IBM PL/I for VSE/ESA

Installation and Customization Guide

Release 1

SC26-8057-00

IBM PL/I for VSE/ESA

Installation and Customization Guide

Release 1

SC26-8057-00

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page v.

First Edition (April 1995)

This edition applies to Version 1 Release 1 of IBM PL/I for VSE/ESA, 5686-069, and to any subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

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

A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation, Department J58
P.O. Box 49023
San Jose, CA, 95161-9023
United States of America

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

© Copyright International Business Machines Corporation 1993, 1995. All rights reserved.
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 1993, 1995

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

Trademarks

The following terms, denoted by an asterisk (*) in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

AD/Cycle
C/370
CICS
CICS/VSE
IBM
Language Environment
OS/2
SAA
SQL/DS
VSE/ESA

© Copyright IBM Corp. 1993, 1995

About this book

This book provides information on planning for, installing, and customizing IBM* PL/I for VSE/ESA* (PL/I VSE) Version 1 Release 1.

Using your documentation

The publications provided with PL/I VSE are designed to help you do PL/I programming under VSE. Each publication helps you perform a different task.

Where to look for more information

For information about the PL/I VSE library, see Table 1.

Table 1. How to use the publications you receive with PL/I VSE

To...	Use...
Evaluate the product	*Fact Sheet*
Understand warranty information	*Licensed Program Specifications*
Install the compiler	*Installation and Customization Guide*
Understand product changes and adapt programs to PL/I VSE	*Migration Guide*
Prepare and test your programs and get details on compiler options	*Programming Guide*
Get details on PL/I syntax and specifications of language elements	*Language Reference* *Reference Summary*
Diagnose compiler problems and report them to IBM	*Diagnosis Guide*
Get details on compile-time messages[1]	*Compile-Time Messages and Codes*

Note:

1. For details on run-time messages, see the LE/VSE library.

You might also require information about IBM* Language Environment* for VSE/ESA* (LE/VSE). For information about the LE/VSE library, see Table 2.

Table 2. How to use the publications you receive with LE/VSE

To...	Use...
Evaluate Language Environment	*Fact Sheet* *Concepts Guide*
Install LE/VSE	*Installation and Customization Guide*
Understand the LE/VSE program models and concepts	*Concepts Guide* *Programming Guide*
Prepare your LE/VSE-conforming applications and find syntax for run-time options and callable services	*Programming Guide* *Reference Summary*
Debug your LE/VSE-conforming application and get details on run-time messages	*Debugging Guide and Run-Time Messages*
Diagnose problems that occur in your LE/VSE-conforming application	*Diagnosis Guide*
Understand warranty information	*Licensed Program Specifications*

© Copyright IBM Corp. 1993, 1995

For the complete titles and order numbers of these and other related publications, see the "Bibliography" on page 37.

What is new in PL/I VSE

This is a major new release of PL/I, containing many new features and facilities. It brings to VSE many of the functions of the MVS & VM version of PL/I (IBM SAA* AD/Cycle* PL/I MVS & VM), while retaining close source compatibility with the DOS PL/I Optimizing Compiler (DOS PL/I).

PL/I VSE enables you to integrate your PL/I applications into IBM Language Environment for VSE/ESA (LE/VSE). In addition to PL/I's already impressive features, you gain access to LE/VSE's rich set of library routines and enhanced interlanguage communication (ILC) with IBM COBOL for VSE/ESA (COBOL/VSE).

IBM Language Environment for VSE/ESA support

PL/I VSE provides the following functions in the LE/VSE area:

Interlanguage communication (ILC) support:

- Object code produced by PL/I VSE Release 1 can be linked with object code produced by other LE/VSE-conforming compilers (currently only COBOL/VSE).
- PL/I VSE programs can fetch COBOL/VSE phases.
- COBOL/VSE programs can fetch PL/I VSE phases.

Note: PL/I VSE does not support ILC with:

- FORTRAN
- RPG
- DOS/VS COBOL
- C/370*

Limited ILC support is provided for VS COBOL II at Release 3.2 or later.

Common support for multiple operating environments:

- Some of the restrictions on PL/I coding in the CICS* environment have been lifted.
- Procedure OPTIONS option FETCHABLE can be used to specify the procedure that gets control within a fetched phase.
- CEETDLI is supported in addition to PLITDLI and EXEC DLI.
- LE/VSE services provide storage management and condition handling support, as well as PLIDUMP and MSGFILE support for PL/I messages and other output.
- By default, only user-generated output is written to SYSLST. All run-time generated messages are written to MSGFILE.
- ERROR conditions now get control of all system abends. The PL/I message is issued only if there is no ERROR on-unit or if the ERROR on-unit does not recover from the condition via a GOTO.
- Selected items from PL/I Package/2 (the PL/I product for OS/2*) are implemented to allow better coexistence.
  - Limited support of OPTIONS(BYVALUE and BYADDR)

- Limited support of EXTERNAL(environment-name) allowing alternate external names
- Limited support of OPTIONAL arguments/parameters
- Support for %PROCESS statement
- NOT and OR compiler options

Product packaging:

- All PL/I VSE resident library routines are now packaged with LE/VSE, and are loaded at run time rather than link-edited with the application program. Changes to the resident library no longer require PL/I programs to be re-linked.
- At link-edit time, you have the option of getting math results that are compatible with LE/VSE or with DOS PL/I.
- Installation enhancements are provided to ease product installation and migration.

For migration considerations, see the *PL/I VSE Migration Guide.*

Usability enhancements

These enhancements expand the PL/I language statements and options, PL/I data types, and compiler options, to make the language easier to use.

Enhanced double-byte character set (DBCS) support: This support introduces many enhancements that facilitate processing of GRAPHIC and mixed-character data and allows the source of the PL/I program to be in DBCS and/or the single-byte character set (SBCS), rather than only in SBCS.

Hexadecimal data constants: Constants for bit and character data can now be defined in hexadecimal notation, such that each *character* (0-9 and A-F) represents 4 bits.

Interface improvements for all (sub)systems: A new compiler option, SYSTEM, lets the programmer specify the target operating environment (of the generated object code), and the format of the parameters for the MAIN procedure.

Specification of compile-time options: You can specify compile-time options on the *PROCESS statement, a new %PROCESS statement, and in the PARM option of the EXEC IEL1AA JCL statement.

Linking after errors: The COMPILE compile-time option has been enhanced to allow linking to proceed after a severe error.

Run-time options: You can specify program run-time options in the PARM option of the EXEC JCL statement. PL/I VSE and LE/VSE will use these to control the execution of PL/I programs.

Passing parameters to the MAIN procedure: VSE JCL can also be used to pass a parameter to the MAIN PL/I procedure. A slash (/) separates the run-time options from the program parameter.

OPEN statement enhancements:

- There are new parameters on the PL/I OPEN statement that allow additional file attributes to be specified at file open time. These attributes are added to those in the file declaration.
- A vendor exit on the PL/I OPEN statement can be used to change the system logical unit number of the PL/I spill file.
- Data set name sharing for VSAM files, using the DSN option of the ENVIRONMENT attribute.

Date and time enhancements: A new built-in function, DATETIME, returns consistent date and time, including the four-digit year.

PL/I statement numbering options: A new compiler option, NUMBER, specifies that PL/I statement numbers will be derived from the sequence numbers in the program source deck, instead of being allocated sequentially.

Dynamic loading of external procedures: PL/I now supports the FETCH and RELEASE statements, to load external procedures into main storage at run time instead of having them link-edited with the MAIN procedure. (If these external procedures are PL/I, they must be compiled with PL/I VSE.)

New I/O facilities: PL/I VSE provides the following new I/O facilities:

- Support for REGIONAL(2) files
- Support for V and VS formats on REGIONAL(3) files
- Support for DELETE statement on REGIONAL files
- Support for multitrack search on REGIONAL files, using the LIMCT option of the ENVIRONMENT attribute
- Support for VSAM variable-length relative-record data sets (VRDS)
- Support for V format on consecutive unbuffered files
- Support for VS and VBS formats on consecutive buffered files

System programmer functions: A number of significant new features enhance PL/I as a system programming language:

- Support of additional program execution environments.

  PL/I can now be used for some system exit routines, such as the LE/VSE initialization exit.

- Additional support for pointers.

  PL/I built-in functions are now available to perform extended operations on pointers, including pointer arithmetic.

- Additional support for entry variables.

  A new built-in function and pseudovariable, ENTRYADDR, allows programmers to manipulate entry point addresses of procedures.

Extended addressing enhancements

These enhancements exploit the large amounts of storage available in the VSE/ESA environment, making programming easier.

Addressing mode: PL/I VSE programs can be link-edited with AMODE(31) and RMODE(ANY).

Location of variables: PL/I variables can now be located above the 16-megabyte line.

Fullword array subscripts: Array bounds can now be in the range -2^{31} (-2,147,483,648) through $+2^{31}-1$ (+2,147,483,647). The associated built-in functions (such as LBOUND and HBOUND) now return FIXED BINARY(31) values.

AREA and aggregate sizes: An AREA can now have a maximum size of 2,147,483,647 ($2^{31}-1$) bytes.

An aggregate can now have a maximum size of 2,147,483,647 ($2^{31}-1$) bytes. For unaligned BIT arrays and aggregates that contain any unaligned BIT data (arrays or non-arrays), the maximum size is 268,435,455 ($2^{28}-1$) bytes.

These numbers include any control information bytes that might be needed.

Notation conventions used in this book

Throughout this book, syntax is described using the following structure:

- 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

- 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, the items appear vertically, in a stack. If you **must** choose one of the items, one item of the stack appears on the main path. The default, if any, appears above the main path and is chosen by the compiler if you do not specify another choice.

  Note: In some cases, the default is affected by the:

  - System in which the program is being run

- Environmental parameters specified

```
                  ┌─default-item─┐
►►──STATEMENT─────┼─choice 1─────┼─(1)──►◄
                  └─choice 2─────┘
```

Note:

[1] Because *choice 1* appears on the horizontal bar, one of the items must be included in the statement. If you don't specify either *choice 1* or *choice 2*, the compiler implements the default for you.

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 is a *repeat arrow*, and it 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.

- If there is a comma as part of the repeat arrow, you must use a comma to separate items in a series.

```
                  ┌─,───────────────┐
►►──STATEMENT─────▼─repeatable-item─┴───►◄
```

If the comma appears below the repeat arrow line instead of on the line as shown in the previous example, the comma is optional as a separator of items in a series.

- A syntax fragment is delimited in the main syntax diagram by a set of vertical lines. The corresponding meaning of the fragment begins with the name of the fragment followed by the syntax, which starts and ends with a vertical line.

```
►►──STATEMENT──┤ fragment ├──►◄
```

fragment:

```
├──syntax items──┤
```

- Keywords appear in uppercase (for example, STATEMENT). 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 other symbols are shown, you must enter them as part of the syntax.

Chapter 1. Planning for PL/I VSE

This chapter helps you plan for installing and customizing PL/I VSE.

Planning for installation

This section contains planning information to help you properly install PL/I VSE:

- What you receive with PL/I VSE
- Choosing required and optional software
- Checking you have enough DASD storage
- Planning where to install PL/I VSE
- Checking service updates
- Ensuring you have the right publications

What you receive with PL/I VSE

When you order PL/I VSE, you receive:

- Basic machine-readable material
- Basic unlicensed publications
- Optional unlicensed materials (if you specify the required feature numbers when ordering PL/I VSE)

The following sections describe these materials and publications.

Basic machine-readable material

You receive PL/I VSE basic machine-readable material on one of these distribution media:

- 9-track magnetic tape written at 6250 BPI
- 3480 tape cartridge
- ¼-inch tape cartridge

The distribution medium contains all the programs and data you need to install PL/I VSE.

Table 3 describes the distribution media.

Table 3. Basic machine-readable material: distribution media

Medium	Feature number	Physical volume	External label	Volser
6250 tape	5801	1	PL/I VSE RELEASE 1	unlabeled
3480 cart.	5802	1	PL/I VSE RELEASE 1	unlabeled
¼-inch tape	5804	1	PL/I VSE RELEASE 1	unlabeled

Table 4 on page 2 describes the contents of files on the distribution media.

© Copyright IBM Corp. 1993, 1995

Table 4. Distribution media: file contents

File	Contents
1	Header file containing PL/I VSE copyright statement
2	Backup file ID PLI.VSE....1.1.0 followed by an MSHP System History File
3	PL/I VSE library file containing the PL/I VSE product
4	Null (Tape Mark)
5	End of backup (EOB) record
6	Null (Tape Mark)

Table 5 shows the component identifier (COMP ID) and component level code (CLC) for PL/I VSE.

Table 5. Component ID and CLC

Component ID	CLC	Description
5686-069-00	18P	PL/I COMP BASE

Optional material

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

Basic unlicensed publications

Table 6 lists PL/I VSE basic unlicensed publications. When you order PL/I VSE, you receive one of each of these publications.

Table 6. Basic unlicensed publications

Publication title	Form number
PL/I VSE Licensed Program Specifications	GC26-8055
PL/I VSE Installation and Customization Guide	SC26-8057
PL/I VSE Compile-Time Messages and Codes	SC26-8059

Optional unlicensed publications

Table 7 lists PL/I VSE optional unlicensed publications. You can order one free copy of an optional unlicensed publication by specifying its feature number.

Table 7. Optional unlicensed publications

Publication title	Form number	Feature number
PL/I VSE Diagnosis Guide	SC26-8058	7174
PL/I VSE Fact Sheet	GC26-8052	7171
PL/I VSE Language Reference	SC26-8054	7173
PL/I VSE Migration Guide	SC26-8056	7175
PL/I VSE Programming Guide	SC26-8053	7172
PL/I VSE Reference Summary	SX26-3836	7176

For a list of publications for related products, see “Bibliography” on page 37.

What you need to install PL/I VSE

The following sections describe system requirements for installing PL/I VSE.

Licensed programs

Table 8 lists licensed programs you need to install and use PL/I VSE.

Table 9 lists optional licensed programs you can use with PL/I VSE.

Use the minimum release listed or any subsequent release of these licensed programs.

Table 8. Required licensed programs

Licensed program	Minimum release
VSE/ESA	Version 1 Release 4 Version 2 Release 1
IBM Language Environment for VSE/ESA	Version 1 Release 1
High Level Assembler/MVS & VM & VSE	Release 1

Table 9. Optional licensed programs

Licensed program	Minimum release
CICS/VSE*	Version 2 Release 3
SQL/DS*	Version 3 Release 4
DFSORT/VSE	Version 3 Release 1
DOS/VS Sort/Merge (VSE 1.4 only)	Version 2 Release 5
DL/I DOS/VS	Release 10
BookManager/Read	Release 2 (to view softcopy documentation)

DASD storage

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

- VSE Librarian library
- System history file

Table 10 lists the minimum DASD storage that PL/I VSE requires for a VSE Librarian library. This does not include storage for other licensed programs installed in the library.

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

Table 10. Minimum library storage required

LIBR BLKS[1]	3350 CYL	3375 CYL	3380 CYL	3390 CYL	9345 CYL	FBA BLKS
4200	14	15	10	9	11	8416

Note:

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

If you plan to use a separate system history (MSHP) file for PL/I VSE, see Table 11 on page 4 for the DASD storage required.

Table 11. Storage required for system history (MHSP) file IJSYSHF

3350 CYL	3375 CYL	3380 CYL	3390 CYL	9345 CYL	FBA BLKS
1	2	1	1	1	900

Planning where to install PL/I VSE

All PL/I VSE installation jobs assume you are using the default sublibrary PRD2.PROD. If you decide to install PL/I VSE elsewhere, you need to change the installation jobs.

Other products that work with PL/I VSE also use PRD2 as a default library, so you may already have a PRD2 library. If you plan to install PL/I VSE in an existing library, make sure there is enough free storage for PL/I VSE.

To check free storage, use the LIBR program LISTDIR command to list the directory information for the library.

Checking service updates

Before installing PL/I VSE, check with your IBM Support Center or use either Information/Access or SoftwareXcel Extended to see whether there is additional service information you need. To get this information, specify the following UPGRADE and SUBSET values: PLIVSE110 and 06918P.

Publications useful during installation

For a list of VSE/ESA publications you might find useful during installation, see "Bibliography" on page 37.

Planning for customizing PL/I VSE

This section contains planning information to help you customize PL/I VSE:

- Deciding whether or not to customize PL/I VSE
- Planning to customize the IBM-supplied defaults for PL/I VSE compile-time options

Deciding whether to customize

You need to consider whether the IBM-supplied defaults for PL/I VSE compile-time options suit the needs of your site. Compile-time options control features such as:

- Suppressing part of the assembly listing
- Generating symbol cross-reference information

Make sure that PL/I VSE serves the needs of the application programmers at your site. Confer with them while you evaluate the customization options for PL/I VSE, particularly those options that are also available to the application programmers. This ensures the modifications you make best support the application programs being developed at your site.

Planning to customize PL/I VSE options

Table 12 lists the IBM-supplied default values for the options in the PL/I VSE customization assembler macro. (The options in this assembler macro determine the defaults for PL/I VSE compile-time options.) If you plan to customize a default, write the new value in the **Customized installation default** column. The options marked '*' can only be specified during installation customization—they are not available to application programmers.

These options are described in more detail in Chapter 3, "Customizing PL/I VSE" on page 16.

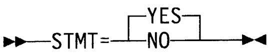
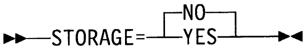
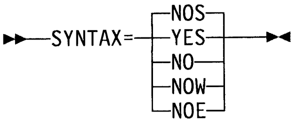
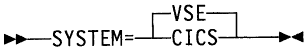
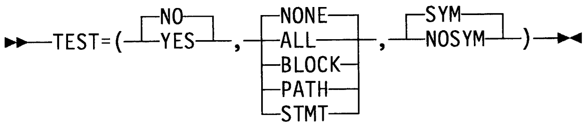
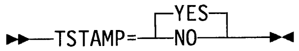
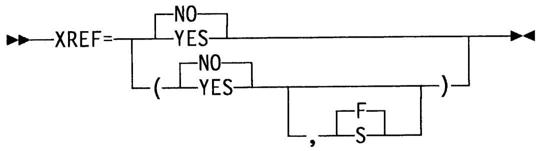
Table 12 (Page 1 of 3). Compile-time option defaults

Option		Customized installation default	IBM-supplied default	Description
┌─NO──┐ ►►──AGGREGA=──┴─YES─┴──►◄		______	NO	Lists aggregates and their size
┌─NO──┐ ►►──ATTRIBU=──┴─YES─┴──►◄		______	NO	Lists atrributes of identifiers
┌─NOS─┐ ►►──COMPILE=──┼─YES─┼──►◄ ├─NO──┤ ├─NOW─┤ └─NOE─┘		______	NOS	Stops processing after errors are found in syntax checking
┌─'OPTIMIZE'─────────┐ ►►──CONTROL=──┴─'*character-string*'─┴──►◄		______	OPTIMIZE	Specifies that any compile-time options previously deleted are available
┌─,──┐ ►►──DELETE=(─▼*item*─┴─)──►◄	*	______	-	Disallows overriding default options at compile-time
┌─NO──┐ ►►──ESD=──┴─YES─┴──►◄		______	NO	Lists external symbol dictionary
┌─I─┐ ►►──FLAG=──┼─W─┼──►◄ ├─E─┤ └─S─┘		______	I	Suppresses diagnostic messages below a certain severity
┌─2─┐ ┌─72─┐ ►►──FMARGIN=(──┴─*m*─┴─,──┴─*n*──┴──┬──────┬─)──►◄ └─,*c*─┘	*	______	(2,72)	Identifies position of PL/I source and a carriage control character
┌─73─┐ ┌─80─┐ ►►──FSEQUEN=──┬─(──┴─*m*──┴─,──┴─*n*──┴─)─┬──►◄ └─NO──────────────────┘	*	______	(73,80)	Specifies the columns used for sequence numbers
┌─NO──┐ ►►──GONUMBE=──┴─YES─┴──►◄		______	NO	Includes line numbers in run-time messages
┌─NO──┐ ►►──GOSTMT=──┴─YES─┴──►◄		______	NO	Includes statement numbers in run-time messages
┌─NO──┐ ►►──GRAPHIC=──┴─YES─┴──►◄		______	NO	Specifies that DBCS is used in source
┌─NO──┐ ►►──INCLUDE=──┴─YES─┴──►◄		______	NO	Allows secondary input to be included without using preprocessor

Table 12 (Page 2 of 3). Compile-time option defaults

Option	Customized installation default	IBM-supplied default	Description
┌YES┐ ►►─INSOURC=─┴NO─┴──►◄	_______	YES	Lists preprocessor input
┌NOSPROG┐ ►►─LANGLVL=(OS,─┴SPROG──┴─)─►◄	_______	(OS,NOSPROG)	Defines the level of the language supported
┌55┐ ►►─LINECOU=─┴*n*─┴───►◄	_______	55	Specifies number of lines per page on listing
┌NO─┐ ►►─LIST=─┴YES┴───►◄	_______	NO	List object code produced by compiler
┌YES┐ ►►─LMESSAG=─┴NO─┴──►◄	_______	YES	Specifies concise or full message format
┌NO─┐ ►►─MACRO=─┴YES┴───►◄	_______	NO	Allows preprocessor to be used
┌NO─┐ ►►─MAP=─┴YES┴───►◄	_______	NO	Lists offsets of variables in static control section and DSAs
┌NO──────────┐ ►►─MARGINI=─┴'*character*'┴───►◄	_______	NO	Highlights any source outside margins
┌NO─┐ ►►─MDECK=─┴YES┴───►◄	_______	NO	Produces a source deck from preprocessor output
┌NO─┐ ►►─NEST=─┴YES┴───►◄	_______	NO	Indicates do-group and block level by numbering in margin
►►─NOT='*character-string*'─►◄	_______	-	Specifies alternate symbols to be used as the logical NOT operator
┌NO─┐ ►►─NUMBER=─┴YES┴───►◄	_______	NO	Numbers statements according to line in which they start
┌NO─┐ ►►─OFFSET=─┴YES┴───►◄	_______	NO	Generates a listing associating statement numbers with offsets
┌NO───┐ ►►─OPTIMIZ=─┼TIME─┼───►◄ ├0────┤ └2────┘	_______	NO	Improve run-time performance
┌YES┐ ►►─OPTIONS=─┴NO─┴──►◄	_______	YES	Lists options used
►►─OR='*character-string*'─►◄	_______	-	Specifies alternate symbols to be used as the logical OR symbol and in the string concatenation operator
┌MAX─┐ ►►─SIZE=─┼*n*───┼───►◄ ├*n*K──┤ ├─*n*──┤ └─*n*K─┘	_______	MAX	Controls the amount of storage used by the compiler
┌YES┐ ►►─SOURCE=─┴NO─┴──►◄	_______	YES	Lists source program or preprocessor output

Table 12 (Page 3 of 3). Compile-time option defaults

Option	Customized installation default	IBM-supplied default	Description
	______	YES	Numbers statements sequentially
	______	NO	Lists storage requirements for the compiled object module
	______	NOS	Stops processing after errors are found in preprocessing
	______	VSE	Specifies the target system (VSE, for example) under which the object program runs
 TEST=(NO YES , NONE ALL BLOCK PATH STMT , SYM NOSYM)	______	(NO,NONE,SYM)	Specifies which debug hooks will be inserted in the object module
	* ______	YES	Specifies whether the compiler places the compilation time and date in the STATIC INTERNAL CSECT
	______	(NO,F)	Lists statements in which each identifier is used

Notes:

* You can specify these options during installation customization only. These options are not available to application programmers.

See the *PL/I VSE Programming Guide* for a list of options available to the programmer.

Chapter 2. Installing PL/I VSE

This chapter describes the installation method and the step-by-step procedures you use to install and activate the functions of PL/I VSE.

Overview of installation

You install PL/I VSE using the Maintain System History Program (MSHP).

Checklist for installing PL/I VSE

Table 13 lists the steps and associated jobs for installing PL/I VSE. The remaining sections in this chapter describe each step. You can use Table 13 as a checklist.

Table 13. Summary of steps for installing PL/I VSE

Step	Description	Installation job	Page
__ 1	Backup the original system.	—	8
__ 2	Allocate space for the library. (Omit if using the default sublibrary.)	LIBRDEF	8
__ 3	Install PL/I VSE.		10
	Method 1. Install PL/I VSE using the Interactive Interface.	—	10
	Method 2. Install PL/I VSE using a batch job.	IEL1INST	12
__ 4	Verify the installation of PL/I VSE.	IEL1EIVP	14

Step 1: Backup the original system

Make a backup copy of your current PL/I VSE library or the library you intend to install PL/I VSE into, and the system history file.

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

Step 2: Allocate space for the Library (omit if using the default sublibrary)

By default, PL/I VSE is installed into the PRD2.PROD sublibrary.

If you want to install PL/I VSE into the default PRD2.PROD sublibrary, go to the next step.

If you want to install PL/I VSE into a sublibrary other than PRD2.PROD, proceed with this step.

Decide where to allocate space for the PL/I VSE sublibrary. 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 1 on page 9 shows JCL to list the VTOC for the volume with serial number SYSWK1.

© Copyright IBM Corp. 1993, 1995

```
// JOB    LVTOC
*
*  List volume table of contents
*
// ASSGN  SYS004,DISK,TEMP,VOL=SYSWK1,SHR
// ASSGN  SYS005,SYSLST
// EXEC   LVTOC
/*
/&
```

Figure 1. Job to list the contents of a DASD volume

The disk space selected for PL/I VSE is used in the LIBR installation job to allocate the VSE Librarian library in the sample job shown in Figure 2.

```
// JOB    LIBRDEF
*
*  Create a library for PL/I VSE
*
// OPTION LOG
*  Label for the PL/I VSE library    1
// DLBL   PLIVSE,'PLIVSE.LIBRARY',1999/365,SD
// EXTENT SYS002,volser,,,rtrk,ntrk
// ASSGN  SYS002,DISK,VOL=volser,SHR
// EXEC   LIBR
 DELETE LIB=PLIVSE         2
 DEFINE LIB=PLIVSE
/*
/&
```

Figure 2. Job to allocate the PL/I VSE library space

In area 1 change the *filename* (*PLIVSE* in the example) and *file-id* (*PLIVSE.LIBRARY* in the example) of the PL/I VSE sublibrary to suit your installation.

The variable *ntrk* represents the number of blocks or tracks required; this is the size of the extent needed in Figure 2. If you are using an FBA device (such as an IBM 3370) or a CKD device (such as an IBM 3380), see "DASD storage" on page 3 for the number of blocks/tracks required. The variable *rtrk* represents the start position of the extent. Change *rtrk* and *ntrk* to suit the type of device being used for the current installation.

The variable *volser* represents the name of the DASD volume where the library will be placed. Change this to the name of the volume that you are using for your PL/I VSE library.

The Librarian job step in area 2 includes a DELETE statement before the DEFINE statement so you can rerun the job. When you run the job for the first time, the DELETE statement causes these messages:

```
L101I  LIBRARY PLIVSE DOES NOT EXIST
L027I  ABNORMAL END DURING DELETE COMMAND
       PROCESSING
L113I  RETURN CODE OF DELETE IS 8
```

Ignore these messages; the job will continue, and allocate the library.

Step 3: Install PL/I VSE

You can install PL/I VSE using either the Interactive Interface of VSE or a batch installation job.

Method 1: Install PL/I VSE using the Interactive Interface

To install PL/I VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (If you would like information about the functions of the Interactive Interface, refer to *VSE/ESA Administration*.)

Mount the PL/I 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:

   a. If you received PL/I VSE as a stacked tape, which contains one or more optional products, select:
      ===> ***1*** (Install Programs - Stacked V2 Format)

      INSTALL PROGRAMS - STACKED V2 FORMAT menu:
      ===> ***1*** (Prepare for Installation (Stacked Tapes Only))

      PREPARE FOR INSTALLATION (STACKED TAPES ONLY) menu:
      ===> ***cuu***
      (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 identifier for PL/I VSE is `PLI.VSE....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 the identifier:

      PLI.VSE....1.1.0 (PL/I VSE)

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.

b. If you received a tape that contains only PL/I VSE, select:

   ===> ***2***
   (Install Programs - Non-stacked V2 Format or V1 Format)

   INSTALL PROGRAMS - NON-STACKED V2 OR V1 FORMAT menu:

   ===> ***PLI.VSE....1.1.0*** (Tape Label)

   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.

c. **VSE/ESA INSTALL PRODUCT(S) TAPE SPECIFICATION** menu:

   ===> ***cuu***
   (the address of the tape drive where you mounted the
   PL/I VSE tape)

d. **JOB DISPOSITION** menu:

   Make any changes required and press ENTER to submit the job and install PL/I VSE.

Messages

Check the list output for error conditions. If there are any error messages:

- See *VSE/ESA Administration* for corrective action
- Correct the error(s)
- Rerun the job
- Recheck the list output

List the directories and components

If you wish to print the directory entries from the sublibrary where PL/I VSE is installed, and the component records from the system history file, you can now submit a batch job to do this. The last two steps of the job shown in Figure 3 on page 13 will produce these lists.

Method 2: Install PL/I VSE using a batch job

The batch installation job stream for installing PL/I VSE 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 statements, with the necessary information for the system history file, are included in the job stream.

Depending on how you request the PL/I VSE product you may receive different installation tapes. One may contain only the PL/I VSE product, the other may be a stacked tape containing one or more optional program products. The job shown in Figure 3 on page 13 will handle both types of tape without any specific change depending on the contents of the tape.

Create and tailor the following job stream, mount the distribution tape, and run the installation job.

Figure 3 on page 13 provides the JCL required to install PL/I VSE. Tailor this JCL to suit the requirements of your installation.

As many as five modifications may be required to tailor the JCL. The keys within Figure 3 on page 13 are explained individually and refer to the sections that accompany the JCL description.

```
// JOB    IEL1INST
*
*  Install PL/I VSE
*
// OPTION LOG
// DLBL   PLIVSE,'PLIVSE.LIBRARY'       1
// EXTENT SYS005
// ASSGN  SYS005,DISK,VOL=volser,SHR
// ASSGN  SYS006,cuu           2
// MTC    REW,SYS006
* ------------------------------------------
*  This step installs PL/I VSE
*  from the distribution tape
*  using the VSE system history file
* ------------------------------------------
// EXEC   MSHP,SIZE=900K,PARM='PIDSTACKED'
 INSTALL PROD FROMTAPE ID='PLI.VSE....1.1.0' -
    PROD INTO=PRD2.PROD          3
/*
* ------------------------------------------
*  List the PL/I VSE Library
* ------------------------------------------
// EXEC   LIBR
 LISTDIR SUBLIB=PRD2.PROD  -       4
     OUTPUT=NORMAL -
     UNIT=SYSLST
/*
* ------------------------------------------
*  Retrace the PL/I VSE product  5
* ------------------------------------------
// EXEC   MSHP,SIZE=900K
RETRACE COMPONENT IDENTIFIER=5686-069-00
/*
// MTC    RUN,SYS006
/*
/&
```

Figure 3. Job to install PL/I VSE

Specify the label information: In area 1 if you are installing PL/I VSE into a sublibrary other than the default then insert DLBL, EXTENT, and ASSGN information as specified in Figure 2 on page 9. The library name must match the name used in the allocation job in Figure 2.

If you are installing PL/I VSE into the default sublibrary, omit these DLBL, EXTENT, and ASSGN statements.

Assign the distribution tape: Assign the distribution tape in area 2 to logical unit SYS006. Replace *cuu* with the address of the tape drive where the distribution tape is mounted.

Install PL/I VSE: Area 3 of the job calls MSHP to install PL/I VSE into the sublibrary identified on the INTO operand of the INSTALL statement. If you are installing PL/I VSE into a sublibrary other than the default then the name of the sublibrary on the INTO operand of the INSTALL statement must be changed to reflect this sublibrary. For more information about the install options, see “Maintain System History Program (MSHP)” in *VSE/ESA System Control Statements.*

Note: The EXEC JCL statement should specify PARM='PIDSTACKED' if PL/I VSE is delivered as part of an Optional Products tape. It is not an error to specify it if PL/I VSE is delivered as a single component, although a message will be produced.

List the directory entries: The step in area 4 of the job lists the directory entries of the sublibrary where PL/I VSE was installed. Remove this step if a directory list is not required. If you have installed PL/I VSE into a sublibrary other than the default then the name of the sublibrary must be changed to reflect that used in Figure 2 on page 9.

Entries for PL/I VSE have a three character prefix of ***IEL*** to distinguish them from other products; there is one exception to this rule: `$$$C018P.Z`

Retrace the PL/I VSE product in the system history file: The final step in area 5 of the job prints the component records from the system history file for PL/I VSE. Remove this step if a retrace listing is not required.

If this job has to be rerun, remember first to restore the system history file, which should have been backed up before running this install job, and second to re-run the library allocation step, if applicable.

Step 4: Verify the installation of PL/I VSE

The installation verification program IEL1ESO1.P is a sample program provided on the distribution tape. It lets you check that your installation is successful by exercising representative features of PL/I VSE.

Run the sample program

Figure 4 shows the job IEL1EIVP.Z provided in the installed sublibrary. It runs the sample program from the same sublibrary. Note that for successful execution, this job requires IBM Language Environment for VSE/ESA to be installed. If you installed PL/I VSE or LE/VSE in a sublibrary other than the default, modify the IBM-supplied JCL that runs the verification program.

Run the job listed in Figure 4 to verify the success of your installation.

```
// JOB    IEL1EIVP
*
*  Sample JCL to verify installation of PL/I VSE
*
// LIBDEF *,SEARCH=(PRD2.PROD,PRD2.SCEEBASE)          1
// EXEC   PROC=IEL1EIVQ,
                 SUBLIB='PRD2.PROD',            2
                 VOL1=vvvvvv,START1=nnnnn,LENGTH1=30,   3
                 VOL2=vvvvvv,START2=nnnnn,LENGTH2=30
// EXEC   PROC=IEL1EIVR
/&
```

Figure 4. Job to verify the success of your installation

Note: Since the IEL1EIVQ procedure contains a permanent assignment for SYSPCH, this job cannot be run in a dynamic partition.

In area 1 specify the sublibrary where PL/I VSE resides, and the sublibrary where LE/VSE resides.

In area 2 specify the sublibrary where PL/I VSE resides.

The job requires two work files, each approximately 30 tracks of 3390 DASD. In area 3, cataloged procedure IEL1EIVQ requires six parameters, three for each work file:

VOL1 The volume ID of a DASD volume for the first work file

START1 The starting track number for the first work file

LENGTH1 The number of tracks for the first work file

VOL2 The volume ID for the second work file

START2 The starting track number for the second work file

LENGTH2 The number of tracks for the second work file

A return code of 2 (two) for the job indicates successful completion of the sample program.

Note: You can find a complete listing of this sample program and its output in the *PL/I VSE Programming Guide*.

Chapter 3. Customizing PL/I VSE

This chapter describes how to customize PL/I VSE compile-time options after installation.

This chapter consists of two sections:

"Compile-time option descriptions"
Describes the compile-time options, their defaults and other values.

"Customizing compile-time option defaults" on page 29
Describes how to change compile-time options using the supplied assembler macro.

Compile-time option descriptions

Table 12 on page 5 lists PL/I VSE compile-time options, their IBM-supplied defaults and your planned customizations.

You can specify some options only during installation customization (they are not available to application programmers). These options are marked with an asterisk (*) in Table 12 on page 5 and described in "Compile-time options with restricted use" on page 27.

PL/I VSE compile-time options are described on the following pages in alphabetic order.

Options that generate a compiler listing are described only briefly. The generated listing is described in the *PL/I VSE Programming Guide.*

In these syntax diagrams, IBM-supplied defaults are shown above the main path.

For detailed descriptions of the option formats, see the *PL/I VSE Programming Guide.*

© Copyright IBM Corp. 1993, 1995

AGGREGA

```
                ┌─NO──┐
►►──AGGREGA=──┴─YES─┴────►◄
```

Specifies whether the compiler includes an aggregate length table in the compiler listing. This table shows the lengths of all arrays and major structures in the source program.

ATTRIBU

```
                ┌─NO──┐
►►──ATTRIBU=──┴─YES─┴────►◄
```

Specifies whether the compiler includes in the compiler listing a table of source-program identifiers and their attributes. If both ATTRIBU=YES and XREF=YES are in effect, the two tables are combined.

COMPILE

```
                ┌─NOS─┐
►►──COMPILE=──┼─YES─┼────►◄
                ├─NO──┤
                ├─NOW─┤
                └─NOE─┘
```

Specifies how the compiler responds to error conditions. The compiler continues as follows, depending on the severity of errors detected:

NOS Do not compile if a severe error or an unrecoverable error is detected. This is the default.

YES Compile, unless an unrecoverable error occurs.

NO Do not compile after syntax checking.

NOW Do not compile if a warning, an error, a severe error, or an unrecoverable error is detected.

NOE Do not compile if an error, a severe error, or an unrecoverable error is detected.

CONTROL

```
                ┌─'OPTIMIZE'──────────┐
►►──CONTROL=──┴─'character-string'─┴────►◄
```

Specifies whether any compile-time options deleted for your site are available for this compilation. The application programmer must still specify the appropriate keywords in order to use the options. The CONTROL option must be specified with a password established for your site. The password can be up to eight SBCS characters. An incorrect password causes processing to end.

DELETE

```
             ┌─,────┐
►►──DELETE=(─▼item─┴─)──►◄
```

See “Compile-time options with restricted use” on page 27.

ESD

```
          ┌─NO──┐
►►──ESD=──┴─YES─┴─────►◄
```

Specifies whether the external symbol dictionary (ESD) is included in the compiler listing.

FLAG

```
           ┌─I─┐
►►──FLAG=──┼─W─┼─────►◄
           ├─E─┤
           └─S─┘
```

Specifies the minimum severity of error that requires a message to be listed in the compiler listing.

I List all informational messages.

W List all except informational messages.

E List all except warning and informational messages.

S List only severe error and unrecoverable error messages.

FMARGIN

```
               ┌─2─┐   ┌─72─┐
►►──FMARGIN=(──┴─m─┴─,─┴─n──┴──┬──────┬─)──►◄
                               └─,c───┘
```

See “Compile-time options with restricted use” on page 27.

FSEQUEN

```
                  ┌─73─┐   ┌─80─┐
►►──FSEQUEN=──┬─(─┴─m──┴─,─┴─n──┴─)─┬──────►◄
              └─NO──────────────────┘
```

See “Compile-time options with restricted use” on page 27.

GONUMBE

```
               ┌─NO──┐
►►──GONUMBE=───┴─YES─┴──────►◄
```

Specifies whether the compiler produces additional information in run-time messages. This information includes line numbers from the source program.

Alternatively, these line numbers can be derived by using both the offset address (which is always included in run-time messages) and the table produced by OFFSET=YES. (NUMBER must equal YES.)

Using GONUMBE=YES requires you to set NUMBER=YES, STMT=NO, and GOSTMT=NO.

GOSTMT

```
              ┌─NO──┐
►►──GOSTMT=───┴─YES─┴──────►◄
```

Specifies whether the compiler produces additional information in run-time messages. This information includes statement numbers from the source program.

Alternatively, these statement numbers can be derived by using the offset address, which is always included in run-time messages, and the table produced by OFFSET=YES. (STMT must equal YES.)

Using GOSTMT=YES requires you to set STMT=YES, NUMBER=NO, and GONUMBE=NO.

GRAPHIC

```
               ┌─NO──┐
►►──GRAPHIC=───┴─YES─┴──────►◄
```

Specifies whether source code can include double-byte character set (DBCS) data. With GRAPHIC=YES, the compiler recognizes (X'0E' and X'0F') as DBCS shift codes.

INCLUDE

```
               ┌─NO──┐
►►──INCLUDE=───┴─YES─┴──────►◄
```

Specifies whether the syntax-checking stage of the compiler handles the inclusion of PL/I source statements for programs that use the %INCLUDE statement. This method is faster than using the preprocessor for programs that use the %INCLUDE statement but no other PL/I preprocessor statements. The INCLUDE option has no effect if you specify MACRO=YES.

INSOURC

```
             ┌─YES─┐
►►──INSOURC=─┴─NO──┴─────►◄
```

Specifies whether the compiler includes a listing of the source program (including preprocessor statements) in the compiler listing. This option is applicable only when using the preprocessor; therefore, MACRO must equal YES.

LANGLVL

```
                  ┌─NOSPROG─┐
►►──LANGLVL=(OS,──┴─SPROG───┴─)──►◄
```

specifies the level of PL/I language the compiler is to support. OS is the only level currently available.

Two LANGLVL suboptions tell the compiler how to support pointers used in expressions.

NOSPROG — Specifies that the compiler is ***not*** to allow the additional support for pointers allowed under SPROG. This is the IBM-supplied default.

SPROG — Specifies that the compiler is to allow comparison of pointers, arithmetic manipulation of pointers, and use of the POINTERADD, BINARYVALUE, and POINTERVALUE built-in functions.

For more information on the effects of this suboption, see the *PL/I VSE Language Reference.*

LINECOU

```
             ┌─55─┐
►►──LINECOU=─┴─n──┴─────►◄
```

Specifies the number of lines to include in each page of the compiler listing, including heading lines and blank lines.

n The number of lines

n must be in the range 1 through 32767, but only headings are generated if you specify less than 7. If *n* is 100 or greater, the output produced by the LIST and MAP options is in one-column format.

LIST

```
          ┌─NO──┐
►►──LIST=─┴─YES─┴─────►◄
```

Specifies whether the compiler includes a listing of the object module in the compiler listing. This is given in a syntax similar to assembler language instructions.

LMESSAG

```
              ┌─YES─┐
►►──LMESSAG=──┴─NO──┴──────►◄
```

Produces messages in long form (YES) or in short form (NO).

MACRO

```
            ┌─NO──┐
►►──MACRO=──┴─YES─┴──────►◄
```

Specifies whether the preprocessor processed the source program. MACRO overrides INCLUDE, when both are specified.

MAP

```
          ┌─NO──┐
►►──MAP=──┴─YES─┴──────►◄
```

Specifies whether the compiler produces tables showing the organization of the static storage for the object module. These tables show how variables are mapped in the static internal control section and in DSAs, thus enabling STATIC INTERNAL and AUTOMATIC variables to be found in PLIDUMP. If LIST=YES, then MAP=YES produces tables showing constants, control blocks, and INITIAL variable values.

MARGINI

```
              ┌─NO──────────┐
►►──MARGINI=──┴─'character'─┴──────►◄
```

Specifies whether the compiler includes a specified character both in the column preceding the left-hand margin, and in the column following the right-hand margin, of the listings resulting from the INSOURC and SOURCE options. Any text in the source input that precedes the left-hand margin shifts left one column, and any text that follows the right-hand margin shifts right one column. Thus text outside the source margins can be easily detected.

The MARGINI option has one operand:

character The character that prints as the margin indicator

MDECK

```
            ┌─NO──┐
►►──MDECK=──┴─YES─┴──────►◄
```

Specifies whether the preprocessor produces a copy of its output on SYSPCH.

This option allows you to retain the output from the preprocessor as a file of 80-column records.

NEST

```
         ┌─NO──┐
►►──NEST=─┴─YES─┴──────►◄
```

Specifies whether the listing resulting from SOURCE=YES indicates the block level and the DO-group level for each statement.

NOT

```
►►──NOT='character-string'──►◄
```

Specifies up to seven alternate symbols, any one of which may be used as the logical NOT operator. If you do not supply an alternate value, the default symbol (the logical NOT operator) applies.

For example, if you specify `NOT='˜'` then the tilde character, (˜, X'A1') is recognized as the logical NOT operator. The standard NOT symbol (¬) is not recognized.

Similarly, if you specify `NOT='˜¬'` then either the tilde character or the standard NOT symbol is recognized as the logical NOT operator.

NUMBER

```
           ┌─NO──┐
►►──NUMBER=─┴─YES─┴──────►◄
```

Specifies whether the numbers specified in the sequence fields in the source input records are used to derive the statement numbers in listings resulting from AGGREGA=YES, ATTRIBU=YES, LIST=YES, OFFSET=YES, SOURCE=YES, and XREF=YES.

If NUMBER=NO, then STMT=YES and GONUMBE=NO are required. If STMT=NO or GONUMBE=YES, then NUMBER=YES is required.

OFFSET

```
           ┌─NO──┐
►►──OFFSET=─┴─YES─┴──────►◄
```

Specifies whether the compiler prints a table of statement or line numbers for each procedure, with their offset addresses relative to the primary entry point of the procedure. This information is useful in identifying the statement being processed when an error occurs, and when a listing of the object module is available. The object module is obtained by using the LIST option.

If GOSTMT=YES, statement numbers, as well as offset addresses, are included in run-time messages. If GONUMBE=YES, line numbers, as well as offset addresses, are included in run-time messages.

OPTIMIZ

```
               ┌─NO───┐
►►──OPTIMIZ=──┼─TIME─┼────►◄
               ├─0────┤
               └─2────┘
```

Specifies the type of optimization required:

NO Specifies fast compilation speed, but inhibits optimization.

TIME Specifies that the compiler optimizes the machine instructions generated to produce a more efficient object program. A secondary benefit to this type of optimization can be a reduction in the amount of storage required for the object module. Using OPTIMIZ=TIME generally requires more compile time than OPTIMIZ=NO. In this case, statement numbers in the program listings might not correspond to the statement numbers issued in messages.

0 The equivalent of OPTIMIZ=NO.

2 The equivalent of OPTIMIZ=TIME.

OPTIONS

```
               ┌─YES─┐
►►──OPTIONS=──┴─NO──┴────►◄
```

Specifies whether the compiler includes in the compiler listing the compile-time options used during the compilation.

This list includes all the compile-time options applied by default, those specified in the PARM parameter of an EXEC statement, and those specified in a PROCESS statement.

OR

```
►►──OR='character-string'──►◄
```

Specifies up to seven alternate symbols, any one of which can be used as the logical OR operator and the string concatenation operator. If you do not supply an alternate value, the default symbol (the logical OR operator) applies.

For example, if you specify `OR='\'` then the backslash character, (\, X'E0') is recognized as the logical OR operator and two consecutive backslashes (\\) are recognized as the concatenation operator. The standard OR symbol (|) is not recognized.

Similarly, if you specify `OR='\|'` then either the backslash or the standard OR symbol is recognized as the logical OR operator, and either or both can be used to form the concatenation operator.

SIZE

```
          ┌─MAX─┐
►►──SIZE=─┼─n───┼───────►◄
          ├─nK──┤
          ├──n──┤
          └──nK─┘
```

Limits the amount of storage the compiler uses. For example, use the SIZE option when you are dynamically invoking the compiler to ensure that space is left for other purposes.

The SIZE option can be expressed in five forms:

n Requests *n* bytes of storage. Leading zeros are not required.

*n*K Requests *n*K bytes of storage. (1K bytes=1024 bytes.) Leading zeros are not required.

MAX Requests as much storage in the virtual machine as possible, less 55K for system overheads.

-*n* Requests all but *n* bytes of storage. This leaves *n* bytes of storage for use by the operating system or other programs. Leading zeros are not required.

-*n*K Requests all but *n*K bytes of storage. This leaves *n*K bytes of storage for use by the operating system or other programs. Leading zeros are not required.

The IBM-supplied default is SIZE=MAX, which allows the compiler to use as much storage as it can obtain, less 55K for operating system overheads.

When you specify a limit, the amount of storage used by the compiler depends on how the operating system was generated, and the method used for storage allocation. The compiler assumes that buffers, data management routines, and the compiler phases take up a fixed amount of storage, but this amount can vary.

When you specify SIZE=MAX, the compiler obtains all available space (after allowance for subsequent data management storage areas). If you specify a limit, the compiler requests this amount. If the amount available is less than specified, but is more than the minimum workspace required, compilation proceeds. If insufficient storage is available, compilation terminates. The amount of storage for compilation cannot be changed after processing has begun.

The size established at invocation of the compiler cannot be changed for later programs in the batch. Thus, after the first invocation, the compiler ignores a respecification of this option in any PROCESS statement. Specify SIZE(-n) to reserve sufficient storage space, where *n* is at least 55K bytes.

SOURCE

```
            ┌─YES─┐
►►──SOURCE=─┴─NO──┴────►◄
```

Specifies whether the compiler includes a listing of the source program in the compiler listing. The source program listed is either the original source input or, if MACRO=YES, the output from the preprocessor.

STMT

```
          ┌─YES─┐
►►──STMT=─┴─NO──┴────►◄
```

Specifies whether the compiler counts statements in the source program. The statement number produced by the compiler is used to identify statements in the compiler listings resulting from AGGREGA=YES, ATTRIBU=YES, LIST=YES, OFFSET=YES, SOURCE=YES, and XREF=YES. STMT=YES is required if NUMBER=NO or GOSTMT=YES. If STMT=NO, then NUMBER=YES and GOSTMT=NO are required.

STORAGE

```
             ┌─NO──┐
►►──STORAGE=─┴─YES─┴────►◄
```

Specifies whether the compiler listing includes a table that gives the storage requirements for the object module.

SYNTAX

```
            ┌─NOS─┐
►►──SYNTAX=─┼─YES─┼────►◄
            ├─NO──┤
            ├─NOW─┤
            └─NOE─┘
```

Specifies whether the compiler continues syntax checking after initialization (or after preprocessing if MACRO=YES) unless an unrecoverable error is detected. Continuation depends on the severity of errors detected so far, as follows:

NOS No syntax checking if a severe error or unrecoverable error is detected

YES Checks syntax unless an unrecoverable error occurs

NO No syntax checking after initialization or preprocessing

NOW No syntax checking if a warning, error, severe error, or unrecoverable error is detected

NOE No syntax checking if an error, severe error, or unrecoverable error is detected

If SOURCE=YES, the compiler generates a source listing even if syntax checking is not performed.

SYNTAX=NO terminates the compilation, and the system does not produce the cross-reference listing, attribute listing, and other listings that follow the source program.

The use of this option can prevent wasted runs when testing or debugging a PL/I program that uses the preprocessor.

SYSTEM

```
              ┌─VSE──┐
►►──SYSTEM=───┴─CICS─┴────►◄
```

Specifies the format used to pass parameters to the MAIN PL/I procedure and indicates the host system under which the program can run. The parameters are:

`VSE` Indicates that the PL/I MAIN procedure expects to receive a VSE parameter list and the object code can run under VSE.

`CICS` Indicates that the PL/I MAIN procedure expects to run under CICS as an application program. The PL/I MAIN procedure expects to receive 1 or 2 parameters that are pointers.

TEST

```
            ┌─NO──┐     ┌─NONE──┐    ┌─SYM───┐
►►──TEST=(──┴─YES─┴─,─┬─ALL───┼─,──┴─NOSYM─┴─)──►◄
                      ├─BLOCK─┤
                      ├─PATH──┤
                      └─STMT──┘
```

The TEST option specifies which debugging hooks will be inserted into the compiled object code.

Programmers should specify NOTEST to reduce the size of the compiled object code.

Five TEST suboptions tell the compiler where in the application program to insert debugging hooks:

`BLOCK` Insert hooks at block boundaries (block entry and block exit).

`STMT` Insert hooks at every statement boundary and at block boundaries. STMT generates a statement table.

`PATH` Insert hooks:

- Before the first statement enclosed by an iterative DO statement
- Before the first statement of the true part of an IF statement
- Before the first statement of the false part of an IF statement
- Before the first statement following a user label
- At every CALL or function reference—both before and after control is passed to the routine and after control returns

- At block boundaries
- At WHEN or OTHERWISE clauses that appear within SELECT statements

PATH generates a statement table.

`ALL` Insert hooks at all possible locations and generate a statement table.

`NONE` Do not insert hooks in the program.

The suboption SYM tells the compiler to create a symbol table.

For more information on the TEST option, and for some sample programs that make use of the debug hooks, refer to the *PL/I VSE Programming Guide.*

TSTAMP

```
                ┌─YES─┐
►►──TSTAMP=─────┴─NO──┴─────►◄
```

See “Compile-time options with restricted use.”

XREF

```
              ┌─NO──┐
►►──XREF=──┬──┴─YES─┴──────────────────────────┬─────►◄
           │     ┌─NO──┐                       │
           └─(───┴─YES─┴──┬─────────────────┬──)─┘
                          │       ┌─F─┐     │
                          └─,─────┴─S─┴─────┘
```

Specifies whether the compiler includes in the compiler listing a cross-reference table of names used in the program together with the numbers of the statements in cases where they are declared or referenced.

`F` Specifies the default suboption is full
`S` Specifies the suboption is short

For a description of the format and content of the cross-reference table, see the *PL/I VSE Programming Guide.*

Compile-time options with restricted use

Application programmers cannot use the compile-time options marked with an asterisk (*) in Table 12 on page 5. These options are used at the time of installation **only**. This section describes these restricted compile-time options.

FMARGIN

```
                 ┌─2─┐      ┌─72─┐
►►──FMARGIN=(────┴─m─┴──,───┴─n──┴───┬──────┬──)──►◄
                                     └─,c───┘
```

Specifies MARGINS(*m*,*n*,*c*) as the compile-time option default when reading the source program.

m The column number of the leftmost character to be processed by the compiler. The column number *m* must not exceed 80.

n The column number of the rightmost character to be processed by the compiler. The column number of *n* must be greater than *m*, but not greater than 80.

c The column number of the ANS printer control character. It must not exceed 80 and must be outside of the values specified for *m* and *n*.

FSEQUEN

```
                  ┌─73─┐    ┌─80─┐
►►──FSEQUEN=──┬─(─┴─m──┴─,──┴─n──┴─)─┬──────►◄
              └─NO───────────────────┘
```

Specifies defaults for the SEQUENCE compile-time option.

NO Specifies that NOSEQUENCE is the compile-time option default.

m Specifies that SEQUENCE(*m*,*n*) is the compile-time option default. *m* is the column number of the left-hand margin of the sequence field.

n Specifies that SEQUENCE(*m*,*n*) is the compile-time option default. *n* is the column number of the right-hand margin of the sequence field.

TSTAMP

```
              ┌─YES─┐
►►──TSTAMP=───┴─NO──┴─────►◄
```

Specifies whether the compiler places the compilation time and date in the STATIC INTERNAL CSECT, in a location indicated by the first word in the STATIC INTERNAL CSECT. This provides a time-stamping facility if you prefer not to use the compile-time preprocessor function in all your programs.

YES Specifies that the time and date of compilation be placed in the STATIC INTERNAL CSECT. This is the IBM-supplied default.

NO Specifies that the time and date of compilation is not to be placed in the STATIC INTERNAL CSECT.

DELETE

```
              ┌─,────┐
►►──DELETE=(──▼─item─┴─)──►◄
```

Specifies that the options in the list cannot be used at compile time in the *PROCESS statement or on the EXEC statement to override the default options established by the IEL1COPV macro instruction.

You can, however, use the CONTROL option to specify whether or not any options deleted can be used for this compilation. One or more of the following options can be specified. All are shown in Figure 5 on page 30.

AGGREGA
ATTRIBU
COMPILE
ESD
FLAG
FMARGIN
FSEQUEN
GONUMBE
GOSTMT
GRAPHIC
INCLUDE
INSOURC
LANGLVL
LINCOU
LIST
LMESSAG
MACRO
MAP
MARGINI
MDECK
NAME
NEST
NOT
NUMBER
OFFSET
OPTIMIZ
OPTIONS
OR
SIZE
SOURCE
STMT
STORAGE
SYNTAX
SYSTEM
TEST
XREF

For more detailed information on the compile-time options used for application programming, see *PL/I VSE Programming Guide.*

For more information on the options used only at the time of installation, see "Compile-time options with restricted use" on page 27.

To change the IBM-supplied default values of the compile-time options, modify the macro operands that control the defaults. The macro that governs compilation is IEL1COPV in the IEL1AV compiler module.

Allowing for compile-time option interdependencies

Some compile-time options, such as STMT, GOSTMT, and NUMBER, are interdependent at compile-time. If, for example, a programmer specifies GOSTMT at compile-time to override the IBM-supplied default of GOSTMT=NO, the compiler performs the compilation as if the programmer also specified STMT and NONUMBER, even though these are not defaults. The *PL/I VSE Programming Guide* fully describes these interdependencies.

The adjustment of one compile-time option based on the setting of another compile-time option does **not** occur at installation time. If you change the IBM-supplied default value for GOSTMT from GOSTMT=NO to GOSTMT=YES, you must make sure the defaults for NUMBER and STMT are compatible.

Customizing compile-time option defaults

Note: You can customize PL/I VSE only after installing the product. This section assumes that you have already installed PL/I VSE, as described in Chapter 2, "Installing PL/I VSE" on page 8.

If you decided during planning to change any of the PL/I VSE compile-time option defaults, you can do this now.

A sample job IEL1OPTV.Z is supplied to help you modify and assemble the IBM-supplied default assembler options. Figure 5 on page 30 shows this job.

The IEL1COPV macro in IEL1OPTV.Z contains the options that you can modify. "Compile-time option descriptions" on page 16 describes the possible values for these options. For more detailed information on these options see the PL/I VSE Programming Guide.

Once the changes have been made, submit the job for assembly on your VSE system. This JCL assembles and catalogs a new IEL1AV.OBJ module in the user sublibrary. Once this has been completed successfully, continue with the section on re-linking PL/I VSE phase IEL1AV. This completes the customization of the PL/I VSE options.

```
// JOB     IEL1OPTV
*
*  Assemble PL/I VSE default options
*
* ---------------------------------------------------------------
// SETPARM VOLUME='VSE111'         * Volume to use
// SETPARM START='????'            * Starting extent trk/blk  1
// SETPARM LENGTH='0020'           * Length of work file
* ---------------------------------------------------------------
*
// LIBDEF  *,SEARCH=(PRD2.PROD)                    2
// DLBL    IJSYSPH,'ASSEMBLE.OUTPUT',0
// EXTENT  SYSPCH,&VOLUME,1,0,&START,&LENGTH
ASSGN      SYSPCH,DISK,VOL=&VOLUME,SHR
// OPTION  DECK
// EXEC    ASMA90
         PUNCH 'CATALOG IEL1AV.OBJ REPLACE=YES'
         PRINT ON,GEN
         IEL1COPV AGGREGA=NO,                                          X
               ATTRIBU=NO,                                             X
               COMPILE=NOS,                                            X
               CONTROL='OPTIMIZE',                                     X
               DELETE=,                                                X
               ESD=NO,                                                 X
               FLAG=I,                                                 X
               FMARGIN=(2,72),                                         X
               FSEQUEN=(73,80),                                        X
               GONUMBE=NO,                                             X
               GOSTMT=NO,                                              X
               GRAPHIC=NO,                                             X
               INCLUDE=NO,                                             X
               INSOURC=YES,                                            X
               LANGLVL=(OS,NOSPROG),                                   X
               LINECOU=55,                                             X
               LIST=NO,                                                X
               LMESSAG=YES,                                            X
               MACRO=NO,                                               X
               MAP=NO,                                                 X
               MARGINI=NO,                                             X
               MDECK=NO,                                               X
               NEST=NO,                                                X
               NOT=,                                                   X
               NUMBER=NO,                                              X
               OFFSET=NO,                                              X
               OPTIMIZ=NO,                                             X
               OPTIONS=YES,                                            X
               OR=,                                                    X
               SIZE=MAX,                                               X
               SOURCE=YES,                                             X
               STMT=YES,                                               X
               STORAGE=NO,                                             X
               SYNTAX=NOS,                                             X
               SYSTEM=VSE,                                             X
               TEST=(NO,NONE,SYM),                                     X
               TSTAMP=YES,                                             X
               XREF=(NO,F)
         END
```

Figure 5 (Part 1 of 2). Sample JCL to assemble the default options

```
/*
CLOSE      SYSPCH,PUNCH
// DLBL    IJSYSIN,'ASSEMBLE.OUTPUT',0
// EXTENT  SYSIPT
ASSGN      SYSIPT,DISK,VOL=&VOLUME,SHR
// EXEC    LIBR,PARM='MSHP;ACCESS SUBLIB=LIB.USERLIB'     3
/*
CLOSE      SYSIPT,SYSRDR
/*
/&
```

Figure 5 (Part 2 of 2). Sample JCL to assemble the default options

Note: In Figure 5 on page 30, all the option lines — except for the last option (in the figure, XREF=(NO,F)) — must end with a comma (for example, ESD=NO,) and have a non-blank continuation character in column 72.

In area 1 change the SETPARM values to those applicable to your installation.

In area 2 if PL/I VSE has been installed in a different sublibrary from the default change this to reflect your sublibrary.

In area 3 change the sublibrary to reflect a user sublibrary where changes to the PL/I VSE default options will be catalogued. This should be in a user library to ensure that the original IBM-supplied defaults are not overwritten.

Re-link PL/I VSE phase IEL1AV

The PL/I VSE phase IEL1AV needs to be re-linked to successfully change the default options. A sample JCL member IEL1OPTL.Z is provided which will re-link the PL/I VSE phase IEL1AV.

```
// JOB    IEL1OPTL
*
*  Linkedit PL/I VSE default options phase IEL1AV
*
// LIBDEF *,SEARCH=(LIB.USERLIB,PRD2.PROD)  1
// LIBDEF PHASE,CATALOG=LIB.USERLIB         2
// OPTION CATAL
 INCLUDE IEL$AV                             3
// EXEC   LNKEDT,PARM='MSHP'
/*
/&
```

Figure 6. Sample JCL to linkedit the default options phase

In area 1 the library search chain points first to the sublibrary where IEL1AV.OBJ was cataloged when you assembled the changed options. It then points to the sublibrary where PL/I VSE was installed, and where the link book IEL$AV.OBJ can be found.

In area 2 change the library and sublibrary to reflect the user sublibrary where IEL1AV.PHASE is to be located.

In area 3 a link book is used to supply the phase statement and include values.

Chapter 4. Maintaining PL/I VSE

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

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

Re-installing PL/I VSE

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

You do not need to remove PL/I VSE from your system before re-installing PL/I VSE unless you intend to re-install the product in a different sublibrary from the previous installation. In this case you must remove PL/I VSE from the system history file before you can re-install it. Figure 10 on page 36 shows a job to remove PL/I VSE from the system history file.

To re-install PL/I VSE you follow the same steps as for installing PL/I VSE. See Chapter 2, "Installing PL/I VSE" on page 8.

Applying service updates

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

What you receive

If you report a problem with PL/I 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 PL/I VSE or any other licensed product you have installed, including VSE/ESA.

You apply service to PL/I VSE 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 PL/I VSE.

Checklist for applying service

Table 14 on page 33 lists the steps for installing corrective service on PL/I VSE. You can use Table 14 as a checklist.

© Copyright IBM Corp. 1993, 1995

Table 14. Summary of steps for installing service on PL/I VSE

Step	Description	MSHP Command or Jobname	Page
__ 1	Ensure prerequisite APARs or PTFs are applied.	RETRACE	33
__ 2	Backup existing system	___	33
__ 3	Apply service	INSTALL	33
__ 4	Run the installation verification program	IEL1EIVP	35

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 PL/I VSE or any licensed program you have installed at your installation.

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 7 shows how to retrace APARs and PTFs in the system history file.

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

Figure 7. Job to retrace APARs and PTFs

Use this listing 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.

Step 2. Backup original system

Make a backup copy of your current PL/I VSE library and the system history file. For information about backing up libraries and the system history file, see *VSE/ESA System Control Statements.*

Step 3. Apply service

You can apply service to PL/I VSE from the provided service tape using either the Interactive Interface or a batch job.

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

Method 1: Apply Service Using the Interactive Interface

To apply service to PL/I VSE using the Interactive Interface, logon to the VSE/ESA Interactive Interface as the system administrator. (If you would like information about the functions of the Interactive Interface, refer to *VSE/ESA Administration.*)

Mount the 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)

   a. **IBM SERVICE** menu:
      ===> ***2*** (PTF Handling)

   b. **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***
        (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***
        (the address of the tape drive where you mounted the service tape)

   c. **JOB DISPOSITION** menu:

      Make any changes required and press ENTER to submit the job and apply the service.

Method 2: Apply Service Using a Batch Job

The batch job to apply service to PL/I VSE uses the MSHP system history file where PL/I VSE was installed.

A sample job to apply service using the Maintain System History Program (MSHP) is shown in Figure 8 on page 35. For more information on MSHP see *VSE/ESA System Control Statements.*

```
// JOB   MSHPSERV
*
*  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 PL/I VSE is installed. You do not need to supply this information.

Step 4. Run the installation verification program (IVP)

After you have applied all the files on the service tape, run the installation verification program IEL1EIVP to ensure that PL/I VSE functions properly. See the description of IEL1EIVP in Figure 4 on page 14.

Removing PL/I VSE

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

If you do have to remove PL/I VSE from your system for any reason, you must delete all the PL/I VSE entries from your sublibrary and remove PL/I VSE from the system history file. Figure 10 on page 36 shows the JCL needed to remove PL/I VSE from the system history file.

To delete all PL/I VSE entries from your sublibrary, use the DELETE command of the LIBR program. The job shown in Figure 9 shows the JCL needed to delete PL/I VSE from the default sublibrary PRD2.PROD.

```
// JOB   LIBRDEL
*
*  Delete PL/I VSE
*
* Label for the PL/I VSE library         1
// EXEC  LIBR,SIZE=200K
ACCESS S=PRD2.PROD                       2
DELETE IEL1*.*
DELETE IEL$*.*
DELETE $$$C018P.Z
/*
/&
```

Figure 9. Job to delete PL/I VSE from a sublibrary

If you have installed PL/I VSE into a sublibrary other than the default (PRD2.PROD) then insert the required DLBL, EXTENT and ASSGN information for the PL/I VSE library in area 1 .

If you have installed PL/I VSE in a sublibrary other than the default, change the statement in area 2 .

To remove PL/I VSE from the system history file, use the REMOVE command of the Maintain System History Program (MSHP). The sample job shown in Figure 10 shows the JCL needed to remove PL/I VSE from the system history file.

```
// JOB   MSHPREM
*
*  Remove the PL/I VSE product
*
// EXEC  MSHP,SIZE=900K
REMOVE 5686-069-00-18P                          1
/*
/&
```

Figure 10. Job to remove PL/I VSE from the system history file

Area 1 shows the component ID for PL/I VSE.

How to report a problem with PL/I VSE

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

Bibliography

IBM PL/I for VSE/ESA publications

Fact Sheet, GC26-8052

Installation and Customization Guide, SC26-8057

Licensed Program Specifications, GC26-8055

Language Reference, SC26-8054

Compile-Time Messages and Codes, SC26-8059

Diagnosis Guide, SC26-8058

Migration Guide, SC26-8056

Programming Guide, SC26-8053

Reference Summary, SX26-3836

IBM Language Environment for VSE/ESA publications

Concepts Guide, GC26-8063

Fact Sheet, GC26-8062

Debugging Guide and Run-Time Messages, SC26-8066

Diagnosis Guide, SC26-8060

Installation and Customization Guide, SC26-8064

Licensed Program Specifications, GC26-8061

Programming Guide, SC26-8065

Reference Summary, SX26-3835

VSE/ESA publications

VSE/ESA Version 1

Administration, SC33-6505

Messages and Codes, SC33-6507

System Control Statements, SC33-6513

System Utilities, SC33-6517

System Macros Reference, SC33-6516

Guide to System Functions, SC33-6511

VSE/VSAM Commands and Macros, SC33-6532

VSE/VSAM User's Guide, SC33-6535

VSE/ESA Version 2

Administration, SC33-6605

Messages and Codes, SC33-6607

System Control Statements, SC33-6613

System Utilities, SC33-6617

System Macros Reference, SC33-6616

Guide to System Functions, SC33-6611

VSE/VSAM Commands and Macros, SC33-6631

VSE/VSAM User's Guide, SC33-6632

Related publications

CICS/VSE

Application Programming Guide, SC33-0712

Application Programming Reference, SC33-0713

System Definition and Operations Guide, SC33-0706

Resource Definition, SC33-0708

DFSORT for VSE/ESA

Application Programming Guide, SC26-7040

Sort/Merge II

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

DL/I DOS/VS

Application Programming: High Level Programming Interface, SH24-5009

Application Programming: CALL and RQDLI Interfaces, SH12-5411

SQL/DS

SQL/Data System Application Programming Guide for VSE, SH09-8098

Softcopy publications

These collections contain the LE/VSE and LE/VSE-conforming language product publications:

VSE Collection, SK2T-0060

Application Development Collection, SK2T-1237

© Copyright IBM Corp. 1993, 1995

Index

© Copyright IBM Corp. 1993, 1995

We'd Like to Hear from You

IBM PL/I for VSE/ESA
Installation and Customization Guide
Release 1

Publication No. SC26-8057-00

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

- Mail—Use the Readers' Comments form on the next page. 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—Use the Readers' Comments form on the next page and fax it to this U.S. number: 800-426-7773.
- Electronic mail—Use one of the following network IDs:
  - Internet: COMMENTS@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.

Readers' Comments

IBM PL/I for VSE/ESA
Installation and Customization Guide
Release 1

Publication No. SC26-8057-00

How satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Technically accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□
Grammatically correct and consistent	□	□	□	□	□
Graphically well designed	□	□	□	□	□
Overall satisfaction	□	□	□	□	□

Please tell us how we can improve this book:

May we contact you to discuss your comments? □ Yes □ No

Name

Address

Company or Organization

Phone No.

Readers' Comments
SC26-8057-00

IBM®

Fold and Tape

Please do not staple

Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

Department J58
International Business Machines Corporation
PO BOX 49023
SAN JOSE CA 95161-9945

Fold and Tape

Please do not staple

Fold and Tape

IBM®

Program Number: 5686-069

IBM PL/I for VSE/ESA Publications

GC26-8052	Fact Sheet
GC26-8055	Licensed Program Specifications
SC26-8056	Migration Guide
SC26-8057	Installation and Customization Guide
SC26-8053	Programming Guide
SC26-8054	Language Reference
SX26-3836	Reference Summary
SC26-8058	Diagnosis Guide
SC26-8059	Compile-Time Messages and Codes

SC26-8057-00