Description : Sample job to install VISUALAGE COBOL/VSE MLE  *
*                See the Installation & Customization Guide    *
*                for the tailoring requirements.               *
*                                                              *
* ------------------------------------------------------------ *
// OPTION LOG
*
*  Label information for the COBOL/VSE library if required
*
*  ____________________________________________ 1
*
// DLBL COBVSE,'COBVSE.LIBRARY',99/365,SD
// EXTENT SYS002,volser,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volser,SHR
*
*  Assign for the distribution tape.
*
```

```
// ASSGN SYS006,cuu  _____________________________  2
// MTC REW,SYS006
*  ---------------------------------------------------------------
*  Install VisualAge COBOL MLE for VSE from the distribution tape
*  ---------------------------------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'  _________  3
 INSTALL PROD FROMTAPE ID='COBOLMLE...1.1.0' -
   PROD INTO=PRD2.PROD
/*
*
*  Retrace VisualAge COBOL MLE
*
// EXEC MSHP,SIZE=900K  _________________________  4
RETRACE COMPONENT IDENTIFIER=5686-MLE-00
/*
// MTC RUN,SYS006
/*
/&
```

Figure 11. Job to Install VisualAge COBOL MLE for VSE

1 Specify the label information.

You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a library other than the default (PRD2), code the DLBL job control statement (and EXTENT and ASSGN statements if necessary) of the library in which you installed COBOL/VSE.

Usually, you do not need to specify label information for the system history file. Your installation should have a permanent system standard label for this, with IJSYSHF as the filename. (IJSYSHF is the default filename that MSHP looks for in a label statement.)

2 Assign the distribution tape.

Replace *cuu* with the address of the tape drive on which you have mounted the distribution tape.

3 Install VisualAge COBOL MLE for VSE.

This job step invokes MSHP to install VisualAge COBOL MLE for VSE into the sublibrary identified on the INTO operand of the INSTALL statement. You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a sublibrary other than the default PRD2.PROD, change the name of the sublibrary on the INTO operand of each INSTALL statement to the name of the sublibrary into which you installed COBOL/VSE. For example, if you installed COBOL/VSE into sublibrary COBVSE.BASE, change references to PRD2.PROD to COBVSE.BASE.

For more information about the MSHP install options, see *VSE/ESA System Control Statements*.

4 Retrace the VisualAge COBOL MLE for VSE product in the system history file.

This step prints the component records from the system history file for VisualAge COBOL MLE for VSE. Remove this step if an MSHP retrace listing is not required.

If the install job runs successfully, it should finish with a return code of zero.

If you need to rerun this install job, make sure you first restore the system history file, which you should have backed up before you started the install process.

IBM Library Server

| 2.2.5.1 Verify Date Processing in COBOL/VSE

| The sample job IGYMLIV1, in the installation sublibrary member IGYMLIV1.Z,
| compiles, link-edits, and runs the COBOL program CALLIVPM installed from
| your distribution tape as sublibrary member IGYMLIVP.C. The object code
| produced by the COBOL/VSE compiler is link-edited and run using the
| libraries created when LE/VSE was installed. You must install LE/VSE and
| apply the appropriate PTFs before running IGYMLIV1.

| **Note:** If you installed VisualAge COBOL MLE for VSE using the VSE/ESA
| Interactive Interface dialogs, IGYMLIV1 is also available in ICCF library
| 62.

| **Modifying the JCL for IGYMLIV1**

| 1. Modify the job card as appropriate for your site.

| 2. Add POWER JECL statements if your site requires them.

| 3. If necessary, change the LIBDEF statement to match the sublibraries
| where you have installed COBOL/VSE, VisualAge COBOL MLE for VSE, and
| LE/VSE.

| After you modify the IGYMLIV1 job, submit it. The job finishes with a
| return code of 2 if it runs successfully. The return code of 2 indicates
| there were unresolved weak external references during the link-edit step.
| This return code is normal and does not indicate a problem.

| IGYMLIV1 prints the following messages to SYSLST:

```
|    ***** START OF CALLIVPM *****
|    ***** CALLIVPM SUCCESSFUL *****
```

| **Note:** If VisualAge COBOL MLE for VSE was properly installed, IGYMLIV1
| should not produce any LE/VSE run-time messages.

IBM Library Server

2.2.4.2 Method 2: Install VisualAge COBOL MLE for VSE Using a Batch Job

A sample batch job to install VisualAge COBOL MLE for VSE using MSHP is shown in Figure 11. This sample job uses the MSHP system history file that already exists as part of the VSE system. This system history file may already be defined in the system standard labels; if not, make sure that DLBL and EXTENT job control statements with the necessary information for the system history file, are included in the job stream.

Depending on how you ordered the VisualAge COBOL MLE for VSE licensed program, you will have received a stacked tape containing one or more optional licensed programs, or a non-stacked tape containing only the VisualAge COBOL MLE for VSE licensed program. The JCL shown in Figure 11 will handle both stacked and non-stacked tapes.

To install VisualAge COBOL MLE for VSE using a batch job, create and tailor the job stream shown in Figure 11, mount the distribution tape, and run the installation job.

There are four main sections in the sample job that you will need to tailor to suit the requirements of your installation. The tailoring requirements for each section are discussed in the notes that follow Figure 11.

```
// JOB IGYMLINS
* ------------------------------------------------------------ *
*   LICENSED MATERIALS - PROPERTY OF IBM                       *
*                                                              *
*   5686-MLE                                                   *
*                                                              *
*   (C) COPYRIGHT IBM CORP. 1998. ALL RIGHTS RESERVED.         *
*                                                              *
*   US GOVERNMENT USERS RESTRICTED RIGHTS - USE,               *
*   DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP            *
*   SCHEDULE CONTRACT WITH IBM CORP.                           *
*                                                              *
* ------------------------------------------------------------ *
* -------------------- Description --------------------------- *
*  Name       : IGYMLINS                                      *
*  Description : Sample job to install VISUALAGE COBOL/VSE MLE  *
*                See the Installation & Customization Guide    *
*                for the tailoring requirements.               *
*                                                              *
* ------------------------------------------------------------ *
// OPTION LOG
*
*  Label information for the COBOL/VSE library if required
*
*  _____________________________________________ 1
*
// DLBL COBVSE,'COBVSE.LIBRARY',99/365,SD
// EXTENT SYS002,volser,,,rtrk,ntrk
// ASSGN SYS002,DISK,VOL=volser,SHR
*
*  Assign for the distribution tape.
*
```

```
// ASSGN SYS006,cuu  _____________________________  2
// MTC REW,SYS006
*  ---------------------------------------------------------------
*  Install VisualAge COBOL MLE for VSE from the distribution tape
*  ---------------------------------------------------------------
// EXEC MSHP,SIZE=900K,PARM='PIDSTACKED'  _________  3
 INSTALL PROD FROMTAPE ID='COBOLMLE...1.1.0' -
   PROD INTO=PRD2.PROD
/*
*
*  Retrace VisualAge COBOL MLE
*
// EXEC MSHP,SIZE=900K  _________________________  4
RETRACE COMPONENT IDENTIFIER=5686-MLE-00
/*
// MTC RUN,SYS006
/*
/&
```

Figure 11. Job to Install VisualAge COBOL MLE for VSE

1 Specify the label information.

You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a library other than the default (PRD2), code the DLBL job control statement (and EXTENT and ASSGN statements if necessary) of the library in which you installed COBOL/VSE.

Usually, you do not need to specify label information for the system history file. Your installation should have a permanent system standard label for this, with IJSYSHF as the filename. (IJSYSHF is the default filename that MSHP looks for in a label statement.)

2 Assign the distribution tape.

Replace *cuu* with the address of the tape drive on which you have mounted the distribution tape.

3 Install VisualAge COBOL MLE for VSE.

This job step invokes MSHP to install VisualAge COBOL MLE for VSE into the sublibrary identified on the INTO operand of the INSTALL statement. You should install VisualAge COBOL MLE for VSE in the same sublibrary as COBOL/VSE. If you installed COBOL/VSE into a sublibrary other than the default PRD2.PROD, change the name of the sublibrary on the INTO operand of each INSTALL statement to the name of the sublibrary into which you installed COBOL/VSE. For example, if you installed COBOL/VSE into sublibrary COBVSE.BASE, change references to PRD2.PROD to COBVSE.BASE.

For more information about the MSHP install options, see *VSE/ESA System Control Statements*.

4 Retrace the VisualAge COBOL MLE for VSE product in the system history file.

This step prints the component records from the system history file for VisualAge COBOL MLE for VSE. Remove this step if an MSHP retrace listing is not required.

If the install job runs successfully, it should finish with a return code of zero.

If you need to rerun this install job, make sure you first restore the system history file, which you should have backed up before you started the install process.

IBM Library Server

| 2.2.5 Step 4: Verify the Installation of VisualAge COBOL MLE for VSE

| After you install VisualAge COBOL MLE for VSE, run sample job IGYMLIV1 to
| verify automatic date processing in COBOL/VSE.

Subtopics:

- 2.2.5.1 Verify Date Processing in COBOL/VSE

IBM Library Server

| 2.2.5 Step 4: Verify the Installation of VisualAge COBOL MLE for VSE

| After you install VisualAge COBOL MLE for VSE, run sample job IGYMLIV1 to
| verify automatic date processing in COBOL/VSE.

Subtopics:

- 2.2.5.1 Verify Date Processing in COBOL/VSE

IBM Library Server

2.2.5.1 Verify Date Processing in COBOL/VSE

The sample job IGYMLIV1, in the installation sublibrary member IGYMLIV1.Z, compiles, link-edits, and runs the COBOL program CALLIVPM installed from your distribution tape as sublibrary member IGYMLIVP.C. The object code produced by the COBOL/VSE compiler is link-edited and run using the libraries created when LE/VSE was installed. You must install LE/VSE and apply the appropriate PTFs before running IGYMLIV1.

Note: If you installed VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface dialogs, IGYMLIV1 is also available in ICCF library 62.

Modifying the JCL for IGYMLIV1

---

1. Modify the job card as appropriate for your site.
2. Add POWER JECL statements if your site requires them.
3. If necessary, change the LIBDEF statement to match the sublibraries where you have installed COBOL/VSE, VisualAge COBOL MLE for VSE, and LE/VSE.

After you modify the IGYMLIV1 job, submit it. The job finishes with a return code of 2 if it runs successfully. The return code of 2 indicates there were unresolved weak external references during the link-edit step. This return code is normal and does not indicate a problem.

IGYMLIV1 prints the following messages to SYSLST:

```
   ***** START OF CALLIVPM *****
   ***** CALLIVPM SUCCESSFUL *****
```

Note: If VisualAge COBOL MLE for VSE was properly installed, IGYMLIV1 should not produce any LE/VSE run-time messages.

IBM Library Server

| 2.2.5.1 Verify Date Processing in COBOL/VSE

| The sample job IGYMLIV1, in the installation sublibrary member IGYMLIV1.Z,
| compiles, link-edits, and runs the COBOL program CALLIVPM installed from
| your distribution tape as sublibrary member IGYMLIVP.C. The object code
| produced by the COBOL/VSE compiler is link-edited and run using the
| libraries created when LE/VSE was installed. You must install LE/VSE and
| apply the appropriate PTFs before running IGYMLIV1.

| **Note:** If you installed VisualAge COBOL MLE for VSE using the VSE/ESA
| Interactive Interface dialogs, IGYMLIV1 is also available in ICCF library
| 62.

| **Modifying the JCL for IGYMLIV1**

| 1. Modify the job card as appropriate for your site.

| 2. Add POWER JECL statements if your site requires them.

| 3. If necessary, change the LIBDEF statement to match the sublibraries
| where you have installed COBOL/VSE, VisualAge COBOL MLE for VSE, and
| LE/VSE.

| After you modify the IGYMLIV1 job, submit it. The job finishes with a
| return code of 2 if it runs successfully. The return code of 2 indicates
| there were unresolved weak external references during the link-edit step.
| This return code is normal and does not indicate a problem.

| IGYMLIV1 prints the following messages to SYSLST:

```
|    ***** START OF CALLIVPM *****
|    ***** CALLIVPM SUCCESSFUL *****
```

| **Note:** If VisualAge COBOL MLE for VSE was properly installed, IGYMLIV1
| should not produce any LE/VSE run-time messages.

IBM Library Server

2.2.6 Step 5: Place VisualAge COBOL MLE for VSE in the SVA

After you have verified the successful installation of VisualAge COBOL MLE for VSE, you might want to place VisualAge COBOL MLE for VSE in the SVA.

Note: Placing VisualAge COBOL MLE for VSE in the SVA must be done by a system administrator.

To assist you in placing the VisualAge COBOL MLE for VSE phase in the SVA, a sample job is provided in member IGYMLSV1.Z. This sample job is shown in Figure 12. See the notes following Figure 12 for information about tailoring the sample job.

```
// JOB IGYMLSV1 LOAD SVA-ELIGIBLE PHASE
*
*  Load SVA
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
IGYCMLE,SVA
/*
/&
```

Figure 12. Job to Place VisualAge COBOL MLE for VSE in the SVA

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

Alternatively, as shown by the sample job in Figure 13, you can place the VisualAge COBOL MLE for VSE phase in the SVA using a load list. To help you do this, the load list $SVAIGYM is provided. You can also specify this load list in the BG startup procedure. If you choose to specify the load list in the BG startup procedure, the sublibrary where you installed VisualAge COBOL MLE for VSE must be in the LIBDEF search chain within the LIBSDL procedure.

```
// JOB IGYMLSVA LOAD SVA-ELIGIBLE PHASES
*
```

```
*  Load SVA using a load list
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
LIST=$SVAIGYM
/*
/&
```

Figure 13. Job to Place VisualAge COBOL MLE for VSE in the SVA Using a Load List

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

For more information about placing phases in the SVA, see *VSE/ESA System Control Statements*.

IBM Library Server

2.3.1 Reinstalling VisualAge COBOL MLE for VSE

You do not need to remove VisualAge COBOL MLE for VSE from your system before reinstalling VisualAge COBOL MLE for VSE unless you intend to reinstall the product in a different sublibrary from the previous installation. In this case you must remove VisualAge COBOL MLE for VSE from the system history file before you can reinstall it. However, if you are reinstalling in the same sublibrary, you might need to delete VisualAge COBOL MLE for VSE from the sublibrary and release the space to ensure that there is sufficient library space available for the reinstallation.

If you do need to remove VisualAge COBOL MLE for VSE Release 1 from your system, sample jobs, IGYMLDLV.Z and IGYMLDLH.Z are provided in PRD2.PROD to help you do this.

If you installed VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface dialogs, these jobs will also be available in ICCF library 62.

To reinstall VisualAge COBOL MLE for VSE, follow the installation steps described in Chapter 7, "Installing VisualAge COBOL MLE for VSE" in topic 2.2.

IBM Library Server

| 2.3.2 Applying Service Updates

| You might need to apply maintenance or service updates to VisualAge COBOL
| MLE for VSE periodically. There are two types of formally supported
| software fixes. One is the program temporary fix (PTF) applied as
| corrective service or as preventive maintenance. The other is the
| authorized program analysis report (APAR) fix applied as a code
| replacement in a corrective maintenance mode.

Subtopics:

IBM Library Server

| 2.3.3 To Report a Problem with VisualAge COBOL MLE for VSE

| Report any difficulties you have using this product to your IBM Support
| Center. Table 30 identifies the component ID and Release Level for
| VisualAge COBOL MLE for VSE.

Table 30. Component IDs

Component Id	Component Description	REL
5686-MLE-00	VA COBOL MLE for VSE	1JQ

IBM Library Server

2.2.6 Step 5: Place VisualAge COBOL MLE for VSE in the SVA

After you have verified the successful installation of VisualAge COBOL MLE for VSE, you might want to place VisualAge COBOL MLE for VSE in the SVA.

Note: Placing VisualAge COBOL MLE for VSE in the SVA must be done by a system administrator.

To assist you in placing the VisualAge COBOL MLE for VSE phase in the SVA, a sample job is provided in member IGYMLSV1.Z. This sample job is shown in Figure 12. See the notes following Figure 12 for information about tailoring the sample job.

```
// JOB IGYMLSV1 LOAD SVA-ELIGIBLE PHASE
*
*  Load SVA
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
IGYCMLE,SVA
/*
/&
```

Figure 12. Job to Place VisualAge COBOL MLE for VSE in the SVA

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

Alternatively, as shown by the sample job in Figure 13, you can place the VisualAge COBOL MLE for VSE phase in the SVA using a load list. To help you do this, the load list $SVAIGYM is provided. You can also specify this load list in the BG startup procedure. If you choose to specify the load list in the BG startup procedure, the sublibrary where you installed VisualAge COBOL MLE for VSE must be in the LIBDEF search chain within the LIBSDL procedure.

```
// JOB IGYMLSVA LOAD SVA-ELIGIBLE PHASES
*
```

```
*  Load SVA using a load list
*
// LIBDEF *,SEARCH=(PRD2.PROD)   _______ 1
SET SDL
LIST=$SVAIGYM
/*
/&
```

Figure 13. Job to Place VisualAge COBOL MLE for VSE in the SVA Using a Load List

1 If you installed VisualAge COBOL MLE for VSE into a sublibrary other than the default (PRD2.PROD), modify the LIBDEF job control statement to specify the sublibrary in which you installed VisualAge COBOL MLE for VSE.

For more information about placing phases in the SVA, see *VSE/ESA System Control Statements*.

IBM Library Server

2.3 Chapter 8. Maintaining VisualAge COBOL MLE for VSE

This chapter describes how to replace or reinstall VisualAge COBOL MLE for VSE, and how to apply service updates to VisualAge COBOL MLE for VSE. To effectively use the maintenance procedures, you must have already installed VisualAge COBOL MLE for VSE and any required products.

In addition, this chapter describes how to remove VisualAge COBOL MLE for VSE from your system.

Subtopics:

IBM Library Server

2.3 Chapter 8. Maintaining VisualAge COBOL MLE for VSE

This chapter describes how to replace or reinstall VisualAge COBOL MLE for VSE, and how to apply service updates to VisualAge COBOL MLE for VSE. To effectively use the maintenance procedures, you must have already installed VisualAge COBOL MLE for VSE and any required products.

In addition, this chapter describes how to remove VisualAge COBOL MLE for VSE from your system.

Subtopics:

IBM Library Server

2.3.1 Reinstalling VisualAge COBOL MLE for VSE

You do not need to remove VisualAge COBOL MLE for VSE from your system before reinstalling VisualAge COBOL MLE for VSE unless you intend to reinstall the product in a different sublibrary from the previous installation. In this case you must remove VisualAge COBOL MLE for VSE from the system history file before you can reinstall it. However, if you are reinstalling in the same sublibrary, you might need to delete VisualAge COBOL MLE for VSE from the sublibrary and release the space to ensure that there is sufficient library space available for the reinstallation.

If you do need to remove VisualAge COBOL MLE for VSE Release 1 from your system, sample jobs, IGYMLDLV.Z and IGYMLDLH.Z are provided in PRD2.PROD to help you do this.

If you installed VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface dialogs, these jobs will also be available in ICCF library 62.

To reinstall VisualAge COBOL MLE for VSE, follow the installation steps described in Chapter 7, "Installing VisualAge COBOL MLE for VSE" in topic 2.2.

IBM Library Server

| 2.3.2.1 What You Receive

| If you report a problem with VisualAge COBOL MLE for VSE to your IBM
| Support Center, you will receive a tape containing one or more APARs or
| PTFs which have been created to solve your problem.

| You may also receive a list of prerequisite APARs or PTFs which should
| have been applied to your system before applying the current service.
| These prerequisite APARs or PTFs may relate to VisualAge COBOL MLE for VSE
| or any other licensed product you have installed, including VSE/ESA.

| You apply service to VisualAge COBOL MLE for VSE with MSHP, using either
| the VSE/ESA Interactive Interface or a batch job.

| The following checklist provides a summary of steps you should use to
| apply service to VisualAge COBOL MLE for VSE.

IBM Library Server

| 2.3.2.2 Checklist for Applying Service

| Table 29 lists the steps for installing corrective service on VisualAge COBOL MLE for VSE. You can use Table 29 as a checklist.

Table 29. Summary of Steps for Installing Service on VisualAge COBOL MLE for VSE

Step	Description	MSHP Command or Jobname	Topic
1	Ensure prerequisite APARs or PTFs are applied	IGYMLRTR	2.3.2.3
2	Backup existing system	-	2.3.2.4
3	Apply service	IGYMLSVC	2.3.2.5
4	Run the installation verification programs	IGYMLIV1	2.3.2.6

IBM Library Server

2.3.2.3 Step 1: Check Prerequisite APARs or PTFs

Prerequisite APARs or PTFs are APARs or PTFs that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to VisualAge COBOL MLE for VSE or any licensed program you have installed at your site.

Your IBM Support Center will have given you a list of any relevant prerequisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 14 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYMLRTR Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 14. Job to Retrace APARs and PTFs

Use the listing produced when you run this job to check that you have already applied any prerequisite APARs or PTFs. If you have not, your IBM Support Center will arrange to send them to you and you should apply them before applying other service.

| 2.3.2.4 Step 2: Backup Existing System

| Make a backup copy of the library in which VisualAge COBOL MLE for VSE is
| installed, and the system history file. For information about backing up
| libraries and the system history file, see *VSE/ESA System Control
| Statements*.

IBM Library Server

2.3.2.5 Step 3: Apply Service

You can apply service to VisualAge COBOL MLE for VSE from the provided service tape using either the Interactive Interface dialogs or a batch job.

You will receive detailed instructions for applying service with the service tape.

Subtopics:

IBM Library Server

| 2.3.2.6 Step 4: Run the Installation Verification Program (IVP)

| After you have applied all the files on the service tape, run the
| appropriate installation verification programs to ensure that VisualAge
| COBOL MLE for VSE functions properly. For information about how to run
| the installation verification programs, see "Step 4: Verify the Installation of VisualAge COBOL MLE for VSE" in topic 2.2.5.

IBM Library Server

2.3.1 Reinstalling VisualAge COBOL MLE for VSE

You do not need to remove VisualAge COBOL MLE for VSE from your system before reinstalling VisualAge COBOL MLE for VSE unless you intend to reinstall the product in a different sublibrary from the previous installation. In this case you must remove VisualAge COBOL MLE for VSE from the system history file before you can reinstall it. However, if you are reinstalling in the same sublibrary, you might need to delete VisualAge COBOL MLE for VSE from the sublibrary and release the space to ensure that there is sufficient library space available for the reinstallation.

If you do need to remove VisualAge COBOL MLE for VSE Release 1 from your system, sample jobs, IGYMLDLV.Z and IGYMLDLH.Z are provided in PRD2.PROD to help you do this.

If you installed VisualAge COBOL MLE for VSE using the VSE/ESA Interactive Interface dialogs, these jobs will also be available in ICCF library 62.

To reinstall VisualAge COBOL MLE for VSE, follow the installation steps described in Chapter 7, "Installing VisualAge COBOL MLE for VSE" in topic 2.2.

IBM Library Server

| 2.3.2 Applying Service Updates

| You might need to apply maintenance or service updates to VisualAge COBOL
| MLE for VSE periodically. There are two types of formally supported
| software fixes. One is the program temporary fix (PTF) applied as
| corrective service or as preventive maintenance. The other is the
| authorized program analysis report (APAR) fix applied as a code
| replacement in a corrective maintenance mode.

Subtopics:

IBM Library Server

| 2.3.2 Applying Service Updates

| You might need to apply maintenance or service updates to VisualAge COBOL
| MLE for VSE periodically. There are two types of formally supported
| software fixes. One is the program temporary fix (PTF) applied as
| corrective service or as preventive maintenance. The other is the
| authorized program analysis report (APAR) fix applied as a code
| replacement in a corrective maintenance mode.

Subtopics:

IBM Library Server

| 2.3.2.1 What You Receive

| If you report a problem with VisualAge COBOL MLE for VSE to your IBM
| Support Center, you will receive a tape containing one or more APARs or
| PTFs which have been created to solve your problem.

| You may also receive a list of prerequisite APARs or PTFs which should
| have been applied to your system before applying the current service.
| These prerequisite APARs or PTFs may relate to VisualAge COBOL MLE for VSE
| or any other licensed product you have installed, including VSE/ESA.

| You apply service to VisualAge COBOL MLE for VSE with MSHP, using either
| the VSE/ESA Interactive Interface or a batch job.

| The following checklist provides a summary of steps you should use to
| apply service to VisualAge COBOL MLE for VSE.

IBM Library Server

| 2.3.2.1 What You Receive

| If you report a problem with VisualAge COBOL MLE for VSE to your IBM
| Support Center, you will receive a tape containing one or more APARs or
| PTFs which have been created to solve your problem.

| You may also receive a list of prerequisite APARs or PTFs which should
| have been applied to your system before applying the current service.
| These prerequisite APARs or PTFs may relate to VisualAge COBOL MLE for VSE
| or any other licensed product you have installed, including VSE/ESA.

| You apply service to VisualAge COBOL MLE for VSE with MSHP, using either
| the VSE/ESA Interactive Interface or a batch job.

| The following checklist provides a summary of steps you should use to
| apply service to VisualAge COBOL MLE for VSE.

IBM Library Server

| 2.3.2.2 Checklist for Applying Service

| Table 29 lists the steps for installing corrective service on VisualAge COBOL MLE for VSE. You can use Table 29 as a checklist.

Table 29. Summary of Steps for Installing Service on VisualAge COBOL MLE for VSE

Step	Description	MSHP Command or Jobname	Topic
1	Ensure prerequisite APARs or PTFs are applied	IGYMLRTR	2.3.2.3
2	Backup existing system	-	2.3.2.4
3	Apply service	IGYMLSVC	2.3.2.5
4	Run the installation verification programs	IGYMLIV1	2.3.2.6

IBM Library Server

2.3.2.2 Checklist for Applying Service

Table 29 lists the steps for installing corrective service on VisualAge COBOL MLE for VSE. You can use Table 29 as a checklist.

Table 29. Summary of Steps for Installing Service on VisualAge COBOL MLE for VSE

Step	Description	MSHP Command or Jobname	Topic
1	Ensure prerequisite APARs or PTFs are applied	IGYMLRTR	2.3.2.3
2	Backup existing system	-	2.3.2.4
3	Apply service	IGYMLSVC	2.3.2.5
4	Run the installation verification programs	IGYMLIV1	2.3.2.6

IBM Library Server

2.3.2.3 Step 1: Check Prerequisite APARs or PTFs

Prerequisite APARs or PTFs are APARs or PTFs that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to VisualAge COBOL MLE for VSE or any licensed program you have installed at your site.

Your IBM Support Center will have given you a list of any relevant prerequisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 14 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYMLRTR Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 14. Job to Retrace APARs and PTFs

Use the listing produced when you run this job to check that you have already applied any prerequisite APARs or PTFs. If you have not, your IBM Support Center will arrange to send them to you and you should apply them before applying other service.

IBM Library Server

2.3.2.3 Step 1: Check Prerequisite APARs or PTFs

Prerequisite APARs or PTFs are APARs or PTFs that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to VisualAge COBOL MLE for VSE or any licensed program you have installed at your site.

Your IBM Support Center will have given you a list of any relevant prerequisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 14 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYMLRTR Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 14. Job to Retrace APARs and PTFs

Use the listing produced when you run this job to check that you have already applied any prerequisite APARs or PTFs. If you have not, your IBM Support Center will arrange to send them to you and you should apply them before applying other service.

IBM Library Server

| 2.3.2.4 Step 2: Backup Existing System

| Make a backup copy of the library in which VisualAge COBOL MLE for VSE is
| installed, and the system history file. For information about backing up
| libraries and the system history file, see *VSE/ESA System Control
| Statements*.

IBM Library Server

| 2.3.2.5.1 Method 1: Apply Service Using the Interactive Interface

| The VSE/ESA Interactive Interface enables you to use dialog requests to
| apply service to VisualAge COBOL MLE for VSE. For more information about
| the functions of the Interactive Interface, see *VSE/ESA Administration*.
| For more information on how to use the dialogs to apply service, see
| *VSE/ESA System Upgrade and Service*.

IBM Library Server

| 2.3.2.5.2 Method 2: Apply Service Using a Batch Job

| The batch job to apply service to VisualAge COBOL MLE for VSE uses the
| MSHP system history file to determine where VisualAge COBOL MLE for VSE
| was installed.

| A sample job to apply service using the Maintain System History Program
| (MSHP) is shown in Figure 15. For more information on MSHP see *VSE/ESA*
| *System Control Statements*.

```
| // JOB IGYMLSVC
| // ASSGN SYS006,cuu _____________________________ 1
| // EXEC MSHP,SIZE=900K
| INSTALL SERVICE FROMTAPE  _____________________ 2
| /*
| /&
```

| Figure 15. Job to Apply Service

| **1** Specify the address of the tape drive.

| Change *cuu* to the address of the tape drive where you have mounted
| the service tape.

| **2** Install the service from tape.

| This MSHP statement indicates that the service will be installed from
| tape.

| The information in the system history file will direct MSHP to apply
| the service to the sublibrary in which VisualAge COBOL MLE for VSE is
| installed. You do not need to supply this information.

IBM Library Server

| 2.3.2.4 Step 2: Backup Existing System

| Make a backup copy of the library in which VisualAge COBOL MLE for VSE is
| installed, and the system history file. For information about backing up
| libraries and the system history file, see *VSE/ESA System Control
| Statements*.

| 2.3.2.5.2 Method 2: Apply Service Using a Batch Job

| The batch job to apply service to VisualAge COBOL MLE for VSE uses the MSHP system history file to determine where VisualAge COBOL MLE for VSE was installed.

| A sample job to apply service using the Maintain System History Program (MSHP) is shown in Figure 15. For more information on MSHP see *VSE/ESA System Control Statements*.

```
| // JOB IGYMLSVC
| // ASSGN SYS006,cuu _____________________________ 1
| // EXEC MSHP,SIZE=900K
| INSTALL SERVICE FROMTAPE  _____________________ 2
| /*
| /&
```

| Figure 15. Job to Apply Service

| **1** Specify the address of the tape drive.

| Change *cuu* to the address of the tape drive where you have mounted the service tape.

| **2** Install the service from tape.

| This MSHP statement indicates that the service will be installed from tape.

| The information in the system history file will direct MSHP to apply the service to the sublibrary in which VisualAge COBOL MLE for VSE is installed. You do not need to supply this information.

IBM Library Server

| 2.3.2.5.2 Method 2: Apply Service Using a Batch Job

| The batch job to apply service to VisualAge COBOL MLE for VSE uses the
| MSHP system history file to determine where VisualAge COBOL MLE for VSE
| was installed.

| A sample job to apply service using the Maintain System History Program
| (MSHP) is shown in Figure 15. For more information on MSHP see *VSE/ESA*
| *System Control Statements*.

```
| // JOB IGYMLSVC
| // ASSGN SYS006,cuu _____________________________ 1
| // EXEC MSHP,SIZE=900K
| INSTALL SERVICE FROMTAPE  _____________________ 2
| /*
| /&
```

| Figure 15. Job to Apply Service

| **1** Specify the address of the tape drive.

| Change *cuu* to the address of the tape drive where you have mounted
| the service tape.

| **2** Install the service from tape.

| This MSHP statement indicates that the service will be installed from
| tape.

| The information in the system history file will direct MSHP to apply
| the service to the sublibrary in which VisualAge COBOL MLE for VSE is
| installed. You do not need to supply this information.

IBM Library Server

| 2.3.2.6 Step 4: Run the Installation Verification Program (IVP)

| After you have applied all the files on the service tape, run the
| appropriate installation verification programs to ensure that VisualAge
| COBOL MLE for VSE functions properly. For information about how to run
| the installation verification programs, see "Step 4: Verify the
| Installation of VisualAge COBOL MLE for VSE" in topic 2.2.5.

IBM Library Server

| 2.3.2.6 Step 4: Run the Installation Verification Program (IVP)

| After you have applied all the files on the service tape, run the
| appropriate installation verification programs to ensure that VisualAge
| COBOL MLE for VSE functions properly. For information about how to run
| the installation verification programs, see "Step 4: Verify the Installation of VisualAge COBOL MLE for VSE" in topic 2.2.5.

IBM Library Server

| 2.3.3 To Report a Problem with VisualAge COBOL MLE for VSE

| Report any difficulties you have using this product to your IBM Support
| Center. Table 30 identifies the component ID and Release Level for
| VisualAge COBOL MLE for VSE.

Table 30. Component IDs

Component Id	Component Description	REL
5686-MLE-00	VA COBOL MLE for VSE	1JQ

IBM Library Server

BACK_1.1 IBM COBOL for VSE/ESA

General Information, GC26-8068

Migration Guide, GC26-8070

Installation and Customization Guide, SC26-8071

Programming Guide, SC26-8072

Language Reference, SC26-8073

Reference Summary, SX26-3834

Diagnosis Guide, SC26-8528

Licensed Program Specifications, GC26-8069

IBM Library Server

BACK_1.2 IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

Installation and Customization Guide, SC26-8071

COBOL Millennium Language Extensions Guide, GC26-9266

Fact Sheet, GC26-9321

Licensed Program Specifications, GC26-9417

IBM Library Server

BACK_1.3 Language Environment Publications

Fact Sheet, GC33-6679

Concepts Guide, GC33-6680

Installation and Customization Guide, SC33-6682

Programming Guide, SC33-6684

Programming Reference, SC33-6685

Debugging Guide and Run-Time Messages, SC33-6681

Licensed Program Specifications, GC33-6683

Run-Time Migration Guide, SC33-6687

Writing Interlanguage Communication Applications, SC33-6686

C Run-Time Library Reference, SC33-6689

C Run-Time Programming Guide, SC33-6688

IBM Library Server

BACK_1.4 Related Publications

CICS/VSE

Application Programming Guide, SC33-0712

Application Programming Reference, SC33-0713

System Definition and Operations Guide, SC33-0706

Resource Definition (Online), SC33-0708

Debug Tool for VSE/ESA

User's Guide and Reference, SC26-8797

Installation and Customization Guide, SC26-8798

DFSORT for VSE/ESA

Application Programming Guide, SC26-7040

DL/I DOS/VS

Application Programming: High-Level Programming Interface, SH24-5009

Application Programming: CALL and RQDLI Interfaces, SH12-5411

Sort/Merge II

DOS/VS VM/SP Sort/Merge Version 2 Application Programming Guide, SC33-4044

SQL/DS

Application Programming for VSE, SH09-8098

VSE/ESA Version 1 Release 4

Planning, SC33-6503

Installation and Service, SC33-6504

Administration, SC33-6505

Guide to System Functions, SC33-6511

System Control Statements, SC33-6513

System Macros User's Guide, SC33-6515

System Macros Reference, SC33-6516

System Utilities, SC33-6517

Messages and Codes Vol.1 & 2, SC33-6507

VSE/ESA Version 2

System Upgrade and Service, SC33-6602

Planning, SC33-6603

Installation, SC33-6604

Administration, SC33-6605

Guide to System Functions, SC33-6611

System Control Statements, SC33-6613

System Macros User's Guide, SC33-6615

System Macros Reference, SC33-6616

System Utilities, SC33-6617

Messages and Codes Vol. 1, SC33-6698

Messages and Codes Vol. 2, SC33-6699

VSE/ESA VSAM

VSE/VSAM Commands and Macros, SC33-6532

VSE/VSAM User's Guide, SC33-6535

IBM Library Server

BACK_1.5 Softcopy Publications

These collection kit contains the COBOL/VSE and LE/VSE-conforming language product publications:

VSE Collection, SK2T-0060

IBM Library Server

| 2.3.3 To Report a Problem with VisualAge COBOL MLE for VSE

| Report any difficulties you have using this product to your IBM Support
| Center. Table 30 identifies the component ID and Release Level for
| VisualAge COBOL MLE for VSE.

Table 30. Component IDs

Component Id	Component Description	REL
5686-MLE-00	VA COBOL MLE for VSE	1JQ

IBM Library Server

BACK_1 Bibliography

Subtopics:

IBM Library Server

BACK_1 Bibliography

Subtopics:

IBM Library Server

BACK_1.1 IBM COBOL for VSE/ESA

General Information, GC26-8068

Migration Guide, GC26-8070

Installation and Customization Guide, SC26-8071

Programming Guide, SC26-8072

Language Reference, SC26-8073

Reference Summary, SX26-3834

Diagnosis Guide, SC26-8528

Licensed Program Specifications, GC26-8069

IBM Library Server

BACK_1.1 IBM COBOL for VSE/ESA

General Information, GC26-8068

Migration Guide, GC26-8070

Installation and Customization Guide, SC26-8071

Programming Guide, SC26-8072

Language Reference, SC26-8073

Reference Summary, SX26-3834

Diagnosis Guide, SC26-8528

Licensed Program Specifications, GC26-8069

IBM Library Server

BACK_1.2 IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

Installation and Customization Guide, SC26-8071

COBOL Millennium Language Extensions Guide, GC26-9266

Fact Sheet, GC26-9321

Licensed Program Specifications, GC26-9417

IBM Library Server

BACK_1.2 IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

Installation and Customization Guide, SC26-8071

COBOL Millennium Language Extensions Guide, GC26-9266

Fact Sheet, GC26-9321

Licensed Program Specifications, GC26-9417

IBM Library Server

BACK_1.3 Language Environment Publications

Fact Sheet, GC33-6679

Concepts Guide, GC33-6680

Installation and Customization Guide, SC33-6682

Programming Guide, SC33-6684

Programming Reference, SC33-6685

Debugging Guide and Run-Time Messages, SC33-6681

Licensed Program Specifications, GC33-6683

Run-Time Migration Guide, SC33-6687

Writing Interlanguage Communication Applications, SC33-6686

C Run-Time Library Reference, SC33-6689

C Run-Time Programming Guide, SC33-6688

IBM Library Server

BACK_1.3 Language Environment Publications

Fact Sheet, GC33-6679

Concepts Guide, GC33-6680

Installation and Customization Guide, SC33-6682

Programming Guide, SC33-6684

Programming Reference, SC33-6685

Debugging Guide and Run-Time Messages, SC33-6681

Licensed Program Specifications, GC33-6683

Run-Time Migration Guide, SC33-6687

Writing Interlanguage Communication Applications, SC33-6686

C Run-Time Library Reference, SC33-6689

C Run-Time Programming Guide, SC33-6688

IBM Library Server

BACK_1.4 Related Publications

CICS/VSE

Application Programming Guide, SC33-0712

Application Programming Reference, SC33-0713

System Definition and Operations Guide, SC33-0706

Resource Definition (Online), SC33-0708

Debug Tool for VSE/ESA

User's Guide and Reference, SC26-8797

Installation and Customization Guide, SC26-8798

DFSORT for VSE/ESA

Application Programming Guide, SC26-7040

DL/I DOS/VS

Application Programming: High-Level Programming Interface, SH24-5009

Application Programming: CALL and RQDLI Interfaces, SH12-5411

Sort/Merge II

DOS/VS VM/SP Sort/Merge Version 2 Application Programming Guide, SC33-4044

SQL/DS

Application Programming for VSE, SH09-8098

VSE/ESA Version 1 Release 4

Planning, SC33-6503

Installation and Service, SC33-6504

Administration, SC33-6505

Guide to System Functions, SC33-6511

System Control Statements, SC33-6513

System Macros User's Guide, SC33-6515

System Macros Reference, SC33-6516

System Utilities, SC33-6517

Messages and Codes Vol.1 & 2, SC33-6507

VSE/ESA Version 2

System Upgrade and Service, SC33-6602

Planning, SC33-6603

Installation, SC33-6604

Administration, SC33-6605

Guide to System Functions, SC33-6611

System Control Statements, SC33-6613

System Macros User's Guide, SC33-6615

System Macros Reference, SC33-6616

System Utilities, SC33-6617

Messages and Codes Vol. 1, SC33-6698

Messages and Codes Vol. 2, SC33-6699

VSE/ESA VSAM

VSE/VSAM Commands and Macros, SC33-6532

VSE/VSAM User's Guide, SC33-6535

IBM Library Server

BACK_1.4 Related Publications

CICS/VSE

Application Programming Guide, SC33-0712

Application Programming Reference, SC33-0713

System Definition and Operations Guide, SC33-0706

Resource Definition (Online), SC33-0708

Debug Tool for VSE/ESA

User's Guide and Reference, SC26-8797

Installation and Customization Guide, SC26-8798

DFSORT for VSE/ESA

Application Programming Guide, SC26-7040

DL/I DOS/VS

Application Programming: High-Level Programming Interface, SH24-5009

Application Programming: CALL and RQDLI Interfaces, SH12-5411

Sort/Merge II

DOS/VS VM/SP Sort/Merge Version 2 Application Programming Guide, SC33-4044

SQL/DS

Application Programming for VSE, SH09-8098

VSE/ESA Version 1 Release 4

Planning, SC33-6503

Installation and Service, SC33-6504

Administration, SC33-6505

Guide to System Functions, SC33-6511

System Control Statements, SC33-6513

System Macros User's Guide, SC33-6515

System Macros Reference, SC33-6516

System Utilities, SC33-6517

Messages and Codes Vol.1 & 2, SC33-6507

VSE/ESA Version 2

System Upgrade and Service, SC33-6602

Planning, SC33-6603

Installation, SC33-6604

Administration, SC33-6605

Guide to System Functions, SC33-6611

System Control Statements, SC33-6613

System Macros User's Guide, SC33-6615

System Macros Reference, SC33-6616

System Utilities, SC33-6617

Messages and Codes Vol. 1, SC33-6698

Messages and Codes Vol. 2, SC33-6699

VSE/ESA VSAM

VSE/VSAM Commands and Macros, SC33-6532

VSE/VSAM User's Guide, SC33-6535

IBM Library Server

BACK_1.5 Softcopy Publications

These collection kit contains the COBOL/VSE and LE/VSE-conforming language product publications:

VSE Collection, SK2T-0060

IBM Library Server

FRONT_2 About This Book

This book is for systems programmers who are responsible for installing and customizing either or both of the products:

IBM COBOL for VSE/ESA
IBM VisualAge COBOL Millennium Language Extensions for VSE/ESA

for their location. This book provides information needed to plan for, install, and customize COBOL/VSE and VisualAge COBOL Millennium Language Extensions (MLE) for VSE/ESA under VSE/ESA. In addition, this book may help you to assess the value of COBOL/VSE and VisualAge COBOL MLE for VSE to your organization.

In this book, the generic term operating system is used when referring to VSE/ESA.

You should have a knowledge of COBOL/VSE and of your system's operating environment to use this book and ensure a successful installation of COBOL/VSE and VisualAge COBOL MLE for VSE.

Important

COBOL/VSE is distributed as two offerings:

- A full function offering, which includes Debug Tool/VSE
- An alternate function offering, which does not include Debug Tool/VSE

This book does **not** provide information needed to plan for, install, or customize Debug Tool/VSE. For information about installing Debug Tool/VSE, see *Debug Tool/VSE Installation and Customization Guide*.

Subtopics:

IBM Library Server

BACK_1.5 Softcopy Publications

These collection kit contains the COBOL/VSE and LE/VSE-conforming language product publications:

VSE Collection, SK2T-0060

IBM Library Server

Index

A

B

C

D

E

F

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Y

Z

IBM Library Server

Index

A

B

C

D

E

F

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Y

Z

IBM Library Server

BACK_2 We'd Like to Hear from You

COBOL for VSE/ESA
VisualAge COBOL Millennium Language Extensions
for VSE/ESA
Installation and Customization Guide

Publication No. SC26-8071-01

Please use one of the following ways to send us your comments about this book:

- Mail--Print and use the Readers' Comments form on the next page. To print the form, select **Print** or **Copy** from the **Services** pull-down menu. Enter *COMMENTS* as the topic to be printed or copied. Mail the completed form to:

  IBM Corporation
  555 Bailey Avenue, W92/H3
  P.O. Box 49023
  San Jose, CA 95161-9023

  If you are sending the form from a country other than the United States, give it to your local IBM branch office or IBM representative for mailing.

- Fax--Print and use the Readers' Comments form on the next page and fax it to this U.S. number: 800-426-7773. To print the form, follow the instructions under "Mail."

- Electronic mail--Use one of the following network IDs:

  - IBMMail: USIB2VVG at IBMMAIL
  - IBMLink: COBPUBS at STLVM27
  - Internet: COBPUBS@VNET.IBM.COM

  Be sure to include the following with your comments:

  - Title and publication number of this book
  - Your name, address, and telephone number if you would like a reply

Your comments should pertain only to the information in this book and the way the information is presented. To request additional publications, or to comment on other IBM information or the function of IBM products, please give your comments to your IBM representative or to your IBM authorized remarketer.

IBM may use or distribute your comments without obligation.

IBM Library Server

BACK_2 We'd Like to Hear from You

COBOL for VSE/ESA
VisualAge COBOL Millennium Language Extensions
for VSE/ESA
Installation and Customization Guide

Publication No. SC26-8071-01

Please use one of the following ways to send us your comments about this book:

- Mail--Print and use the Readers' Comments form on the next page. To print the form, select **Print** or **Copy** from the **Services** pull-down menu. Enter *COMMENTS* as the topic to be printed or copied. Mail the completed form to:

  IBM Corporation
  555 Bailey Avenue, W92/H3
  P.O. Box 49023
  San Jose, CA 95161-9023

  If you are sending the form from a country other than the United States, give it to your local IBM branch office or IBM representative for mailing.

- Fax--Print and use the Readers' Comments form on the next page and fax it to this U.S. number: 800-426-7773. To print the form, follow the instructions under "Mail."

- Electronic mail--Use one of the following network IDs:

  - IBMMail: USIB2VVG at IBMMAIL
  - IBMLink: COBPUBS at STLVM27
  - Internet: COBPUBS@VNET.IBM.COM

  Be sure to include the following with your comments:

  - Title and publication number of this book
  - Your name, address, and telephone number if you would like a reply

Your comments should pertain only to the information in this book and the way the information is presented. To request additional publications, or to comment on other IBM information or the function of IBM products, please give your comments to your IBM representative or to your IBM authorized remarketer.

IBM may use or distribute your comments without obligation.

IBM Library Server

1.2.2.2 Syntax Format for Modifying Compiler Options and Phases

If you plan to modify compiler option values and compiler phases, use the IGYCOPT syntax format shown in Figure 1. The IBM-supplied default values are shown both on the planning worksheets and immediately following the syntax diagrams.

Compiler options and phases, and their defaults, are discussed in the following sections. You should review these options and phases, and their default values to determine the values most suitable for your applications.

IGYCOPT format

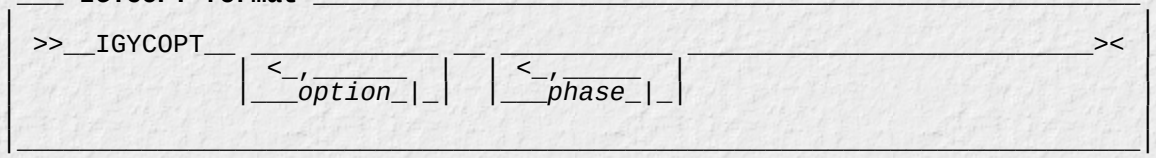


Figure 1. Syntax Format for IGYCOPT Compiler Options and Phases Macro

Subtopics:

- 1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

IBM Library Server

1.2.2.2 Syntax Format for Modifying Compiler Options and Phases

If you plan to modify compiler option values and compiler phases, use the IGYCOPT syntax format shown in Figure 1. The IBM-supplied default values are shown both on the planning worksheets and immediately following the syntax diagrams.

Compiler options and phases, and their defaults, are discussed in the following sections. You should review these options and phases, and their default values to determine the values most suitable for your applications.

IGYCOPT format

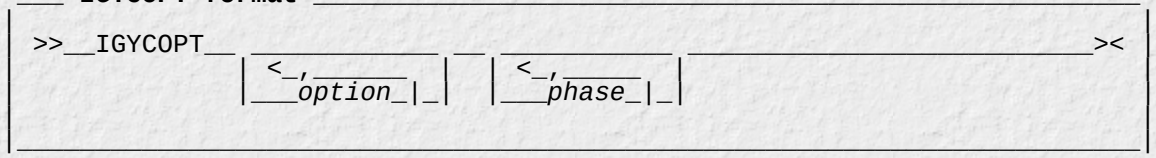


Figure 1. Syntax Format for IGYCOPT Compiler Options and Phases Macro

Subtopics:

- 1.2.2.2.1 IGYCOPT Macro Worksheet for Compiler Options

IBM Library Server

1.2.3.21 IGYCXREF

IGYCXREF is the cross-reference phase. It sorts user-names and procedure-names in EBCDIC collating sequence.

Syntax

Subtopics:

- 1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

IBM Library Server

1.2.3.21 IGYCXREF

IGYCXREF is the cross-reference phase. It sorts user-names and procedure-names in EBCDIC collating sequence.

Syntax

Subtopics:

- 1.2.3.21.1 IGYCOPT Macro Worksheet for Compiler Phases

IBM Library Server

| 2.1.1.1 Distribution Media

| You will receive VisualAge COBOL MLE for VSE on one of the following:

| - 9-track magnetic tape written at 6250 BPI
| - IBM 3480 or IBM 3490 cartridge
| - 4mm DAT cartridge

| The tape or cartridge contains all the programs and data needed for
| installation.

Subtopics:

IBM Library Server

| 2.1.1.1 Distribution Media

| You will receive VisualAge COBOL MLE for VSE on one of the following:

- 9-track magnetic tape written at 6250 BPI
- IBM 3480 or IBM 3490 cartridge
- 4mm DAT cartridge

| The tape or cartridge contains all the programs and data needed for installation.

Subtopics:

IBM Library Server

| 2.1.1.2 Program Documentation

| This section identifies the basic and optional VisualAge COBOL MLE for VSE
| documentation you receive.

Subtopics:

IBM Library Server

| 2.1.1.2 Program Documentation

| This section identifies the basic and optional VisualAge COBOL MLE for VSE
| documentation you receive.

Subtopics:

IBM Library Server

| 2.1.1.2.2 Optional Unlicensed Program Publications and Softcopy

| Table 23 in topic 2.1.1.2.3 identifies the optional unlicensed program
| publications for VisualAge COBOL MLE for VSE. You receive one free
| printed copy of each publication listed if you specify the feature number
| 7014 (or 7035 for Japanese publications) when you order VisualAge COBOL
| MLE for VSE.

IBM Library Server

| 2.1.1.2.2 Optional Unlicensed Program Publications and Softcopy

| Table 23 in topic 2.1.1.2.3 identifies the optional unlicensed program
| publications for VisualAge COBOL MLE for VSE. You receive one free
| printed copy of each publication listed if you specify the feature number
| 7014 (or 7035 for Japanese publications) when you order VisualAge COBOL
| MLE for VSE.

1.3.2.3.1 Step 3a: Install COBOL/VSE using the Interactive Interface

VSE/ESA provides dialogs (Interactive Interface) to install IBM licensed programs.

To install COBOL/VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (Additional information about the functions of the Interactive Interface are in *VSE/ESA Administration*, and for further information about installing licensed programs using the Interactive Interface see *VSE/ESA Installation and Service*.)

Mount the COBOL/VSE tape on an available tape drive.

In the following menus specify the **highlighted** items that appear after the ===> symbol.

1. **VSE/ESA FUNCTION SELECTION** menu:
   ===> ***1*** (Installation)

2. **INSTALLATION** menu:

   ===> ***1*** (Install Programs - Stacked V2 Format)

INSTALL PROGRAMS - STACKED V2 FORMAT menu:
===> ***1*** (Prepare for Installation (Stacked Tapes Only))

Note: Despite the comment which says you can only use option 1 for stacked tapes, you can use option 1 for both a stacked distribution tape containing one or more optional licensed programs, and a nonstacked tape containing just COBOL/VSE.

PREPARE FOR INSTALLATION (STACKED TAPES ONLY) menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the distribution tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job.

The output listing from this job will give a list of the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifiers for COBOL/VSE are COB.BASE...1.1.0, COB.ENU....1.1.0, and COB.JPN....1.1.0.

The program identifiers of the optional programs on the distribution

tape are also automatically entered on the **INSTALL PRODUCT(S) FROM TAPE** menu.

Return to the **VSE/ESA FUNCTION SELECTION** menu:
===> ***1*** (Installation)

INSTALLATION menu:
===> ***1*** (Install Programs - Stacked V2 Format)

INSTALL PROGRAMS - STACKED V2 FORMAT menu:
===> ***2*** (Install Product(s) from Tape)

INSTALL PRODUCT(S) FROM TAPE menu:

Enter **1** (install) in the OPT field against each of the identifiers below that you intend to install:

COB.BASE...1.1.0 (COBOL/VSE)

COB.ENU....1.1.0 (COBOL/VSE US English Language Feature)

COB.JPN....1.1.0 (COBOL/VSE Japanese Language Feature)

and **2** (skip installation) against any other optional products you do not intend to install at this time.

If you did not use the default library PRD2.PROD, enter the name of your library and sublibrary on this screen.

Press PF5 to generate the installation job.

VSE/ESA INSTALL PRODUCT(S) TAPE SPECIFICATION menu:

===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the COBOL/VSE tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job and install COBOL/VSE.

Check the output from the batch job created by the previous steps to ensure that the installation was successful. If you are installing COBOL/VSE from a nonstacked tape, you will receive the following message:
IESI0083I TAPE IS NOT PID-V2-STACKED

This message is for information only and can be ignored. The Librarian RESTORE job ends with a return code of 4. This is not an error: Installation was successful.

1.3.2.3.1 Step 3a: Install COBOL/VSE using the Interactive Interface

VSE/ESA provides dialogs (Interactive Interface) to install IBM licensed programs.

To install COBOL/VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (Additional information about the functions of the Interactive Interface are in *VSE/ESA Administration*, and for further information about installing licensed programs using the Interactive Interface see *VSE/ESA Installation and Service*.)

Mount the COBOL/VSE tape on an available tape drive.

In the following menus specify the **highlighted** items that appear after the ===> symbol.

1. **VSE/ESA FUNCTION SELECTION** menu:
   ===> ***1*** (Installation)

2. **INSTALLATION** menu:

   ===> ***1*** (Install Programs - Stacked V2 Format)

   INSTALL PROGRAMS - STACKED V2 FORMAT menu:
   ===> ***1*** (Prepare for Installation (Stacked Tapes Only))

   Note: Despite the comment which says you can only use option 1 for stacked tapes, you can use option 1 for both a stacked distribution tape containing one or more optional licensed programs, and a nonstacked tape containing just COBOL/VSE.

   PREPARE FOR INSTALLATION (STACKED TAPES ONLY) menu:
   ===> ***cuu***
   (*cuu* is the address of the tape drive where you mounted the distribution tape)

   JOB DISPOSITION menu:

   Make any changes required and press ENTER to submit the job.

   The output listing from this job will give a list of the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifiers for COBOL/VSE are COB.BASE...1.1.0, COB.ENU....1.1.0, and COB.JPN....1.1.0.

   The program identifiers of the optional programs on the distribution

tape are also automatically entered on the **INSTALL PRODUCT(S) FROM TAPE** menu.

Return to the **VSE/ESA FUNCTION SELECTION** menu:
===> ***1*** (Installation)

INSTALLATION menu:
===> ***1*** (Install Programs - Stacked V2 Format)

INSTALL PROGRAMS - STACKED V2 FORMAT menu:
===> ***2*** (Install Product(s) from Tape)

INSTALL PRODUCT(S) FROM TAPE menu:

Enter **1** (install) in the OPT field against each of the identifiers below that you intend to install:

COB.BASE...1.1.0 (COBOL/VSE)

COB.ENU....1.1.0 (COBOL/VSE US English Language Feature)

COB.JPN....1.1.0 (COBOL/VSE Japanese Language Feature)

and **2** (skip installation) against any other optional products you do not intend to install at this time.

If you did not use the default library PRD2.PROD, enter the name of your library and sublibrary on this screen.

Press PF5 to generate the installation job.

VSE/ESA INSTALL PRODUCT(S) TAPE SPECIFICATION menu:

===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the COBOL/VSE tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job and install COBOL/VSE.

Check the output from the batch job created by the previous steps to ensure that the installation was successful. If you are installing COBOL/VSE from a nonstacked tape, you will receive the following message:
IESI0083I TAPE IS NOT PID-V2-STACKED

This message is for information only and can be ignored. The Librarian RESTORE job ends with a return code of 4. This is not an error: Installation was successful.

1.4.4.1 Procedure for Creating or Modifying a Reserved Word Table

To create or modify a reserved word table you must edit the appropriate source member from the compiler sublibrary:

- Member IGY8RWRD.Z (default reserved word table supplied by IBM)
- Member IGY8CICS.Z (CICS reserved word table supplied by IBM)
- A user member (user-supplied reserved word table)

You must also modify and invoke the appropriate JCL.

The JCL for the default reserved word table is IGYWERWD.Z

The JCL for the CICS reserved word table is IGYWERWC.Z

Your source member should have four parts: Parts I, II, III, and IV. Modify the member and JCL as follows:

1. Make a private copy of the member.
2. Do not edit Part I (all lines up to and including the line with the keyword MOD).
3. Edit Part II by placing asterisks in column 1 of the lines that contain CODASYL reserved words for which you do not want informational messages issued.
4. Edit Part III by placing asterisks in column 1 of the lines that contain obsolete reserved words for which you do not want severe messages issued.
5. Edit Part IV by coding additional reserved word control statements that create the modifications you want, as described under "Coding Control Statements" in topic 1.4.4.1.1.
6. Modify and run the JCL, as discussed under "Modifying and Running JCL to Create a New Reserved Word Table" in topic 1.4.4.5. You will also have to create a unique 1- to 4-character identification for the new reserved word table and supply it in the JCL.

Subtopics:

- 1.4.4.1.3 Coding Operands in Control Statements
- 1.4.4.1.4 Rules for Coding Control Statement Operands

IBM Library Server

1.4.4.1 Procedure for Creating or Modifying a Reserved Word Table

To create or modify a reserved word table you must edit the appropriate source member from the compiler sublibrary:

- Member IGY8RWRD.Z (default reserved word table supplied by IBM)
- Member IGY8CICS.Z (CICS reserved word table supplied by IBM)
- A user member (user-supplied reserved word table)

You must also modify and invoke the appropriate JCL.

The JCL for the default reserved word table is IGYWERWD.Z

The JCL for the CICS reserved word table is IGYWERWC.Z

Your source member should have four parts: Parts I, II, III, and IV. Modify the member and JCL as follows:

1. Make a private copy of the member.
2. Do not edit Part I (all lines up to and including the line with the keyword MOD).
3. Edit Part II by placing asterisks in column 1 of the lines that contain CODASYL reserved words for which you do not want informational messages issued.
4. Edit Part III by placing asterisks in column 1 of the lines that contain obsolete reserved words for which you do not want severe messages issued.
5. Edit Part IV by coding additional reserved word control statements that create the modifications you want, as described under "Coding Control Statements" in topic 1.4.4.1.1.
6. Modify and run the JCL, as discussed under "Modifying and Running JCL to Create a New Reserved Word Table" in topic 1.4.4.5. You will also have to create a unique 1- to 4-character identification for the new reserved word table and supply it in the JCL.

Subtopics:

- 1.4.4.1.3 Coding Operands in Control Statements
- 1.4.4.1.4 Rules for Coding Control Statement Operands

IBM Library Server

1.5.2.2 Checklist for Applying Service

Table 18 lists the steps for installing corrective service on COBOL/VSE. You can use this table as a checklist.

Table 18. Summary of Steps for Installing Service on COBOL/VSE

Step	Description	MSHP Command or Jobname	Topic
1	Check pre-requisite APARs or PTFs	RETRACE	1.5.2.2.1
2	Backup the Original System	----	1.5.2.2.2
3a	Apply Service using the Interactive Interface	INSTALL	1.5.2.2.3
3b	Apply Service using a batch job	INSTALL	1.5.2.2.3
4	Run the Installation Verification Program	IGYWEIVP	1.5.2.2.6

Subtopics:

IBM Library Server

1.5.2.2 Checklist for Applying Service

Table 18 lists the steps for installing corrective service on COBOL/VSE. You can use this table as a checklist.

Table 18. Summary of Steps for Installing Service on COBOL/VSE

Step	Description	MSHP Command or Jobname	Topic
1	Check pre-requisite APARs or PTFs	RETRACE	1.5.2.2.1
2	Backup the Original System	----	1.5.2.2.2
3a	Apply Service using the Interactive Interface	INSTALL	1.5.2.2.3
3b	Apply Service using a batch job	INSTALL	1.5.2.2.3
4	Run the Installation Verification Program	IGYWEIVP	1.5.2.2.6

Subtopics:

1.5.2.2.4 Step 3a: Apply Service Using the Interactive Interface

To apply service to COBOL/VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (Additional information about the functions of the Interactive Interface are in *VSE/ESA Administration*.)

Mount the COBOL/VSE service tape on an available tape drive.

In the following menus specify the **highlighted** items that appear after the ===> symbol.

1. **VSE/ESA FUNCTION SELECTION** menu:
   ===> ***1*** (Installation)

2. **INSTALLATION** menu:
   ===> ***4*** (IBM Service)

PTF HANDLING menu

- If you want to print the documentation about the supplied PTFs before applying the service, select:
  ===> ***1*** (Print Service Document)

PRINT SERVICE DOCUMENT menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the service tape)

- If you want to apply the service directly, select:
  ===> ***3*** (Apply PTFs from Service Tape)

APPLY PTF menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the service tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job to apply the service.

IBM Library Server

1.5.2.2.4 Step 3a: Apply Service Using the Interactive Interface

To apply service to COBOL/VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (Additional information about the functions of the Interactive Interface are in *VSE/ESA Administration*.)

Mount the COBOL/VSE service tape on an available tape drive.

In the following menus specify the **highlighted** items that appear after the ===> symbol.

1. **VSE/ESA FUNCTION SELECTION** menu:
   ===> ***1*** (Installation)

2. **INSTALLATION** menu:
   ===> ***4*** (IBM Service)

PTF HANDLING menu

- If you want to print the documentation about the supplied PTFs before applying the service, select:
  ===> ***1*** (Print Service Document)

PRINT SERVICE DOCUMENT menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the service tape)

- If you want to apply the service directly, select:
  ===> ***3*** (Apply PTFs from Service Tape)

APPLY PTF menu:
===> ***cuu***
(*cuu* is the address of the tape drive where you mounted the service tape)

JOB DISPOSITION menu:

Make any changes required and press ENTER to submit the job to apply the service.

IBM Library Server

| 2.3.2.5.1 Method 1: Apply Service Using the Interactive Interface

| The VSE/ESA Interactive Interface enables you to use dialog requests to
| apply service to VisualAge COBOL MLE for VSE. For more information about
| the functions of the Interactive Interface, see *VSE/ESA Administration*.
| For more information on how to use the dialogs to apply service, see
| *VSE/ESA System Upgrade and Service*.

IBM Library Server

| 2.3.2.5.1 Method 1: Apply Service Using the Interactive Interface

| The VSE/ESA Interactive Interface enables you to use dialog requests to
| apply service to VisualAge COBOL MLE for VSE. For more information about
| the functions of the Interactive Interface, see *VSE/ESA Administration*.
| For more information on how to use the dialogs to apply service, see
| *VSE/ESA System Upgrade and Service*.

IBM Library Server

1.2.2.1 Why Make Compiler Options Fixed?

COBOL/VSE can aid in setting up your site's programming standards. For example, many sites select APOST or QUOTE as their preferred compiler option for literal delimiters, but have no easy way to enforce its use. In COBOL/VSE, the IGYCOPT macro can be used to specify that an option may not be changed or replaced at compile time; that is, the option is fixed. At compile time, an attempt to replace a fixed option will not be allowed and will result in a diagnostic message with a nonzero compiler return code.

When certain options are fixed for consistent use, there might be times when special conditions require the ability to bypass such a fixed option. This can be done by assembling a temporary copy of the IGYCOPT macro with different parameters. At compile time, by selectively using a LIBDEF JCL statement specifying a sublibrary containing the required IGYCDOPT phase, you can bypass the fixed option. For example, if you select the OPT (OPTIMIZE) option to be fixed (indicating you always want the COBOL compiler to generate optimized object code), and then need to exempt an application from this requirement, you must reassemble the IGYCOPT macro after removing the asterisk parameter from the option. Then place the resulting IGYCDOPT phase in a temporary sublibrary to be accessed using the LIBDEF JCL statement at compile time.

Subtopics:

- 1.2.2.1.1 Sample Installation Job

IBM Library Server

1.2.2.1 Why Make Compiler Options Fixed?

COBOL/VSE can aid in setting up your site's programming standards. For example, many sites select APOST or QUOTE as their preferred compiler option for literal delimiters, but have no easy way to enforce its use. In COBOL/VSE, the IGYCOPT macro can be used to specify that an option may not be changed or replaced at compile time; that is, the option is fixed. At compile time, an attempt to replace a fixed option will not be allowed and will result in a diagnostic message with a nonzero compiler return code.

When certain options are fixed for consistent use, there might be times when special conditions require the ability to bypass such a fixed option. This can be done by assembling a temporary copy of the IGYCOPT macro with different parameters. At compile time, by selectively using a LIBDEF JCL statement specifying a sublibrary containing the required IGYCDOPT phase, you can bypass the fixed option. For example, if you select the OPT (OPTIMIZE) option to be fixed (indicating you always want the COBOL compiler to generate optimized object code), and then need to exempt an application from this requirement, you must reassemble the IGYCOPT macro after removing the asterisk parameter from the option. Then place the resulting IGYCDOPT phase in a temporary sublibrary to be accessed using the LIBDEF JCL statement at compile time.

Subtopics:

- 1.2.2.1.1 Sample Installation Job

1.2.4.3.1 Default Reserved Word Table (IGYCRWT)

The default reserved word table is described in an appendix in the *COBOL/VSE Language Reference*.

1.2.4.3.1 Default Reserved Word Table (IGYCRWT)

The default reserved word table is described in an appendix in the *COBOL/VSE Language Reference*.

IBM Library Server

1.2.4.3 Reserved Word Tables Supplied with COBOL/VSE

Three reserved word tables come on the installation tape: the default reserved word table, the CICS reserved word table, and the SAA reserved word table.

Subtopics:

IBM Library Server

1.2.4.3 Reserved Word Tables Supplied with COBOL/VSE

Three reserved word tables come on the installation tape: the default reserved word table, the CICS reserved word table, and the SAA reserved word table.

Subtopics:

1.2.4.3.2 CICS Reserved Word Table (IGYCCICS)

COBOL/VSE provides an alternate reserved word table specifically for CICS application programs. It is set up so that COBOL words not supported under CICS are flagged by the compiler with an error message.

Contents of the Table: The CICS reserved word table is the same as the default reserved word table except that the following COBOL words are marked as restricted (RSTR):

ACCEPT	FILE-CONTROL	RERUN
CLOSE	INPUT-OUTPUT	REWRITE
DELETE	I-O-CONTROL	SD
DISPLAY	MERGE	SORT
FD	OPEN	START
FILE	READ	WRITE

SORT Users

If you intend to use the SORT statement under CICS (COBOL/VSE supports an interface for the SORT statement under CICS), you must modify the CICS reserved word table before using it. The words highlighted above must be removed from the list of words marked as restricted, because they are required for the SORT function.

Using the Table: In order to use the CICS reserved word table, the compiler option WORD(CICS) must be specified. If you want to use the CICS reserved word table as the default reserved word table, then you must set the default value of the WORD compiler option to WORD=CICS.

Location of the Table: The data used to create the CICS reserved word table is in member IGY8CICS.Z in the COBOL/VSE sublibrary.

1.2.4.3.2 CICS Reserved Word Table (IGYCCICS)

COBOL/VSE provides an alternate reserved word table specifically for CICS application programs. It is set up so that COBOL words not supported under CICS are flagged by the compiler with an error message.

Contents of the Table: The CICS reserved word table is the same as the default reserved word table except that the following COBOL words are marked as restricted (RSTR):

ACCEPT	FILE-CONTROL	RERUN
CLOSE	INPUT-OUTPUT	REWRITE
DELETE	I-O-CONTROL	SD
DISPLAY	MERGE	SORT
FD	OPEN	START
FILE	READ	WRITE

SORT Users

If you intend to use the SORT statement under CICS (COBOL/VSE supports an interface for the SORT statement under CICS), you must modify the CICS reserved word table before using it. The words highlighted above must be removed from the list of words marked as restricted, because they are required for the SORT function.

Using the Table: In order to use the CICS reserved word table, the compiler option WORD(CICS) must be specified. If you want to use the CICS reserved word table as the default reserved word table, then you must set the default value of the WORD compiler option to WORD=CICS.

Location of the Table: The data used to create the CICS reserved word table is in member IGY8CICS.Z in the COBOL/VSE sublibrary.

IBM Library Server

1.3.2.3 Step 3: Install COBOL/VSE

You can use the Interactive Interface (Step 3a) or a batch job (Step 3b) to install COBOL/VSE.

Depending on how you ordered the COBOL/VSE product, you will receive one of the following types of distribution tape:

- A distribution tape containing only the COBOL/VSE product
- A VSE stacked tape containing one or more optional products

The installation methods shown will handle both types of distribution tape.

Continue with either Step 3a or Step 3b.

Subtopics:

IBM Library Server

1.3.2.3 Step 3: Install COBOL/VSE

You can use the Interactive Interface (Step 3a) or a batch job (Step 3b) to install COBOL/VSE.

Depending on how you ordered the COBOL/VSE product, you will receive one of the following types of distribution tape:

- A distribution tape containing only the COBOL/VSE product
- A VSE stacked tape containing one or more optional products

The installation methods shown will handle both types of distribution tape.

Continue with either Step 3a or Step 3b.

Subtopics:

1.4.4.1.1 Coding Control Statements

To create a reserved word table, you must understand the syntax rules for the control statements and for the operands within the control statements.

Figure 6 illustrates the format for coding reserved word processor control statements.

```
ABBR   reserved-word: user-word [comments]
     [reserved-word: user-word [comments]]

.
.
.
INFO   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
RSTR   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
```

Figure 6. Syntax Format for Reserved Word Processor Control Statements

As shown in Figure 6, keywords you can use are:

ABBR to specify an alternative form of an existing reserved word

INFO to specify words that are to be flagged with an informational message whenever they are used in a program

RSTR to specify words that are to be flagged with an error message whenever a program employs them

Note: All words you identify with the control statement keywords INFO and RSTR will be flagged with a message in the source listing of the COBOL/VSE program that uses them. Words that are abbreviated will not be flagged in the source listing unless you have also specified them on the INFO or RSTR control statements.

IBM Library Server

1.4.4.1.1 Coding Control Statements

To create a reserved word table, you must understand the syntax rules for the control statements and for the operands within the control statements.

Figure 6 illustrates the format for coding reserved word processor control statements.

```
ABBR   reserved-word: user-word [comments]
     [reserved-word: user-word [comments]]

.
.
.
INFO   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
RSTR   COBOL-word [comments]
     [COBOL-word [comments]]

.
.
.
```

Figure 6. Syntax Format for Reserved Word Processor Control Statements

As shown in Figure 6, keywords you can use are:

ABBR to specify an alternative form of an existing reserved word

INFO to specify words that are to be flagged with an informational message whenever they are used in a program

RSTR to specify words that are to be flagged with an error message whenever a program employs them

Note: All words you identify with the control statement keywords INFO and RSTR will be flagged with a message in the source listing of the COBOL/VSE program that uses them. Words that are abbreviated will not be flagged in the source listing unless you have also specified them on the INFO or RSTR control statements.

1.4.4.1.2 Rules for Coding Control Statements

When you code your control statements, do the following:

- Begin the control statement in column 1.
- Place one or more spaces between the keyword and the first operand.
- When specifying a second operand, include a colon (:) and one or more spaces after the first operand.
- Continue a control statement by putting blanks in columns 1 through 5, followed by the operand or operands, to make additional specifications.
- Specify comments by putting an asterisk (*) in column 1 of the control statements. You can also place comments on the same line as the control statement. In that case, however, there must be at least one space following the operand(s) before a comment begins.
- To specify more than one change within a single control statement, put each additional specification on a separate line.
- Do not add any blank lines.

1.4.4.1.2 Rules for Coding Control Statements

When you code your control statements, do the following:

- Begin the control statement in column 1.
- Place one or more spaces between the keyword and the first operand.
- When specifying a second operand, include a colon (:) and one or more spaces after the first operand.
- Continue a control statement by putting blanks in columns 1 through 5, followed by the operand or operands, to make additional specifications.
- Specify comments by putting an asterisk (*) in column 1 of the control statements. You can also place comments on the same line as the control statement. In that case, however, there must be at least one space following the operand(s) before a comment begins.
- To specify more than one change within a single control statement, put each additional specification on a separate line.
- Do not add any blank lines.

1.4.4.1.3 Coding Operands in Control Statements

The following list shows the types of operands you will be coding in the control statements:

reserved-word is an existing reserved word.

user-word is a user-defined COBOL word that is not a reserved word.

comments are any comments you want to put on the same line with the control statement, or on a separate line that has an asterisk in column 1.

COBOL-word is a word of up to 30 characters that may be a system name, a reserved word, or a user-defined word.

1.4.4.1.3 Coding Operands in Control Statements

The following list shows the types of operands you will be coding in the control statements:

reserved-word is an existing reserved word.

user-word is a user-defined COBOL word that is not a reserved word.

comments are any comments you want to put on the same line with the control statement, or on a separate line that has an asterisk in column 1.

COBOL-word is a word of up to 30 characters that may be a system name, a reserved word, or a user-defined word.

IBM Library Server

1.4.4.2 ABBR Statement

```
ABBR  reserved-word:  user-word [ comments ]
    Defines an alternative symbol for the reserved word specified.  The
    symbol can be used when coding a program.
```

Notes:

1. *User-word* becomes a reserved word and can be used in place of the reserved word specified in this statement.

2. *Reserved-word* remains a reserved word with its original definition.

3. The source listing will show the original source--the symbol as you coded it.

4. The reserved word will be used in compiler output--other listings, some messages, and so forth.

Subtopics:

- 1.4.4.2.1 Example of an ABBR Statement

IBM Library Server

1.4.4.2 ABBR Statement

```
ABBR  reserved-word:  user-word [ comments ]
    Defines an alternative symbol for the reserved word specified.  The
    symbol can be used when coding a program.
```

Notes:

1. *User-word* becomes a reserved word and can be used in place of the reserved word specified in this statement.

2. *Reserved-word* remains a reserved word with its original definition.

3. The source listing will show the original source--the symbol as you coded it.

4. The reserved word will be used in compiler output--other listings, some messages, and so forth.

Subtopics:

- 1.4.4.2.1 Example of an ABBR Statement

1.4.4.3 INFO Statement

```
INFO  COBOL-word  [ comments ]
    Specifies a COBOL word that is to be flagged by the compiler.


The messages will be handled as information (I) messages.  The compilation
condition is not changed.
```

Subtopics:

- 1.4.4.3.1 Example of an INFO Statement

1.4.4.3 INFO Statement

```
INFO  COBOL-word  [ comments ]
    Specifies a COBOL word that is to be flagged by the compiler.


The messages will be handled as information (I) messages.  The compilation
condition is not changed.
```

Subtopics:

- 1.4.4.3.1 Example of an INFO Statement

1.4.4.4 RSTR Statement

```
RSTR  COBOL-word  [ comments ]
```

Specifies a COBOL word that cannot be used in a program.

Note: The following reserved words may not be restricted:

IDENTIFICATION	FD
ENVIRONMENT	DATA
WORKING-STORAGE	PROCEDURE
DIVISION	SECTION
PROGRAM-ID	

Subtopics:

- 1.4.4.4.1 Examples of RSTR Statements

IBM Library Server

1.4.4.4 RSTR Statement

```
RSTR  COBOL-word  [ comments ]
    Specifies a COBOL word that cannot be used in a program.
```

Note: The following reserved words may not be restricted:

IDENTIFICATION	FD
ENVIRONMENT	DATA
WORKING-STORAGE	PROCEDURE
DIVISION	SECTION
PROGRAM-ID	

Subtopics:

- 1.4.4.4.1 Examples of RSTR Statements

IBM Library Server

1.4.4.5 Modifying and Running JCL to Create a New Reserved Word Table

The JCL you use to create a new reserved word table contains three steps:

STEP 1 Runs the reserved word table utility with your modified table.

STEP 2 Assembles your modified reserved word table.

STEP 3 Produces a run-time loadable phase from the object module.

After you run the job, a new reserved word table will be created, the sublibrary you specified will contain the new table, and the name of the table will be IGYC plus the 1- to 4-character identification you specified.

Subtopics:

1.4.4.5 Modifying and Running JCL to Create a New Reserved Word Table

The JCL you use to create a new reserved word table contains three steps:

STEP 1 Runs the reserved word table utility with your modified table.

STEP 2 Assembles your modified reserved word table.

STEP 3 Produces a run-time loadable phase from the object module.

After you run the job, a new reserved word table will be created, the sublibrary you specified will contain the new table, and the name of the table will be IGYC plus the 1- to 4-character identification you specified.

Subtopics:

1.5.2.2.1 Step 1. Check Pre-requisite APARs or PTFs

Pre-requisite APARs or PTFs are those that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to COBOL/VSE or any licensed program you have installed.

Your IBM Support Center will have given you a list of any relevant pre-requisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 7 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYRETR  Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 7. Job to Retrace APARs and PTFs

Use the resulting listing to check that you have already applied any pre-requisite APARs or PTFs. If you have not, your IBM Support Center will send them to you, and you should apply them before applying any other service.

 *Copyright IBM Corp. 1983,1998*

1.5.2.2.1 Step 1. Check Pre-requisite APARs or PTFs

Pre-requisite APARs or PTFs are those that need to be applied to your system before you can apply the current maintenance. These APARs or PTFs may apply to COBOL/VSE or any licensed program you have installed.

Your IBM Support Center will have given you a list of any relevant pre-requisite APARs or PTFs. Most probably they will already be applied to your system. You can verify this by retracing the APARs and PTFs in your system history file. The job shown in Figure 7 shows how to retrace APARs and PTFs in the system history file.

```
// JOB IGYRETR  Retrace APARs and PTFs
// EXEC MSHP,SIZE=900K
RETRACE APARS
RETRACE PTFS
/*
/&
```

Figure 7. Job to Retrace APARs and PTFs

Use the resulting listing to check that you have already applied any pre-requisite APARs or PTFs. If you have not, your IBM Support Center will send them to you, and you should apply them before applying any other service.

 *Copyright IBM Corp. 1983,1998*

1.5.2.2.2 Step 2. Backup the Original System

Make a backup copy of your current COBOL/VSE sublibrary and the system history file. For information about backing up, see *VSE/ESA System Control Statements*.

1.5.2.2.2 Step 2. Backup the Original System

Make a backup copy of your current COBOL/VSE sublibrary and the system history file. For information about backing up, see *VSE/ESA System Control Statements*.

1.5.2.2.5 Step 3b: Apply Service Using a Batch Job

The batch job to apply service to COBOL/VSE uses the MSHP system history file where COBOL/VSE was installed.

A sample job to apply service using the Maintain System History Program (MSHP) is shown in Figure 8. For more information on MSHP see *VSE/ESA System Control Statements*.

```
// JOB IGYAPP   Apply Service
// ASSGN SYS006,cuu              1
// EXEC MSHP,SIZE=900K
INSTALL SERVICE FROMTAPE         2
/*
/&
```

Figure 8. Job to Apply Service

In area **1** change *cuu* to the address of the tape drive where you have mounted the service tape.

Area **2** shows the MSHP statement to install service from a tape. The information in the system history file will direct MSHP to apply the service to the sublibrary in which COBOL/VSE is installed.

IBM Library Server

1.5.2.2.5 Step 3b: Apply Service Using a Batch Job

The batch job to apply service to COBOL/VSE uses the MSHP system history file where COBOL/VSE was installed.

A sample job to apply service using the Maintain System History Program (MSHP) is shown in Figure 8. For more information on MSHP see *VSE/ESA System Control Statements*.

```
// JOB IGYAPP   Apply Service
// ASSGN SYS006,cuu              1
// EXEC MSHP,SIZE=900K
INSTALL SERVICE FROMTAPE         2
/*
/&
```

Figure 8. Job to Apply Service

In area **1** change *cuu* to the address of the tape drive where you have mounted the service tape.

Area **2** shows the MSHP statement to install service from a tape. The information in the system history file will direct MSHP to apply the service to the sublibrary in which COBOL/VSE is installed.

IBM Library Server

1.5.2.2.3 Step 3. Apply Service

You can apply service to COBOL/VSE from the provided service tape using either the Interactive Interface or a batch job.

You will receive instructions for applying service with the service tape.

Continue either with Step 3a or Step 3b.

IBM Library Server

1.5.2.2.3 Step 3. Apply Service

You can apply service to COBOL/VSE from the provided service tape using either the Interactive Interface or a batch job.

You will receive instructions for applying service with the service tape.

Continue either with Step 3a or Step 3b.

IBM Library Server

2.1.1.1.1 Basic Machine-Readable Material

Table 20 describes the separately-ordered licensed program distribution tape.

Table 20. Basic Material: Separately-Ordered Distribution Tape

National Language	Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
US English(1)	6250 tape	5801	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5802	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6300	1 of 1	VA CBL MLE V1R1	unlabeled
Japanese(1)	6250 tape	5811	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5812	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6301	1 of 1	VA CBL MLE V1R1	unlabeled

Note:

1. There is no difference between the program tapes for the US English Language Feature and the Japanese Language Feature. The different feature numbers are used to specify the national language of the program documentation.

The file content of the distribution tape you receive depends upon the

form in which you receive VisualAge COBOL MLE for VSE. If you receive VisualAge COBOL MLE for VSE on a VSE/ESA optional program tape, there might be other licensed programs on the tape. If you ordered VisualAge COBOL MLE for VSE separately, the distribution tape will contain only VisualAge COBOL MLE for VSE. Table 21 describes the file content of the separately-ordered licensed program distribution tape.

Table 21. File Content: Separately-Ordered Licensed Program Distribution Tape

File	Description
1	Header file containing the VisualAge COBOL MLE for VSE copyright statement
2	Backup file ID COBOLMLE...1.1.0. followed by an MSHP System History File
3	VisualAge COBOL MLE for VSE product
4	Tape Mark
5	End of backup (EOB) record
6	Tape Mark

IBM Library Server

2.1.1.1.1 Basic Machine-Readable Material

Table 20 describes the separately-ordered licensed program distribution tape.

Table 20. Basic Material: Separately-Ordered Distribution Tape

National Language	Medium	Feature Number	Physical Volume	External Label Identification	VOLSER
US English(1)	6250 tape	5801	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5802	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6300	1 of 1	VA CBL MLE V1R1	unlabeled
Japanese(1)	6250 tape	5811	1 of 1	VA CBL MLE V1R1	unlabeled
	3480 or 3490 cartridge	5812	1 of 1	VA CBL MLE V1R1	unlabeled
	4mm DAT	6301	1 of 1	VA CBL MLE V1R1	unlabeled

Note:

1. There is no difference between the program tapes for the US English Language Feature and the Japanese Language Feature. The different feature numbers are used to specify the national language of the program documentation.

The file content of the distribution tape you receive depends upon the

form in which you receive VisualAge COBOL MLE for VSE. If you receive VisualAge COBOL MLE for VSE on a VSE/ESA optional program tape, there might be other licensed programs on the tape. If you ordered VisualAge COBOL MLE for VSE separately, the distribution tape will contain only VisualAge COBOL MLE for VSE. Table 21 describes the file content of the separately-ordered licensed program distribution tape.

Table 21. File Content: Separately-Ordered Licensed Program Distribution Tape

File	Description
1	Header file containing the VisualAge COBOL MLE for VSE copyright statement
2	Backup file ID COBOLMLE...1.1.0. followed by an MSHP System History File
3	VisualAge COBOL MLE for VSE product
4	Tape Mark
5	End of backup (EOB) record
6	Tape Mark

IBM Library Server

2.1.1.2.1 Basic Unlicensed Program Publications

Table 22 identifies the basic program publications for VisualAge COBOL MLE for VSE. Unless otherwise noted, you receive one copy of each of these publications when you receive the basic materials for VisualAge COBOL MLE for VSE. For additional copies, contact your IBM representative.

Table 22. Basic VisualAge COBOL MLE for VSE Documentation

Title	Order Number
VisualAge COBOL MLE for VSE Licensed Program Specifications	GC26-9417
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

2.1.1.2.1 Basic Unlicensed Program Publications

Table 22 identifies the basic program publications for VisualAge COBOL MLE for VSE. Unless otherwise noted, you receive one copy of each of these publications when you receive the basic materials for VisualAge COBOL MLE for VSE. For additional copies, contact your IBM representative.

Table 22. Basic VisualAge COBOL MLE for VSE Documentation

Title	Order Number
VisualAge COBOL MLE for VSE Licensed Program Specifications	GC26-9417
COBOL/VSE and VisualAge COBOL MLE for VSE Installation and Customization Guide	SC26-8071

IBM Library Server

2.2.4.1 Method 1: Install VisualAge COBOL MLE for VSE Using the Interactive Interface

The VSE/ESA Interactive Interface enables you to use dialog requests to install VisualAge COBOL MLE for VSE. For more information about installing licensed programs using the Interactive Interface, see *VSE/ESA Installation*.

To install VisualAge COBOL MLE for VSE using the Interactive Interface, do the following:

1. Logon to the Interactive Interface as the system administrator.

2. Mount the VisualAge COBOL MLE for VSE distribution tape on an available tape drive.

3. In the following menus, specify the items that appear following the ===> symbol.

   a. **VSE/ESA FUNCTION SELECTION** menu:

      ===> 1 (Installation)

   b. **INSTALLATION** menu:

      ===> 1 (Install Programs - V2 Format)

      Note: You can use option 1 for both a stacked distribution tape containing one or more optional licensed programs, and a non-stacked tape containing just VisualAge COBOL MLE for VSE. Both are in Librarian V2 format.

   c. **INSTALL PROGRAMS - V2 FORMAT** menu:

      ===> 1 (Prepare for Installation)

   d. **PREPARE FOR INSTALLATION** menu:

      ===> cuu (address of drive with distribution tape)

   e. **JOB DISPOSITION** menu:

Make any changes required and press ENTER to submit the job. While the job is running, reply to any system console messages as necessary.

The output listing from this job lists the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifier for the VisualAge COBOL MLE for VSE component has the format COBOLMLE...1.1.0.

f. Check the output from the batch job created by the previous steps to ensure that the install was successful. Once the batch job created by this step has successfully run, the program identifiers of the optional programs on the distribution tape are also automatically entered on the **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu.

g. After the batch job created by the previous step has successfully run, return to the **VSE/ESA FUNCTION SELECTION** menu:

===> 1 (Installation)

h. **INSTALLATION** menu:

===> 1 (Install Programs - V2 Format)

i. **INSTALL PROGRAMS - V2 FORMAT** menu:

===> 2 (Install Program(s) from Tape)

j. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

Enter 1 (install) in the OPT field against the required identifier of the format COBOLMLE...1.1.0, and 2 (skip installation) against any other optional licensed programs you do not intend to install at this time.

VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE. If you did not install COBOL/VSE in the default sublibrary PRD2.PROD, enter the name of the library and sublibrary where COBOL/VSE is installed on this screen.

k. Press PF5 to generate the installation job.

l. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> 1 (Save the list)

m. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> cuu (address of drive with distribution tape)

n. **JOB DISPOSITION** menu:

```
|          Make any changes required and press ENTER to submit the job and
|          install VisualAge COBOL MLE for VSE.  While the job is running,
|          reply to any system console messages as necessary.


|      o.  Check the output from the batch job created by the previous steps
|          to ensure that the install was successful.  If you are installing
|          VisualAge COBOL MLE for VSE from a non-stacked tape, you received
|          the following message:


|            IESI0083I  TAPE IS NOT V2-STACKED


|          This message is for information only and can be ignored.  The
|          Librarian RESTORE job ends with a return code of 4.  This is not
|          an error.  Installation was successful.
```

Subtopics:

- 2.2.4.1.1 Condition Code and Messages

 *Copyright IBM Corp. 1983,1998*

IBM Library Server

| 2.2.4.1 Method 1: Install VisualAge COBOL MLE for VSE Using the Interactive Interface

| The VSE/ESA Interactive Interface enables you to use dialog requests to
| install VisualAge COBOL MLE for VSE. For more information about
| installing licensed programs using the Interactive Interface, see *VSE/ESA Installation*.

| To install VisualAge COBOL MLE for VSE using the Interactive Interface, do
| the following:

| 1. Logon to the Interactive Interface as the system administrator.

| 2. Mount the VisualAge COBOL MLE for VSE distribution tape on an
| available tape drive.

| 3. In the following menus, specify the items that appear following the
| ===> symbol.

| a. **VSE/ESA FUNCTION SELECTION** menu:

| ===> 1 (Installation)

| b. **INSTALLATION** menu:

| ===> 1 (Install Programs - V2 Format)

| **Note:** You can use option 1 for both a stacked distribution tape
| containing one or more optional licensed programs, and a
| non-stacked tape containing just VisualAge COBOL MLE for VSE.
| Both are in Librarian V2 format.

| c. **INSTALL PROGRAMS - V2 FORMAT** menu:

| ===> 1 (Prepare for Installation)

| d. **PREPARE FOR INSTALLATION** menu:

| ===> cuu (address of drive with distribution tape)

| e. **JOB DISPOSITION** menu:

Make any changes required and press ENTER to submit the job. While the job is running, reply to any system console messages as necessary.

The output listing from this job lists the optional programs on the distribution tape with program identifiers and recommended library sizes. The program identifier for the VisualAge COBOL MLE for VSE component has the format COBOLMLE...1.1.0.

f. Check the output from the batch job created by the previous steps to ensure that the install was successful. Once the batch job created by this step has successfully run, the program identifiers of the optional programs on the distribution tape are also automatically entered on the **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu.

g. After the batch job created by the previous step has successfully run, return to the **VSE/ESA FUNCTION SELECTION** menu:

===> 1 (Installation)

h. **INSTALLATION** menu:

===> 1 (Install Programs - V2 Format)

i. **INSTALL PROGRAMS - V2 FORMAT** menu:

===> 2 (Install Program(s) from Tape)

j. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

Enter 1 (install) in the OPT field against the required identifier of the format COBOLMLE...1.1.0, and 2 (skip installation) against any other optional licensed programs you do not intend to install at this time.

VisualAge COBOL MLE for VSE should be installed in the same sublibrary as COBOL/VSE. If you did not install COBOL/VSE in the default sublibrary PRD2.PROD, enter the name of the library and sublibrary where COBOL/VSE is installed on this screen.

k. Press PF5 to generate the installation job.

l. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> 1 (Save the list)

m. **INSTALL ADDITIONAL PROGRAM(S) FROM TAPE** menu:

===> cuu (address of drive with distribution tape)

n. **JOB DISPOSITION** menu:

Make any changes required and press ENTER to submit the job and install VisualAge COBOL MLE for VSE. While the job is running, reply to any system console messages as necessary.

o. Check the output from the batch job created by the previous steps to ensure that the install was successful. If you are installing VisualAge COBOL MLE for VSE from a non-stacked tape, you received the following message:

```
IESI0083I  TAPE IS NOT V2-STACKED
```

This message is for information only and can be ignored. The Librarian RESTORE job ends with a return code of 4. This is not an error. Installation was successful.

Subtopics:

- 2.2.4.1.1 Condition Code and Messages

IBM Library Server

| 2.3.2.5 Step 3: Apply Service

| You can apply service to VisualAge COBOL MLE for VSE from the provided
| service tape using either the Interactive Interface dialogs or a batch
| job.

| You will receive detailed instructions for applying service with the
| service tape.

Subtopics:

IBM Library Server

| 2.3.2.5 Step 3: Apply Service

| You can apply service to VisualAge COBOL MLE for VSE from the provided
| service tape using either the Interactive Interface dialogs or a batch
| job.

| You will receive detailed instructions for applying service with the
| service tape.

Subtopics: