Advanced Function Presentation

Application Programming Interface: PL/1 Language Reference

S544-3874-01

Advanced Function Presentation

Application Programming Interface: PL/1 Language Reference

S544-3874-01

Note!

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

Second Edition (February 1996)

This edition applies to Print Services Facility/MVS Version 2 Release 2 Modification 0, Print Services Facility/VM Version 2 Release 1 Modification 1, and to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters. See the Summary of Changes for the changes made to this publication. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change. Be sure to use 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 given below.

The IBM Printing Systems Company welcomes your comments. For your convenience, a form for reader's comments is provided at the back of this publication. You may either send your comments by fax to 1-800-524-1519, or mail to:

INFORMATION DEVELOPMENT
THE IBM PRINTING SYSTEMS COMPANY
DEPARTMENT H7FE BUILDING 003G
PO BOX 1900
BOULDER CO 80301-9191

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.

Contents

Notices v
Programming Interface v
Trademarks v

Summary of Changes vii

Chapter 1. Introduction 1
Copy Book Files Shipped With the Product 1
Other Files Shipped With the Product 2

Chapter 2. PL/1 Language Bindings 3
AFPBDOC (Begin Document) 5
AFPBFLD (Begin Field) 6
AFPBGRP (Begin Group) 7
AFPBPAG (Begin Page) 8
AFPBPAR (Begin Paragraph) 9
AFPBROW (Begin Row) 11
AFPBTBL (Begin Table) 12
AFPCARE (Create Area) 14
AFPDFLD (Define Field) 15
AFPDFNT (Define Font by Attributes) 17
AFPDROW (Define Row) 19
AFPEARE (End Area) 21
AFPEDOC (End Document) 22
AFPEFLD (End Field) 23
AFPEGRP (End Group) 24
AFPEND (End AFPAPI) 25
AFPEPAG (End Page) 26
AFPEPAR (End Paragraph) 27
AFPEROW (End Row) 28
AFPETBL (End Table) 29
AFPGBUF (Get Output Buffer) 30
AFPINIT (Initialize AFPAPI) 31
AFPINVM (Invoke Medium Map) 32
AFPIOBJ (Include Object) 33
AFPIOVL (Include Page Overlay) 35
AFPIPSG (Include Page Segment) 36
AFPPARE (Put Area) 37
AFPPBOX (Put Box) 38
AFPPCHS (Put Character String) 39
AFPPRUL (Put Rule) 40
AFPPTAG (Put Tag) 41
AFPPTXT (Put Text) 42
AFPQATT (Query Current Attributes) 43
AFPQPOS (Query Current Position) 44
AFPQSTR (Query Character String Size) 45
AFPSCLR (Set Color) 46
AFPSFNT (Set Font) 47
AFPSICS (Set Intercharacter Spacing) 48
AFPSLIB (Set Resource Library Names) 49
AFPSOUT (Set Output Characteristics) 50

AFPSPOS (Set Position) 51
AFPSRTH (Set Rule Thickness) 52
AFPSUNI (Set Units) 53
AFPSWSP (Set Word Spacing) 54
AFPTERM (Terminate AFPAPI) 55
AFPXARE (Destroy Area) 56

Chapter 3. PL/1 Sample Code 57
Sample Document 58
APQPSAMP 59
APQPSMP2 68

Appendix A. PL/1 Macros Used as Programming Interfaces 85
APQPCON 86
APQPRCS 88
APQPVAR 90
APQPPRF 92

Appendix B. Related Publications 101

Notices

References in this publication to products or services of IBM do not suggest or imply that IBM will make them available in all countries where IBM does business or that only products or services of IBM may be used. Noninfringing equivalents may be substituted, but the user must verify that such substitutes, unless expressly designated by IBM, work correctly. No license, expressed or implied, to patents or copyrights of IBM is granted by furnishing this document.

Programming Interface

This manual is intended to help the customer program AFP applications with the PL/1 programming language. This manual documents General-Use Programming Interface and Associated Guidance Information provided by AFP Application Programming Interface.

General-Use programming interfaces allow the customer to write programs that obtain the services of AFP Application Programming Interface.

Trademarks

The following terms appear in this publication and are trademarks or registered trademarks of the IBM Corporation:

Advanced Function Presentation
AFP
BookManager
IBM
Print Services Facility
PSF

Summary of Changes

In addition to editorial changes, the following changes are included in this edition of the publication, S544-3874-01:

- Changes documented in the *Print Services Facility/MVS: Update Guide*, G544-3984-00, are incorporated.
- Support for output buffering is added. Changes include:
  - The Set Output Characteristics procedure call allows you to request output buffering in the Output Filename parameter. You can also request that AFP API discard the output.
  - The new Get Output Buffer procedure call returns the AFP buffered output to your application program.
  - A new perform, AFPGBUF, is included in APQPPRF.
  - New return codes 280, 281, and 282 are defined in APQPRCS.
  - New variables are defined in APQPVAR.
  - New constants are defined in APQPCON.
- Support for querying the size of a character string is added. Changes include:
  - The new Query Character String Size procedure call returns the size of the area required to print a character string in the current font.
  - A new perform, AFPQSTR, is included in APQPPRF.
  - New return codes 284 and 285 are defined in APQPRCS.
  - New variables are defined in APQPVAR.
- You can now issue the Define Row and Define Field procedure calls *only* in the document state. You can no longer issue these calls in the page or area state.
- You can issue the Put Area procedure call *only* in the page state, not in the area state.
- You must set the Concatenate parameter to TRU on the first Put Text procedure call in a field or paragraph.
- The trace function is removed from the Initialize AFP procedure call.
- The minimum valid value for several measurement parameters is corrected; the minimum value is .0002 millimeters instead of .0001 millimeters.
- The sample program APQPSAMP is included.
- An appendix listing related publications is included.

Technical changes made to the text in this edition are indicated by a vertical line to the left of the changes.

Chapter 1. Introduction

Please Read

If you are not familiar with IBM's Advanced Function Presentation or the AFP API, refer to *AFP API: Programming Guide and Reference* before using this book. You must understand AFP concepts and the principles of AFP API before you can understand and use these language bindings.

This publication contains the following:

- PL/1 language bindings for the AFP Application Programming Interface.
- The copy books shipped with the product.
- The PL/1 source code for the sample document shipped with the product.

Copy Book Files Shipped With the Product

Several copy book files are shipped with the product to aid you in developing your programs. They are:

Copy Book	***Description***
APQPCON	Contains constants that AFP API uses.
APQPRCS	Contains return codes that AFP API generates.
APQPVAR	Contains variables that AFP API uses.
APQPPRF	Contains subroutines that invoke the AFP API procedures. Upon return from an AFP API procedure, the program examines the severity code. If the code is SEVERE or FATAL, the program does the following: • At SYSOUT, displays the name of the AFP API procedure call in error and its associated return code and severity code. • Issues AFPTERM to terminate AFP API. This prints any partial page and generates a STOP statement that terminates the PL/1 program. **Note:** APQPPRF ignores WARNING severity codes. The user program must check for these codes.

Other Files Shipped With the Product

In addition to the copy files described earlier, the following AFP API files are shipped with the product and are used in the sample described in *AFP API: Programming Guide and Reference*.

Note: Because of rounding, account totals in the sample output produced by PL/1 may be slightly different than the totals shown in the sample output printed in AFP API publications.

APQPSAMP PL/1 source calls to APQPPRF subroutines which in turn invoke AFP API procedures

APQPSMP2 PL/1 source calls which invoke AFP API procedures

APQDATA Data file for APQPSAMP and APQPSMP2

APQCOPLI JCL to compile and link APQPSAMP

APQIVPLI JCL to run the APQPSAMP

APQPSEG Page segment (PRIMO artwork)

O1APQL2 Overlay (shaded summary box)

IOCAMMR Image object used to describe AFPIOBJ (Include Object). IOCAMMR is not included in the sample output. It is distributed with PSF but is not distributed on the AFP API tape.

Chapter 2. PL/1 Language Bindings

This chapter contains information on the PL/1 syntax rules, constants, and definitions for each of the AFP API procedure calls. Parameters for each call must be in the order given and all are required. If any are missing or out of order, an addressing exception will probably occur. All procedure calls must end with a semicolon (;). The generic data types are mapped to PL/I data types, with valid value ranges given (unless specified otherwise in the tables) as follows:

HANDLE FIXED BINARY (31)

TOKEN CHAR (8)

FILENAME_TOKEN CHAR (8)

FILETYPE_TOKEN CHAR (8)

FILEMODE_TOKEN CHAR (2)

BOOLEAN FIXED BINARY (31)

STRING CHAR (maximum string length)

CHARACTER CHAR (1)

REAL FIXED BINARY (31)

The valid values for parameters of type FIXED BINARY (31) are:

- 0.0 to 5918.0 when using millimeters
- 0.0 to 591.0 when using centimeters
- 0.0 to 233.0 when using inches
- 0.0 to 55924.0 when using 240 units per inch
- 0.0 to 335544.0 when using 1440 units per inch

Notes:

1. AFP API interprets a FIXED BINARY (31) as a REAL number with four implied decimal positions. Therefore, multiply all REAL parameters by a factor of 10,000. Once converted by AFP API, the parameters contain the proper four decimal places.

   Example:

   ```
   DCL AFP_DOC_PAGE_WIDTH FIXED BINARY(31);

            AFP_DOC_PAGE_WIDTH = 215.0 * 10000;
   ```

2. Multiply the value being set first, or the decimal portion is truncated. For example, the following code does not yield the desired results, because the PL/1 compiler truncates the value to 215 prior to multiplication:

   ```
   AFP_DOC_PAGE_WIDTH = 215.95;
   AFP_DOC_PAGE_WIDTH = AFP_DOC_PAGE_WIDTH * 10000;
   ```

3. The maximum value for a FIXED BINARY (31) is 99999.9999.

4. In PL/1, many decimal fractions do not have an exact representation in binary. This could affect your results if you do not allow for it when using small decimal fractions.

INTEGER4 FIXED BINARY (31)

SGLARRAY A one-dimensional array of FIXED BINARY (31) whose size varies depending upon input parameters.

Note: AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000.

MULTARRAY A two-dimensional array of FIXED BINARY (31) whose size varies depending upon input parameters.

Column headings in the tables mean the following:

Parameters AFP API input and output procedure calls. APQPVAR contains the data declarations for this column, and copy file APQPPRF uses them. You can invoke these subroutines in your program, so that you don't need to write the code for these calls yourself.

Data Attribute General characteristics and editing requirements of an elementary item. See APQPVAR for the initial values assigned to the data declarations.

Description Additional information about the procedure calls. APQPCON contains the constants for the items in bold in this column.

Notes:

1. For parameters that have an identifier as a valid value (for example, ORIENT0 in AFPBDOC), you can use the numeric literal for that identifier (for example, 0) instead of the identifier. Or, you can use another identifier whose numeric literal is the same (for example, ROTATE0 in the APQPCON copy book also has a numeric literal of 0).
2. Whenever an AFP API procedure call requires the *Current* handle, the application program must copy the value returned by the appropriate AFPINIT, AFPBDOC, AFPBPAGE, AFPBPAR, or AFPBTBL procedure calls to the AFP-CURRENT-HANDLE parameter. Refer to *AFP API: Programming Guide and Reference* for a description of current handle.

AFPBDOC (Begin Document)

```
DCL AFPBDOC EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPBDOC (AFPAPI_HANDLE,
              AFP_UNIT_OF_MEASURE,
              AFP_DOC_PAGE_WIDTH,
              AFP_DOC_PAGE_DEPTH,
              AFP_PAGE_ORIENTATION,
              AFP_DOCUMENT_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_UNIT_OF_ MEASURE	FIXED BINARY (31)	The unit of measure: **INCH** Inches **MM** Millimeters **CM** Centimeters **U240** 240 units per inch **U1440** 1440 units per inch **Note:** The output file generated by AFP API is in logical units of 1440 per inch, even if you specify a different unit of measure in this parameter.
AFP_DOC_PAGE_ WIDTH	FIXED BINARY (31)	The width of the logical page. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_DOC_PAGE_ DEPTH	FIXED BINARY (31)	The depth of the logical page. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_PAGE_ ORIENTATION	FIXED BINARY (31)	The orientation of the logical page: **ORIENT0** 0 degree orientation **ORIENT90** 90 degree orientation **ORIENT180** 180 degree orientation **ORIENT270** 270 degree orientation

Output Parameters	Data Attribute	Description
AFP_ DOCUMENT_ HANDLE	FIXED BINARY (31)	Returned by AFPBDOC and required on various subsequent procedure calls such as AFPBPAG.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPBFLD (Begin Field)

```
DCL AFPBFLD EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPBFLD (AFPAPI_HANDLE,
              AFP_TABLE_HANDLE,
              AFP_FIELD_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_TABLE_ HANDLE	FIXED BINARY (31)	Table handle (returned by AFPBTBL).
AFP_FIELD_ID	FIXED BINARY (31)	ID for the field (returned by AFPDFLD).

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPBGRP (Begin Group)

```
DCL AFPBGRP EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPBGRP (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_GROUP_NAME,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_DOCUMENT_ HANDLE	FIXED BINARY (31)	Document state handle (returned by AFPBDOC).
AFP_GROUP_ NAME	CHAR (64)	A 64-byte character string that's padded on the right with blanks if the actual group name is less than 64 characters.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPBPAG (Begin Page)

```
DCL AFPBPAG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPBPAG (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_PAGE_WIDTH,
              AFP_PAGE_DEPTH,
              AFP_PAGE_ORIENTATION,
              AFP_PAGE_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_DOCUMENT_ HANDLE	FIXED BINARY (31)	Document handle (returned by AFPBDOC).
AFP_PAGE_ WIDTH	FIXED BINARY (31)	The width of the logical page. **AFP_DEFAULT** Use the page width specified on AFPBDOC Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_PAGE_ DEPTH	FIXED BINARY (31)	The depth of the logical page. **AFP_DEFAULT** Use the page depth specified on AFPBDOC Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_PAGE_ ORIENTATION	FIXED BINARY (31)	The orientation of the logical page: **ORIENTDOC** Use the orientation specified on AFPBDOC. **ORIENT0** 0 degree orientation **ORIENT90** 90 degree orientation **ORIENT180** 180 degree orientation **ORIENT270** 270 degree orientation

Output Parameters	Data Attribute	Description
AFP_PAGE_ HANDLE	FIXED BINARY (31)	Returned by AFPBPAG and required on various subsequent procedure calls such as AFPPCHS.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPBPAR (Begin Paragraph)

```
DCL AFPBPAR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPBPAR (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_FIRST_LINE_INDENT,
              AFP_FORMAT_OPTION,
              AFP_FIRST_LINE_OFFSET,
              AFP_LEFT_MARGIN,
              AFP_LINE_LENGTH,
              AFP_LINE_SPACING,
              AFP_PARAGRAPH_FRAME,
              AFP_RT_RULE_OFFSET,
              AFP_BOT_RULE_OFFSET,
              AFP_SHADING_PATTERN,
              AFP_SHADING_INTENSITY,
              AFP_PARAGRAPH_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).
AFP_FIRST_ LINE_ INDENT	FIXED BINARY (31)	The amount to indent text. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_FORMAT_ OPTION	FIXED BINARY (31)	The type of formatting: **FOLEFT** Left aligned paragraph (ragged right) **FOCENTER** Centered paragraph **FORIGHT** Right aligned paragraph (ragged left) **FOJUSTIFY** Justified paragraph
AFP_FIRST_ LINE_ OFFSET	FIXED BINARY (31)	The offset of the first line of text. **AFP_DEFAULT** Use the default for the first line offset. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_LEFT_ MARGIN	FIXED BINARY (31)	The offset of the left margin from the paragraph origin. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_LINE_LENGTH	FIXED BINARY (31)	The length of a line. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Input Parameters	Data Attribute	Description
AFP_LINE_ SPACING	FIXED BINARY (31)	The spacing between lines. **AFP_DEFAULT** Use the default linespace. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_ PARAGRAPH_ FRAME	FIXED BINARY (31)	**TRU** Enclose the paragraph in a frame **FALS** Don't enclose the paragraph in a frame
AFP_RT_RULE_ OFFSET	FIXED BINARY (31)	The offset of the right vertical rule from the paragraph origin. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_BOT_RULE_ OFFSET	FIXED BINARY (31)	The offset of the bottom horizontal rule from the last line in the paragraph. **AFP_DEFAULT** Use the default bottom rule offset. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_SHADING_ PATTERN	FIXED BINARY (31)	The shading pattern: **STNDARD** Use the standard shading pattern **SCREEN** Use the screen shading pattern **NOSHADE** Don't use a shading pattern
AFP_SHADING_ INTENSITY	FIXED BINARY (31)	The shading intensity: 0-100.

Output Parameters	Data Attribute	Description
AFP_ PARAGRAPH_ HANDLE	FIXED BINARY (31)	Returned by AFPBPAR and required on various subsequent procedure calls such as AFPPTXT.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPBROW (Begin Row)

```
DCL AFPBROW EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPBROW (AFPAPI_HANDLE,
              AFP_TABLE_HANDLE,
              AFP_ROW_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_TABLE_ HANDLE	FIXED BINARY (31)	Table handle (returned by AFPBTBL).
AFP_ROW_ID	CHAR (9)	ID for the row. (Returned by AFPDROW.)

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPBTBL (Begin Table)

```
DCL AFPBTBL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPBTBL (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_TABLE_WIDTH,
              AFP_MAX_TABLE_DEPTH,
              AFP_TABLE_ROTATION,
              AFP_TOP_THICKNESS,
              AFP_BOTTOM_THICKNESS,
              AFP_LEFT_THICKNESS,
              AFP_RIGHT_THICKNESS,
              AFP_TABLE_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).
AFP_TABLE_ WIDTH	FIXED BINARY (31)	The width of the table. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_MAX_ TABLE_ DEPTH	FIXED BINARY (31)	The depth of the table. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_TABLE_ ROTATION	FIXED BINARY (31)	The rotation of the table: **ROTATE0** 0 degree rotation **ROTATE90** Rotate the table 90 degrees **ROTATE180** Rotate the table 180 degrees **ROTATE270** Rotate the table 270 degrees
AFP_TOP_ THICKNESS	FIXED BINARY (31)	The thickness of the top horizontal rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_BOTTOM_ THICKNESS	FIXED BINARY (31)	The thickness of the bottom horizontal rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_LEFT_ THICKNESS	FIXED BINARY (31)	The thickness of the left vertical rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_RIGHT_ THICKNESS	FIXED BINARY (31)	The thickness of the right vertical rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Output Parameters	Data Attribute	Description
AFP_TABLE_ HANDLE	FIXED BINARY (31)	Returned by AFPBTBL and required on various subsequent procedure calls such as AFPBROW.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPCARE (Create Area)

```
DCL AFPCARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPCARE (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_AREA_WIDTH,
              AFP_MAX_AREA_DEPTH,
              AFP_AREA_FRAME,
              AFP_SHADING_PATTERN,
              AFP_SHADING_INTENSITY,
              AFP_AREA_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC or AFPBPAG).
AFP_AREA_ WIDTH	FIXED BINARY (31)	The width of the area. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_MAX_AREA_ DEPTH	FIXED BINARY (31)	The depth of the area. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_AREA_ FRAME	FIXED BINARY (31)	**TRU** Enclose the area in a frame. **FALS** Don't enclose the area in a frame.
AFP_SHADING_ PATTERN	FIXED BINARY (31)	The shading pattern: **STNDARD** Use the standard shading pattern **SCREEN** Use the screen shading pattern **NOSHADE** Don't use a shading pattern
AFP_SHADING_ INTENSITY	FIXED BINARY (31)	The shading intensity: 0-100.

Output Parameters	Data Attribute	Description
AFP_AREA_ HANDLE	FIXED BINARY (31)	Returned by AFPCARE and required on various subsequent procedure calls such as AFPPCHS.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPDFLD (Define Field)

```
DCL AFPDFLD EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPDFLD (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_FORMAT_OPTION,
              AFP_ALIGNMENT_POSITION,
              AFP_VERTICAL_FORMAT,
              AFP_LEFT_MARGIN,
              AFP_RIGHT_MARGIN,
              AFP_LINE_SPACING,
              AFP_TEXT_ORIENTATION,
              AFP_SHADING_PATTERN,
              AFP_SHADING_INTENSITY,
              AFP_TOP_THICKNESS,
              AFP_BOTTOM_THICKNESS,
              AFP_LEFT_THICKNESS,
              AFP_RIGHT_THICKNESS,
              AFP_FIELD_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_DOCUMENT_ HANDLE	FIXED BINARY (31)	Document handle (returned by AFPBDOC).
AFP_FORMAT_ OPTION	FIXED BINARY (31)	The type of formatting: **FOLEFT** Left aligned text (ragged right) **FOCENTER** Centered text **FORIGHT** Right aligned text (ragged left) **FOJUSTIFY** Justified text
AFP_ ALIGNMENT_ POSITION	FIXED BINARY (31)	The position for character alignment. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_VERTICAL_ FORMAT	FIXED BINARY (31)	The vertical alignment option: **VERTOP** Align the text at the top of the field **VERCENTER** Align the text in the center of the field **VERBOTTOM** Align the text at the bottom of the field
AFP_LEFT_ MARGIN	FIXED BINARY (31)	The left margin for the lines of text. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_RIGHT_ MARGIN	FIXED BINARY (31)	The right margin for the lines of text. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_LINE_ SPACING	FIXED BINARY (31)	The spacing between lines of text. **AFP_DEFAULT** Use the default linespace. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_TEXT_ ORIENTATION	FIXED BINARY (31)	The orientation of the text: **TXTOR0_0** 0, 0 text orientation **TXTOR90_180** 90, 180 text orientation **TXTOR180_270** 180, 270 text orientation **TXTOR270_0** 270, 0 text orientation

Input Parameters	Data Attribute	Description
AFP_SHADING_ PATTERN	FIXED BINARY (31)	The shading pattern: **STNDARD** Use the standard shading pattern **SCREEN** Use the screen shading pattern **NOSHADE** Don't use a shading pattern
AFP_SHADING_ INTENSITY	FIXED BINARY (31)	The shading intensity: 0-100.
AFP_TOP_ THICKNESS	FIXED BINARY (31)	The thickness of the top horizontal rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_BOTTOM_ THICKNESS	FIXED BINARY (31)	The thickness of the bottom horizontal rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_LEFT_ THICKNESS	FIXED BINARY (31)	The thickness of the left vertical rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_RIGHT_ THICKNESS	FIXED BINARY (31)	The thickness of the right vertical rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Output Parameters	Data Attribute	Description
AFP_FIELD_ID	FIXED BINARY (31)	Returned by AFPDFLD and required on subsequent AFPBFLD procedure calls.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPDFNT (Define Font by Attributes)

```
DCL AFPDFNT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPDFNT (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_CODE_PAGE,
              AFP_DESC_NAME_LENGTH,
              AFP_DESCRIPTIVE_NAME,
              AFP_POINT_SIZE,
              AFP_WEIGHT,
              AFP_FONT_WIDTH,
              AFP_ROTATION,
              AFP_STYLE,
              AFP_FONT_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG or AFPCARE).
AFP_CODE_PAGE	CHAR (8)	An 8-byte character string that's left-aligned and padded on the right with blanks if the actual code page name is less than 8.
AFP_DESC_ NAME_LENGTH	FIXED BINARY (31)	Length of descriptive name: 1–32
AFP_ DESCRIPTIVE_ NAME	CHAR (32)	A 32-byte character string that's left-aligned and padded on the right with blanks if the actual descriptive name is less than 32.
AFP_POINT_ SIZE	FIXED BINARY (31)	The height of the character.
AFP_WEIGHT	FIXED BINARY (31)	The weight of the font: **ULTRALIGHT** UltraLight **EXTRALIGHT** ExtraLight **LIGHT** Light **SEMILIGHT** SemiLight **MEDIUM** Medium **SEMIBOLD** SemiBold **BOLD** Bold **EXTRABOLD** ExtraBold **ULTRABOLD** UltraBold
AFP_FONT_ WIDTH	FIXED BINARY (31)	The thickness of the font: **ULTRACOND** UltraCondensed **EXTRACOND** ExtraCondensed **CONDENSED** Condensed **SEMICOND** SemiCondensed **NORMAL** Normal **SEMIEXP** SemiExpanded **EXPANDED** Expanded **EXTRAEXP** ExtraExpanded **ULTRAEXP** UltraExpanded
AFP_ROTATION	FIXED BINARY (31)	The character rotation: **ROTATE0** 0 degree character rotation **ROTATE90** 90 degree character rotation **ROTATE180** 180 degree character rotation **ROTATE270** 270 degree character rotation
AFP_STYLE	FIXED BINARY (31)	The style of the font: **ROMAN** Roman **ITALIC** Italic

Output Parameters	Data Attribute	Description
AFP_FONT_ID	CHAR (9)	ID for the font, returned by AFPDFNT, and used on subsequent procedure calls such as AFPSFNT.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPDROW (Define Row)

```
DCL AFPDROW EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPDROW (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_MIN_SUBROW_DEPTH_ARRAY,
              AFP_TOP_THICKNESS,
              AFP_BOTTOM_THICKNESS,
              AFP_NUMBER_COLUMNS,
              AFP_NUMBER_SUBROWS,
              AFP_ROW_ARRANGE_ARRAY,
              AFP_COLUMN_WIDTH_ARRAY,
              AFP_ROW_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_DOCUMENT_ HANDLE	FIXED BINARY (31)	Document handle (returned by AFPBDOC).
AFP_MIN_ SUBROW_ DEPTH_ ARRAY	DCL AFP_MIN_SUBROW_DEPTH_ARRAY (64) FIXED BINARY (31) INIT((64)0);	The minimum depth of each subrow in the row. AFP-MIN-SUBROW-DEPTH-ARRAY (n) contains the minimum depth of the nth subrow in the row. **AFP_DEFAULT** - Use the default subrow depth. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as an array of REAL numbers with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_TOP_ THICKNESS	FIXED BINARY (31)	The thickness of the top horizontal rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_BOTTOM_ THICKNESS	FIXED BINARY (31)	The thickness of the bottom horizontal rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_NUMBER_ COLUMNS	FIXED BINARY (31)	Number of columns: 1–64.
AFP_NUMBER_ SUBROWS	FIXED BINARY (31)	Number of subrows: 1–64

Input Parameters	Data Attribute	Description
AFP_ROW_ ARRANGE_ ARRAY	DCL AFP_ROW_ARRANGE_ARRAY (s,c) FIXED BINARY (31) INIT((64)0)	The FIELD_IDs returned by the AFPDFLD calls for the fields that are to be used in each position of the row. AFP_ROW_ARRANGE_ARRAY(n,m) contains the field ID of the nth sub_row in the mth column. **Notes:** 1. The total number of array elements in the AFP_ROW_ARRANGE_ARRAY is the product of AFP_NUMBER_SUBROWS and AFP_NUMBER_COLUMNS. This number can not exceed 64. 2. This declaration is not part of the APQPVAR copy book. Therefore, declare this variable in your PL/1 program.
AFP_COLUMN_ WIDTH_ ARRAY	DCL AFP_COLUMN_WIDTH_ARRAY (64) FIXED BINARY (31) INIT((64)0);	The width of each column in the row. COLUMN_WIDTH_ARRAY(n) contains the width of the nth column in the row. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as an array of REAL numbers with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Output Parameters	Data Attribute	Description
AFP_ROW_ID	CHAR (9)	ID returned for the row. It is used on subsequent AFPBROW procedure calls.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEARE (End Area)

```
DCL AFPEARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEARE (AFPAPI_HANDLE,
              AFP_AREA_HANDLE,
              AFP_AREA_DEPTH,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_AREA_ HANDLE	FIXED BINARY (31)	Handle of the area to be ended (returned from AFPCARE).

Output Parameters	Data Attribute	Description
AFP_AREA_ DEPTH	FIXED BINARY (31)	The depth of the area, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEDOC (End Document)

```
DCL AFPEDOC EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEDOC (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.

Input/Output Parameters	Data Attribute	Description
AFP_DOCUMENT_HANDLE	FIXED BINARY (31)	Document handle (returned from AFPBDOC).

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEFLD (End Field)

```
DCL AFPEFLD EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEFLD (AFPAPI_HANDLE,
              AFP_TABLE_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_TABLE_ HANDLE	FIXED BINARY (31)	Table handle (returned by AFPBTBL).

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEGRP (End Group)

```
DCL AFPEGRP EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEGRP (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_GROUP_NAME,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_DOCUMENT_ HANDLE	FIXED BINARY (31)	Document state handle (returned by AFPBDOC).
AFP_GROUP_ NAME	CHAR (64)	A 64-byte character string that's padded on the right with blanks if the actual group name is less than 64 characters.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEND (End AFPAPI)

```
DCL AFPEND EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEND (AFPAPI_HANDLE,
             AFP_RET_CODE,
             AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEPAG (End Page)

```
DCL AFPEPAG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEPAG (AFPAPI_HANDLE,
              AFP_PAGE_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.

Input/Output Parameters	Data Attribute	Description
AFP_PAGE_ HANDLE	FIXED BINARY (31)	Page handle (returned from AFPBPAG).

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEPAR (End Paragraph)

```
DCL AFPEPAR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEPAR (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              AFP_PARAGRAPH_DEPTH,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.

Input/Output Parameters	Data Attribute	Description
AFP_ PARAGRAPH_ HANDLE	FIXED BINARY (31)	Paragraph handle (returned from AFPBPAR).

Output Parameters	Data Attribute	Description
AFP_ PARAGRAPH_ DEPTH	FIXED BINARY (31)	The depth of the paragraph, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPEROW (End Row)

```
DCL AFPEROW EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPEROW (AFPAPI_HANDLE,
              AFP_TABLE_HANDLE,
              AFP_CURRENT_TABLE_DEPTH,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_TABLE_ HANDLE	FIXED BINARY (31)	Table handle (returned by AFPBTBL).

Output Parameters	Data Attribute	Description
AFP_CURRENT_ TABLE_DEPTH	FIXED BINARY (31)	The depth of the table, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPETBL (End Table)

```
DCL AFPETBL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPETBL (AFPAPI_HANDLE,
              AFP_TABLE_HANDLE,
              AFP_TABLE_DEPTH,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.

Input/Output Parameters	Data Attribute	Description
AFP_TABLE_ HANDLE	FIXED BINARY (31)	Table handle (returned from AFPBTBL).

Output Parameters	Data Attribute	Description
AFP_TABLE_ DEPTH	FIXED BINARY (31)	The depth of the table, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPGBUF (Get Output Buffer)

```
DCL AFPGBUF EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPGBUF (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_BUFFER,
              AFP_BUFFER_LENGTH,
              AFP_MORE_RECORDS,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_DOCUMENT_HANDLE	FIXED BINARY (31)	Document state handle (returned from AFPBDOC).

Output Parameters	Data Attribute	Description
AFP_BUFFER	CHAR (max buffer length)	A buffer in which AFP API returns the next record (structured field) in the page. This parameter is defined in APQPVAR as CHAR(8205).
AFP_BUFFER_LENGTH	FIXED BINARY (31)	The length of the structured field returned in AFP_BUFFER.
AFP_MORE_RECORDS	FIXED BINARY (31)	**TRU** Another output record exists. Issue AFPGBUF again to get the next record. **FALS** No more records exist.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPINIT (Initialize AFPAPI)

```
DCL AFPINIT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPINIT (AFPAPI_HANDLE,
              AFP_TRACE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description	
AFP_TRACE	FIXED BINARY (31)	**TRU**	TRU is not supported; always set this parameter to FALS.
		FALS	Don't generate a trace file for this AFP API session.

Output Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Returned by AFPINIT and required on each subsequent AFPAPI call
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPINVM (Invoke Medium Map)

```
DCL AFPINVM EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPINVM (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_MEDIUM_MAP_NAME,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Returned by AFPINIT and required on each subsequent AFPAPI call
AFP_DOCUMENT_HANDLE	FIXED BINARY (31)	Returned by AFPBDOC.
AFP_MEDIUM_MAP_NAME	CHAR (8)	An 8-byte character string that's padded on the right with blanks if the actual medium map name is less than 8.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPIOBJ (Include Object)

```
DCL AFPIOBJ EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPIOBJ (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_OBJECT_NAME,
              AFP_OBJECT_WIDTH,
              AFP_OBJECT_DEPTH,
              AFP_OBJECT_ROTATION,
              AFP_OBJECT_MAPPING_OPTION,
              AFP_OBJECT_X_OFFSET,
              AFP_OBJECT_Y_OFFSET,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_HANDLE	FIXED BINARY (31)	Current state handle (returned from AFPBPAG or AFPCARE).
AFP_OBJECT_NAME	CHAR (8)	An 8-byte character string that's left-aligned and padded on the right with blanks if the actual object name is less than 8.
AFP_OBJECT_WIDTH	FIXED BINARY (31)	The width of the object area. **AFP_DEFAULT**: Use the width value specified in the object. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_OBJECT_DEPTH	FIXED BINARY (31)	The depth of the object area. **AFP_DEFAULT**: Use the depth value specified in the object. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_OBJECT_ROTATION	FIXED BINARY (31)	The rotation of the object area: **ROTATE_DEFAULT** Use the rotation specified in the object. **ROTATE0** 0 degree rotation **ROTATE90** Rotate the object 90 degrees **ROTATE180** Rotate the object 180 degrees **ROTATE270** Rotate the object 270 degrees

Input Parameters	Data Attribute	Description
AFP_OBJECT_MAPPING_OPTION	FIXED BINARY (31)	**DEFAULT_MAP** Use the mapping option specified in the object. **SCALE_TO_FIT** Center the object within the area dimensions specified in AFP_OBJECT_WIDTH and AFP_OBJECT_DEPTH and scale the object to fit within the area. **CENTER_AND_TRIM** Center the object within the area dimensions specified in AFP_OBJECT_WIDTH and AFP_OBJECT_DEPTH and trim what falls outside the area. **POSITION_AND_TRIM** Position the object at the location specified in AFP_OBJECT_X_OFFSET and AFP_OBJECT_Y_OFFSET within the dimensions specified in AFP_OBJECT_WIDTH and AFP_OBJECT_DEPTH and trim what falls outside the area. **POINT-TO-PEL** Position the object at the object area origin within the dimensions specified in AFP_OBJECT_WIDTH and AFP_OBJECT_DEPTH, and trim what falls outside the area. No resolution correction is done; that is, each image point is mapped to a pel. **DOUBLE-DOT** Position the object at the object area origin within the dimensions specified in AFP_OBJECT_WIDTH and AFP_OBJECT_DEPTH, and trim what falls outside the area. Each image point is doubled. No resolution correction is done; that is, each of the new image points is mapped to a pel.
AFP_OBJECT_X_OFFSET	FIXED BINARY (31)	The X offset of the object. **AFP_DEFAULT** Use the x offset value specified in the object. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_OBJECT_Y_OFFSET	FIXED BINARY (31)	The Y offset of the object. **AFP_DEFAULT** Use the y offset value specified in the object. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPIOVL (Include Page Overlay)

```
DCL AFPIOVL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPIOVL (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_OVLY_NAME,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).
AFP_OVLY_ NAME	CHAR (8)	An 8-byte character string that's left aligned and padded on the right with blanks if the actual overlay name is less than 8.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPIPSG (Include Page Segment)

```
DCL AFPIPSG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPIPSG (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_PSEG_NAME,
              AFP_INLINE_OPTION,
              AFP-REUSE_OPTION,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).
AFP_PSEG_NAME	CHAR (8)	An 8-byte character string that's left aligned and padded on the right with blanks if the actual page segment name is less than 8.
AFP_INLINE_ OPTION	FIXED BINARY (31)	**TRU** Bring the page segment inline as part of the page. **FALS** Don't bring the page segment inline, but simply reference it.
AFP_REUSE_ OPTION	FIXED BINARY (31)	**TRU** The page segment will be reused on multiple pages. **FALS** The page segment will not be reused on multiple pages.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPPARE (Put Area)

```
DCL AFPPARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPPARE (AFPAPI_HANDLE,
              AFP_PAGE_HANDLE,
              AFP_AREA_HANDLE,
              AFP_AREA_ROTATION,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_PAGE_ HANDLE	FIXED BINARY (31)	Page handle (returned by AFPBPAG).
AFP_AREA_ HANDLE	FIXED BINARY (31)	Area handle (returned from AFPCARE).
AFP_AREA_ ROTATION	FIXED BINARY (31)	The rotation of the area: **ROTATE0** 0 degree rotation **ROTATE90** Rotate the area 90 degrees **ROTATE180** Rotate the area 180 degrees **ROTATE270** Rotate the area 270 degrees

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPPBOX (Put Box)

```
DCL AFPPBOX EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPPBOX (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_BOX_WIDTH,
              AFP_BOX_DEPTH,
              AFP_SHADING_PATTERN,
              AFP_SHADING_INTENSITY,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).
AFP_BOX_WIDTH	FIXED BINARY (31)	The width of the box. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_BOX_DEPTH	FIXED BINARY (31)	The depth of the box. Valid values for this parameter are: A value between 0.0002 and 5918.0 when using millimeters A value between 0.0001 and 591.0 when using centimeters A value between 0.0001 and 233.0 when using inches A value between 0.0017 and 55924.0 when using 240 units per inch A value between 0.0101 and 99999.9999 when using 1440 units per inch **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_SHADING_ PATTERN	FIXED BINARY (31)	The shading pattern: **STNDARD** Use the standard shading pattern **SCREEN** Use the screen shading pattern **NOSHADE** Don't use a shading pattern
AFP_SHADING_ INTENSITY	FIXED BINARY (31)	The shading intensity: 0-100.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPPCHS (Put Character String)

```
DCL AFPPCHS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPPCHS (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_STRING_LENGTH,
              AFP_CHARACTER_STRING,
              AFP_ALIGNMENT_OPTION,
              AFP_ALIGNMENT_CHAR,
              AFP_POSITION_OPTION,
              AFP_UNDERLINE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG, AFPCARE, or AFPBTBL).
AFP_STRING_ LENGTH	FIXED BINARY (31)	The length of the character string: 1 – limit supported by your compiler.
AFP_ CHARACTER_ STRING	CHAR (max string length)	A character string of length AFP-STRING-LENGTH. This parameter is defined in APQPVAR as CHAR(160).
AFP_ ALIGNMENT_ OPTION	FIXED BINARY (31)	The horizontal alignment option: **R_GHT** Right align the string **L_FT** Left align the string **CENTER** Center the string **CHAR** Align the string around the character specified in AFP_ALIGNMENT_CHAR
AFP_ALIGNMENT_ CHAR	CHAR (1)	The character for character alignment: any character except a blank.
AFP_POSITION_ OPTION	FIXED BINARY (31)	**TRU** Keep the position at the origin of the character string. **FALS** Don't keep the position at the origin of the character string, but move it to the end of the character string. **Note:** This parameter causes a line break when you place a character string in a table field. If set to TRU, the character string begins at the start of a new line.
AFP_UNDERLINE	FIXED BINARY (31)	**TRU** Underline the character string. **FALS** Don't underline the character string.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPPRUL (Put Rule)

```
DCL AFPPRUL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPPRUL (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_DIRECTION,
              AFP_RULE_LENGTH,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).
AFP_DIRECTION	FIXED BINARY (31)	The direction of the rule: **XDIRECTION** Rule parallel to the X (inline) axis. **YDIRECTION** Rule parallel to the Y (baseline) axis.
AFP_RULE_ LENGTH	FIXED BINARY (31)	The length of the rule. See REAL on page 3 for a list of valid values.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPPTAG (Put Tag)

```
DCL AFPPTAG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPPTAG (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_TAG_NAME,
              AFP_TAG_VALUE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC or AFPBPAG).
AFP_TAG_NAME	CHAR (64)	A 64-byte character string that's padded on the right with blanks if the actual tag name is less than 64 characters.
AFP_TAG_VALUE	CHAR (64)	A 64-byte character string that's padded on the right with blanks if the actual tag value is less than 64 characters.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPPTXT (Put Text)

```
DCL AFPPTXT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPPTXT (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_STRING_LENGTH,
              AFP_CHARACTER_STRING,
              AFP_CONCATENATE,
              AFP_UNDERLINE,
              AFP_REMAINING_LENGTH,
              AFP_REMAINING_STRING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAR or AFPBTBL).
AFP_STRING_ LENGTH	FIXED BINARY (31)	The length of the character string: 1 – limit supported by your compiler.
AFP_ CHARACTER_ STRING	CHAR (max string length)	A character string of length AFP_STRING_LENGTH. This parameter is defined in APQPVAR as CHAR(160).
AFP_ CONCATENATE	FIXED BINARY (31)	On the first AFPPTXT call in a field or paragraph, always set this parameter to TRU. On subsequent calls, set the parameter to TRU or FALS, depending on whether concatenation is desired. **TRU** Concatenate the character string in this AFPPTXT on the same line as the character string in the previous AFPPTXT (space permitting). **FALS** Don't concatenate the character string in this AFPPTXT with the string in the previous AFPPTXT. Instead, start a new line.
AFP_UNDERLINE	FIXED BINARY (31)	**TRU** Underline the character string. **FALS** Don't underline the character string.

Output Parameters	Data Attribute	Description
AFP_ REMAINING_ LENGTH	FIXED BINARY (31)	If the entire string would not fit, specifies the remaining length (characters not placed).
AFP_ REMAINING_ STRING	FIXED BINARY (31)	Characters in the string that would not fit.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPQATT (Query Current Attributes)

```
DCL AFPQATT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPQATT (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_UNIT_OF_MEASURE,
              AFP_X_COORDINATE,
              AFP_Y_COORDINATE,
              AFP_COLOR,
              AFP_RULE_THICKNESS,
              AFP_FONT_ID,
              AFP_CHARACTER_SPACING,
              AFP_WORD_SPACING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG or AFPCARE).

Output Parameters	Data Attribute	Description
AFP_UNIT_OF_ MEASURE	FIXED BINARY (31)	The current unit of measure: **INCH** Inches **MM** Millimeters **CM** Centimeters **U240** 240 units per inch **U1440** 1440 units per inch
AFP_X_ COORDINATE	FIXED BINARY (31)	The X coordinate of the current position, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_Y_ COORDINATE	FIXED BINARY (31)	The Y coordinate of the current position, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_COLOR	FIXED BINARY (31)	The current color: **BLACK** Black **BLUE** Blue **RED** Red **MAGENTA** Magenta **GREEN** Green **CYAN** Cyan **YELLOW** Yellow **BROWN** Brown **MEDIA** Color of medium
AFP_RULE_ THICKNESS	FIXED BINARY (31)	The current rule thickness. See REAL on page 3 for a list of valid values.
AFP_FONT_ID	FIXED BINARY (31)	The ID of the current font (returned by AFPDFNT)
AFP_ CHARACTER_ SPACING	FIXED BINARY (31)	The current character spacing, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_WORD_ SPACING	FIXED BINARY (31)	The current word spacing, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPQPOS (Query Current Position)

```
DCL AFPQPOS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPQPOS (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_X_COORDINATE,
              AFP_Y_COORDINATE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).

Output Parameters	Data Attribute	Description
AFP_X_ COORDINATE	FIXED BINARY (31)	The current X coordinate, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_Y_ COORDINATE	FIXED BINARY (31)	The current Y coordinate, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPQSTR (Query Character String Size)

```
DCL AFPQSTR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPQSTR (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_CHARACTER_STRING,
              AFP_STRING_LENGTH,
              AFP_MEASURED_WIDTH,
              AFP_LINE_SPACING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG, AFPCARE, AFPBPAR, or AFPBTBL).
AFP_CHARACTER_ STRING	CHAR (max string length)	A character string of length AFP_STRING_LENGTH. This parameter is defined in APQPVAR as CHAR(160).
AFP_STRING_ LENGTH	FIXED BINARY (31)	The length of the character string: 1 – limit supported by your compiler.

Output Parameters	Data Attribute	Description
AFP_MEASURED_ WIDTH	FIXED BINARY (31)	The width of the string in the current font, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_LINE_ SPACING	FIXED BINARY (31)	The depth of the string in the current font, returned in the current unit of measure. See REAL on page 3 for a list of valid values.
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSCLR (Set Color)

```
DCL AFPSCLR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSCLR (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_COLOR,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG, AFPCARE, AFPBPAR, or AFPBTBL).
AFP_COLOR	FIXED BINARY (31)	The color to be printed: **BLACK** Black **BLUE** Blue **RED** Red **MAGENTA** Magenta **GREEN** Green **CYAN** Cyan **YELLOW** Yellow **BROWN** Brown **MEDIA** Color of medium

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSFNT (Set Font)

```
DCL AFPSFNT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSFNT (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_FONT_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG, AFPCARE, AFPBPAR, or AFPBTBL).
AFP_FONT_ID	CHAR (9)	ID of the font (returned by AFPDFNT).

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSICS (Set Intercharacter Spacing)

```
DCL AFPSICS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSICS (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_CHARACTER_SPACING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG, AFPCARE, AFPBPAR, AFPBTBL).
AFP_ CHARACTER_ SPACING	FIXED BINARY (31)	The amount of space to be inserted between characters. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSLIB (Set Resource Library Names)

```
DCL AFPSLIB EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSLIB (AFPAPI_HANDLE,
              AFP_PSEG_LIBRARY,
              AFP_OBJECT_LIBRARY,
              AFP_FONT_LIBRARY,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_PSEG_ LIBRARY	CHAR (8)	An 8-byte character string that's padded on the right with blanks if the actual page segment library name is less than 8.
AFP_OBJECT_ LIBRARY	CHAR (8)	An 8-byte character string that's padded on the right with blanks if the actual object library name is less than 8.
AFP_FONT_ LIBRARY	CHAR (8)	An 8-byte character string that's padded on the right with blanks if the actual font library name is less than 8.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSOUT (Set Output Characteristics)

```
DCL AFPSOUT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSOUT (AFPAPI_HANDLE,
              AFP_OUTPUT_RECORD_SIZE,
              AFP_OUTPUT_FILENAME,
              AFP_OUTPUT_FILETYPE,
              AFP_OUTPUT_FILEMODE,
              AFP_REPLACE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_OUTPUT_ RECORD_SIZE	FIXED BINARY (31)	If API is writing output records to an output file, valid values are 512 bytes to 8205 bytes; for MVS, the value cannot exceed the DCB length for the file. If API is writing output records to an output buffer, valid values are 512 bytes to 32767 bytes.
AFP_OUTPUT_ FILENAME	CHAR (8)	One of the following values: An 8-byte character string that's left-aligned and padded on the right with blanks if the actual output filename is less than 8 characters. **BUFFERED** API is to write output to a buffer instead of to an output file. **DISCBUFF** API is to discard the output instead of writing it to an output buffer or output file.
AFP_OUTPUT_ FILETYPE	CHAR (8)	An 8-byte character string that's left-aligned and padded on the right with blanks if the actual output filetype is less than 8 characters. This parameter is ignored on MVS. This parameter is ignored on VM if AFP_OUTPUT_FILENAME is set to BUFFERED or DISCBUFF.
AFP_OUTPUT_ FILEMODE	CHAR (2)	A 2-byte character string that's left-aligned and padded on the right with blanks if the actual output filemode is less than 2 characters. This parameter is ignored on MVS. This parameter is ignored on VM if AFP_OUTPUT_FILENAME is set to BUFFERED or DISCBUFF.
AFP_REPLACE	FIXED BINARY (31)	**TRU** Replace the file if it already exists. **FALS** Do not replace an existing file. This parameter is ignored on MVS. This parameter is ignored on VM if AFP_OUTPUT_FILENAME is set to BUFFERED or DISCBUFF.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSPOS (Set Position)

```
DCL AFPSPOS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSPOS (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_X_COORDINATE,
              AFP_X_REF_COORD_SYS,
              AFP_Y_COORDINATE,
              AFP_Y_REF_COORD_SYS,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBPAG or AFPCARE).
AFP_X_ COORDINATE	FIXED BINARY (31)	The X coordinate of the position. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_X_REF_ COORD_SYS	FIXED BINARY (31)	**XABS** The X_COORDINATE value is an absolute offset from the X coordinate system origin. **XREL** The X_COORDINATE value is a relative offset from the current X position.
AFP_Y_ COORDINATE	FIXED BINARY (31)	The Y coordinate of the position. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.
AFP_Y_REF_ COORD_SYS	FIXED BINARY (31)	**YABS** The Y_COORDINATE value is an absolute offset from the Y coordinate system origin. **YREL** The Y_COORDINATE value is a relative offset from the current Y position. **YLINES** Number of lines from the current Y position.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSRTH (Set Rule Thickness)

```
DCL AFPSRTH EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSRTH (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_RULE_THICKNESS,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG or AFPCARE).
AFP_RULE_ THICKNESS	FIXED BINARY (31)	The thickness of the rule. See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSUNI (Set Units)

```
DCL AFPSUNI EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSUNI (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_UNIT_OF_MEASURE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG or AFPCARE).
AFP_UNIT_OF_ MEASURE	FIXED BINARY (31)	The unit of measure: **INCH** Inches **MM** Millimeters **CM** Centimeters **U240** 240 units per inch **U1440** 1440 units per inch **Note:** The output file generated by AFP API is in logical units of 1440 per inch, even if you specify a different unit of measure in this parameter.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPSWSP (Set Word Spacing)

```
DCL AFPSWSP EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPSWSP (AFPAPI_HANDLE,
              AFP_CURRENT_HANDLE,
              AFP_WORD_SPACING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.
AFP_CURRENT_ HANDLE	FIXED BINARY (31)	Current state handle (returned by AFPBDOC, AFPBPAG, AFPCARE, AFPBPAR, or AFPBTBL).
AFP_WORD_ SPACING	FIXED BINARY (31)	The width of the space between words. **AFP_DEFAULT** Use the default wordspacing See REAL on page 3 for a list of valid values. **Note:** AFP API interprets this parameter as a REAL number with 4 decimal positions. Therefore, multiply the value for this parameter by 10,000 prior to invoking this procedure call.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPTERM (Terminate AFPAPI)

```
DCL AFPTERM EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPTERM (AFPAPI_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

AFPXARE (Destroy Area)

```
DCL AFPXARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER);

CALL AFPXARE (AFPAPI_HANDLE,
              AFP_AREA_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

Input Parameters	Data Attribute	Description
AFPAPI_HANDLE	FIXED BINARY (31)	Value returned from AFPINIT.

Input Parameters	Data Attribute	Description
AFP_AREA_ HANDLE	FIXED BINARY (31)	Handle of the area to be destroyed (returned from AFPCARE).

Output Parameters	Data Attribute	Description
AFP_RET_CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the return codes.
AFP_SEVERITY_ CODE	FIXED BINARY (31)	Refer to *AFP API: Programming Guide and Reference* for a list of the severity codes.

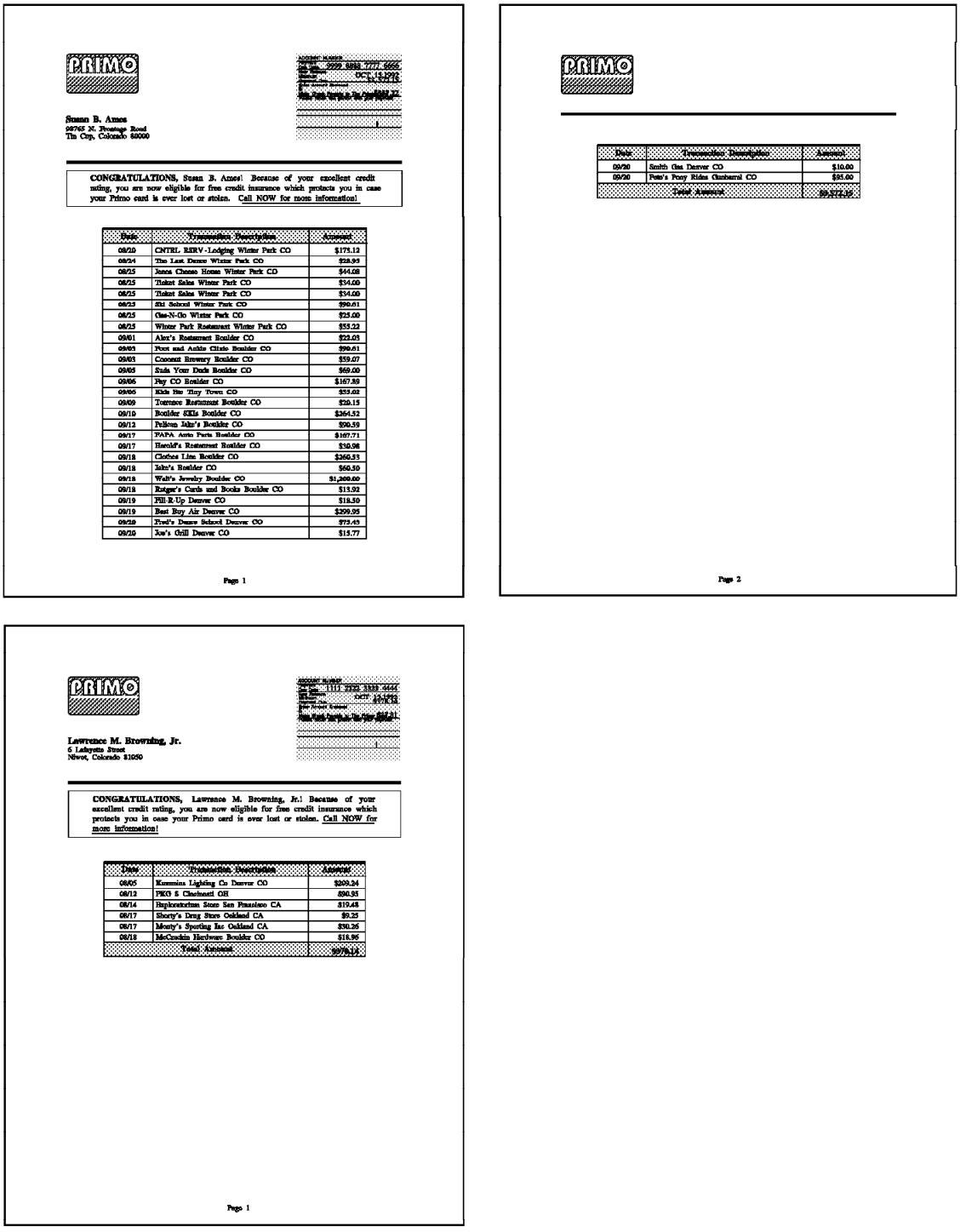
Chapter 3. PL/1 Sample Code

This is the source code, shipped with the product, that produces the sample document shown in Figure 1 on page 58. Two forms of the code are shipped:

- APQPSAMP, which invokes PL/1 subroutines provided in APQPPRF
- APQPSMP2, which uses direct calls from the PL/1 program

The sample code is printed in 2-column format throughout this chapter.

Sample Document



PRIMO

Susan B. Ames
98765 N. Frontage Road
Tin City, Colorado 80000

ACCOUNT NUMBER
5599 8888 7777 6666
OCT. 15 19XX
$4,327.15

CONGRATULATIONS, Susan B. Ames! Because of your excellent credit rating, you are now eligible for free credit insurance which protects you in case your Primo card is ever lost or stolen. Call NOW for more information!

Date	Transaction Description	Amount
08/20	CNTRL RSRV - Lodging Winter Park CO	$175.12
08/24	The Last Dance Winter Park CO	$28.93
08/25	Janes Cheese House Winter Park CO	$44.08
08/25	Ticket Sales Winter Park CO	$34.00
08/25	Ticket Sales Winter Park CO	$34.00
08/23	Ski School Winter Park CO	$90.61
08/25	Gas-N-Go Winter Park CO	$25.00
08/25	Winter Park Restaurant Winter Park CO	$53.22
09/01	Alex's Restaurant Boulder CO	$23.05
09/03	Foot and Ankle Clinic Boulder CO	$90.61
09/03	Crocent Brewery Boulder CO	$59.07
09/05	Suds Your Duds Boulder CO	$69.00
09/06	Pay CO Boulder CO	$167.80
09/06	Kids Rec Tiny Town CO	$53.02
09/09	Torrance Restaurant Boulder CO	$20.15
09/10	Boulder CEIs Boulder CO	$264.52
09/12	Falcon Jake's Boulder CO	$90.59
09/17	PAPA Auto Parts Boulder CO	$167.71
09/17	Harold's Restaurant Boulder CO	$30.98
09/18	Clothes Line Boulder CO	$260.55
09/18	Jake's Boulder CO	$60.50
09/18	Walt's Jewelry Boulder CO	$1,200.60
09/18	Ratgar's Cards and Books Boulder CO	$13.92
09/19	Fill-R-Up Denver CO	$18.50
09/19	Best Buy Air Denver CO	$299.95
09/20	Fred's Dance School Denver CO	$73.45
09/20	Joe's Grill Denver CO	$13.77

Page 1

PRIMO

Date	Transaction Description	Amount
09/20	Smith Gas Denver CO	$10.00
09/20	Pete's Pony Rides Cimharral CO	$95.00
	Total Amount	$9,272.35

Page 2

PRIMO

Lawrence M. Browning, Jr.
6 Lafayette Street
Niwot, Colorado 81050

ACCOUNT NUMBER
1111 2222 3333 4444
OCT. 15 19XX
$4,327.15

CONGRATULATIONS, Lawrence M. Browning, Jr.! Because of your excellent credit rating, you are now eligible for free credit insurance which protects you in case your Primo card is ever lost or stolen. Call NOW for more information!

Date	Transaction Description	Amount
08/05	Kunmins Lighting Co Denver CO	$209.24
08/12	PKO 5 Cincinnati OH	$90.95
08/14	Exploratorium Store San Francisco CA	$19.48
08/17	Shorty's Drug Store Oakland CA	$9.25
08/17	Monty's Sporting Inc Oakland CA	$90.95
08/18	McCaskin Hardware Boulder CO	$14.95
	Total Amount	$978.12

Page 1

Figure 1. Sample Document

APQPSAMP

```
/*************************************************************/
/*   PL/I PROGRAM -- AFPAPI APQPSMP                          */
/*                                                           */
/* This PL/I program invokes the AFP API to produce a sample */
/* customer billing statement.  See the "AFP API Programming */
/* Guide and Reference" for a picture of the print output    */
/* produced by this program.                                 */
/*************************************************************/

APQPSMP: PROCEDURE OPTIONS (MAIN);

DCL DATAIN FILE INPUT ENV(FB RECSIZE(80));

DCL 1 TRAN_IN,
    2 POST_DATE_IN               PIC '(4)9',
    2 TRANS_DESC                 CHAR(40),
    2 TRANSACTION_AMOUNT_IN      PIC '(7)9V99',
    2 TRAN_FILLER                CHAR(27);

DCL CUST_IN;
  DCL CUST_NAME                  CHAR(30) VARYING;
  DCL CUST_ST_ADDR               CHAR(30) VARYING;
  DCL CUST_CITY_STATE            CHAR(35) VARYING;
  DCL CUSTOMER_BALANCE_IN        PIC '(5)9V99';

DCL 1 ACCOUNT_NUM,
    2 ACCT_1                     CHAR (4),
    2 ACCT_2                     CHAR (4),
    2 ACCT_3                     CHAR (4),
    2 ACCT_4                     CHAR (4);

DCL TRANSACTION_PROCESSING_VARS;
  DCL PAGE_HEADER_DEPTH          FIXED BIN(31)   INIT(0);
  DCL PAGE_BODY                  FIXED BIN(31)   INIT(0);
  DCL BOTTOM_MARGIN              FIXED BIN(31)   INIT(200000);
  DCL TABLE_WHITE_SPACE          FIXED BIN(31)   INIT(100000);
  DCL PARAGRAPH_WHITE_SPACE      FIXED BIN(31)   INIT(0);
  DCL NUM_CUSTOMER_PAGES         FIXED DEC(2)    INIT(1);
  DCL RECORD_COUNT               FIXED DEC(2)    INIT(0);
  DCL TEST_VALUE                 CHAR (2)        INIT('  ');
  DCL BLK                        CHAR (1)        INIT(' ');
  DCL CUST_LENGTH                FIXED DEC(3)    INIT(0);
  DCL TABLE_DEPTH                FIXED DEC(9,4)  INIT(156.6544);
  DCL DUE_DATE                   CHAR (12) INIT('OCT 15, 1992');
  DCL STRING_FIELD               CHAR (160) VARYING;
  DCL STRING_1                   CHAR (80) VARYING;
  DCL STRING_2                   CHAR (80) VARYING;

DCL CUST_OUT;
  DCL OLD_BALANCE_OUT            PIC '$,$$$,$$$V.99';
  DCL MIN_AMOUNT_DUE_OUT         PIC '$$$,$$$V.99';
  DCL POST_DATE_OUT              PIC '99/99';
  DCL TRANSACTION_DATE_OUT       PIC '99/99';
  DCL TRANSACTION_AMOUNT_OUT     PIC '$,$$$,$$$V.99';
  DCL CUSTOMER_BALANCE_OUT       PIC '$,$$$,$$$V.99';
  DCL EXCLAMATION                CHAR(1) INIT ('4F'X);
  DCL NUM_CUSTOMER_PAGES_OUT     PIC 'Z9';

DCL PROCESSING_SWITCHES;
  DCL DATA_REMAINS               BIT (1) INIT ('1'B);
  DCL NEW_CUSTOMER               BIT (1) INIT ('0'B);
  DCL YES                        BIT (1) INIT ('1'B);
  DCL NO                         BIT (1) INIT ('0'B);
  DCL MIN_AMOUNT_DUE_COMP        FIXED DEC(7,2);

DCL FONT_IDS;
  DCL TIM1OMED                   CHAR(9);
  DCL TIM12MED                   CHAR(9);
  DCL TIM12MEDITAL               CHAR(9);
  DCL TIM12BOLD                  CHAR(9);

DCL FIELD_IDS;
  DCL FIELDH1                    FIXED BIN(31);
  DCL FIELDH2                    FIXED BIN(31);
  DCL FIELDH3                    FIXED BIN(31);
  DCL FIELDT1                    FIXED BIN(31);
  DCL FIELDT2                    FIXED BIN(31);
  DCL FIELDT3                    FIXED BIN(31);
  DCL FIELDS1                    FIXED BIN(31);
  DCL FIELDS2                    FIXED BIN(31);

DCL ROW_IDS;
  DCL ROW1                       CHAR(9);
  DCL ROW2                       CHAR(9);
  DCL ROW3                       CHAR(9);

DCL AFP_ROW_ARRANGE_ARRAY (1,3) FIXED BIN (31) INIT ((3)0);

DCL AFP_ROW_ARRANGE_ARRAY2 (2,3) FIXED BIN (31) INIT ((6)0);

%PAGE;
%INCLUDE DDSYS (APQPCON);
%PAGE;
%INCLUDE DDSYS (APQPVAR);

%PAGE;
/*************************************************************/
/*                                                           */
/* This program produces a sample customer statement.  The   */
/* statement contains a logo stored in a page segment, a     */
/* paragraph enclosed in a box with variable data, account   */
/* summary information which is highlighted in a shaded area, */
/* and the account transactions contained in a variable depth */
/* table.                                                    */
/*                                                           */
/* The mainline logic is as follows:                         */
/*                                                           */
/*   Initialize the API and define the fonts, fields, and    */
/*     rows.                                                 */
/*   Load the sample data for the first customer.            */
/*   Process a customer until no more customers to process.  */
/*   End the API.                                            */
/*************************************************************/

  ON ENDFILE (DATAIN)
      DATA_REMAINS = NO;

  STRING_ARRAY = ' ';

  OPEN FILE (DATAIN) RECORD SEQUENTIAL;

  CALL SETUP_AFPAPI;

  CALL READ_DATA;

  DO WHILE (DATA_REMAINS);
    CALL PROCESS_A_CUSTOMER;
  END;

  CALL END_PROCESSING;

  PUT SKIP LIST(' ');
  PUT SKIP LIST('APQPSMP COMPLETED');

%PAGE;
/*************************************************************/
/*    SETUP_AFPAPI.                                          */
/*        Initialize the AFP API.                            */
/*        Begin a document.                                  */
/*        Set the output characteristics.                    */
/*        Define the fonts.                                  */
/*        Define the fields and rows of a table.             */
/*************************************************************/

SETUP_AFPAPI:  PROC;

   AFP_TRACE             = FALS;

   CALL DO_AFPINIT;

/*                                                           */
/*    SET THE OUTPUT CHARACTERISTICS                         */
/*                                                           */

   AFP_OUTPUT_RECORD_SIZE = 8205;
   AFP_REPLACE            = TRU;
   AFP_OUTPUT_FILENAME    = 'APQSAMP';

/* the following two fields are not used in MVS */
```

```
   AFP_OUTPUT_FILETYPE    = 'LISTAFP';
   AFP_OUTPUT_FILEMODE    = 'A1';

   CALL DO_AFPSOUT;

/*                                                          */
/*    Begin the document                                    */
/*                                                          */

   AFP_UNIT_OF_MEASURE    = MM;
   AFP_DOC_PAGE_WIDTH     = 215.0 * 10000;
   AFP_DOC_PAGE_DEPTH     = 280.0 * 10000;
   AFP_PAGE_ORIENTATION   = ORIENT0;

   CALL DO_AFPBDOC;

   AFP_CURRENT_HANDLE     = AFP_DOCUMENT_HANDLE;

/*                                                          */
/*    DEFINE THE FONTS                                      */
/*                                                          */

   AFP_CODE_PAGE        = 'T1V10500';
   AFP_DESC_NAME_LENGTH = 22;
   AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
   AFP_POINT_SIZE       = 10;
   AFP_WEIGHT           = MEDIUM;
   AFP_FONT_WIDTH       = NORMAL;
   AFP_ROTATION         = ORIENT0;
   AFP_STYLE            = ROMAN;

   CALL DO_AFPDFNT;

   TIM10MED             = AFP_FONT_ID;
   AFP_CODE_PAGE        = 'T1V10500';
   AFP_DESC_NAME_LENGTH = 22;
   AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
   AFP_POINT_SIZE       = 12;
   AFP_WEIGHT           = MEDIUM;
   AFP_FONT_WIDTH       = NORMAL;
   AFP_ROTATION         = ORIENT0;
   AFP_STYLE            = ROMAN;

   CALL DO_AFPDFNT;

   TIM12MED             = AFP_FONT_ID;
   AFP_CODE_PAGE        = 'T1V10500';
   AFP_DESC_NAME_LENGTH = 22;
   AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
   AFP_POINT_SIZE       = 12;
   AFP_WEIGHT           = MEDIUM;
   AFP_FONT_WIDTH       = NORMAL;
   AFP_ROTATION         = ORIENT0;
   AFP_STYLE            = ITALIC;

   CALL DO_AFPDFNT;

   TIM12MEDITAL         = AFP_FONT_ID;
   AFP_CODE_PAGE        = 'T1V10500';
   AFP_DESC_NAME_LENGTH = 22;
   AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
   AFP_POINT_SIZE       = 12;
   AFP_WEIGHT           = BOLD;
   AFP_FONT_WIDTH       = NORMAL;
   AFP_ROTATION         = ORIENT0;
   AFP_STYLE            = ROMAN;

   CALL DO_AFPDFNT;

   TIM12BOLD              = AFP_FONT_ID;

/*                                                          */
/*   THIS IS THE START OF THE FIELD AND ROW DEFINITIONS     */
/*                                                          */

   AFP_FORMAT_OPTION        = FOCENTER;
   AFP_ALIGNMENT_POSITION   = 0.0 * 10000;
   AFP_VERTICAL_FORMAT      = VERCENTER;
   AFP_LEFT_MARGIN          = 0.0 * 10000;
   AFP_RIGHT_MARGIN         = 0.0 * 10000;
   AFP_LINE_SPACING         = AFP_DEFAULT * 10000;
   AFP_TEXT_ORIENTATION     = TXTORO_0;
   AFP_SHADING_PATTERN      = SCREEN;
   AFP_SHADING_INTENSITY    = 18;
   AFP_TOP_THICKNESS        = .5 * 10000;
   AFP_BOTTOM_THICKNESS     = .5 * 10000;
   AFP_LEFT_THICKNESS       = .5 * 10000;
   AFP_RIGHT_THICKNESS      = .5 * 10000;

   CALL DO_AFPDFLD;

   FIELDH1                  = AFP_FIELD_ID;

   CALL DO_AFPDFLD;

   FIELDH2                  = AFP_FIELD_ID;

   CALL DO_AFPDFLD;

   FIELDH3                        = AFP_FIELD_ID;
   AFP_NUMBER_COLUMNS             = 3;
   AFP_NUMBER_SUBROWS             = 1;
   AFP_MIN_SUBROW_DEPTH_ARRAY (1) = AFP_DEFAULT * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,1) = FIELDH1;
   AFP_COLUMN_WIDTH_ARRAY     (1) = 25.0 * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,2) = FIELDH2;
   AFP_COLUMN_WIDTH_ARRAY     (2) = 70.0 * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,3) = FIELDH3;
   AFP_COLUMN_WIDTH_ARRAY     (3) = 30.0 * 10000;

   CALL DO_AFPDROW;

   ROW1                     = AFP_ROW_ID;

/*                                                          */
/*   Define the transaction row                             */
/*                                                          */

   AFP_FORMAT_OPTION        = FOCENTER;
   AFP_ALIGNMENT_POSITION   = 0.0 * 10000;
   AFP_VERTICAL_FORMAT      = VERCENTER;
   AFP_LEFT_MARGIN          = 1.0 * 10000;
   AFP_RIGHT_MARGIN         = 1.0 * 10000;
   AFP_LINE_SPACING         = AFP_DEFAULT * 10000;
   AFP_TEXT_ORIENTATION     = TXTORO_0;
   AFP_SHADING_PATTERN      = NOSHADE;
   AFP_SHADING_INTENSITY    = 0;
   AFP_TOP_THICKNESS        = .5 * 10000;
   AFP_BOTTOM_THICKNESS     = .5 * 10000;
   AFP_LEFT_THICKNESS       = .5 * 10000;
   AFP_RIGHT_THICKNESS      = .5 * 10000;

   CALL DO_AFPDFLD;

   FIELDT1                  = AFP_FIELD_ID;

   CALL DO_AFPDFLD;

   FIELDT2                  = AFP_FIELD_ID;
   AFP_LEFT_MARGIN          = 0.0 * 10000;
   AFP_ALIGNMENT_POSITION   = 20 * 10000;

   CALL DO_AFPDFLD;

   FIELDT3                        = AFP_FIELD_ID;
   AFP_TOP_THICKNESS              = 0.5 * 10000;
   AFP_BOTTOM_THICKNESS           = 0.5 * 10000;
   AFP_NUMBER_COLUMNS             = 3;
   AFP_NUMBER_SUBROWS             = 1;
   AFP_MIN_SUBROW_DEPTH_ARRAY (1) = AFP_DEFAULT * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,1) = FIELDT1;
   AFP_COLUMN_WIDTH_ARRAY     (1) = 25.0 * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,2) = FIELDT2;
   AFP_COLUMN_WIDTH_ARRAY     (2) = 70.0 * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,3) = FIELDT3;
   AFP_COLUMN_WIDTH_ARRAY     (3) = 30.0 * 10000;

   CALL DO_AFPDROW;

   ROW2                     = AFP_ROW_ID;

/*                                                          */
/*   Define the summary row                                 */
/*                                                          */

   AFP_ALIGNMENT_POSITION   = 0.0 * 10000;
```

```
   AFP_LEFT_MARGIN          = 1.0 * 10000;
   AFP_RIGHT_MARGIN         = 1.0 * 10000;
   AFP_LINE_SPACING         = AFP_DEFAULT * 10000;
   AFP_SHADING_PATTERN      = SCREEN;
   AFP_SHADING_INTENSITY    = 18;
   AFP_TOP_THICKNESS        = 0.5 * 10000;
   AFP_BOTTOM_THICKNESS     = 0.5 * 10000;
   AFP_LEFT_THICKNESS       = 0.5 * 10000;
   AFP_RIGHT_THICKNESS      = 0.5 * 10000;

   CALL DO_AFPDFLD;

   FIELDS1                  = AFP_FIELD_ID;
   AFP_LEFT_MARGIN          = 0.0 * 10000;
   AFP_ALIGNMENT_POSITION   = 20 * 10000;

   CALL DO_AFPDFLD;

   FIELDS2                        = AFP_FIELD_ID;
   AFP_TOP_THICKNESS              = 0.0 * 10000;
   AFP_BOTTOM_THICKNESS           = 0.0 * 10000;
   AFP_NUMBER_COLUMNS             = 2;
   AFP_NUMBER_SUBROWS             = 1;
   AFP_MIN_SUBROW_DEPTH_ARRAY (1) = AFP_DEFAULT * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,1) = FIELDS1;
   AFP_COLUMN_WIDTH_ARRAY     (1) = 95.0 * 10000;
   AFP_ROW_ARRANGE_ARRAY    (1,2) = FIELDS2;
   AFP_COLUMN_WIDTH_ARRAY     (2) = 30.0 * 10000;

   CALL DO_AFPDROW;

   ROW3                            = AFP_ROW_ID;

END SETUP_AFPAPI;

%PAGE;
/*************************************************************/
/*                                                           */
/*    READ SAMPLE DATA                                       */
/*                                                           */
/*************************************************************/

READ_DATA:  PROC;

   READ FILE(DATAIN) INTO(TRAN_IN);

   ON ENDFILE (DATAIN)
       DATA_REMAINS = NO;

/*                                                           */
/* If this is a new customer, read the customer address and  */
/* account balance.                                          */
/*                                                           */

   IF POST_DATE_IN = 0 THEN
     DO;
       NEW_CUSTOMER        = YES;

       ACCT_1              = SUBSTR(TRANS_DESC,1,4);
       ACCT_2              = SUBSTR(TRANS_DESC,5,4);
       ACCT_3              = SUBSTR(TRANS_DESC,9,4);
       ACCT_4              = SUBSTR(TRANS_DESC,13,4);

/*                                                           */
/*   Read the customer name                                  */
/*                                                           */

       READ FILE(DATAIN) INTO(TRAN_IN);

       ON ENDFILE (DATAIN)
         DATA_REMAINS = NO;

       CUST_NAME           = TRANS_DESC;

/*                                                           */
/*   Read the customer street addres                         */
/*                                                           */

       READ FILE(DATAIN) INTO(TRAN_IN);

       ON ENDFILE (DATAIN)
         DATA_REMAINS = NO;

       CUST_ST_ADDR        = TRANS_DESC;

/*                                                           */
/*   Read the customer city and state                        */
/*                                                           */

       READ FILE(DATAIN) INTO(TRAN_IN);

       ON ENDFILE (DATAIN)
         DATA_REMAINS = NO;

       CUST_CITY_STATE     = TRANS_DESC;

/*                                                           */
/*   Read the customer balance                               */
/*                                                           */

       READ FILE(DATAIN) INTO(TRAN_IN);

       ON ENDFILE (DATAIN)
         DATA_REMAINS = NO;

       CUSTOMER_BALANCE_IN = TRANSACTION_AMOUNT_IN;

/*                                                           */
/*   Read the first customer transaction                     */
/*                                                           */

       READ FILE(DATAIN) INTO(TRAN_IN);

       ON ENDFILE (DATAIN)
         DATA_REMAINS = NO;
     END;

END READ_DATA;

%PAGE;
/*************************************************************/
/*   PROCESS THE CUSTOMER                                    */
/*                                                           */
/*     Begin a page.                                         */
/*     Write the page header.                                */
/*     Write the paragraph.                                  */
/*     Process the customer transactions.                    */
/*     Write the page footer.                                */
/*     End the page.                                         */
/*                                                           */
/*************************************************************/

PROCESS_A_CUSTOMER:  PROC;

   CUSTOMER_BALANCE_OUT      = CUSTOMER_BALANCE_IN;

/*                                                           */
/* Initialize this customer's number of transactions to 0    */
/*                                                           */

   NUM_CUSTOMER_PAGES        = 1;
   AFP_PAGE_WIDTH            = AFP_DEFAULT * 10000;
   AFP_PAGE_DEPTH            = AFP_DEFAULT * 10000;
   AFP_PAGE_ORIENTATION      = ORIENTDOC;

   CALL DO_AFPBPAG;

   CALL CREATE_THE_HEADER;

/*                                                           */
/* Calculate the page body size as the page size less the    */
/* page header and bottom margin.                            */
/*                                                           */

GE_HEADER_DEPTH = PAGE_HEADER_DEPTH * 10000;

GE_BODY = AFP_DOC_PAGE_DEPTH - PAGE_HEADER_DEPTH - BOTTOM_MARGIN;

  CALL PROCESS_THE_PARAGRAPH;

/*                                                           */
/* Calculate the remaining page body after the paragraph.    */
/*                                                           */

  PAGE_BODY = PAGE_BODY - AFP_PARAGRAPH_DEPTH;
```

```
  CALL PROCESS_TRANSACTIONS;

  CALL CREATE_THE_FOOTER;

  CALL DO_AFPEPAG;

END PROCESS_A_CUSTOMER;

%PAGE;
/*************************************************************/
/*                                                           */
/*    CREATE THE HEADER.                                     */
/*                                                           */
/*************************************************************/

CREATE_THE_HEADER:  PROC;

  CALL PROCESS_THE_AREA;

/*                                                           */
/* Include the Page Segment                                  */
/*                                                           */

  AFP_CURRENT_HANDLE          = AFP_PAGE_HANDLE;
  AFP_X_COORDINATE            = 29 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = 23 * 10000;
  AFP_Y_REF_COORD_SYS         = YABS;

  CALL DO_AFPSPOS;

  AFP_PSEG_NAME               = 'APQPSEG';
  AFP_INLINE_OPTION           = TRU;
  AFP_REUSE_OPTION            = FALS;

  CALL DO_AFPIPSG;

  CALL PROCESS_THE_ADDRESS;

/*                                                           */
/* Draw a rule underneath the address                        */
/*                                                           */

  AFP_RULE_THICKNESS          = 1.5 * 10000;

  CALL DO_AFPSRTH;

  AFP_X_COORDINATE            = 29 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = 73 * 10000;
  AFP_Y_REF_COORD_SYS         = YABS;

  CALL DO_AFPSPOS;

  AFP_DIRECTION               = XDIRECTION;
  AFP_RULE_LENGTH             = 158 * 10000;

  CALL DO_AFPPRUL;

/*                                                           */
/* Leave space after the rule                                */
/*                                                           */

  AFP_Y_COORDINATE            = 4 * 10000;
  AFP_Y_REF_COORD_SYS         = YREL;

  CALL DO_AFPSPOS;

/*                                                           */
/* Query the position and calculate the page header depth    */
/*                                                           */

  CALL DO_AFPQPOS;

END CREATE_THE_HEADER;

%PAGE;
/*************************************************************/
/*                                                           */
/*    PROCESS THE AREA                                       */
/*                                                           */
/*************************************************************/

PROCESS_THE_AREA:  PROC;

  AFP_CURRENT_HANDLE          = AFP_PAGE_HANDLE;
  AFP_AREA_WIDTH              = 50.0 * 10000;
  AFP_MAX_AREA_DEPTH          = 65.0 * 10000;
  AFP_SHADING_PATTERN         = NOSHADE;
  AFP_SHADING_INTENSITY       = 0;

  CALL DO_AFPCARE;

  AFP_CURRENT_HANDLE          = AFP_AREA_HANDLE;

/*                                                           */
/* Include the Page Overlay                                  */
/*                                                           */

  AFP_OVLY_NAME               = 'O1APQL2';

  CALL DO_AFPIOVL;

/*                                                           */
/* Write the account number                                  */
/*                                                           */

  AFP_FONT_ID                 = TIM10MED;

  CALL DO_AFPSFNT;

  AFP_CURRENT_HANDLE          = AFP_AREA_HANDLE;
  AFP_X_COORDINATE            = 49 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = 7 * 10000;
  AFP_Y_REF_COORD_SYS         = YABS;

  CALL DO_AFPSPOS;

  STRING_FIELD =
     ACCT_1 || BLK || ACCT_2 || BLK || ACCT_3 || BLK || ACCT_4;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_OPTION        = R_GHT;
  AFP_POSITION_OPTION         = FALS;
  AFP_UNDERLINE               = FALS;

  CALL DO_AFPPCHS;

/*                                                           */
/* Write the due date                                        */
/*                                                           */

  AFP_X_COORDINATE            = 49 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = 12 * 10000;
  AFP_Y_REF_COORD_SYS         = YABS;

  CALL DO_AFPSPOS;

  STRING_FIELD                = DUE_DATE;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_OPTION        = R_GHT;
  AFP_POSITION_OPTION         = FALS;
  AFP_UNDERLINE               = FALS;

  CALL DO_AFPPCHS;

/*                                                           */
/* Write the customer balance                                */
/*                                                           */

  AFP_X_COORDINATE            = 49 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = 19 * 10000;
  AFP_Y_REF_COORD_SYS         = YABS;

  CALL DO_AFPSPOS;

  CUSTOMER_BALANCE_OUT        = CUSTOMER_BALANCE_IN;
  STRING_FIELD                = CUSTOMER_BALANCE_OUT;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_OPTION        = R_GHT;
  AFP_POSITION_OPTION         = FALS;
```

```
  AFP_UNDERLINE              = FALS;

  CALL DO_AFPPCHS;

/*                                                              */
/* Write the customer payment                                   */
/*                                                              */

  AFP_X_COORDINATE           = 49 * 10000;
  AFP_X_REF_COORD_SYS        = XABS;
  AFP_Y_COORDINATE           = 24 * 10000;
  AFP_Y_REF_COORD_SYS        = YABS;

  CALL DO_AFPSPOS;

  MIN_AMOUNT_DUE_OUT         = CUSTOMER_BALANCE_IN * .1;
  STRING_FIELD               = MIN_AMOUNT_DUE_OUT;
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;
  AFP_ALIGNMENT_OPTION       = R_GHT;
  AFP_POSITION_OPTION        = FALS;
  AFP_UNDERLINE              = FALS;

  CALL DO_AFPPCHS;

  CALL DO_AFPEARE;

/*                                                              */
/* Place the area on the page                                   */
/*                                                              */

  AFP_CURRENT_HANDLE         = AFP_PAGE_HANDLE;
  AFP_X_COORDINATE           = 137 * 10000;
  AFP_X_REF_COORD_SYS        = XABS;
  AFP_Y_COORDINATE           = 23 * 10000;
  AFP_Y_REF_COORD_SYS        = YABS;

  CALL DO_AFPSPOS;

  AFP_AREA_ROTATION          = ORIENT0;

  CALL DO_AFPPARE;

/*                                                              */
/* Destroy the area from AFP API storage                        */
/*                                                              */

  CALL DO_AFPXARE;

END PROCESS_THE_AREA;

%PAGE;
/*************************************************************/
/*                                                           */
/*    PROCESS THE ADDRESS.                                   */
/*                                                           */
/*************************************************************/

PROCESS_THE_ADDRESS:  PROC;

/*                                                              */
/* Write the customer name                                      */
/*                                                              */

  AFP_FONT_ID                = TIM12BOLD;

  CALL DO_AFPSFNT;

  AFP_X_COORDINATE           = 29 * 10000;
  AFP_X_REF_COORD_SYS        = XABS;
  AFP_Y_COORDINATE           = 56 * 10000;
  AFP_Y_REF_COORD_SYS        = YABS;

  CALL DO_AFPSPOS;

  AFP_STRING_LENGTH          = INDEX(CUST_NAME,TEST_VALUE);
  AFP_STRING_LENGTH          = AFP_STRING_LENGTH - 1;
  CUST_LENGTH                = LENGTH(CUST_NAME);
  AFP_CHARACTER_STRING       = CUST_NAME;
  AFP_ALIGNMENT_OPTION       = L_FT;
  AFP_POSITION_OPTION        = FALS;
  AFP_UNDERLINE              = FALS;

  CALL DO_AFPPCHS;

/*                                                              */
/* Write the customer address                                   */
/*                                                              */

  AFP_FONT_ID                = TIM10MED;

  CALL DO_AFPSFNT;

  AFP_Y_COORDINATE           = 1 * 10000;
  AFP_Y_REF_COORD_SYS        = YLINES;

  CALL DO_AFPSPOS;

  AFP_STRING_LENGTH          = LENGTH(CUST_ST_ADDR);
  AFP_CHARACTER_STRING       = CUST_ST_ADDR;
  AFP_ALIGNMENT_OPTION       = L_FT;
  AFP_POSITION_OPTION        = FALS;
  AFP_UNDERLINE              = FALS;

  CALL DO_AFPPCHS;

  AFP_Y_COORDINATE           = 1 * 10000;
  AFP_Y_REF_COORD_SYS        = YLINES;

  CALL DO_AFPSPOS;

  AFP_STRING_LENGTH          = LENGTH(CUST_CITY_STATE);
  AFP_CHARACTER_STRING       = CUST_CITY_STATE;
  AFP_ALIGNMENT_OPTION       = L_FT;
  AFP_POSITION_OPTION        = FALS;
  AFP_UNDERLINE              = FALS;

  CALL DO_AFPPCHS;

END PROCESS_THE_ADDRESS;

%PAGE;
/*************************************************************/
/*                                                           */
/*    PROCESS THE PARAGRAPH.                                 */
/*                                                           */
/*************************************************************/

PROCESS_THE_PARAGRAPH:  PROC;

  AFP_CURRENT_HANDLE         = AFP_PAGE_HANDLE;
  AFP_X_COORDINATE           = 29 * 10000;
  AFP_X_REF_COORD_SYS        = XABS;
  AFP_Y_COORDINATE           = PARAGRAPH_WHITE_SPACE * 10000;
  AFP_Y_REF_COORD_SYS        = YREL;

  CALL DO_AFPSPOS;

  AFP_RULE_THICKNESS         = 0.5 * 10000;

  CALL DO_AFPSRTH;

  AFP_FIRST_LINE_INDENT      = 0 * 10000;
  AFP_FORMAT_OPTION          = FOJUSTIFY;
  AFP_FIRST_LINE_OFFSET      = AFP_DEFAULT * 10000;
  AFP_LEFT_MARGIN            = 10.0 * 10000;
  AFP_LINE_LENGTH            = 135.0 * 10000;
  AFP_LINE_SPACING           = AFP_DEFAULT * 10000;
  AFP_PARAGRAPH_FRAME        = TRU;
  AFP_RT_RULE_OFFSET         = 158.0 * 10000;
  AFP_BOT_RULE_OFFSET        = 0.0 * 10000;
  AFP_SHADING_PATTERN        = NOSHADE;
  AFP_SHADING_INTENSITY      = 0;

  CALL DO_AFPBPAR;

  AFP_CURRENT_HANDLE         = AFP_PARAGRAPH_HANDLE;
  AFP_FONT_ID                = TIM12BOLD;

  CALL DO_AFPSFNT;

  STRING_FIELD               = 'CONGRATULATIONS, ';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;
  AFP_UNDERLINE              = FALS;
  AFP_CONCATENATE            = TRU;
```

```
  CALL DO_AFPPTXT;

  AFP_FONT_ID                 = TIM12MED;

  CALL DO_AFPSFNT;

  AFP_STRING_LENGTH           = INDEX(CUST_NAME,TEST_VALUE);
  AFP_STRING_LENGTH           = AFP_STRING_LENGTH - 1;
  AFP_CHARACTER_STRING        = CUST_NAME;
  AFP_UNDERLINE               = FALS;

  CALL DO_AFPPTXT;

  STRING_1 = '  Because of your excellent credit rating, you ';
  STRING_2 = 'are now eligible for free credit insurance which ';

  STRING_FIELD = EXCLAMATION || STRING_1 || STRING_2;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_UNDERLINE               = FALS;

  CALL DO_AFPPTXT;

  STRING_1 = 'protects you in case your Primo card is ever ';
  STRING_2 = 'lost or stolen.  ';

  STRING_FIELD                = STRING_1 || STRING_2;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_UNDERLINE               = FALS;

  CALL DO_AFPPTXT;

  STRING_FIELD = 'Call NOW for more information' || EXCLAMATION;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_UNDERLINE               = TRU;

  CALL DO_AFPPTXT;

  CALL DO_AFPEPAR;

/*                                                              */
/* Calculate the amount of space taken up by the paragraph      */
/*                                                              */

  PARAGRAPH_WHITE_SPACE = PARAGRAPH_WHITE_SPACE * 10000;

  AFP_PARAGRAPH_DEPTH =
      AFP_PARAGRAPH_DEPTH + PARAGRAPH_WHITE_SPACE;

  AFP_CURRENT_HANDLE          = AFP_PAGE_HANDLE;

END PROCESS_THE_PARAGRAPH;

%PAGE;
/*************************************************************/
/*  PROCESS TRANSACTIONS.                                    */
/*                                                           */
/*      Begin a table.                                       */
/*      Write the header rows.                               */
/*      Write a transaction row until no more data for this  */
/*         customer or no more data.                         */
/*      Write the summary row.                               */
/*      End the table.                                       */
/*      Compute the table depth.                             */
/*************************************************************/

PROCESS_TRANSACTIONS:  PROC;

  NEW_CUSTOMER                = NO;

/*                                                              */
/* Start the table whose maximum depth is the remaining page    */
/* body space after the white space preceeding the table.       */
/*                                                              */

  AFP_X_COORDINATE            = 45 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = TABLE_WHITE_SPACE;
  AFP_Y_REF_COORD_SYS         = YREL;

  CALL DO_AFPSPOS;

  AFP_MAX_TABLE_DEPTH         = TABLE_DEPTH * 10000;
  AFP_TABLE_WIDTH             = 125.0 * 10000;
  AFP_TABLE_ROTATION          = ORIENT0;
  AFP_TOP_THICKNESS           = 1.0 * 10000;
  AFP_BOTTOM_THICKNESS        = 0.5 * 10000;
  AFP_LEFT_THICKNESS          = 0.5 * 10000;
  AFP_RIGHT_THICKNESS         = 0.5 * 10000;

  CALL DO_AFPBTBL;

  AFP_CURRENT_HANDLE          = AFP_TABLE_HANDLE;

/*                                                              */
/* Write the table header rows                                  */
/*                                                              */

  CALL WRITE_HEADER_ROWS;

/*                                                              */
/* Write the transaction rows for this customer                 */
/*                                                              */

  IF NEW_CUSTOMER = NO THEN
    DO WHILE(NEW_CUSTOMER = NO);
      CALL WRITE_TRANSACTIONS;
    END;

/*                                                              */
/* Write the table summary rows                                 */
/*                                                              */

  CALL WRITE_SUMMARY_ROW;

/*                                                              */
/* If the end of the table was reached, end this page, start    */
/* a new page, and write the summary row that didn't fit.       */
/*                                                              */

  IF AFP_SEVERITY_CODE = WARNING THEN
    DO;
      CALL END_CUST_PAGE;
      CALL WRITE_SUMMARY_ROW;
    END;

/*                                                              */
/* End the table                                                */
/*                                                              */

  CALL DO_AFPETBL;

  AFP_CURRENT_HANDLE          = AFP_PAGE_HANDLE;

/*                                                              */
/* Calculate the amount of space taken up by the table          */
/*                                                              */

  AFP_TABLE_DEPTH = AFP_TABLE_DEPTH + TABLE_WHITE_SPACE;

END PROCESS_TRANSACTIONS;

%PAGE;
/*************************************************************/
/*    WRITE_HEADER_ROWS.                                     */
/*                                                           */
/*            WRITE THE HEADER ROW FOR THE TABLE.            */
/*************************************************************/

WRITE_HEADER_ROWS:  PROC;

  AFP_ROW_ID                  = ROW1;

  CALL DO_AFPBROW;

/*                                                              */
/* Write the first field in column 1                            */
/*                                                              */

  AFP_FIELD_ID                = FIELDH1;

  CALL DO_AFPBFLD;
```

```
  AFP_FONT_ID                = TIM12MED;

  CALL DO_AFPSFNT;

  STRING_FIELD               = 'Date';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;
  AFP_ALIGNMENT_OPTION       = CENTER;
  AFP_UNDERLINE              = FALS;

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

/*                                                          */
/* Write the 2nd field in column 2                          */
/*                                                          */

  AFP_FIELD_ID               = FIELDH2;

  CALL DO_AFPBFLD;

  STRING_FIELD               = 'Transaction Description';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

/*                                                          */
/* Write the 3rd field in column 3                          */
/*                                                          */

  AFP_FIELD_ID               = FIELDH3;

  CALL DO_AFPBFLD;

  STRING_FIELD               = 'Amount';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

  CALL DO_AFPEROW;

END WRITE_HEADER_ROWS;

%PAGE;
/************************************************************/
/*    WRITE_TRANSACTIONS.                                   */
/*                                                          */
/*            Write a transaction row.                      */
/*            Read another input data record.               */
/************************************************************/

WRITE_TRANSACTIONS:  PROC;

  IF DATA_REMAINS THEN
     CALL WRITE_TRANSACTION_ROW;

/* If the end of the table was reached, end this page, start  */
/* a new page, and write the transaction row that didn't fit. */

  IF AFP_SEVERITY_CODE = WARNING THEN
    DO;
      CALL END_CUST_PAGE;

      CALL WRITE_TRANSACTION_ROW;
    END;

  IF DATA_REMAINS THEN
    CALL READ_DATA;
  ELSE
    NEW_CUSTOMER = YES;

END WRITE_TRANSACTIONS;

%PAGE;
/************************************************************/
/* WRITE_TRANSACTION_ROW.                                   */
/*                                                          */
/*     Begin a row.                                         */
/*     Write the post date to the first field in the table. */
/*     Write the transaction description to the 2nd field.  */
/*     Write the transaction amount to the 3rd field.       */
/*     End the row.                                         */
/************************************************************/

WRITE_TRANSACTION_ROW:  PROC;

  AFP_ROW_ID                 = ROW2;

  CALL DO_AFPBROW;

/*                                                          */
/* Write the transaction date in column 1                   */
/*                                                          */

  AFP_FIELD_ID               = FIELDT1;

  CALL DO_AFPBFLD;

  AFP_FONT_ID                = TIM10MED;

  CALL DO_AFPSFNT;

  POST_DATE_OUT              = POST_DATE_IN;

  STRING_FIELD               = POST_DATE_OUT;
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;
  AFP_ALIGNMENT_OPTION       = CENTER;

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

/*                                                          */
/* Write the transaction description in column 2            */
/*                                                          */

  AFP_FIELD_ID               = FIELDT2;

  CALL DO_AFPBFLD;

  STRING_FIELD               = TRANS_DESC;
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;
  AFP_ALIGNMENT_OPTION       = L_FT;
  AFP_POSITION_OPTION        = FALS;

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

/*                                                          */
/* Write the transaction amount in column 3                 */
/*                                                          */

  AFP_FIELD_ID               = FIELDT3;

  CALL DO_AFPBFLD;

  TRANSACTION_AMOUNT_OUT     = TRANSACTION_AMOUNT_IN;
  STRING_FIELD               = TRANSACTION_AMOUNT_OUT;
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;
  AFP_ALIGNMENT_OPTION       = CHAR;
  AFP_ALIGNMENT_CHAR         = '.';

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

  CALL DO_AFPEROW;

END WRITE_TRANSACTION_ROW;

%PAGE;
/************************************************************/
/*    WRITE_SUMMARY_ROW.                                    */
/*                                                          */
```

```
/*           Write the customer summary row.                */
/*************************************************************/

WRITE_SUMMARY_ROW:  PROC;

  AFP_ROW_ID                  = ROW3;

  CALL DO_AFPBROW;

  AFP_FIELD_ID                = FIELDS1;

  CALL DO_AFPBFLD;

  AFP_FONT_ID                 = TIM12MED;

  CALL DO_AFPSFNT;

  STRING_FIELD                = 'Total Amount';
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_OPTION        = CENTER;

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

  AFP_FIELD_ID                = FIELDS2;

  CALL DO_AFPBFLD;

  STRING_FIELD                = CUSTOMER_BALANCE_OUT;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_OPTION        = CHAR;

  CALL DO_AFPPCHS;

  CALL DO_AFPEFLD;

  CALL DO_AFPEROW;

END WRITE_SUMMARY_ROW;

%PAGE;
/*************************************************************/
/*    CREATE_THE_FOOTER.                                     */
/*                                                           */
/*            WRITE THE PAGE FOOTER.                         */
/*************************************************************/

CREATE_THE_FOOTER:  PROC;

  AFP_X_COORDINATE            = 108 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = 270 * 10000;
  AFP_Y_REF_COORD_SYS         = YABS;

  CALL DO_AFPSPOS;

  AFP_FONT_ID                 = TIM10MED;

  CALL DO_AFPSFNT;

  STRING_FIELD                = 'Page ';
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_OPTION        = CENTER;
  AFP_POSITION_OPTION         = FALS;

  CALL DO_AFPPCHS;

  NUM_CUSTOMER_PAGES_OUT      = NUM_CUSTOMER_PAGES;
  STRING_FIELD                = NUM_CUSTOMER_PAGES_OUT;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_OPTION        = L_FT;
  AFP_POSITION_OPTION         = FALS;

  CALL DO_AFPPCHS;

END CREATE_THE_FOOTER;

%PAGE;
/*************************************************************/
/*    END_CUST_PAGE.                                         */
/*                                                           */
/*            End the table.                                 */
/*            Write the page footer.                         */
/*            End the page.                                  */
/*            Start a new page.                              */
/*            Begin a table.                                 */
/*            Write the header rows.                         */
/*************************************************************/

END_CUST_PAGE:  PROC;

  CALL DO_AFPETBL;

  AFP_CURRENT_HANDLE          = AFP_PAGE_HANDLE;

/*                                                           */
/*            Write the footer                               */
/*                                                           */

  CALL CREATE_THE_FOOTER;

  CALL DO_AFPEPAG;

/*                                                           */
/*            Start a new page                               */
/*                                                           */

  NUM_CUSTOMER_PAGES          = NUM_CUSTOMER_PAGES + 1;
  AFP_PAGE_WIDTH              = AFP_DEFAULT * 10000;
  AFP_PAGE_DEPTH              = AFP_DEFAULT * 10000;
  AFP_PAGE_ORIENTATION        = ORIENTDOC;

  CALL DO_AFPBPAG;

  AFP_CURRENT_HANDLE          = AFP_PAGE_HANDLE;

  CALL CREATE_THE_CONT_HEADER;

/*                                                           */
/* Calculate the page body space as the page size less the   */
/* continuation page header size and bottom margin.          */
/*                                                           */

GE_HEADER_DEPTH = PAGE_HEADER_DEPTH * 10000;

GE_BODY = AFP_DOC_PAGE_DEPTH - PAGE_HEADER_DEPTH - BOTTOM_MARGIN;

/*                                                           */
/*            begin a table                                  */
/*                                                           */

  AFP_X_COORDINATE            = 45 * 10000;
  AFP_X_REF_COORD_SYS         = XABS;
  AFP_Y_COORDINATE            = TABLE_WHITE_SPACE;
  AFP_Y_REF_COORD_SYS         = YREL;

  CALL DO_AFPSPOS;

  AFP_MAX_TABLE_DEPTH         = 196 * 10000;
  AFP_TABLE_WIDTH             = 125.0 * 10000;
  AFP_TABLE_ROTATION          = ROTATE0;
  AFP_TOP_THICKNESS           = 1.0 * 10000;
  AFP_BOTTOM_THICKNESS        = 0.5 * 10000;
  AFP_LEFT_THICKNESS          = 0.5 * 10000;
  AFP_RIGHT_THICKNESS         = 0.5 * 10000;

  CALL DO_AFPBTBL;

  AFP_CURRENT_HANDLE          = AFP_TABLE_HANDLE;

/*                                                           */
/*            write the header rows                          */
/*                                                           */

  CALL WRITE_HEADER_ROWS;

END END_CUST_PAGE;

%PAGE;
/*************************************************************/
```

```
/*    CREATE THE HEADER FOR THE CONTINUATION PAGES           */
/*************************************************************/

CREATE_THE_CONT_HEADER:  PROC;

/*                                                           */
/*  Include the Page Segment                                 */
/*                                                           */

  PAGE_BODY                  = 260 * 10000;
  AFP_X_COORDINATE           = 29 * 10000;
  AFP_X_REF_COORD_SYS        = XABS;
  AFP_Y_COORDINATE           = 23 * 10000;
  AFP_Y_REF_COORD_SYS        = YABS;

  CALL DO_AFPSPOS;

  AFP_PSEG_NAME              = 'APQPSEG';
  AFP_INLINE_OPTION          = TRU;
  AFP_REUSE_OPTION           = FALS;

  CALL DO_AFPIPSG;

/*                                                           */
/*  Draw a rule underneath the Page Segment                  */
/*                                                           */

  AFP_RULE_THICKNESS         = 1.5 * 10000;

  CALL DO_AFPSRTH;

  AFP_X_COORDINATE           = 29 * 10000;
  AFP_X_REF_COORD_SYS        = XABS;
  AFP_Y_COORDINATE           = 50 * 10000;
  AFP_Y_REF_COORD_SYS        = YABS;

  CALL DO_AFPSPOS;

  AFP_DIRECTION              = XDIRECTION;
  AFP_RULE_LENGTH            = 158 * 10000;

  CALL DO_AFPPRUL;

/*                                                           */
/*  Leave space after the rule                               */
/*                                                           */

  AFP_Y_COORDINATE           = 4 * 10000;
  AFP_Y_REF_COORD_SYS        = YREL;

  CALL DO_AFPSPOS;

/*                                                           */
/*  Query the position and calculate the page header depth   */
/*                                                           */

  CALL DO_AFPQPOS;

END CREATE_THE_CONT_HEADER;

%PAGE;
/*************************************************************/
/*                                                           */
/*    TERMINATE THE AFPAPI AND THE PROGRAM                   */
/*                                                           */
/*************************************************************/

END_PROCESSING:  PROC;

 CALL DO_AFPEDOC;

 CALL DO_AFPEND;

 CLOSE FILE(DATAIN);

END END_PROCESSING;

%PAGE;
%INCLUDE DDSYS (APQPPRF);

END APQPSMP;
```

APQPSMP2

```
/*************************************************************/
/*   PL/I PROGRAM -- AFPAPI APQPSM2                          */
/*                                                           */
/* This PL/I program invokes the AFP API to produce a sample  */
/* customer billing statement.  See the "AFP API Programming  */
/* Guide and Reference" for a picture of the print output     */
/* produced by this program.                                 */
/* Note: This program invokes the API directly instead of     */
/*       calling the subroutines provided in the APQPPRF     */
/*       copy book.                                          */
/*************************************************************/

APQPSM2: PROCEDURE OPTIONS (MAIN);

DCL DATAIN FILE INPUT ENV(FB BLKSIZE(32720) RECSIZE(80));

DCL 1 TRAN_IN,
    2 POST_DATE_IN               PIC '(4)9',
    2 TRANS_DESC                 CHAR(40),
    2 TRANSACTION_AMOUNT_IN      PIC '(7)9V99',
    2 TRAN_FILLER                CHAR(27);

DCL CUST_IN;
  DCL CUST_NAME                  CHAR(30) VARYING;
  DCL CUST_ST_ADDR               CHAR(30) VARYING;
  DCL CUST_CITY_STATE            CHAR(35) VARYING;
  DCL CUSTOMER_BALANCE_IN        PIC '(5)9V99';

DCL 1 ACCOUNT_NUM,
    2 ACCT_1                     CHAR (4),
    2 ACCT_2                     CHAR (4),
    2 ACCT_3                     CHAR (4),
    2 ACCT_4                     CHAR (4);

DCL TRANSACTION_PROCESSING_VARS;
  DCL PAGE_HEADER_DEPTH          FIXED BIN(31)   INIT(0);
  DCL PAGE_BODY                  FIXED BIN(31)   INIT(0);
  DCL BOTTOM_MARGIN              FIXED BIN(31)   INIT(200000);
  DCL TABLE_WHITE_SPACE          FIXED BIN(31)   INIT(100000);
  DCL PARAGRAPH_WHITE_SPACE      FIXED BIN(31)   INIT(0);
  DCL NUM_CUSTOMER_PAGES         FIXED DEC(2)    INIT(1);
  DCL RECORD_COUNT               FIXED DEC(2)    INIT(0);
  DCL TEST_VALUE                 CHAR (2)        INIT('  ');
  DCL BLK                        CHAR (1)        INIT(' ');
  DCL CUST_LENGTH                FIXED DEC(3)    INIT(0);
  DCL TABLE_DEPTH                FIXED DEC(9,4)  INIT(156.6544);
  DCL DUE_DATE                   CHAR (12) INIT('OCT 15, 1992');
  DCL STRING_FIELD               CHAR (160) VARYING;
  DCL STRING_1                   CHAR (80) VARYING;
  DCL STRING_2                   CHAR (80) VARYING;

DCL CUST_OUT;
  DCL OLD_BALANCE_OUT            PIC '$,$$$,$$$V.99';
  DCL MIN_AMOUNT_DUE_OUT         PIC '$$$,$$$V.99';
  DCL POST_DATE_OUT              PIC '99/99';
  DCL TRANSACTION_DATE_OUT       PIC '99/99';
  DCL TRANSACTION_AMOUNT_OUT     PIC '$,$$$,$$$V.99';
  DCL CUSTOMER_BALANCE_OUT       PIC '$,$$$,$$$V.99';
  DCL EXCLAMATION                CHAR(1) INIT ('4F'X);
  DCL NUM_CUSTOMER_PAGES_OUT     PIC 'Z9';

DCL PROCESSING_SWITCHES;
  DCL DATA_REMAINS               BIT (1) INIT ('1'B);
  DCL NEW_CUSTOMER               BIT (1) INIT ('0'B);
  DCL YES                        BIT (1) INIT ('1'B);
  DCL NO                         BIT (1) INIT ('0'B);
  DCL MIN_AMOUNT_DUE_COMP        FIXED DEC(7,2);

DCL FONT_IDS;
  DCL TIM10MED                   CHAR(9);
  DCL TIM12MED                   CHAR(9);
  DCL TIM12MEDITAL               CHAR(9);
  DCL TIM12BOLD                  CHAR(9);

DCL FIELD_IDS;
  DCL FIELDH1                    FIXED BIN(31);
  DCL FIELDH2                    FIXED BIN(31);
  DCL FIELDH3                    FIXED BIN(31);
  DCL FIELDT1                    FIXED BIN(31);
  DCL FIELDT2                    FIXED BIN(31);
  DCL FIELDT3                    FIXED BIN(31);
  DCL FIELDS1                    FIXED BIN(31);
  DCL FIELDS2                    FIXED BIN(31);

DCL ROW_IDS;
  DCL ROW1                       CHAR(9);
  DCL ROW2                       CHAR(9);
  DCL ROW3                       CHAR(9);

DCL AFP_ROW_ARRANGE_ARRAY (1,3) FIXED BIN (31) INIT ((3)0);

DCL AFP_ROW_ARRANGE_ARRAY2 (2,3) FIXED BIN (31) INIT ((6)0);

%PAGE;
%INCLUDE DDSYS (APQPCON);
%PAGE;
%INCLUDE DDSYS (APQPVAR);

%PAGE;
/*************************************************************/
/*                                                           */
/* This program produces a sample customer statement.  The   */
/* statement contains a logo stored in a page segment, a     */
/* paragraph enclosed in a box with variable data, account   */
/* summary information which is highlighted in a shaded area, */
/* and the account transactions contained in a variable depth */
/* table.                                                    */
/*                                                           */
/* The mainline logic is as follows:                         */
/*                                                           */
/*   Initialize the API and define the fonts, fields, and    */
/*     rows.                                                 */
/*   Load the sample data for the first customer.            */
/*   Process a customer until no more customers to process.  */
/*   End the API.                                            */
/*************************************************************/

  ON ENDFILE (DATAIN)
      DATA_REMAINS = NO;

  STRING_ARRAY = ' ';

  OPEN FILE (DATAIN) RECORD SEQUENTIAL;

  CALL SETUP_AFPAPI;

  CALL READ_DATA;

  DO WHILE (DATA_REMAINS);
    CALL PROCESS_A_CUSTOMER;
  END;

  CALL END_PROCESSING;

  PUT SKIP LIST(' ');
  PUT SKIP LIST('APQPSM2 COMPLETED');
```

```
%PAGE;
/*************************************************************/
/*    SETUP_AFPAPI.                                          */
/*        Initialize the AFP API.                            */
/*        Begin a document.                                  */
/*        Set the output characteristics.                    */
/*        Define the fonts.                                  */
/*        Define the fields and rows of a table.             */
/*************************************************************/

SETUP_AFPAPI:  PROC;

  CALL AFPINIT (AFPAPI_HANDLE,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPINIT';
  CALL CHKSUCC;

/*                                                           */
/*    SET THE OUTPUT CHARACTERISTICS                         */
/*                                                           */

  AFP_OUTPUT_RECORD_SIZE = 8205;
  AFP_OUTPUT_FILENAME    = 'APQPSMP2';

/* the following two fields are not used in MVS */

  AFP_OUTPUT_FILETYPE    = 'LISTAFP';
  AFP_OUTPUT_FILEMODE    = 'A1';

  CALL AFPSOUT (AFPAPI_HANDLE,
                AFP_OUTPUT_RECORD_SIZE,
                AFP_OUTPUT_FILENAME,
                AFP_OUTPUT_FILETYPE,
                AFP_OUTPUT_FILEMODE,
                TRU,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSOUT';
  CALL CHKSUCC;

/*                                                           */
/*    Begin the document                                     */
/*                                                           */

  AFP_DOC_PAGE_WIDTH     = 215.0 * 10000;
  AFP_DOC_PAGE_DEPTH     = 280.0 * 10000;

  CALL AFPBDOC (AFPAPI_HANDLE,
                MM,
                AFP_DOC_PAGE_WIDTH,
                AFP_DOC_PAGE_DEPTH,
                ORIENTO,
                AFP_DOCUMENT_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA  = 'AFPBDOC';
  CALL CHKSUCC;

/*                                                           */
/*    DEFINE THE FONTS                                       */
/*                                                           */

  AFP_CODE_PAGE        = 'T1V10500';
  AFP_DESC_NAME_LENGTH = 22;
  AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
  AFP_POINT_SIZE       = 10;

  CALL AFPDFNT (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_CODE_PAGE,
                AFP_DESC_NAME_LENGTH,
                AFP_DESCRIPTIVE_NAME,
                AFP_POINT_SIZE,
                MEDIUM,
                NORMAL,
                ORIENTO,
                ROMAN,
                AFP_FONT_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDFNT';
  CALL CHKSUCC;

  TIM10MED             = AFP_FONT_ID;
  AFP_CODE_PAGE        = 'T1V10500';
  AFP_DESC_NAME_LENGTH = 22;
  AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
  AFP_POINT_SIZE       = 12;

  CALL AFPDFNT (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_CODE_PAGE,
                AFP_DESC_NAME_LENGTH,
                AFP_DESCRIPTIVE_NAME,
                AFP_POINT_SIZE,
                MEDIUM,
                NORMAL,
                ORIENTO,
                ROMAN,
                AFP_FONT_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDFNT';
  CALL CHKSUCC;

  TIM12MED             = AFP_FONT_ID;
  AFP_CODE_PAGE        = 'T1V10500';
  AFP_DESC_NAME_LENGTH = 22;
  AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
  AFP_POINT_SIZE       = 12;

  CALL AFPDFNT (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_CODE_PAGE,
                AFP_DESC_NAME_LENGTH,
                AFP_DESCRIPTIVE_NAME,
                AFP_POINT_SIZE,
                MEDIUM,
                NORMAL,
                ORIENTO,
                ROMAN,
                ITALIC,
                AFP_FONT_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDFNT';
  CALL CHKSUCC;

  TIM12MEDITAL         = AFP_FONT_ID;
  AFP_CODE_PAGE        = 'T1V10500';
  AFP_DESC_NAME_LENGTH = 22;
  AFP_DESCRIPTIVE_NAME = 'TIMES NEW ROMAN LATIN1';
  AFP_POINT_SIZE       = 12;

  CALL AFPDFNT (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_CODE_PAGE,
                AFP_DESC_NAME_LENGTH,
                AFP_DESCRIPTIVE_NAME,
                AFP_POINT_SIZE,
                BOLD,
                NORMAL,
                ORIENTO,
                ROMAN,
                AFP_FONT_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'AFPDFNT';
  CALL CHKSUCC;

  TIM12BOLD               = AFP_FONT_ID;

/*                                                              */
/*   THIS IS THE START OF THE FIELD AND ROW DEFINITIONS         */
/*                                                              */

  AFP_ALIGNMENT_POSITION   = 0.0 * 10000;
  AFP_LEFT_MARGIN          = 0.0 * 10000;
  AFP_RIGHT_MARGIN         = 0.0 * 10000;
  AFP_LINE_SPACING         = AFP_DEFAULT * 10000;
  AFP_SHADING_INTENSITY    = 18;
  AFP_TOP_THICKNESS        = .5 * 10000;
  AFP_BOTTOM_THICKNESS     = .5 * 10000;
  AFP_LEFT_THICKNESS       = .5 * 10000;
  AFP_RIGHT_THICKNESS      = .5 * 10000;

  CALL AFPDFLD (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                FOCENTER,
                AFP_ALIGNMENT_POSITION,
                VERCENTER,
                AFP_LEFT_MARGIN,
                AFP_RIGHT_MARGIN,
                AFP_LINE_SPACING,
                TXTORO_0,
                SCREEN,
                AFP_SHADING_INTENSITY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_FIELD_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDFLD';
  CALL CHKSUCC;

  FIELDH1                 = AFP_FIELD_ID;

  CALL AFPDFLD (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                FOCENTER,
                AFP_ALIGNMENT_POSITION,
                VERCENTER,
                AFP_LEFT_MARGIN,
                AFP_RIGHT_MARGIN,
                AFP_LINE_SPACING,
                TXTORO_0,
                SCREEN,
                AFP_SHADING_INTENSITY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_FIELD_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDFLD';
  CALL CHKSUCC;

  FIELDH2                 = AFP_FIELD_ID;

  CALL AFPDFLD (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                FOCENTER,
                AFP_ALIGNMENT_POSITION,
                VERCENTER,
                AFP_LEFT_MARGIN,
                AFP_RIGHT_MARGIN,
                AFP_LINE_SPACING,
                TXTORO_0,
                SCREEN,
                AFP_SHADING_INTENSITY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_FIELD_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDFLD';
  CALL CHKSUCC;

  FIELDH3                          = AFP_FIELD_ID;
  AFP_NUMBER_COLUMNS               = 3;
  AFP_NUMBER_SUBROWS               = 1;
  AFP_MIN_SUBROW_DEPTH_ARRAY (1)   = AFP_DEFAULT * 10000;
  AFP_ROW_ARRANGE_ARRAY    (1,1)   = FIELDH1;
  AFP_COLUMN_WIDTH_ARRAY     (1)   = 25.0 * 10000;
  AFP_ROW_ARRANGE_ARRAY    (1,2)   = FIELDH2;
  AFP_COLUMN_WIDTH_ARRAY     (2)   = 70.0 * 10000;
  AFP_ROW_ARRANGE_ARRAY    (1,3)   = FIELDH3;
  AFP_COLUMN_WIDTH_ARRAY     (3)   = 30.0 * 10000;

  CALL AFPDROW (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_MIN_SUBROW_DEPTH_ARRAY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_NUMBER_COLUMNS,
                AFP_NUMBER_SUBROWS,
                AFP_ROW_ARRANGE_ARRAY,
                AFP_COLUMN_WIDTH_ARRAY,
                AFP_ROW_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDROW';
  CALL CHKSUCC;

  ROW1                    = AFP_ROW_ID;

/*                                                              */
/*   Define the transaction row                                 */
/*                                                              */

  AFP_ALIGNMENT_POSITION   = 0.0 * 10000;
  AFP_LEFT_MARGIN          = 1.0 * 10000;
  AFP_RIGHT_MARGIN         = 1.0 * 10000;
  AFP_LINE_SPACING         = AFP_DEFAULT * 10000;
  AFP_SHADING_INTENSITY    = 0;
  AFP_TOP_THICKNESS        = .5 * 10000;
  AFP_BOTTOM_THICKNESS     = .5 * 10000;
  AFP_LEFT_THICKNESS       = .5 * 10000;
  AFP_RIGHT_THICKNESS      = .5 * 10000;

  CALL AFPDFLD (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                FOCENTER,
                AFP_ALIGNMENT_POSITION,
                VERCENTER,
                AFP_LEFT_MARGIN,
                AFP_RIGHT_MARGIN,
                AFP_LINE_SPACING,
                TXTORO_0,
                NOSHADE,
                AFP_SHADING_INTENSITY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_FIELD_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
AFP_ERRDATA = 'AFPDFLD';
CALL CHKSUCC;

FIELDT1                 = AFP_FIELD_ID;

CALL AFPDFLD (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              FOCENTER,
              AFP_ALIGNMENT_POSITION,
              VERCENTER,
              AFP_LEFT_MARGIN,
              AFP_RIGHT_MARGIN,
              AFP_LINE_SPACING,
              TXTORO_O,
              NOSHADE,
              AFP_SHADING_INTENSITY,
              AFP_TOP_THICKNESS,
              AFP_BOTTOM_THICKNESS,
              AFP_LEFT_THICKNESS,
              AFP_RIGHT_THICKNESS,
              AFP_FIELD_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPDFLD';
CALL CHKSUCC;

FIELDT2                 = AFP_FIELD_ID;
AFP_LEFT_MARGIN         = 0.0 * 10000;
AFP_ALIGNMENT_POSITION  = 20 * 10000;

CALL AFPDFLD (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              FOCENTER,
              AFP_ALIGNMENT_POSITION,
              VERCENTER,
              AFP_LEFT_MARGIN,
              AFP_RIGHT_MARGIN,
              AFP_LINE_SPACING,
              TXTORO_O,
              NOSHADE,
              AFP_SHADING_INTENSITY,
              AFP_TOP_THICKNESS,
              AFP_BOTTOM_THICKNESS,
              AFP_LEFT_THICKNESS,
              AFP_RIGHT_THICKNESS,
              AFP_FIELD_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPDFLD';
CALL CHKSUCC;

FIELDT3                       = AFP_FIELD_ID;
AFP_TOP_THICKNESS             = 0.5 * 10000;
AFP_BOTTOM_THICKNESS          = 0.5 * 10000;
AFP_NUMBER_COLUMNS            = 3;
AFP_NUMBER_SUBROWS            = 1;
AFP_MIN_SUBROW_DEPTH_ARRAY (1) = AFP_DEFAULT * 10000;
AFP_ROW_ARRANGE_ARRAY    (1,1) = FIELDT1;
AFP_COLUMN_WIDTH_ARRAY     (1) = 25.0 * 10000;
AFP_ROW_ARRANGE_ARRAY    (1,2) = FIELDT2;
AFP_COLUMN_WIDTH_ARRAY     (2) = 70.0 * 10000;
AFP_ROW_ARRANGE_ARRAY    (1,3) = FIELDT3;
AFP_COLUMN_WIDTH_ARRAY     (3) = 30.0 * 10000;

CALL AFPDROW (AFPAPI_HANDLE,
              AFP_DOCUMENT_HANDLE,
              AFP_MIN_SUBROW_DEPTH_ARRAY,
              AFP_TOP_THICKNESS,
              AFP_BOTTOM_THICKNESS,
              AFP_NUMBER_COLUMNS,
              AFP_NUMBER_SUBROWS,
              AFP_ROW_ARRANGE_ARRAY,
              AFP_COLUMN_WIDTH_ARRAY,
              AFP_ROW_ID,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDROW';
  CALL CHKSUCC;

  ROW2                  = AFP_ROW_ID;

 /*                                                          */
 /*   Define the summary row                                 */
 /*                                                          */

  AFP_ALIGNMENT_POSITION   = 0.0 * 10000;
  AFP_LEFT_MARGIN          = 1.0 * 10000;
  AFP_RIGHT_MARGIN         = 1.0 * 10000;
  AFP_LINE_SPACING         = AFP_DEFAULT * 10000;
  AFP_SHADING_INTENSITY    = 18;
  AFP_TOP_THICKNESS        = 0.5 * 10000;
  AFP_BOTTOM_THICKNESS     = 0.5 * 10000;
  AFP_LEFT_THICKNESS       = 0.5 * 10000;
  AFP_RIGHT_THICKNESS      = 0.5 * 10000;

  CALL AFPDFLD (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                FOCENTER,
                AFP_ALIGNMENT_POSITION,
                VERCENTER,
                AFP_LEFT_MARGIN,
                AFP_RIGHT_MARGIN,
                AFP_LINE_SPACING,
                TXTORO_O,
                SCREEN,
                AFP_SHADING_INTENSITY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_FIELD_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDFLD';
  CALL CHKSUCC;

  FIELDS1                 = AFP_FIELD_ID;
  AFP_LEFT_MARGIN         = 0.0 * 10000;
  AFP_ALIGNMENT_POSITION  = 20 * 10000;

  CALL AFPDFLD (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                FOCENTER,
                AFP_ALIGNMENT_POSITION,
                VERCENTER,
                AFP_LEFT_MARGIN,
                AFP_RIGHT_MARGIN,
                AFP_LINE_SPACING,
                TXTORO_O,
                SCREEN,
                AFP_SHADING_INTENSITY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_FIELD_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'AFPDFLD';
  CALL CHKSUCC;

  FIELDS2                        = AFP_FIELD_ID;
  AFP_TOP_THICKNESS              = 0.0 * 10000;
  AFP_BOTTOM_THICKNESS           = 0.0 * 10000;
  AFP_NUMBER_COLUMNS             = 2;
  AFP_NUMBER_SUBROWS             = 1;
  AFP_MIN_SUBROW_DEPTH_ARRAY (1) = AFP_DEFAULT * 10000;
  AFP_ROW_ARRANGE_ARRAY    (1,1) = FIELDS1;
  AFP_COLUMN_WIDTH_ARRAY     (1) = 95.0 * 10000;
  AFP_ROW_ARRANGE_ARRAY    (1,2) = FIELDS2;
  AFP_COLUMN_WIDTH_ARRAY     (2) = 30.0 * 10000;

  CALL AFPDROW (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_MIN_SUBROW_DEPTH_ARRAY,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_NUMBER_COLUMNS,
                AFP_NUMBER_SUBROWS,
                AFP_ROW_ARRANGE_ARRAY,
                AFP_COLUMN_WIDTH_ARRAY,
                AFP_ROW_ID,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPDROW';
  CALL CHKSUCC;

  ROW3                           = AFP_ROW_ID;

END SETUP_AFPAPI;

%PAGE;
/*************************************************************/
/*                                                           */
/*    READ SAMPLE DATA                                       */
/*                                                           */
/*************************************************************/

READ_DATA:  PROC;

  READ FILE(DATAIN) INTO(TRAN_IN);

  ON ENDFILE (DATAIN)
      DATA_REMAINS = NO;

/*                                                           */
/* If this is a new customer, read the customer address and  */
/* account balance.                                          */
/*                                                           */

  IF POST_DATE_IN = 0 THEN
    DO;
      NEW_CUSTOMER        = YES;

      ACCT_1              = SUBSTR(TRANS_DESC,1,4);
      ACCT_2              = SUBSTR(TRANS_DESC,5,4);
      ACCT_3              = SUBSTR(TRANS_DESC,9,4);
      ACCT_4              = SUBSTR(TRANS_DESC,13,4);

/*                                                           */
/*   Read the customer name                                  */
/*                                                           */

      READ FILE(DATAIN) INTO(TRAN_IN);

      ON ENDFILE (DATAIN)
        DATA_REMAINS = NO;

      CUST_NAME           = TRANS_DESC;

/*                                                           */
/*   Read the customer street addres                         */
/*                                                           */

      READ FILE(DATAIN) INTO(TRAN_IN);

      ON ENDFILE (DATAIN)
        DATA_REMAINS = NO;

      CUST_ST_ADDR        = TRANS_DESC;

/*                                                           */
/*   Read the customer city and state                        */
/*                                                           */

      READ FILE(DATAIN) INTO(TRAN_IN);

      ON ENDFILE (DATAIN)
        DATA_REMAINS = NO;

      CUST_CITY_STATE     = TRANS_DESC;

/*                                                           */
/*   Read the customer balance                               */
/*                                                           */

      READ FILE(DATAIN) INTO(TRAN_IN);

      ON ENDFILE (DATAIN)
        DATA_REMAINS = NO;

      CUSTOMER_BALANCE_IN = TRANSACTION_AMOUNT_IN;

/*                                                           */
/*   Read the first customer transaction                     */
/*                                                           */

      READ FILE(DATAIN) INTO(TRAN_IN);

      ON ENDFILE (DATAIN)
        DATA_REMAINS = NO;
    END;

END READ_DATA;

%PAGE;
/*************************************************************/
/*   PROCESS THE CUSTOMER                                    */
/*                                                           */
/*     Begin a page.                                         */
/*     Write the page header.                                */
/*     Write the paragraph.                                  */
/*     Process the customer transactions.                    */
/*     Write the page footer.                                */
/*     End the page.                                         */
/*                                                           */
/*************************************************************/

PROCESS_A_CUSTOMER:  PROC;

  CUSTOMER_BALANCE_OUT      = CUSTOMER_BALANCE_IN;

/*                                                           */
/* Initialize this customer's number of transactions to 0    */
/*                                                           */

  NUM_CUSTOMER_PAGES        = 1;
  AFP_PAGE_WIDTH            = AFP_DEFAULT * 10000;
  AFP_PAGE_DEPTH            = AFP_DEFAULT * 10000;

  CALL AFPBPAG (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_PAGE_WIDTH,
                AFP_PAGE_DEPTH,
                ORIENTDOC,
                AFP_PAGE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'AFPBPAG';

  CALL CREATE_THE_HEADER;

/*                                                            */
/* Calculate the page body size as the page size less the     */
/* page header and bottom margin.                             */
/*                                                            */

GE_HEADER_DEPTH = PAGE_HEADER_DEPTH * 10000;

GE_BODY = AFP_DOC_PAGE_DEPTH - PAGE_HEADER_DEPTH - BOTTOM_MARGIN;

  CALL PROCESS_THE_PARAGRAPH;

/*                                                            */
/* Calculate the remaining page body after the paragraph.     */
/*                                                            */

  PAGE_BODY = PAGE_BODY - AFP_PARAGRAPH_DEPTH;

  CALL PROCESS_TRANSACTIONS;

  CALL CREATE_THE_FOOTER;

  CALL AFPEPAG (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEPAG';
  CALL CHKSUCC;

END PROCESS_A_CUSTOMER;

%PAGE;
/*************************************************************/
/*                                                           */
/*    CREATE THE HEADER.                                     */
/*                                                           */
/*************************************************************/

CREATE_THE_HEADER:  PROC;

  CALL PROCESS_THE_AREA;

/*                                                            */
/* Include the Page Segment                                   */
/*                                                            */

  AFP_X_COORDINATE            = 29 * 10000;
  AFP_Y_COORDINATE            = 23 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_PSEG_NAME               = 'APQPSEG';

  CALL AFPIPSG (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_PSEG_NAME,
                TRU,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPIPSG';
  CALL CHKSUCC;

  CALL PROCESS_THE_ADDRESS;

/*                                                            */
/* Draw a rule underneath the address                         */
/*                                                            */

  AFP_RULE_THICKNESS          = 1.5 * 10000;

  CALL AFPSRTH (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_RULE_THICKNESS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSRTH';
  CALL CHKSUCC;

  AFP_X_COORDINATE            = 29 * 10000;
  AFP_Y_COORDINATE            = 73 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_RULE_LENGTH             = 158 * 10000;

  CALL AFPPRUL (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                XDIRECTION,
                AFP_RULE_LENGTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPRUL';
  CALL CHKSUCC;

/*                                                            */
/* Leave space after the rule                                 */
/*                                                            */

  AFP_Y_COORDINATE            = 4 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YREL,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

/*                                                            */
/* Query the position and calculate the page header depth     */
/*                                                            */

  CALL AFPQPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                AFP_Y_COORDINATE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPQPOS';
  CALL CHKSUCC;

END CREATE_THE_HEADER;
```

```
%PAGE;
/*************************************************************/
/*                                                           */
/*    PROCESS THE AREA                                       */
/*                                                           */
/*************************************************************/

PROCESS_THE_AREA:  PROC;

  AFP_AREA_WIDTH              = 50.0 * 10000;
  AFP_MAX_AREA_DEPTH          = 65.0 * 10000;
  AFP_SHADING_INTENSITY       = 0;

  CALL AFPCARE (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_AREA_WIDTH,
                AFP_MAX_AREA_DEPTH,
                FALS,
                NOSHADE,
                AFP_SHADING_INTENSITY,
                AFP_AREA_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPCARE';
  CALL CHKSUCC;

/*                                                           */
/* Include the Page Overlay                                  */
/*                                                           */

  AFP_OVLY_NAME               = 'O1APQL2';

  CALL AFPIOVL (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_OVLY_NAME,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPIOVL';
  CALL CHKSUCC;

/*                                                           */
/* Write the account number                                  */
/*                                                           */

  CALL AFPSFNT (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                TIM10MED,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSFNT';
  CALL CHKSUCC;

  AFP_X_COORDINATE            = 49 * 10000;
  AFP_Y_COORDINATE            = 7 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  STRING_FIELD =
     ACCT_1 || BLK || ACCT_2 || BLK || ACCT_3 || BLK || ACCT_4;

  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                R_GHT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

/*                                                           */
/* Write the due date                                        */
/*                                                           */

  AFP_X_COORDINATE            = 49 * 10000;
  AFP_Y_COORDINATE            = 12 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  STRING_FIELD                = DUE_DATE;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                R_GHT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

/*                                                           */
/* Write the customer balance                                */
/*                                                           */

  AFP_X_COORDINATE            = 49 * 10000;
  AFP_Y_COORDINATE            = 19 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  CUSTOMER_BALANCE_OUT        = CUSTOMER_BALANCE_IN;
  STRING_FIELD                = CUSTOMER_BALANCE_OUT;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                R_GHT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

/*                                                          */
/* Write the customer payment                               */
/*                                                          */

  AFP_X_COORDINATE            = 49 * 10000;
  AFP_Y_COORDINATE            = 24 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  MIN_AMOUNT_DUE_OUT          = CUSTOMER_BALANCE_IN * .1;
  STRING_FIELD                = MIN_AMOUNT_DUE_OUT;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                R_GHT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEARE (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_AREA_DEPTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEARE';
  CALL CHKSUCC;

/*                                                          */
/* Place the area on the page                               */
/*                                                          */

  AFP_X_COORDINATE            = 137 * 10000;
  AFP_Y_COORDINATE            = 23 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  CALL AFPPARE (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_AREA_HANDLE,
                ROTATE0,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPARE';
  CALL CHKSUCC;

/*                                                          */
/* Destroy the area from AFP API storage                    */
/*                                                          */

  CALL AFPXARE (AFPAPI_HANDLE,
                AFP_AREA_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPXARE';
  CALL CHKSUCC;

END PROCESS_THE_AREA;

%PAGE;
/************************************************************/
/*                                                          */
/*    PROCESS THE ADDRESS.                                  */
/*                                                          */
/************************************************************/

PROCESS_THE_ADDRESS:  PROC;

/*                                                          */
/* Write the customer name                                  */
/*                                                          */

  CALL AFPSFNT (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                TIM12BOLD,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSFNT';
  CALL CHKSUCC;

  AFP_X_COORDINATE            = 29 * 10000;
  AFP_Y_COORDINATE            = 56 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_STRING_LENGTH           = INDEX(CUST_NAME,TEST_VALUE);
  AFP_STRING_LENGTH           = AFP_STRING_LENGTH - 1;
  CUST_LENGTH                 = LENGTH(CUST_NAME);
  AFP_CHARACTER_STRING        = CUST_NAME;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                L_FT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

/*                                                          */
/* Write the customer address                               */
/*                                                          */

  CALL AFPSFNT (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                TIM10MED,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSFNT';
  CALL CHKSUCC;

  AFP_Y_COORDINATE            = 1 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YLINES,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_STRING_LENGTH           = LENGTH(CUST_ST_ADDR);
  AFP_CHARACTER_STRING        = CUST_ST_ADDR;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                L_FT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  AFP_Y_COORDINATE            = 1 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YLINES,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_STRING_LENGTH           = LENGTH(CUST_CITY_STATE);
  AFP_CHARACTER_STRING        = CUST_CITY_STATE;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                L_FT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

END PROCESS_THE_ADDRESS;

%PAGE;
/************************************************************/
/*                                                          */
/*    PROCESS THE PARAGRAPH.                                */
/*                                                          */
/************************************************************/

PROCESS_THE_PARAGRAPH:  PROC;

  AFP_X_COORDINATE            = 29 * 10000;
  AFP_Y_COORDINATE            = PARAGRAPH_WHITE_SPACE * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YREL,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_RULE_THICKNESS          = 0.5 * 10000;

  CALL AFPSRTH (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_RULE_THICKNESS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
AFP_ERRDATA = 'AFPSRTH';
CALL CHKSUCC;

AFP_FIRST_LINE_INDENT       = 0 * 10000;
AFP_FIRST_LINE_OFFSET       = AFP_DEFAULT * 10000;
AFP_LEFT_MARGIN             = 10.0 * 10000;
AFP_LINE_LENGTH             = 135.0 * 10000;
AFP_LINE_SPACING            = AFP_DEFAULT * 10000;
AFP_RT_RULE_OFFSET          = 158.0 * 10000;
AFP_BOT_RULE_OFFSET         = 0.0 * 10000;
AFP_SHADING_INTENSITY       = 0;

CALL AFPBPAR (AFPAPI_HANDLE,
              AFP_PAGE_HANDLE,
              AFP_FIRST_LINE_INDENT,
              FOJUSTIFY,
              AFP_FIRST_LINE_OFFSET,
              AFP_LEFT_MARGIN,
              AFP_LINE_LENGTH,
              AFP_LINE_SPACING,
              TRU,
              AFP_RT_RULE_OFFSET,
              AFP_BOT_RULE_OFFSET,
              NOSHADE,
              AFP_SHADING_INTENSITY,
              AFP_PARAGRAPH_HANDLE,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPBPAR';
CALL CHKSUCC;

AFP_CURRENT_HANDLE          = AFP_PARAGRAPH_HANDLE;

CALL AFPSFNT (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              TIM12BOLD,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPSFNT';
CALL CHKSUCC;

STRING_FIELD                = 'CONGRATULATIONS, ';
AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
AFP_CHARACTER_STRING        = STRING_FIELD;

CALL AFPPTXT (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              AFP_STRING_LENGTH,
              AFP_CHARACTER_STRING,
              TRU,
              FALS,
              AFP_REMAINING_LENGTH,
              AFP_REMAINING_STRING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPPTXT';
CALL CHKSUCC;

CALL AFPSFNT (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              TIM12MED,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPSFNT';
CALL CHKSUCC;

AFP_STRING_LENGTH           = INDEX(CUST_NAME,TEST_VALUE);
AFP_STRING_LENGTH           = AFP_STRING_LENGTH - 1;
AFP_CHARACTER_STRING        = CUST_NAME;

CALL AFPPTXT (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              AFP_STRING_LENGTH,
              AFP_CHARACTER_STRING,
              TRU,
              FALS,
              AFP_REMAINING_LENGTH,
              AFP_REMAINING_STRING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPPTXT';
CALL CHKSUCC;

STRING_1 = '  Because of your excellent credit rating, you ';
STRING_2 = 'are now eligible for free credit insurance which ';

STRING_FIELD = EXCLAMATION || STRING_1 || STRING_2;
AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
AFP_CHARACTER_STRING        = STRING_FIELD;

CALL AFPPTXT (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              AFP_STRING_LENGTH,
              AFP_CHARACTER_STRING,
              TRU,
              FALS,
              AFP_REMAINING_LENGTH,
              AFP_REMAINING_STRING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPPTXT';
CALL CHKSUCC;

STRING_1 = 'protects you in case your Primo card is ever ';
STRING_2 = 'lost or stolen.  ';

STRING_FIELD                = STRING_1 || STRING_2;
AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
AFP_CHARACTER_STRING        = STRING_FIELD;

CALL AFPPTXT (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              AFP_STRING_LENGTH,
              AFP_CHARACTER_STRING,
              TRU,
              FALS,
              AFP_REMAINING_LENGTH,
              AFP_REMAINING_STRING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPPTXT';
CALL CHKSUCC;

STRING_FIELD = 'Call NOW for more information' || EXCLAMATION;
AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
AFP_CHARACTER_STRING        = STRING_FIELD;

CALL AFPPTXT (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              AFP_STRING_LENGTH,
              AFP_CHARACTER_STRING,
              TRU,
              TRU,
              AFP_REMAINING_LENGTH,
              AFP_REMAINING_STRING,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);

AFP_ERRDATA = 'AFPPTXT';
CALL CHKSUCC;

CALL AFPEPAR (AFPAPI_HANDLE,
              AFP_PARAGRAPH_HANDLE,
              AFP_PARAGRAPH_DEPTH,
              AFP_RET_CODE,
              AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'AFPEPAR';
  CALL CHKSUCC;

/*                                                          */
/* Calculate the amount of space taken up by the paragraph  */
/*                                                          */

  PARAGRAPH_WHITE_SPACE = PARAGRAPH_WHITE_SPACE * 10000;

  AFP_PARAGRAPH_DEPTH =
      AFP_PARAGRAPH_DEPTH + PARAGRAPH_WHITE_SPACE;

END PROCESS_THE_PARAGRAPH;

%PAGE;
/************************************************************/
/*  PROCESS TRANSACTIONS.                                   */
/*                                                          */
/*      Begin a table.                                      */
/*      Write the header rows.                              */
/*      Write a transaction row until no more data for this */
/*         customer or no more data.                        */
/*      Write the summary row.                              */
/*      End the table.                                      */
/*      Compute the table depth.                            */
/************************************************************/

PROCESS_TRANSACTIONS:  PROC;

  NEW_CUSTOMER               = NO;

/*                                                          */
/* Start the table whose maximum depth is the remaining page */
/* body space after the white space preceeding the table.   */
/*                                                          */

  AFP_X_COORDINATE           = 45 * 10000;
  AFP_Y_COORDINATE           = TABLE_WHITE_SPACE;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YREL,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_MAX_TABLE_DEPTH        = TABLE_DEPTH * 10000;
  AFP_TABLE_WIDTH            = 125.0 * 10000;
  AFP_TOP_THICKNESS          = 1.0 * 10000;
  AFP_BOTTOM_THICKNESS       = 0.5 * 10000;
  AFP_LEFT_THICKNESS         = 0.5 * 10000;
  AFP_RIGHT_THICKNESS        = 0.5 * 10000;

  CALL AFPBTBL (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_TABLE_WIDTH,
                AFP_MAX_TABLE_DEPTH,
                ORIENTO,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBTBL';
  CALL CHKSUCC;

  AFP_CURRENT_HANDLE         = AFP_TABLE_HANDLE;

/*                                                          */
/* Write the table header rows                              */
/*                                                          */

  CALL WRITE_HEADER_ROWS;

/*                                                          */
/* Write the transaction rows for this customer             */
/*                                                          */

  IF NEW_CUSTOMER = NO THEN
    DO WHILE(NEW_CUSTOMER = NO);
      CALL WRITE_TRANSACTIONS;
    END;

/*                                                          */
/* Write the table summary rows                             */
/*                                                          */

  CALL WRITE_SUMMARY_ROW;

/*                                                          */
/* If the end of the table was reached, end this page, start */
/* a new page, and write the summary row that didn't fit.   */
/*                                                          */

  IF AFP_SEVERITY_CODE = WARNING THEN
    DO;
      CALL END_CUST_PAGE;
      CALL WRITE_SUMMARY_ROW;
    END;

/*                                                          */
/* End the table                                            */
/*                                                          */

  CALL AFPETBL (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_TABLE_DEPTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPETBL';
  CALL CHKSUCC;

/*                                                          */
/* Calculate the amount of space taken up by the table      */
/*                                                          */

  AFP_TABLE_DEPTH = AFP_TABLE_DEPTH + TABLE_WHITE_SPACE;

END PROCESS_TRANSACTIONS;

%PAGE;
/************************************************************/
/*    WRITE_HEADER_ROWS.                                    */
/*                                                          */
/*            WRITE THE HEADER ROW FOR THE TABLE.           */
/************************************************************/

WRITE_HEADER_ROWS:  PROC;

  CALL AFPBROW (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                ROW1,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBROW';
  CALL CHKSUCC;

/*                                                          */
/* Write the first field in column 1                        */
/*                                                          */

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDH1,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  CALL AFPSFNT (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                TIM12MED,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSFNT';
  CALL CHKSUCC;

  STRING_FIELD               = 'Date';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CENTER,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

/*                                                    */
/* Write the 2nd field in column 2                    */
/*                                                    */

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDH2,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  STRING_FIELD               = 'Transaction Description';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CENTER,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

/*                                                    */
/* Write the 3rd field in column 3                    */
/*                                                    */

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDH3,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  STRING_FIELD               = 'Amount';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CENTER,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

  CALL AFPEROW (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_CURRENT_TABLE_DEPTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'APFEROW';
  CALL CHKSUCC;

END WRITE_HEADER_ROWS;

%PAGE;
/*************************************************************/
/*    WRITE_TRANSACTIONS.                                    */
/*                                                           */
/*            Write a transaction row.                       */
/*            Read another input data record.                */
/*************************************************************/

WRITE_TRANSACTIONS:  PROC;

  IF DATA_REMAINS THEN
     CALL WRITE_TRANSACTION_ROW;

/* If the end of the table was reached, end this page, start  */
/* a new page, and write the transaction row that didn't fit. */

  IF AFP_SEVERITY_CODE = WARNING THEN
    DO;
      CALL END_CUST_PAGE;

      CALL WRITE_TRANSACTION_ROW;
    END;

  IF DATA_REMAINS THEN
    CALL READ_DATA;
  ELSE
    NEW_CUSTOMER = YES;

END WRITE_TRANSACTIONS;
```

```
%PAGE;
/*************************************************************/
/* WRITE_TRANSACTION_ROW.                                    */
/*                                                           */
/*     Begin a row.                                          */
/*     Write the post date to the first field in the table.  */
/*     Write the transaction description to the 2nd field.   */
/*     Write the transaction amount to the 3rd field.        */
/*     End the row.                                          */
/*************************************************************/

WRITE_TRANSACTION_ROW:  PROC;

  CALL AFPBROW (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                ROW2,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBROW';
  CALL CHKSUCC;

/*                                                           */
/* Write the transaction date in column 1                    */
/*                                                           */

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDT1,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  CALL AFPSFNT (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                TIM10MED,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSFNT';
  CALL CHKSUCC;

  POST_DATE_OUT               = POST_DATE_IN;
  STRING_FIELD                = POST_DATE_OUT;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CENTER,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

/*                                                           */
/* Write the transaction description in column 2             */
/*                                                           */

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDT2,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  STRING_FIELD                = TRANS_DESC;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                L_FT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

/*                                                           */
/* Write the transaction amount in column 3                  */
/*                                                           */

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDT3,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  TRANSACTION_AMOUNT_OUT      = TRANSACTION_AMOUNT_IN;
  STRING_FIELD                = TRANSACTION_AMOUNT_OUT;
  AFP_STRING_LENGTH           = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING        = STRING_FIELD;
  AFP_ALIGNMENT_CHAR          = '.';

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CHAR,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

  CALL AFPEROW (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_CURRENT_TABLE_DEPTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);
```

```
  AFP_ERRDATA = 'APFEROW';
  CALL CHKSUCC;

END WRITE_TRANSACTION_ROW;

%PAGE;
/*************************************************************/
/*    WRITE_SUMMARY_ROW.                                     */
/*                                                           */
/*           Write the customer summary row.                 */
/*************************************************************/

WRITE_SUMMARY_ROW:  PROC;

  CALL AFPBROW (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                ROW3,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBROW';
  CALL CHKSUCC;

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDS1,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  CALL AFPSFNT (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                TIM12MED,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSFNT';
  CALL CHKSUCC;

  STRING_FIELD               = 'Total Amount';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CENTER,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

  CALL AFPBFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                FIELDS2,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBFLD';
  CALL CHKSUCC;

  STRING_FIELD               = CUSTOMER_BALANCE_OUT;
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CHAR,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  CALL AFPEFLD (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEFLD';
  CALL CHKSUCC;

  CALL AFPEROW (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_CURRENT_TABLE_DEPTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'APFEROW';
  CALL CHKSUCC;

END WRITE_SUMMARY_ROW;

%PAGE;
/*************************************************************/
/*    CREATE_THE_FOOTER.                                     */
/*                                                           */
/*            WRITE THE PAGE FOOTER.                         */
/*************************************************************/

CREATE_THE_FOOTER:  PROC;

  AFP_X_COORDINATE           = 108 * 10000;
  AFP_Y_COORDINATE           = 270 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  CALL AFPSFNT (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                TIM10MED,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSFNT';
  CALL CHKSUCC;
```

```
  STRING_FIELD               = 'Page ';
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                CENTER,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

  NUM_CUSTOMER_PAGES_OUT     = NUM_CUSTOMER_PAGES;
  STRING_FIELD               = NUM_CUSTOMER_PAGES_OUT;
  AFP_STRING_LENGTH          = LENGTH(STRING_FIELD);
  AFP_CHARACTER_STRING       = STRING_FIELD;

  CALL AFPPCHS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_STRING_LENGTH,
                AFP_CHARACTER_STRING,
                L_FT,
                AFP_ALIGNMENT_CHAR,
                FALS,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPCHS';
  CALL CHKSUCC;

END CREATE_THE_FOOTER;

%PAGE;
/*************************************************************/
/*    END_CUST_PAGE.                                         */
/*                                                           */
/*            End the table.                                 */
/*            Write the page footer.                         */
/*            End the page.                                  */
/*            Start a new page.                              */
/*            Begin a table.                                 */
/*            Write the header rows.                         */
/*************************************************************/

END_CUST_PAGE:  PROC;

  CALL AFPETBL (AFPAPI_HANDLE,
                AFP_TABLE_HANDLE,
                AFP_TABLE_DEPTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPETBL';
  CALL CHKSUCC;

  AFP_CURRENT_HANDLE         = AFP_PAGE_HANDLE;

/*                                                           */
/*            Write the footer                               */
/*                                                           */

  CALL CREATE_THE_FOOTER;

  CALL AFPEPAG (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEPAG';
  CALL CHKSUCC;

/*                                                           */
/*            Start a new page                               */
/*                                                           */

  NUM_CUSTOMER_PAGES         = NUM_CUSTOMER_PAGES + 1;
  AFP_PAGE_WIDTH             = AFP_DEFAULT * 10000;
  AFP_PAGE_DEPTH             = AFP_DEFAULT * 10000;

  CALL AFPBPAG (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_PAGE_WIDTH,
                AFP_PAGE_DEPTH,
                ORIENTDOC,
                AFP_PAGE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBPAG';

  CALL CREATE_THE_CONT_HEADER;

/*                                                           */
/* Calculate the page body space as the page size less the   */
/* continuation page header size and bottom margin.          */
/*                                                           */

  PAGE_HEADER_DEPTH = PAGE_HEADER_DEPTH * 10000;

  PAGE_BODY =
      AFP_DOC_PAGE_DEPTH - PAGE_HEADER_DEPTH - BOTTOM_MARGIN;

/*                                                           */
/*            begin a table                                  */
/*                                                           */

  AFP_X_COORDINATE           = 45 * 10000;
  AFP_Y_COORDINATE           = TABLE_WHITE_SPACE;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YREL,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_MAX_TABLE_DEPTH        = 196 * 10000;
  AFP_TABLE_WIDTH            = 125.0 * 10000;
  AFP_TOP_THICKNESS          = 1.0 * 10000;
  AFP_BOTTOM_THICKNESS       = 0.5 * 10000;
  AFP_LEFT_THICKNESS         = 0.5 * 10000;
  AFP_RIGHT_THICKNESS        = 0.5 * 10000;

  CALL AFPBTBL (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_TABLE_WIDTH,
                AFP_MAX_TABLE_DEPTH,
                ROTATEO,
                AFP_TOP_THICKNESS,
                AFP_BOTTOM_THICKNESS,
                AFP_LEFT_THICKNESS,
                AFP_RIGHT_THICKNESS,
                AFP_TABLE_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPBTBL';
  CALL CHKSUCC;
```

```
/*                                                                  */
/*            write the header rows                                 */
/*                                                                  */

  CALL WRITE_HEADER_ROWS;

END END_CUST_PAGE;

%PAGE;
/*******************************************************************/
/*    CREATE THE HEADER FOR THE CONTINUATION PAGES                 */
/*******************************************************************/

CREATE_THE_CONT_HEADER:  PROC;

/*                                                                  */
/*  Include the Page Segment                                        */
/*                                                                  */

  PAGE_BODY                   = 260 * 10000;
  AFP_X_COORDINATE            = 29 * 10000;
  AFP_Y_COORDINATE            = 23 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_PSEG_NAME               = 'APQPSEG';

  CALL AFPIPSG (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_PSEG_NAME,
                TRU,
                FALS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPIPSG';
  CALL CHKSUCC;

/*                                                                  */
/*  Draw a rule underneath the Page Segment                         */
/*                                                                  */

  AFP_RULE_THICKNESS          = 1.5 * 10000;

  CALL AFPSRTH (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_RULE_THICKNESS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSRTH';
  CALL CHKSUCC;

  AFP_X_COORDINATE            = 29 * 10000;
  AFP_Y_COORDINATE            = 50 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YABS,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

  AFP_RULE_LENGTH             = 158 * 10000;

  CALL AFPPRUL (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                XDIRECTION,
                AFP_RULE_LENGTH,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPPRUL';
  CALL CHKSUCC;

/*                                                                  */
/*  Leave space after the rule                                      */
/*                                                                  */

  AFP_Y_COORDINATE            = 4 * 10000;

  CALL AFPSPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                XABS,
                AFP_Y_COORDINATE,
                YREL,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPSPOS';
  CALL CHKSUCC;

/*                                                                  */
/*  Query the position and calculate the page header depth          */
/*                                                                  */

  CALL AFPQPOS (AFPAPI_HANDLE,
                AFP_PAGE_HANDLE,
                AFP_X_COORDINATE,
                AFP_Y_COORDINATE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPQPOS';
  CALL CHKSUCC;

END CREATE_THE_CONT_HEADER;

%PAGE;
/*******************************************************************/
/*                                                                  */
/*    TERMINATE THE AFPAPI AND THE PROGRAM                          */
/*                                                                  */
/*******************************************************************/

END_PROCESSING:  PROC;

  CALL AFPEDOC (AFPAPI_HANDLE,
                AFP_DOCUMENT_HANDLE,
                AFP_RET_CODE,
                AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEDOC';
  CALL CHKSUCC;

  CALL AFPEND (AFPAPI_HANDLE,
               AFP_RET_CODE,
               AFP_SEVERITY_CODE);

  AFP_ERRDATA = 'AFPEND';
  CALL CHKSUCC;

  CLOSE FILE(DATAIN);

END END_PROCESSING;

%PAGE;
%INCLUDE DDSYS (APQPPRF);

END APQPSM2;
```

Appendix A. PL/1 Macros Used as Programming Interfaces

General-Use Programming Interfaces

The macros identified in this appendix are provided as programming interfaces for customers by AFP API.

Warning: Do not use as programming interfaces any AFP API macros other than those identified in this appendix.

End of General-Use Programming Interfaces

This appendix contains the following COBOL macros:

APQPCON PL/1 constants
APQPRCS PL/1 return codes
APQPVAR PL/1 variables
APQPPRF PL/1 performs

APQPCON

```
/*****************************************************/
/* COPY BOOK --- APQPCON                             */
/*                                                   */
/* This copy book contains the PL/1 constants for    */
/* the variables used by the AFP API.                */
/*****************************************************/

/*****************************************************/
/* The following value indicates that a default      */
/* value is to be used for an AFP API parameter.     */
/*****************************************************/

DCL  AFP_DEFAULT            FIXED BIN(31) INIT (-1.0);

/*****************************************************/
/* The following value indicates that the document   */
/* page orientation is to be used for a page.        */
/* Specified on AFPBPAG (Begin Page).                */
/*****************************************************/

DCL  ORIENTDOC              FIXED BIN(31) INIT (1);

/*****************************************************/
/* Boolean values                                    */
/*****************************************************/

DCL  TRU                    FIXED BIN(31) INIT (1);
DCL  FALS                   FIXED BIN(31) INIT (0);

/*****************************************************/
/* The input units of measurement for AFPSUNI and    */
/* AFPBDOC (Set Units and Begin Document)            */
/*****************************************************/

DCL  INCH                   FIXED BIN(31) INIT (1);
DCL  MM                     FIXED BIN(31) INIT (2);
DCL  CM                     FIXED BIN(31) INIT (3);
DCL  U240                   FIXED BIN(31) INIT (4);
DCL  U1440                  FIXED BIN(31) INIT (5);

/*****************************************************/
/* The colors for AFPSCLR (Set Color)                */
/*****************************************************/

DCL  BLACK                  FIXED BIN(31) INIT (1);
DCL  BLUE                   FIXED BIN(31) INIT (2);
DCL  RED                    FIXED BIN(31) INIT (3);
DCL  MAGENTA                FIXED BIN(31) INIT (4);
DCL  GREEN                  FIXED BIN(31) INIT (5);
DCL  CYAN                   FIXED BIN(31) INIT (6);
DCL  YELLOW                 FIXED BIN(31) INIT (7);
DCL  BROWN                  FIXED BIN(31) INIT (8);
DCL  MEDIA                  FIXED BIN(31) INIT (9);

/*****************************************************/
/* The font weights for AFPDFNT (Define Font)        */
/*****************************************************/

DCL  ULTRALIGHT             FIXED BIN(31) INIT (1);
DCL  EXTRALIGHT             FIXED BIN(31) INIT (2);
DCL  LIGHT                  FIXED BIN(31) INIT (3);
DCL  SEMILIGHT              FIXED BIN(31) INIT (4);
DCL  MEDIUM                 FIXED BIN(31) INIT (5);
DCL  SEMIBOLD               FIXED BIN(31) INIT (6);
DCL  BOLD                   FIXED BIN(31) INIT (7);
DCL  EXTRABOLD              FIXED BIN(31) INIT (8);
DCL  ULTRABOLD              FIXED BIN(31) INIT (9);

/*****************************************************/
/* the font widths for AFPDFNT (Define Font)         */
/*****************************************************/

DCL  ULTRACOND              FIXED BIN(31) INIT (1);
DCL  EXTRACOND              FIXED BIN(31) INIT (2);
DCL  CONDENSED              FIXED BIN(31) INIT (3);
DCL  SEMICOND               FIXED BIN(31) INIT (4);
DCL  NORMAL                 FIXED BIN(31) INIT (5);
DCL  SEMIEXP                FIXED BIN(31) INIT (6);
DCL  EXPANDED               FIXED BIN(31) INIT (7);
DCL  EXTRAEXP               FIXED BIN(31) INIT (8);
DCL  ULTRAEXP               FIXED BIN(31) INIT (9);

/*****************************************************/
/* The font styles for AFPDFNT (Define Font)         */
/*****************************************************/

DCL  ROMAN                  FIXED BIN(31) INIT (1);
DCL  ITALIC                 FIXED BIN(31) INIT (2);

/*****************************************************/
/* The string alignment values for AFPPCHS (Put      */
/* Character String)                                 */
/*****************************************************/

DCL  R_GHT                  FIXED BIN(31) INIT (1);
DCL  L_FT                   FIXED BIN(31) INIT (2);
DCL  CENTER                 FIXED BIN(31) INIT (3);
DCL  CHAR                   FIXED BIN(31) INIT (4);

/*****************************************************/
/* The format options for AFPBPAR and AFPDFLD        */
/* (Begin Paragraph and Define Field)                */
/*****************************************************/

DCL  FOLEFT                 FIXED BIN(31) INIT (1);
DCL  FOCENTER               FIXED BIN(31) INIT (2);
DCL  FORIGHT                FIXED BIN(31) INIT (3);
DCL  FOJUSTIFY              FIXED BIN(31) INIT (4);

/*****************************************************/
/* the line directions for AFPPRUL (Put Rule)        */
/*****************************************************/

DCL  XDIRECTION             FIXED BIN(31) INIT (1);
DCL  YDIRECTION             FIXED BIN(31) INIT (2);

/*****************************************************/
/* The Y Reference Coordinate systems for AFPSPOS    */
/* (Set Position)                                    */
/*****************************************************/

DCL  YABS                   FIXED BIN(31) INIT (1);
DCL  YREL                   FIXED BIN(31) INIT (2);
DCL  YLINES                 FIXED BIN(31) INIT (3);

/*****************************************************/
/* The X Reference Coordinate systems for AFPSPOS    */
/* (Set Position)                                    */
/*****************************************************/

DCL  XABS                   FIXED BIN(31) INIT (1);
DCL  XREL                   FIXED BIN(31) INIT (2);

/*****************************************************/
/* The shading patterns                              */
/*****************************************************/

DCL  STNDARD                FIXED BIN(31) INIT (1);
DCL  SCREEN                 FIXED BIN(31) INIT (2);
DCL  NOSHADE                FIXED BIN(31) INIT (3);
```

```
/*****************************************************/
/* The page orientations for AFPBDOC and AFPBPAG     */
/* (Begin Document and Begin Page)                   */
/*****************************************************/

DCL  ORIENT0                FIXED BIN(31) INIT (0);
DCL  ORIENT90               FIXED BIN(31) INIT (90);
DCL  ORIENT180              FIXED BIN(31) INIT (180);
DCL  ORIENT270              FIXED BIN(31) INIT (270);

/*****************************************************/
/* The font, area, table, and object rotations for   */
/* AFPDFNT, AFPPARE, AFPBTBL, and AFPIOBJ (Define    */
/* Font, Put Area, Begin Table, and Include Object)  */
/*****************************************************/

DCL  ROTATE0                FIXED BIN(31) INIT (0);
DCL  ROTATE90               FIXED BIN(31) INIT (90);
DCL  ROTATE180              FIXED BIN(31) INIT (180);
DCL  ROTATE270              FIXED BIN(31) INIT (270);
DCL  ROTATE_DEFAULT         FIXED BIN(31) INIT (1);

/*****************************************************/
/* The text orientation for AFPDFLD (Define Field)   */
/*****************************************************/

DCL  TXTOR0_0               FIXED BIN(31) INIT (0);
DCL  TXTOR90_180            FIXED BIN(31) INIT (90);
DCL  TXTOR180_270           FIXED BIN(31) INIT (180);
DCL  TXTOR270_0             FIXED BIN(31) INIT (270);

/*****************************************************/
/* The vertical alignment for AFPDFLD (Define Field) */
/*****************************************************/

DCL  VERTOP                 FIXED BIN(31) INIT (1);
DCL  VERCENTER              FIXED BIN(31) INIT (2);
DCL  VERBOTTOM              FIXED BIN(31) INIT (3);

/*****************************************************/
/* The object mapping options for AFPIOBJ (Include   */
/* Object)                                           */
/*****************************************************/

DCL  DEFAULT_MAP            FIXED BIN(31) INIT (1);
DCL  SCALE_TO_FIT           FIXED BIN(31) INIT (2);
DCL  CENTER_AND_TRIM        FIXED BIN(31) INIT (3);
DCL  POSITION_AND_TRIM      FIXED BIN(31) INIT (4);
DCL  POINT_TO_PEL           FIXED BIN(31) INIT (5);
DCL  DOUBLE_DOT             FIXED BIN(31) INIT (6);

/*****************************************************/
/* The severity codes returned by any AFP API call   */
/*****************************************************/

DCL  NOERROR                FIXED BIN(31) INIT (0);
DCL  WARNING                FIXED BIN(31) INIT (4);
DCL  ERRER                  FIXED BIN(31) INIT (8);
DCL  SEVERE                 FIXED BIN(31) INIT (12);
DCL  FATAL                  FIXED BIN(31) INIT (16);

/*****************************************************/
/* Special keywords for requesting buffered output   */
/*****************************************************/

DCL  BUFFERED               CHAR (8) INIT ('@#$VMav)');
DCL  DISCBUFF               CHAR (8) INIT ('@#$VJls)');
```

APQPRCS

```
/*****************************************************************/
/*                                                               */
/* COPY BOOK --- APQPRCS                                         */
/*                                                               */
/* This copy book contains PL/1 constants for the return codes  */
/* generated by the AFP API.                                     */
/*****************************************************************/

DCL AFP_RETURN_CODES;
   DCL  ER_FAIL                  FIXED BINARY (31) INIT (1);
   DCL  ER_QATTS                 FIXED BINARY (31) INIT (2);
   DCL  ER_NOATTS                FIXED BINARY (31) INIT (3);
   DCL  ER_TERM                  FIXED BINARY (31) INIT (4);
   DCL  ER_NOTFOUND              FIXED BINARY (31) INIT (5);
   DCL  ER_SETOUT                FIXED BINARY (31) INIT (6);
   DCL  ER_NOTENDED              FIXED BINARY (31) INIT (7);
   DCL  ER_DOCEXISTS             FIXED BINARY (31) INIT (8);
   DCL  ER_PAGEXISTS             FIXED BINARY (31) INIT (9);
   DCL  ER_END                   FIXED BINARY (31) INIT (11);
   DCL  ER_MBAREA                FIXED BINARY (31) INIT (12);
   DCL  ER_IVLTHICK              FIXED BINARY (31) INIT (13);
   DCL  ER_IVTYPE                FIXED BINARY (31) INIT (14);
   DCL  ER_BLKTYPE               FIXED BINARY (31) INIT (15);
   DCL  ER_IVCOLOR               FIXED BINARY (31) INIT (16);
   DCL  ER_IVUNITS               FIXED BINARY (31) INIT (17);
   DCL  ER_IVROTATE              FIXED BINARY (31) INIT (18);
   DCL  ER_IVNUMROWS             FIXED BINARY (31) INIT (19);
   DCL  ER_IVALIGN               FIXED BINARY (31) INIT (20);
   DCL  ER_DPARENT               FIXED BINARY (31) INIT (21);
   DCL  ER_PPARENT               FIXED BINARY (31) INIT (22);
   DCL  ER_APARENT               FIXED BINARY (31) INIT (23);
   DCL  ER_NOCURSOR              FIXED BINARY (31) INIT (24);
   DCL  ER_IVFONTID              FIXED BINARY (31) INIT (25);
   DCL  ER_IVBLOCK               FIXED BINARY (31) INIT (26);
   DCL  ER_IVCONTROL             FIXED BINARY (31) INIT (27);
   DCL  ER_BACK                  FIXED BINARY (31) INIT (28);
   DCL  ER_NEGATIVE              FIXED BINARY (31) INIT (29);
   DCL  ER_IVCSPACEP             FIXED BINARY (31) INIT (30);
   DCL  ER_BLKMEM                FIXED BINARY (31) INIT (31);
   DCL  ER_IVDATAS               FIXED BINARY (31) INIT (33);
   DCL  ER_IVWSPACE              FIXED BINARY (31) INIT (34);
   DCL  ER_OVERFLOW              FIXED BINARY (31) INIT (35);
   DCL  ER_NULLPTR               FIXED BINARY (31) INIT (36);
   DCL  ER_NULLCONTROL           FIXED BINARY (31) INIT (37);
   DCL  ER_NOTINIT               FIXED BINARY (31) INIT (38);
   DCL  ER_NOTSTRT               FIXED BINARY (31) INIT (39);
   DCL  ER_NOTACST               FIXED BINARY (31) INIT (40);
   DCL  ER_NOTACT                FIXED BINARY (31) INIT (41);
   DCL  ER_NOTACT_MOV            FIXED BINARY (31) INIT (42);
   DCL  ER_NOTACT_PUT            FIXED BINARY (31) INIT (43);
   DCL  ER_NOTEND                FIXED BINARY (31) INIT (44);
   DCL  ER_ATTSMEM               FIXED BINARY (31) INIT (45);
   DCL  ER_NOATTPTR              FIXED BINARY (31) INIT (46);
   DCL  ER_IVOUTDS               FIXED BINARY (31) INIT (47);
   DCL  ER_DCFMEM                FIXED BINARY (31) INIT (48);
   DCL  ER_APPLMEM               FIXED BINARY (31) INIT (49);
   DCL  ER_AREAMEM               FIXED BINARY (31) INIT (50);
   DCL  ER_PUTSTR                FIXED BINARY (31) INIT (51);
   DCL  ER_DCFFENV               FIXED BINARY (31) INIT (52);
   DCL  ER_PUTAREA               FIXED BINARY (31) INIT (53);
   DCL  ER_IVAREAROT             FIXED BINARY (31) INIT (54);
   DCL  ER_NOTACT_PUTA           FIXED BINARY (31) INIT (55);
   DCL  ER_SETFONT               FIXED BINARY (31) INIT (56);
   DCL  ER_SETCSPAC              FIXED BINARY (31) INIT (57);
   DCL  ER_SETLTHCK              FIXED BINARY (31) INIT (58);
   DCL  ER_IVYREF                FIXED BINARY (31) INIT (59);
   DCL  ER_NOFONTPTR             FIXED BINARY (31) INIT (60);
   DCL  ER_FONTMEM               FIXED BINARY (31) INIT (61);
   DCL  ER_FONTATSMEM            FIXED BINARY (31) INIT (62);
   DCL  ER_FONTNOTFND            FIXED BINARY (31) INIT (63);
   DCL  ER_NOTACT_DEF            FIXED BINARY (31) INIT (64);
   DCL  ER_NOTACT_SET            FIXED BINARY (31) INIT (65);
   DCL  ER_PUTLINE               FIXED BINARY (31) INIT (66);
   DCL  ER_IVDIRECTION           FIXED BINARY (31) INIT (67);
   DCL  ER_DCFPOS                FIXED BINARY (31) INIT (68);
   DCL  ER_NOTACT_INC            FIXED BINARY (31) INIT (69);
   DCL  ER_INCOVLY               FIXED BINARY (31) INIT (70);
   DCL  ER_AREANOTFND            FIXED BINARY (31) INIT (71);
   DCL  ER_MARG_OVERF            FIXED BINARY (31) INIT (72);
   DCL  ER_IVFMODE               FIXED BINARY (31) INIT (73);
   DCL  ER_IVCODEPG              FIXED BINARY (31) INIT (74);
   DCL  ER_IVDESCNM              FIXED BINARY (31) INIT (75);
   DCL  ER_IVPTSIZE              FIXED BINARY (31) INIT (76);
   DCL  ER_IVWEIGHT              FIXED BINARY (31) INIT (77);
   DCL  ER_IVWIDTH               FIXED BINARY (31) INIT (78);
   DCL  ER_IVFONTROT             FIXED BINARY (31) INIT (79);
   DCL  ER_IVSTYLE               FIXED BINARY (31) INIT (80);
   DCL  ER_NOTACT_INCPS          FIXED BINARY (31) INIT (81);
   DCL  ER_NOTACT_PUTL           FIXED BINARY (31) INIT (82);
   DCL  ER_SETCOLOR              FIXED BINARY (31) INIT (83);
   DCL  ER_SETPGOR               FIXED BINARY (31) INIT (84);
   DCL  ER_SETUNITS              FIXED BINARY (31) INIT (85);
   DCL  ER_SETWORDSP             FIXED BINARY (31) INIT (86);
   DCL  ER_DEFFIELD              FIXED BINARY (31) INIT (87);
   DCL  ER_DEFFONT               FIXED BINARY (31) INIT (88);
   DCL  ER_DEFROW                FIXED BINARY (31) INIT (89);
   DCL  ER_NOROWPTR              FIXED BINARY (31) INIT (90);
   DCL  ER_ROWMEM                FIXED BINARY (31) INIT (91);
   DCL  ER_NOFLDPTR              FIXED BINARY (31) INIT (92);
   DCL  ER_FLDMEM                FIXED BINARY (31) INIT (93);
   DCL  ER_IVHOR                 FIXED BINARY (31) INIT (94);
   DCL  ER_IVVER                 FIXED BINARY (31) INIT (95);
   DCL  ER_IVSHADE               FIXED BINARY (31) INIT (96);
   DCL  ER_IVSHINT               FIXED BINARY (31) INIT (97);
   DCL  ER_IVFLDOR               FIXED BINARY (31) INIT (98);
   DCL  ER_IVFLDFR               FIXED BINARY (31) INIT (99);
   DCL  ER_IVARSHADE             FIXED BINARY (31) INIT (100);
   DCL  ER_IVARSHINT             FIXED BINARY (31) INIT (101);
   DCL  ER_IVDEPTH               FIXED BINARY (31) INIT (102);
   DCL  ER_IVROWFR               FIXED BINARY (31) INIT (103);
   DCL  ER_FLDATSMEM             FIXED BINARY (31) INIT (104);
   DCL  ER_ROWATSMEM             FIXED BINARY (31) INIT (105);
   DCL  ER_CREATETABLE           FIXED BINARY (31) INIT (106);
   DCL  ER_IVTABLEROT            FIXED BINARY (31) INIT (107);
   DCL  ER_IVTBLRGHT             FIXED BINARY (31) INIT (108);
   DCL  ER_ROWNOTFND             FIXED BINARY (31) INIT (110);
   DCL  ER_FIELDNOTFND           FIXED BINARY (31) INIT (111);
   DCL  ER_NOTACT_PUTF           FIXED BINARY (31) INIT (112);
   DCL  ER_IVNUMCOLS             FIXED BINARY (31) INIT (113);
   DCL  ER_NOTACT_OUT            FIXED BINARY (31) INIT (114);
   DCL  ER_IVDDNAME              FIXED BINARY (31) INIT (115);
   DCL  ER_IVTABLDEP             FIXED BINARY (31) INIT (116);
   DCL  ER_NOTACT_SETCOL         FIXED BINARY (31) INIT (117);
   DCL  ER_IVLMAR                FIXED BINARY (31) INIT (118);
   DCL  ER_IVLINESP              FIXED BINARY (31) INIT (119);
   DCL  ER_IVRMAR                FIXED BINARY (31) INIT (120);
   DCL  ER_IVFORMAT              FIXED BINARY (31) INIT (121);
   DCL  ER_IVBXSHADE             FIXED BINARY (31) INIT (122);
   DCL  ER_IVBXSHINT             FIXED BINARY (31) INIT (123);
   DCL  ER_PUTBOX                FIXED BINARY (31) INIT (124);
   DCL  ER_IVPGWID               FIXED BINARY (31) INIT (125);
   DCL  ER_IVPGDEP               FIXED BINARY (31) INIT (126);
   DCL  ER_IVSTRLEN              FIXED BINARY (31) INIT (127);
   DCL  ER_CREATEPARA            FIXED BINARY (31) INIT (129);
   DCL  ER_IVPRSHADE             FIXED BINARY (31) INIT (130);
   DCL  ER_IVPRSHINT             FIXED BINARY (31) INIT (131);
   DCL  ER_INULLPTR              FIXED BINARY (31) INIT (132);
   DCL  ER_IVPARAFORM            FIXED BINARY (31) INIT (133);
   DCL  ER_NOTACT_PUTD           FIXED BINARY (31) INIT (134);
   DCL  ER_NOTACT_SBOX           FIXED BINARY (31) INIT (135);
   DCL  ER_IVAREAWID             FIXED BINARY (31) INIT (136);
   DCL  ER_IVAREALEN             FIXED BINARY (31) INIT (137);
   DCL  ER_IVXPOS                FIXED BINARY (31) INIT (138);
   DCL  ER_IVYPOS                FIXED BINARY (31) INIT (139);
   DCL  ER_IVTHICK               FIXED BINARY (31) INIT (140);
   DCL  ER_IVSPACE               FIXED BINARY (31) INIT (141);
   DCL  ER_IVXREF                FIXED BINARY (31) INIT (142);
   DCL  ER_IVDESCLEN             FIXED BINARY (31) INIT (143);
   DCL  ER_IVPARAOFF             FIXED BINARY (31) INIT (144);
   DCL  ER_IVPARAMAR             FIXED BINARY (31) INIT (145);
   DCL  ER_IVPARALEN             FIXED BINARY (31) INIT (146);
   DCL  ER_IVPARALSP             FIXED BINARY (31) INIT (147);
   DCL  ER_IVPARALOF             FIXED BINARY (31) INIT (148);
   DCL  ER_IVPARABOF             FIXED BINARY (31) INIT (149);
   DCL  ER_PARAEXISTS            FIXED BINARY (31) INIT (150);
   DCL  ER_IVFONT                FIXED BINARY (31) INIT (151);
   DCL  ER_IVROWID               FIXED BINARY (31) INIT (152);
   DCL  ER_TABLEXISTS            FIXED BINARY (31) INIT (153);
   DCL  ER_BEGINROW              FIXED BINARY (31) INIT (154);
   DCL  ER_ENDTABLE              FIXED BINARY (31) INIT (155);
```

```
DCL  ER_BEGINFLD             FIXED BINARY (31) INIT (156);
DCL  ER_NOTACT_SFLD          FIXED BINARY (31) INIT (157);
DCL  ER_NOTACT_EFLD          FIXED BINARY (31) INIT (158);
DCL  ER_NOTACT_SROW          FIXED BINARY (31) INIT (159);
DCL  ER_NOTACT_EROW          FIXED BINARY (31) INIT (160);
DCL  ER_NOTACT_ETBL          FIXED BINARY (31) INIT (161);
DCL  ER_NOTACT_EPAR          FIXED BINARY (31) INIT (162);
DCL  ER_ENDFLD               FIXED BINARY (31) INIT (163);
DCL  ER_ENDROW               FIXED BINARY (31) INIT (164);
DCL  ER_ENDPARA              FIXED BINARY (31) INIT (165);
DCL  ER_IVPARADEP            FIXED BINARY (31) INIT (166);
DCL  ER_IVBOXWIDTH           FIXED BINARY (31) INIT (167);
DCL  ER_IVBOXDEPTH           FIXED BINARY (31) INIT (168);
DCL  ER_IVRULELEN            FIXED BINARY (31) INIT (169);
DCL  ER_IVTBLTOP             FIXED BINARY (31) INIT (170);
DCL  ER_IVTBLBOT             FIXED BINARY (31) INIT (171);
DCL  ER_IVTBLLFT             FIXED BINARY (31) INIT (172);
DCL  ER_IVFLDBOT             FIXED BINARY (31) INIT (173);
DCL  ER_IVFLDLFT             FIXED BINARY (31) INIT (174);
DCL  ER_IVFLDRGHT            FIXED BINARY (31) INIT (175);
DCL  ER_IVCOLWID             FIXED BINARY (31) INIT (176);
DCL  ER_FONTDEFS             FIXED BINARY (31) INIT (177);
DCL  ER_ENDPAGE              FIXED BINARY (31) INIT (178);
DCL  ER_ENDDOC               FIXED BINARY (31) INIT (179);
DCL  ER_INCPSEG              FIXED BINARY (31) INIT (180);
DCL  ER_IVUNITP              FIXED BINARY (31) INIT (181);
DCL  ER_IVXPOSP              FIXED BINARY (31) INIT (182);
DCL  ER_IVYPOSP              FIXED BINARY (31) INIT (183);
DCL  ER_IVCOLORP             FIXED BINARY (31) INIT (184);
DCL  ER_IVINLINE             FIXED BINARY (31) INIT (185);
DCL  ER_IVBLKP               FIXED BINARY (31) INIT (186);
DCL  ER_IVCSPACE             FIXED BINARY (31) INIT (187);
DCL  ER_NOTACT_SETWSP        FIXED BINARY (31) INIT (188);
DCL  ER_NOTACT_SETISP        FIXED BINARY (31) INIT (189);
DCL  ER_PUTTEXT              FIXED BINARY (31) INIT (190);
DCL  ER_IVPARAIND            FIXED BINARY (31) INIT (191);
DCL  ER_INVMM                FIXED BINARY (31) INIT (192);
DCL  ER_NOTACT_INVMM         FIXED BINARY (31) INIT (193);
DCL  ER_IVROWDEP             FIXED BINARY (31) INIT (194);
DCL  ER_NOTACT_SLIBS         FIXED BINARY (31) INIT (195);
DCL  ER_SETLIBS              FIXED BINARY (31) INIT (196);
DCL  ER_IVPSEGLIB            FIXED BINARY (31) INIT (197);
DCL  ER_IVOBJLIB             FIXED BINARY (31) INIT (198);
DCL  ER_IVFONTLIB            FIXED BINARY (31) INIT (199);
DCL  ER_NO_STORAGE           FIXED BINARY (31) INIT (200);
DCL  ER_READ_LIB             FIXED BINARY (31) INIT (201);
DCL  ER_TOO_WIDE             FIXED BINARY (31) INIT (202);
DCL  ER_NO_SHADE             FIXED BINARY (31) INIT (203);
DCL  ER_IVREQUEST            FIXED BINARY (31) INIT (204);
DCL  ER_QFONT_NOTFOUND       FIXED BINARY (31) INIT (206);
DCL  ER_END_OF_PAGE          FIXED BINARY (31) INIT (207);
DCL  ER_NO_FORMATTER_HANDLE  FIXED BINARY (31) INIT (208);
DCL  ER_ROW_TOO_DEEP         FIXED BINARY (31) INIT (209);
DCL  ER_WRITE_OUTPUT         FIXED BINARY (31) INIT (210);
DCL  ER_TOO_BIG              FIXED BINARY (31) INIT (211);
DCL  ER_FONTINDEX            FIXED BINARY (31) INIT (212);
DCL  ER_DEPTH_EXCEEDED       FIXED BINARY (31) INIT (213);
DCL  ER_STARTFONT            FIXED BINARY (31) INIT (214);
DCL  ER_NO_DEFINITION        FIXED BINARY (31) INIT (215);
DCL  ER_NO_OBJECT            FIXED BINARY (31) INIT (216);
DCL  ER_INVFONT              FIXED BINARY (31) INIT (217);
DCL  ER_CODEPAGE             FIXED BINARY (31) INIT (218);
DCL  ER_OFF_PAGE             FIXED BINARY (31) INIT (219);
DCL  ER_FONTSIZE             FIXED BINARY (31) INIT (220);
DCL  ER_AREA_OFF_PAGE        FIXED BINARY (31) INIT (221);
DCL  ER_INVPSEG              FIXED BINARY (31) INIT (222);
DCL  ER_LOADMOD              FIXED BINARY (31) INIT (223);
DCL  ER_REPLACE              FIXED BINARY (31) INIT (224);
DCL  ER_FIELDNDEF            FIXED BINARY (31) INIT (225);
DCL  ER_NESTGRPS             FIXED BINARY (31) INIT (226);
DCL  ER_NOBEGGRP             FIXED BINARY (31) INIT (227);
DCL  ER_NOACTGRP             FIXED BINARY (31) INIT (228);
DCL  ER_INVSUBROW            FIXED BINARY (31) INIT (229);
DCL  ER_INCOBJ               FIXED BINARY (31) INIT (260);
DCL  ER_TRACE                FIXED BINARY (31) INIT (261);
DCL  ER_IVOBJWIDTH           FIXED BINARY (31) INIT (262);
DCL  ER_IVOBJDEPTH           FIXED BINARY (31) INIT (263);
DCL  ER_IVOBJROT             FIXED BINARY (31) INIT (264);
DCL  ER_IVOBJMAP             FIXED BINARY (31) INIT (265);
DCL  ER_IVOBJXPOS            FIXED BINARY (31) INIT (266);
DCL  ER_IVOBJYPOS            FIXED BINARY (31) INIT (267);
DCL  ER_NOTACT_INCOBJ        FIXED BINARY (31) INIT (268);
DCL  ER_BEGGRP               FIXED BINARY (31) INIT (269);
DCL  ER_IVGRPNAME            FIXED BINARY (31) INIT (270);
DCL  ER_ENDGRP               FIXED BINARY (31) INIT (271);
DCL  ER_PUTTAG               FIXED BINARY (31) INIT (272);
DCL  ER_IVTAGNAME            FIXED BINARY (31) INIT (273);
DCL  ER_IVTAGVALUE           FIXED BINARY (31) INIT (274);
DCL  ER_NOTACT_BGRP          FIXED BINARY (31) INIT (275);
DCL  ER_NOTACT_EGRP          FIXED BINARY (31) INIT (276);
DCL  ER_NOTACT_PTAG          FIXED BINARY (31) INIT (277);
DCL  ER_ABORT                FIXED BINARY (31) INIT (278);
DCL  ER_TERMINATE            FIXED BINARY (31) INIT (278);
DCL  ER_LINELEN_OVERF        FIXED BINARY (31) INIT (279);
DCL  ER_GETOUT               FIXED BINARY (31) INIT (280);
DCL  ER_NOTACT_GBUF          FIXED BINARY (31) INIT (281);
DCL  ER_IVBUFFER             FIXED BINARY (31) INIT (282);
DCL  ER_QSTR                 FIXED BINARY (31) INIT (284);
DCL  ER_QSTR_IVSTRLEN        FIXED BINARY (31) INIT (285);
DCL  ER_FORMATTER_ABEND      FIXED BINARY (31) INIT (255);
```

APQPVAR

```
/*****************************************************************/
/* COPY BOOK --- APQPVAR                                         */
/*                                                               */
/* This copy book contains the PL/1 variables used by the AFP API*/
/*****************************************************************/

    /************************************************/
    /* Declare statements for the AFP API subroutines */
    /************************************************/

    DCL AFPBDOC EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPBFLD EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPBGRP EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPBPAG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPBPAR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPBROW EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPBTBL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPCARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPDFLD EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPDFNT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPDROW EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEDOC EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEFLD EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEGRP EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEND  EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEPAG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEPAR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPEROW EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPETBL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPGBUF EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPINIT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPINVM EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPIOBJ EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPIOVL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPIPSG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPPARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPPBOX EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPPCHS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPPRUL EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPPTAG EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPPTXT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPQATT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPQPOS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPQSTR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSCLR EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSFNT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSICS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSLIB EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSOUT EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSPOS EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSRTH EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSUNI EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPSWSP EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPTERM EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);
    DCL AFPXARE EXTERNAL ENTRY OPTIONS(ASSEMBLER INTER, RETCODE);

                /*** PROGRAM VARIABLES ***/
    DCL INDEX                       BUILTIN;
    DCL LENGTH                      BUILTIN;
    DCL SUBSTR                      BUILTIN;
    DCL AFP_ERRDATA                 CHAR (25) INIT (' ');
    DCL AFP_STRING_IN               CHAR (160) VARYING;

                /*** AFPAPI VARIABLES ***/
    DCL AFP_CURRENT_HANDLE          FIXED BIN(31) INIT(0);
    DCL AFP_RET_CODE                FIXED BIN(31) INIT(0);
    DCL AFP_SEVERITY_CODE           FIXED BIN(31) INIT(0);

                /*** SHADING VARIABLES ***/
    DCL AFP_SHADING_PATTERN         FIXED BIN(31) INIT(0);
    DCL AFP_SHADING_INTENSITY       FIXED BIN(31) INIT(0);

                /*** AFPBDOC VARIABLES ***/
    DCL AFP_UNIT_OF_MEASURE         FIXED BIN(31) INIT(0);
    DCL AFP_DOC_PAGE_WIDTH          FIXED BIN(31) INIT(0);
    DCL AFP_DOC_PAGE_DEPTH          FIXED BIN(31) INIT(0);
    DCL AFP_PAGE_ORIENTATION        FIXED BIN(31) INIT(0);
    DCL AFP_DOCUMENT_HANDLE         FIXED BIN(31) INIT(0);

                /*** AFPBGRP VARIABLES ***/
    DCL AFP_GROUP_NAME              CHAR (64) INIT(' ');

                /*** AFPBPAG VARIABLES ***/
    DCL AFP_PAGE_WIDTH              FIXED BIN(31) INIT(0);
    DCL AFP_PAGE_DEPTH              FIXED BIN(31) INIT(0);
    DCL AFP_PAGE_HANDLE             FIXED BIN(31) INIT(0);

                /*** AFPBPAR VARIABLES ***/
    DCL AFP_FIRST_LINE_INDENT       FIXED BIN(31) INIT(0);
    DCL AFP_FORMAT_OPTION           FIXED BIN(31) INIT(0);
    DCL AFP_FIRST_LINE_OFFSET       FIXED BIN(31) INIT(0);
    DCL AFP_LEFT_MARGIN             FIXED BIN(31) INIT(0);
    DCL AFP_LINE_LENGTH             FIXED BIN(31) INIT(0);
    DCL AFP_LINE_SPACING            FIXED BIN(31) INIT(0);
    DCL AFP_PARAGRAPH_FRAME         FIXED BIN(31) INIT(0);
    DCL AFP_RT_RULE_OFFSET          FIXED BIN(31) INIT(0);
    DCL AFP_BOT_RULE_OFFSET         FIXED BIN(31) INIT(0);
    DCL AFP_PARAGRAPH_HANDLE        FIXED BIN(31) INIT(0);

                /*** AFPBTBL VARIABLES ***/
    DCL AFP_TABLE_WIDTH             FIXED BIN(31) INIT(0);
    DCL AFP_MAX_TABLE_DEPTH         FIXED BIN(31) INIT(0);
    DCL AFP_TABLE_ROTATION          FIXED BIN(31) INIT(0);
    DCL AFP_TABLE_HANDLE            FIXED BIN(31) INIT(0);

                /*** AFPCARE VARIABLES ***/
    DCL AFP_AREA_WIDTH              FIXED BIN(31) INIT(0);
    DCL AFP_MAX_AREA_DEPTH          FIXED BIN(31) INIT(0);
    DCL AFP_AREA_FRAME              FIXED BIN(31) INIT(0);
    DCL AFP_AREA_HANDLE             FIXED BIN(31) INIT(0);

                /*** AFPDFLD VARIABLES ***/
    DCL AFP_ALIGNMENT_POSITION      FIXED BIN(31) INIT(0);
    DCL AFP_VERTICAL_FORMAT         FIXED BIN(31) INIT(0);
    DCL AFP_RIGHT_MARGIN            FIXED BIN(31) INIT(0);
    DCL AFP_TEXT_ORIENTATION        FIXED BIN(31) INIT(0);
    DCL AFP_TOP_THICKNESS           FIXED BIN(31) INIT(0);
    DCL AFP_BOTTOM_THICKNESS        FIXED BIN(31) INIT(0);
    DCL AFP_LEFT_THICKNESS          FIXED BIN(31) INIT(0);
    DCL AFP_RIGHT_THICKNESS         FIXED BIN(31) INIT(0);
    DCL AFP_FIELD_ID                FIXED BIN(31) INIT(0);

                /*** AFPDFNT VARIABLES ***/
    DCL AFP_CODE_PAGE               CHAR (8)      INIT(' ');
    DCL AFP_DESC_NAME_LENGTH        FIXED BIN(31) INIT(0);
    DCL AFP_DESCRIPTIVE_NAME        CHAR (32)     INIT(' ');
    DCL AFP_POINT_SIZE              FIXED BIN(31) INIT(0);
    DCL AFP_WEIGHT                  FIXED BIN(31) INIT(0);
    DCL AFP_FONT_WIDTH              FIXED BIN(31) INIT(0);
    DCL AFP_ROTATION                FIXED BIN(31) INIT(0);
    DCL AFP_STYLE                   FIXED BIN(31) INIT(0);
    DCL AFP_FONT_ID                 CHAR (9)      INIT(' ');

                /*** AFPDROW VARIABLES ***/
    DCL AFP_NUMBER_COLUMNS          FIXED BIN(31) INIT(1);
    DCL AFP_NUMBER_SUBROWS          FIXED BIN(31) INIT(1);
    DCL AFP_ROW_ID                  CHAR (9)      INIT(' ');
    DCL AFP_MIN_SUBROW_DEPTH_ARRAY (64) FIXED BIN(31) INIT((64)0);
    DCL AFP_COLUMN_WIDTH_ARRAY     (64) FIXED BIN(31) INIT ((64)0);
```

```
/* DCL AFP_ROW_ARRANGEMENT_ARRAY (m,n) FIXED BIN(31) INIT((64)0)   */
/*                                                                 */
/* Note - This array must be defined in the user's PL/1 program so */
/* that its size is customized for the particular program.  The    */
/* product of the array dimensions (m times n) cannot exceed 64.   */


                /*** AFPEARE VARIABLES ***/
    DCL AFP_AREA_DEPTH              FIXED BIN(31) INIT(0);

                /*** AFPEPAR VARIABLES ***/
    DCL AFP_PARAGRAPH_DEPTH         FIXED BIN(31) INIT(0);

                /*** AFPEROW VARIABLES ***/
    DCL AFP_CURRENT_TABLE_DEPTH     FIXED BIN(31) INIT(0);

                /*** AFPETBL VARIABLES ***/
    DCL AFP_TABLE_DEPTH             FIXED BIN(31) INIT(0);

                /*** AFPGBUF VARIABLES ***/
    DCL AFP_BUFFER                  CHAR (8205)   INIT(' ');
    DCL AFP_BUFFER_LENGTH           FIXED BIN(31) INIT(0);
    DCL AFP_MORE_RECORDS            FIXED BIN(31) INIT(0);

                /*** AFPINIT VARIABLES ***/
    DCL AFPAPI_HANDLE               FIXED BIN(31) INIT(0);
    DCL AFP_TRACE                   FIXED BIN(31) INIT(0);

                /*** AFPINVM VARIABLES ***/
    DCL AFP_MEDIUM_MAP_NAME         CHAR (8)      INIT(' ');

                /*** AFPIOBJ VARIABLES ***/
    DCL AFP_OBJECT_NAME             CHAR (8)      INIT(' ');
    DCL AFP_OBJECT_WIDTH            FIXED BIN(31) INIT(0);
    DCL AFP_OBJECT_DEPTH            FIXED BIN(31) INIT(0);
    DCL AFP_OBJECT_ROTATION         FIXED BIN(31) INIT(0);
    DCL AFP_OBJECT_MAPPING_OPTION   FIXED BIN(31) INIT(0);
    DCL AFP_OBJECT_X_OFFSET         FIXED BIN(31) INIT(0);
    DCL AFP_OBJECT_Y_OFFSET         FIXED BIN(31) INIT(0);

                /*** AFPIOVL VARIABLES ***/
    DCL AFP_OVLY_NAME               CHAR (8)      INIT(' ');

                /*** AFPIPSG VARIABLES ***/
    DCL AFP_PSEG_NAME               CHAR (8)      INIT(' ');
    DCL AFP_INLINE_OPTION           FIXED BIN(31) INIT(0);
    DCL AFP_REUSE_OPTION            FIXED BIN(31) INIT(0);

                /*** AFPPARE VARIABLES ***/
    DCL AFP_AREA_ROTATION           FIXED BIN(31) INIT(0);

                /*** AFPPBOX VARIABLES ***/
    DCL AFP_BOX_WIDTH               FIXED BIN(31) INIT(0);
    DCL AFP_BOX_DEPTH               FIXED BIN(31) INIT(0);

                /*** AFPPCHS VARIABLES ***/
    DCL AFP_STRING_LENGTH           FIXED BIN(31) INIT(0);
    DCL AFP_CHARACTER_STRING        CHAR (160)    INIT(' ');
    DCL AFP_ALIGNMENT_OPTION        FIXED BIN(31) INIT(0);
    DCL AFP_ALIGNMENT_CHAR          CHAR (1)      INIT (' ');
    DCL AFP_POSITION_OPTION         FIXED BIN(31) INIT(0);
    DCL AFP_UNDERLINE               FIXED BIN(31) INIT(0);

                /*** AFPPRUL VARIABLES ***/
    DCL AFP_DIRECTION               FIXED BIN(31) INIT(0);
    DCL AFP_RULE_LENGTH             FIXED BIN(31) INIT(0);

                /*** AFPPTAG VARIABLES ***/
    DCL AFP_TAG_NAME                CHAR (64)     INIT(' ');
    DCL AFP_TAG_VALUE               CHAR (64)     INIT(' ');

                /*** AFPPTRM VARIABLES ***/
    DCL AFP_STRING_FIELD            CHAR (160)    INIT(' ');
    DCL AFP_STRING_OUT              CHAR (160)    INIT(' ');
    DCL STRING_ARRAY (160)          CHAR(1)       INIT (' ');
    DCL STILL_MORE                  BIT (1)       INIT ('1'B);
    DCL ARRAY_SUBSCRIPT             FIXED BIN(31) INIT(1);

                /*** AFPPTXT VARIABLES ***/
    DCL AFP_CONCATENATE             FIXED BIN(31) INIT(0);
    DCL AFP_REMAINING_LENGTH        FIXED BIN(31) INIT(0);
    DCL AFP_REMAINING_STRING        CHAR (160)    INIT(' ');

                /*** AFPQSTR VARIABLES ***/
    DCL AFP_MEASURED_WIDTH          FIXED BIN(31) INIT(0);

                /*** AFPSCLR VARIABLES ***/
    DCL AFP_COLOR                   FIXED BIN(31) INIT(0);

                /*** AFPSICS VARIABLES ***/
    DCL AFP_CHARACTER_SPACING       FIXED BIN(31) INIT(0);

                /*** AFPSLIB VARIABLES ***/
    DCL AFP_PSEG_LIBRARY            CHAR (8)      INIT(' ');
    DCL AFP_OBJECT_LIBRARY          CHAR (8)      INIT(' ');
    DCL AFP_FONT_LIBRARY            CHAR (8)      INIT(' ');

                /*** AFPSOUT VARIABLES ***/
    DCL AFP_OUTPUT_RECORD_SIZE      FIXED BIN(31) INIT(0);
    DCL AFP_OUTPUT_FILENAME         CHAR (8)      INIT(' ');
    DCL AFP_OUTPUT_FILETYPE         CHAR (8)      INIT(' ');
    DCL AFP_OUTPUT_FILEMODE         CHAR (2)      INIT(' ');
    DCL AFP_REPLACE                 FIXED BIN(31) INIT(0);

                /*** AFPSPOS VARIABLES ***/
    DCL AFP_X_COORDINATE            FIXED BIN(31) INIT(0);
    DCL AFP_X_REF_COORD_SYS         FIXED BIN(31) INIT(0);
    DCL AFP_Y_COORDINATE            FIXED BIN(31) INIT(0);
    DCL AFP_Y_REF_COORD_SYS         FIXED BIN(31) INIT(0);

                /*** AFPSRTH VARIABLES ***/
    DCL AFP_RULE_THICKNESS          FIXED BIN(31) INIT(0);

                /*** AFPSWSP VARIABLES ***/
    DCL AFP_WORD_SPACING            FIXED BIN(31) INIT(0);
```

APQPPRF

```
/*************************************************************/
/*                                                           */
/* COPY BOOK --- APQPPRF                                     */
/*                                                           */
/* This copy book contains the PL/1 subroutines that invoke the */
/* AFP API procedures.  Upon return from an AFP API procedure,  */
/* the severity code is examined - if it is found to contain a  */
/* SEVERE or FATAL code, the name of the AFP API call in error  */
/* and its associated return code and severity code are dis-    */
/* played in SYSPRINT.  In addition, the API is terminated by   */
/* calling AFPTERM, which causes any partial page to be print-  */
/* ed, and the PL/I program is terminated by generating a STOP  */
/* statement.                                                */
/*************************************************************/


 /*********************************************/
 /*                                           */
 /*    BEGIN DOCUMENT                         */
 /*                                           */
 /*********************************************/

   DO_AFPBDOC: PROC;

        CALL AFPBDOC (AFPAPI_HANDLE,
                      AFP_UNIT_OF_MEASURE,
                      AFP_DOC_PAGE_WIDTH,
                      AFP_DOC_PAGE_DEPTH,
                      AFP_PAGE_ORIENTATION,
                      AFP_DOCUMENT_HANDLE,
                      AFP_RET_CODE,
                      AFP_SEVERITY_CODE);

        AFP_ERRDATA  = 'AFPBDOC';

        CALL CHKSUCC;

    END DO_AFPBDOC;

 /*********************************************/
 /*                                           */
 /*    BEGIN FIELD                            */
 /*                                           */
 /*********************************************/

   DO_AFPBFLD: PROC;

        CALL AFPBFLD (AFPAPI_HANDLE,
                      AFP_TABLE_HANDLE,
                      AFP_FIELD_ID,
                      AFP_RET_CODE,
                      AFP_SEVERITY_CODE);

        AFP_ERRDATA = 'AFPBFLD';

        CALL CHKSUCC;

   END DO_AFPBFLD;

 /*********************************************/
 /*                                           */
 /*    BEGIN GROUP                            */
 /*                                           */
 /*********************************************/

   DO_AFPBGRP: PROC;

        CALL AFPBGRP (AFPAPI_HANDLE,
                      AFP_DOCUMENT_HANDLE,
                      AFP_GROUP_NAME,
                      AFP_RET_CODE,
                      AFP_SEVERITY_CODE);

        AFP_ERRDATA = 'AFPBGRP';

        CALL CHKSUCC;

   END DO_AFPBGRP;

 /*********************************************/
 /*                                           */
 /*    BEGIN PAGE                             */
 /*                                           */
 /*********************************************/

   DO_AFPBPAG: PROC;

        CALL AFPBPAG (AFPAPI_HANDLE,
                      AFP_DOCUMENT_HANDLE,
                      AFP_PAGE_WIDTH,
                      AFP_PAGE_DEPTH,
                      AFP_PAGE_ORIENTATION,
                      AFP_PAGE_HANDLE,
                      AFP_RET_CODE,
                      AFP_SEVERITY_CODE);

        AFP_ERRDATA = 'AFPBPAG';

        CALL CHKSUCC;

   END DO_AFPBPAG;

 /*********************************************/
 /*                                           */
 /*    BEGIN A PARAGRAPH                      */
 /*                                           */
 /*********************************************/

   DO_AFPBPAR: PROC;

        CALL AFPBPAR (AFPAPI_HANDLE,
                      AFP_CURRENT_HANDLE,
                      AFP_FIRST_LINE_INDENT,
                      AFP_FORMAT_OPTION,
                      AFP_FIRST_LINE_OFFSET,
                      AFP_LEFT_MARGIN,
                      AFP_LINE_LENGTH,
                      AFP_LINE_SPACING,
                      AFP_PARAGRAPH_FRAME,
                      AFP_RT_RULE_OFFSET,
                      AFP_BOT_RULE_OFFSET,
                      AFP_SHADING_PATTERN,
                      AFP_SHADING_INTENSITY,
                      AFP_PARAGRAPH_HANDLE,
                      AFP_RET_CODE,
                      AFP_SEVERITY_CODE);

        AFP_ERRDATA = 'AFPBPAR';

        CALL CHKSUCC;

   END DO_AFPBPAR;

 /*********************************************/
 /*                                           */
 /*    BEGIN ROW                              */
 /*                                           */
 /*********************************************/

   DO_AFPBROW: PROC;

        CALL AFPBROW (AFPAPI_HANDLE,
                      AFP_TABLE_HANDLE,
                      AFP_ROW_ID,
                      AFP_RET_CODE,
                      AFP_SEVERITY_CODE);

        AFP_ERRDATA = 'AFPBROW';

        CALL CHKSUCC;

   END DO_AFPBROW;
```

```
/*********************************************/
/*                                           */
/*    BEGIN TABLE                            */
/*                                           */
/*********************************************/

  DO_AFPBTBL: PROC;

       CALL AFPBTBL (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_TABLE_WIDTH,
                     AFP_MAX_TABLE_DEPTH,
                     AFP_TABLE_ROTATION,
                     AFP_TOP_THICKNESS,
                     AFP_BOTTOM_THICKNESS,
                     AFP_LEFT_THICKNESS,
                     AFP_RIGHT_THICKNESS,
                     AFP_TABLE_HANDLE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPBTBL';

       CALL CHKSUCC;

  END DO_AFPBTBL;

/*********************************************/
/*                                           */
/*    CREATE AN AREA                         */
/*                                           */
/*********************************************/

  DO_AFPCARE: PROC;

       CALL AFPCARE (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_AREA_WIDTH,
                     AFP_MAX_AREA_DEPTH,
                     AFP_AREA_FRAME,
                     AFP_SHADING_PATTERN,
                     AFP_SHADING_INTENSITY,
                     AFP_AREA_HANDLE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPCARE';

       CALL CHKSUCC;

  END DO_AFPCARE;

/*********************************************/
/*                                           */
/*    DEFINE FIELD                           */
/*                                           */
/*********************************************/

  DO_AFPDFLD: PROC;

       CALL AFPDFLD (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_FORMAT_OPTION,
                     AFP_ALIGNMENT_POSITION,
                     AFP_VERTICAL_FORMAT,
                     AFP_LEFT_MARGIN,
                     AFP_RIGHT_MARGIN,
                     AFP_LINE_SPACING,
                     AFP_TEXT_ORIENTATION,
                     AFP_SHADING_PATTERN,
                     AFP_SHADING_INTENSITY,
                     AFP_TOP_THICKNESS,
                     AFP_BOTTOM_THICKNESS,
                     AFP_LEFT_THICKNESS,
                     AFP_RIGHT_THICKNESS,
                     AFP_FIELD_ID,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPDFLD';

       CALL CHKSUCC;

  END DO_AFPDFLD;

/*********************************************/
/*                                           */
/*    DEFINE FONT BY ATTRIBUTE               */
/*                                           */
/*********************************************/

  DO_AFPDFNT: PROC;

       CALL AFPDFNT (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_CODE_PAGE,
                     AFP_DESC_NAME_LENGTH,
                     AFP_DESCRIPTIVE_NAME,
                     AFP_POINT_SIZE,
                     AFP_WEIGHT,
                     AFP_FONT_WIDTH,
                     AFP_ROTATION,
                     AFP_STYLE,
                     AFP_FONT_ID,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPDFNT';

       CALL CHKSUCC;

  END DO_AFPDFNT;

/*********************************************/
/*                                           */
/*    DEFINE ROW                             */
/*                                           */
/*********************************************/

  DO_AFPDROW: PROC;

       CALL AFPDROW (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_MIN_SUBROW_DEPTH_ARRAY,
                     AFP_TOP_THICKNESS,
                     AFP_BOTTOM_THICKNESS,
                     AFP_NUMBER_COLUMNS,
                     AFP_NUMBER_SUBROWS,
                     AFP_ROW_ARRANGE_ARRAY,
                     AFP_COLUMN_WIDTH_ARRAY,
                     AFP_ROW_ID,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPDROW';

       CALL CHKSUCC;

  END DO_AFPDROW;

/*********************************************/
/*                                           */
/*    END AREA                               */
/*                                           */
/*********************************************/

  DO_AFPEARE: PROC;

       CALL AFPEARE (AFPAPI_HANDLE,
                     AFP_AREA_HANDLE,
                     AFP_AREA_DEPTH,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPEARE';

       CALL CHKSUCC;

  END DO_AFPEARE;
```

```
/*********************************************/
/*                                           */
/*    END DOCUMENT                           */
/*                                           */
/*********************************************/

  DO_AFPEDOC: PROC;

       CALL AFPEDOC (AFPAPI_HANDLE,
                     AFP_DOCUMENT_HANDLE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPEDOC';

       CALL CHKSUCC;

  END DO_AFPEDOC;

/*********************************************/
/*                                           */
/*    END FIELD                              */
/*                                           */
/*********************************************/

  DO_AFPEFLD: PROC;

       CALL AFPEFLD (AFPAPI_HANDLE,
                     AFP_TABLE_HANDLE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPEFLD';

       CALL CHKSUCC;

  END DO_AFPEFLD;

/*********************************************/
/*                                           */
/*    END GROUP                              */
/*                                           */
/*********************************************/

  DO_AFPEGRP: PROC;

       CALL AFPEGRP (AFPAPI_HANDLE,
                     AFP_DOCUMENT_HANDLE,
                     AFP_GROUP_NAME,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPEGRP';

       CALL CHKSUCC;

  END DO_AFPEGRP;

/*********************************************/
/*                                           */
/*    END THE AFPAPI                         */
/*                                           */
/*********************************************/

  DO_AFPEND: PROC;

       CALL AFPEND (AFPAPI_HANDLE,
                    AFP_RET_CODE,
                    AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPEND';

       CALL CHKSUCC;

  END DO_AFPEND;
```

```
/*********************************************/
/*                                           */
/*    END PAGE                               */
/*                                           */
/*********************************************/

  DO_AFPEPAG: PROC;

       CALL AFPEPAG (AFPAPI_HANDLE,
                     AFP_PAGE_HANDLE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPEPAG';

       CALL CHKSUCC;

  END DO_AFPEPAG;

/*********************************************/
/*                                           */
/*    END PARAGRAPH                          */
/*                                           */
/*********************************************/

  DO_AFPEPAR: PROC;

       CALL AFPEPAR (AFPAPI_HANDLE,
                     AFP_PARAGRAPH_HANDLE,
                     AFP_PARAGRAPH_DEPTH,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPEPAR';

       CALL CHKSUCC;

  END DO_AFPEPAR;

/*********************************************/
/*                                           */
/*    END ROW                                */
/*                                           */
/*********************************************/

  DO_AFPEROW: PROC;

       CALL AFPEROW (AFPAPI_HANDLE,
                     AFP_TABLE_HANDLE,
                     AFP_CURRENT_TABLE_DEPTH,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'APFEROW';

       CALL CHKSUCC;

  END DO_AFPEROW;

/*********************************************/
/*                                           */
/*    END TABLE                              */
/*                                           */
/*********************************************/

  DO_AFPETBL: PROC;

       CALL AFPETBL (AFPAPI_HANDLE,
                     AFP_TABLE_HANDLE,
                     AFP_TABLE_DEPTH,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPETBL';

       CALL CHKSUCC;

  END DO_AFPETBL;
```

```
/*************************************************/
/*                                               */
/*    GET BUFFER                                 */
/*                                               */
/*************************************************/

  DO_AFPGBUF: PROC;

       CALL AFPGBUF (AFPAPI_HANDLE,
                     AFP_DOCUMENT_HANDLE,
                     AFP_BUFFER,
                     AFP_BUFFER_LENGTH,
                     AFP_MORE_RECORDS,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPGBUF';

       CALL CHKSUCC;

  END DO_AFPGBUF;

/*************************************************/
/*                                               */
/*    INITIALIZE THE AFPAPI                      */
/*                                               */
/*************************************************/

  DO_AFPINIT: PROC;

       CALL AFPINIT (AFPAPI_HANDLE,
                     AFP_TRACE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPINIT';

       CALL CHKSUCC;

  END DO_AFPINIT;

/*************************************************/
/*                                               */
/*    INVOKE A MEDIUM MAP                        */
/*                                               */
/*************************************************/

  DO_AFPINVM: PROC;

       CALL AFPINVM (AFPAPI_HANDLE,
                     AFP_DOCUMENT_HANDLE,
                     AFP_MEDIUM_MAP_NAME,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPINVM';

       CALL CHKSUCC;

  END DO_AFPINVM;
```

```
/*************************************************/
/*                                               */
/*    INCLUDE AN OBJECT                          */
/*                                               */
/*************************************************/

  DO_AFPIOBJ: PROC;

       CALL AFPIOBJ (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_OBJECT_NAME,
                     AFP_OBJECT_WIDTH,
                     AFP_OBJECT_DEPTH,
                     AFP_OBJECT_ROTATION,
                     AFP_OBJECT_MAPPING_OPTION,
                     AFP_OBJECT_X_OFFSET,
                     AFP_OBJECT_Y_OFFSET,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPIOBJ';

       CALL CHKSUCC;

  END DO_AFPIOBJ;

/*************************************************/
/*                                               */
/*    INCLUDE A PAGE OVERLAY                     */
/*                                               */
/*************************************************/

  DO_AFPIOVL: PROC;

       CALL AFPIOVL (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_OVLY_NAME,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPIOVL';

       CALL CHKSUCC;

  END DO_AFPIOVL;

/*************************************************/
/*                                               */
/*    INCLUDE A PAGE SEGMENT                     */
/*                                               */
/*************************************************/

  DO_AFPIPSG: PROC;

       CALL AFPIPSG (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_PSEG_NAME,
                     AFP_INLINE_OPTION,
                     AFP_REUSE_OPTION,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPIPSG';

       CALL CHKSUCC;

  END DO_AFPIPSG;
```

```
/*********************************************/
/*                                           */
/*    PUT AN AREA                            */
/*                                           */
/*********************************************/

  DO_AFPPARE: PROC;

       CALL AFPPARE (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_AREA_HANDLE,
                     AFP_AREA_ROTATION,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPPARE';

       CALL CHKSUCC;

  END DO_AFPPARE;

/*********************************************/
/*                                           */
/*    PUT A BOX                              */
/*                                           */
/*********************************************/

  DO_AFPPBOX: PROC;

       CALL AFPPBOX (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_BOX_WIDTH,
                     AFP_BOX_DEPTH,
                     AFP_SHADING_PATTERN,
                     AFP_SHADING_INTENSITY,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPPBOX';

       CALL CHKSUCC;

  END DO_AFPPBOX;

/*********************************************/
/*                                           */
/*    PUT A CHARACTER STRING                 */
/*                                           */
/*********************************************/

  DO_AFPPCHS: PROC;

       CALL AFPPCHS (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_STRING_LENGTH,
                     AFP_CHARACTER_STRING,
                     AFP_ALIGNMENT_OPTION,
                     AFP_ALIGNMENT_CHAR,
                     AFP_POSITION_OPTION,
                     AFP_UNDERLINE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPPCHS';

       CALL CHKSUCC;

  END DO_AFPPCHS;
```

```
/*********************************************/
/*                                           */
/*    PUT RULE                               */
/*                                           */
/*********************************************/

  DO_AFPPRUL: PROC;

       CALL AFPPRUL (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_DIRECTION,
                     AFP_RULE_LENGTH,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPPRUL';

       CALL CHKSUCC;

  END DO_AFPPRUL;

/*********************************************/
/*                                           */
/*    PUT TAG                                */
/*                                           */
/*********************************************/

  DO_AFPPTAG: PROC;

       CALL AFPPTAG (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_TAG_NAME,
                     AFP_TAG_VALUE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPPTAG';

       CALL CHKSUCC;

  END DO_AFPPTAG;

/*********************************************/
/*                                           */
/*    PUT TEXT                               */
/*                                           */
/*********************************************/

  DO_AFPPTXT: PROC;

       CALL AFPPTXT (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_STRING_LENGTH,
                     AFP_CHARACTER_STRING,
                     AFP_CONCATENATE,
                     AFP_UNDERLINE,
                     AFP_REMAINING_LENGTH,
                     AFP_REMAINING_STRING,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPPTXT';

       CALL CHKSUCC;

  END DO_AFPPTXT;
```

```
/*********************************************/
/*                                           */
/*    QUERY CURRENT ATTRIBUTES               */
/*                                           */
/*********************************************/

  DO_AFPQATT: PROC;

       CALL AFPQATT (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_UNIT_OF_MEASURE,
                     AFP_X_COORDINATE,
                     AFP_Y_COORDINATE,
                     AFP_COLOR,
                     AFP_RULE_THICKNESS,
                     AFP_FONT_ID,
                     AFP_CHARACTER_SPACING,
                     AFP_WORD_SPACING,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPQATT';

       CALL CHKSUCC;

  END DO_AFPQATT;

/*********************************************/
/*                                           */
/*    Query Character String Area Size       */
/*                                           */
/*********************************************/

  DO_AFPQSTR: PROC;

       CALL AFPQSTR (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_CHARACTER_STRING,
                     AFP_STRING_LENGTH,
                     AFP_MEASURED_WIDTH,
                     AFP_LINE_SPACING,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPQSTR';

       CALL CHKSUCC;

  END DO_AFPQSTR;


/*********************************************/
/*                                           */
/*    QUERY CURRENT POSITION                 */
/*                                           */
/*********************************************/

  DO_AFPQPOS: PROC;

       CALL AFPQPOS (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_X_COORDINATE,
                     AFP_Y_COORDINATE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPQPOS';

       CALL CHKSUCC;

  END DO_AFPQPOS;
```

```
/*********************************************/
/*                                           */
/*    SET COLOR                              */
/*                                           */
/*********************************************/

  DO_AFPSCLR: PROC;

       CALL AFPSCLR (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_COLOR,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSCLR';

       CALL CHKSUCC;

  END DO_AFPSCLR;

/*********************************************/
/*                                           */
/*    SET FONT                               */
/*                                           */
/*********************************************/

  DO_AFPSFNT: PROC;

       CALL AFPSFNT (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_FONT_ID,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSFNT';

       CALL CHKSUCC;

  END DO_AFPSFNT;

/*********************************************/
/*                                           */
/*    SET INTERCHARACTER SPACING             */
/*                                           */
/*********************************************/

  DO_AFPSICS: PROC;

       CALL AFPSICS (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_CHARACTER_SPACING,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSICS';

       CALL CHKSUCC;

  END DO_AFPSICS;
```

```
/*************************************************/
/*                                               */
/*    SET RESOURCE LIBRARY NAMES                 */
/*                                               */
/*************************************************/

  DO_AFPSLIB: PROC;

       CALL AFPSLIB (AFPAPI_HANDLE,
                     AFP_PSEG_LIBRARY,
                     AFP_OBJECT_LIBRARY,
                     AFP_FONT_LIBRARY,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSLIB';

       CALL CHKSUCC;

  END DO_AFPSLIB;

/*************************************************/
/*                                               */
/*    SET OUTPUT CHARACTERISTICS                 */
/*                                               */
/*************************************************/

  DO_AFPSOUT: PROC;

       CALL AFPSOUT (AFPAPI_HANDLE,
                     AFP_OUTPUT_RECORD_SIZE,
                     AFP_OUTPUT_FILENAME,
                     AFP_OUTPUT_FILETYPE,
                     AFP_OUTPUT_FILEMODE,
                     AFP_REPLACE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSOUT';

       CALL CHKSUCC;

  END DO_AFPSOUT;

/*************************************************/
/*                                               */
/*    SET POSITION                               */
/*                                               */
/*************************************************/

  DO_AFPSPOS: PROC;

       CALL AFPSPOS (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_X_COORDINATE,
                     AFP_X_REF_COORD_SYS,
                     AFP_Y_COORDINATE,
                     AFP_Y_REF_COORD_SYS,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSPOS';

       CALL CHKSUCC;

  END DO_AFPSPOS;
```

```
/*************************************************/
/*                                               */
/*    SET RULE THICKNESS                         */
/*                                               */
/*************************************************/

  DO_AFPSRTH: PROC;

       CALL AFPSRTH (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_RULE_THICKNESS,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSRTH';

       CALL CHKSUCC;

  END DO_AFPSRTH;

/*************************************************/
/*                                               */
/*    SET UNITS                                  */
/*                                               */
/*************************************************/

  DO_AFPSUNI: PROC;

       CALL AFPSUNI (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_UNIT_OF_MEASURE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSUNI';

       CALL CHKSUCC;

  END DO_AFPSUNI;

/*************************************************/
/*                                               */
/*    SET WORKSPACING                            */
/*                                               */
/*************************************************/

  DO_AFPSWSP: PROC;

       CALL AFPSWSP (AFPAPI_HANDLE,
                     AFP_CURRENT_HANDLE,
                     AFP_WORD_SPACING,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPSWSP';

       CALL CHKSUCC;

  END DO_AFPSWSP;
```

```
/*************************************************/
/*                                               */
/*    TERMINATE AFPAPI                           */
/*                                               */
/*************************************************/

  DO_AFPTERM: PROC;

       CALL AFPTERM (AFPAPI_HANDLE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPTERM';

       CALL CHKSUCC;

  END DO_AFPTERM;

/*************************************************/
/*                                               */
/*    DESTROY AN AREA                            */
/*                                               */
/*************************************************/

  DO_AFPXARE: PROC;

       CALL AFPXARE (AFPAPI_HANDLE,
                     AFP_AREA_HANDLE,
                     AFP_RET_CODE,
                     AFP_SEVERITY_CODE);

       AFP_ERRDATA = 'AFPXARE';

       CALL CHKSUCC;

  END DO_AFPXARE;
```

```
/*************************************************/
/*                                               */
/*    DETERMINE SUCCESS OR FAILURE OF CALL       */
/*                                               */
/* If the call failed, display the name of the */
/* call in error, the API return code, and the */
/* API severity code in SYSPRINT.  Also, call  */
/* AFPTERM to terminate the API session and    */
/* print any partial pages that may exist.     */
/*************************************************/

  CHKSUCC: PROC;

    IF AFP_SEVERITY_CODE > WARNING THEN
      DO;
        PUT SKIP LIST (' ');
        PUT SKIP LIST (AFP_ERRDATA,' FAILED');
        PUT SKIP LIST ('AFPAPI Return Code  : ',AFP_RET_CODE);
        PUT SKIP LIST ('AFPAPI Severity Code: ',AFP_SEVERITY_CODE);

        IF AFP_ERRDATA ¬= 'AFPEND' & AFP_ERRDATA ¬= 'AFPTERM' THEN
            CALL DO_AFPTERM;

        STOP;
      END;

  END CHKSUCC;
```

Appendix B. Related Publications

The following publications may help you understand the information in this publication.

Advanced Function Presentation

Title	Order Number
Guide to Advanced Function Presentation Contains an overview of AFP concepts and products	G544-3876
AFP Application Programming Interface: COBOL Language Reference Contains COBOL language bindings for using AFP API.	S544-3873
AFP Application Programming Interface: Programming Guide and Reference Contains AFP API concepts and how to use the product.	S544-3872
Advanced Function Presentation: Printer Information Contains details about AFP printers	G544-3290
Page Printer Formatting Aid/370: User's Guide and Reference Contains information about the PPFA/370 product used to create AFP page definitions and form definitions	G544-3181
AFP Workbench for Windows: Using the Viewer Application Contains information about using Workbench with AFP API	G544-3813
AFP Conversion and Indexing Facility: Application Programming Guide Contains information about using AFP Conversion and Indexing Facility	G544-3824
Printing and Publishing Collection Kit Contains the BookManager versions of many AFP publications	SK2T-2921

Fonts

Title	Order Number
IBM AFP Fonts: Introduction to Typography	G544-3122
IBM AFP Fonts: Technical Reference for Code Pages	S544-3802
IBM AFP Fonts: Technical Reference for IBM Expanded Core Fonts	S544-5228
IBM AFP Fonts: Font Samples	S544-3792
Advanced Function Printing: Host Font Data Stream Reference	S544-3289

Architecture

Title	Order Number
Mixed Object Document Content Architecture Reference Contains the definition of the Mixed Object Document Content Architecture and its functions and elements.	SC31-6802
Advanced Function Presentation: Programming Guide and Line Data Reference Contains information about processing line and mixed data, page definitions, and the X'5A' prefix on structured fields.	S544-3884
Font Object Content Architecture Reference	S544-3285
Image Object Content Architecture Reference	SC31-6805
Intelligent Printer Data Stream Reference	S544-3417
Graphics Object Content Architecture Reference	SC31-6804
Presentation Text Object Content Architecture Reference	SC31-6803

PSF/MVS

Title	Order Number
Print Services Facility/MVS: Application Programming Guide	S544-3673
Print Services Facility/MVS: System Programming Guide	S544-3672
Program Directory for Print Services Facility/MVS	None

PSF/VM

Title	Order Number
Print Services Facility/VM: Application Programming Guide	S544-3677
Print Services Facility/VM: System Programming Guide	S544-3680
Program Directory for Print Services Facility/VM	None

Readers′ Comments — We′d Like to Hear from You

Advanced Function Presentation
Application Programming Interface:
PL/1 Language Reference

Publication No. S544-3874-01

Use this form to provide comments about this publication, its organization, or subject matter. Understand that IBM may use the information any way it believes appropriate, without incurring any obligation to you. Your comments will be sent to the author′s department for the appropriate action. Comments may be written in your language.

Note: IBM publications are not stocked at the location to which this form is addressed. Direct requests for publications or for assistance in using your IBM system, to your IBM representative or local IBM branch office.

	Yes	No
• Does the publication meet your needs?	____	____
• Did you find the information:		
Accurate?	____	____
Easy to read and understand?	____	____
Easy to retrieve?	____	____
Organized for convenient use?	____	____
Legible?	____	____
Complete?	____	____
Well illustrated?	____	____
Written for your technical level?	____	____
• Do you use this publication:		
As an introduction to the subject?	____	____
As a reference manual?	____	____
As an instructor in class?	____	____
As a student in class?	____	____
• What is your occupation?		

Thank you for your input and cooperation.

Note: You may either send your comments by fax to 1-800-524-1519, or mail your comments. If mailed in the U.S.A., no postage stamp is necessary. For residents outside the U.S.A., your local IBM office or representative will forward your comments.

Comments:

Name

Address

Company or Organization

Phone No.

Readers′ Comments — We′d Like to Hear from You
S544-3874-01

Cut or Fold
Along Line

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

Information Development
The IBM Printing Systems Company
Department H7FE Building 003G
P O Box 1900
BOULDER CO 80301-9817

Fold and Tape **Please do not staple** Fold and Tape

S544-3874-01

Cut or Fold
Along Line

IBM®

File Number: S370-49

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

S544-3874-01