SAS Certification Prep Guide: Base Programming for SAS 9, Third Edition

by SAS Institute

Chapter 3: Editing and Debugging SAS Programs

Overview

Introduction

Now that you're familiar with the basics, you can learn how to use the SAS programming windows to edit and debug programs effectively.

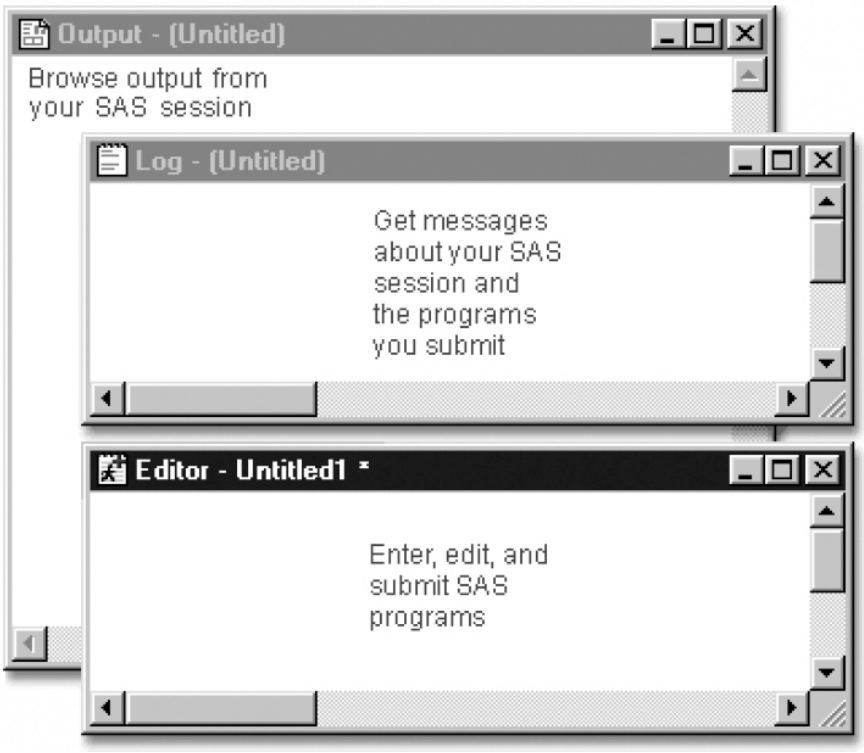

Figure 3.1: SAS Programming Windows

Objectives

In this chapter you learn to

- open a stored SAS program
- edit SAS programs
- clear SAS programming windows
- interpret error messages in the SAS log
- correct errors
- resolve common problems.

Opening a Stored SAS Program

Overview

You can open a stored program in the code editing window, where you can edit the program and submit it again.

You can open a program using

- file shortcuts

- My Favorite Folders
- the Open window
- the INCLUDE command.

Additional Note In operating environments that provide drag-and-drop functionality, you can also open a SAS file by dragging and dropping it into the code editing window.

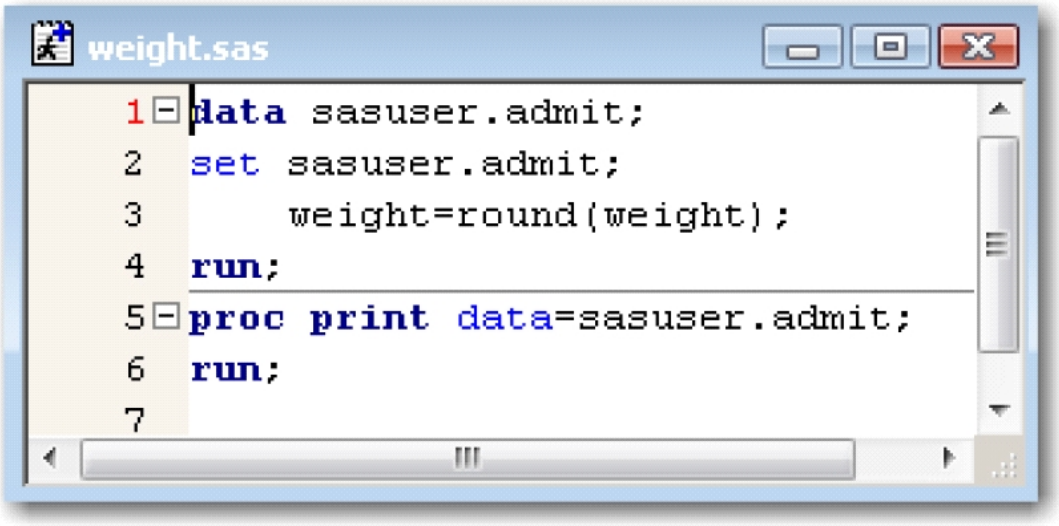

Figure 3.2: Code Editor Window

Using File Shortcuts

File shortcuts are available in the SAS Explorer window. To open a program using file shortcuts:

1. At the top level of the SAS Explorer window, double-click **File Shortcuts**.
2. Double-click the file shortcut that you want to open, or select **Open** from the pop-up menu for the file.

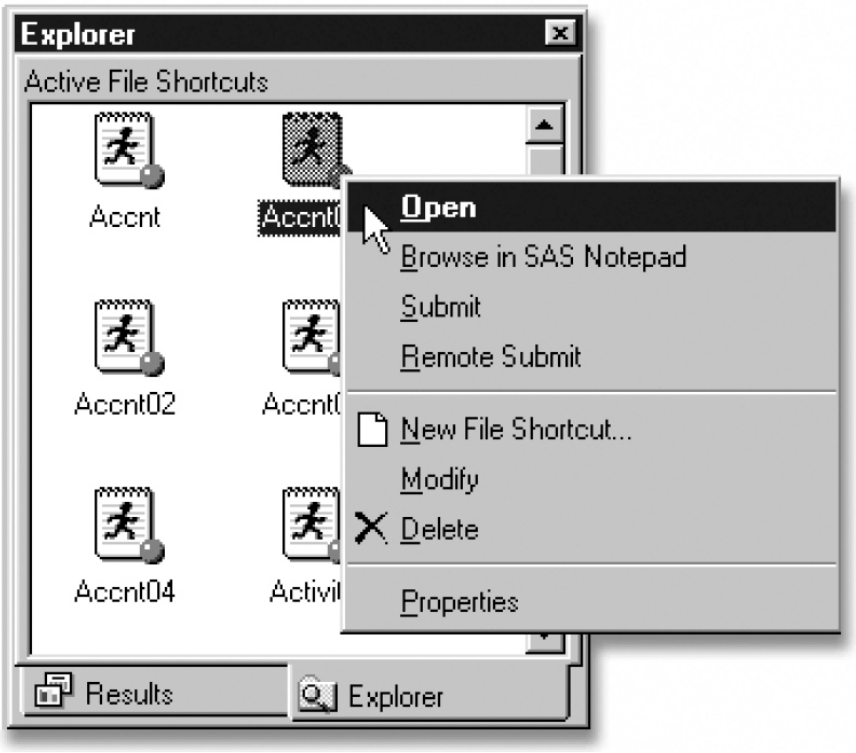

Figure 3.3: File Short Cuts

The file opens in a new code editing window, with the filename shown as the window title.

Additional Note You can select **Submit** from the pop-up menu to execute the file directly from **File Shortcuts**.

Additional Note Remember that you open **File Shortcuts** with a single-click in the z/OS (z/OS) or CMS operating environments. To open a file, you type

- ? beside the item to display pop-up menus
- *S* beside the item to simulate a double-click.

Using My Favorite Folders

To view and manage any files in your operating environment, you can use the My Favorite Folders window.

To open a file that is stored in My Favorite Folders:

1. Select **View** ⇨ **My Favorite Folders**.
2. Double-click the file, or select **Open** from the pop-up menu for the file.

The file opens in a new code editing window, with the filename shown as the window title.

Additional Note You can select **Submit** from the pop-up menu to execute the file directly from My Favorite Folders.

Using the Open Window

To use the Open window:

1. With the code editing window active, select **File** ⇨ **Open** (or **File** ⇨ **Open Program)**.
2. In the Open window, click the file that you want to open (or type the path for the file).
3. Click **Open** or **OK**.

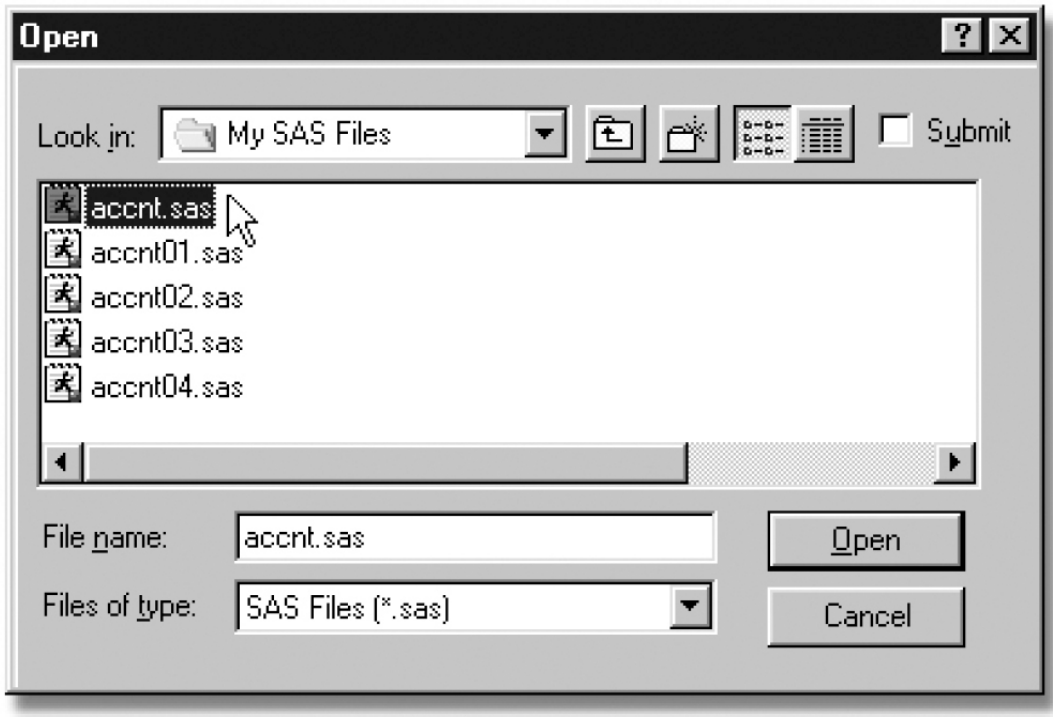

Figure 3.4: The Open WIndow

The file opens in a new code editing window, with the filename shown as the window title.

Additional Note In the Windows environment, you can submit your file directly from the Open window by clicking the **Submit** check box before clicking **Open**.

Additional Note The appearance of the Open window varies by operating environment. In mainframe operating environments, files are not listed, so you must type the path for the file.

If you're not sure of the name of the file in which your program is stored, you can do either of the following:

- use My Favorite Folders to locate the file in your operating environment
- issue the **x** command to temporarily suspend your SAS session.

  The X command enables you to use operating system facilities without ending your SAS session. When you issue the X command, you temporarily exit SAS and enter the host system.

To resume your SAS session after issuing the X command, issue the appropriate host system command, as shown below.

Table 3.1: Resuming a SAS Session

Operating Environment	Host command to resume SAS session
z/OS	RETURN or END
UNIX	exit
Windows	EXIT

Issuing an INCLUDE Command

You can also open a program by issuing an INCLUDE command.

General form, basic INCLUDE command:

INCLUDE *'file-specification'*

where *file-specification* is the physical name by which the host system recognizes the file.

Example

Suppose you want to include the program **`D:\Programs\Sas\Myprog1.sas`** in the Windows operating environment. To do so, you can issue the following INCLUDE command:

```
include 'd:\programs\sas\myprog1.sas'
```

Because this is a command (not a SAS statement), no semicolon is used at the end.

Editing SAS Programs

Now that you know how to open a SAS program, let's review the characteristics of SAS statements and look at enhancing the readability of your SAS programs.

SAS Program Structure

Remember that SAS programs consist of SAS statements. Consider the SAS program that is shown in a code editing window below.

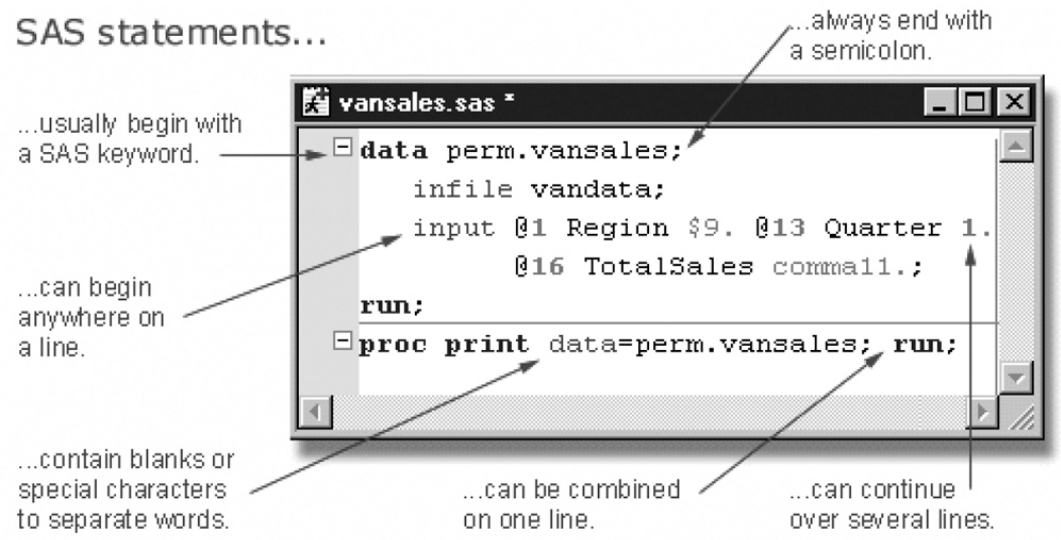

Figure 3.5: An Example SAS Program

Although you can write SAS statements in almost any format, a consistent layout enhances readability and helps you understand the program's purpose. It's a good idea to

- begin DATA and PROC steps in column one
- indent statements within a step
- begin RUN statements in column one
- include a RUN statement after every DATA step or PROC step.

```
data sashelp.prdsale;
   infile jobs;
   input date name $ job $;
run;
proc print data=sashelp.prdsale;
run;
```

Now let's look at the Enhanced Editor window features that you can use to edit your SAS programs.

Additional Note If you are running SAS in an operating environment other than Windows, you can skip theEnhanced Editor window section and see "Program Editor Features" on page 84 instead.

Using the Enhanced Editor

When you edit SAS programs in the Enhanced Editor, you can take advantage of a number of useful features. The following section shows you how to

- use color-coding to identify code elements
- automatically indent the next line when you press the **Enter** key
- collapse and expand sections of SAS procedures, DATA steps, and macros
- bookmark lines of code for easy access to different sections of your program
- open multiple views of a file.

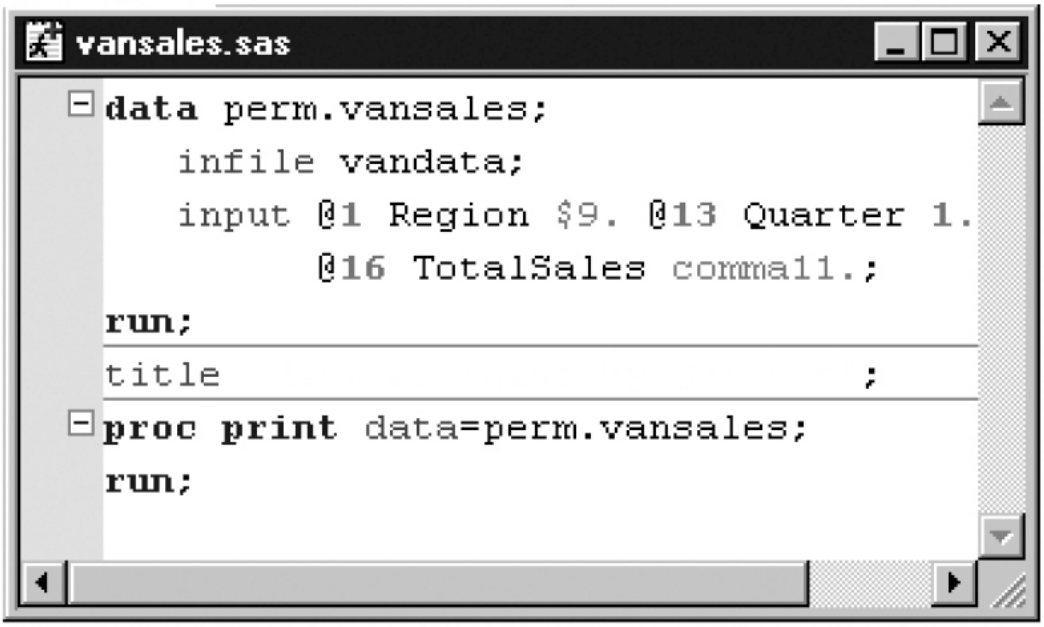

Figure 3.6: Enhanced Editor Window

You can also

- save color-coding settings in a color scheme
- create macros that record and play back program editing commands
- create shortcuts for typing in text using abbreviations
- customize keyboard shortcuts for most Enhanced Editor commands
- create and format your own keywords
- automatically reload modified disk files
- access Help for SAS procedures by placing the insertion point within the procedure name and pressing F1.

Entering and Viewing Text

The Enhanced Editor uses visual aides such as color coding and code sections to help you write and debug your SAS programs.

You can use the margin on the left side of the Editor window to

- select one or more lines of text

- expand and collapse code sections
- display bookmarks.

Finding and Replacing Text

To find text in the Editor window, select **Edit** ⇨ **Find** or press Ctrl+F. You can set commonly used find options, including specifying that the text string is a regular expression. You can also specify whether to search in the code only, in the comments only, or in both the code and comments.

Figure 3.7: The Find Window

Using Abbreviations

You can define a character string so that when you type it and then press the Tab key or the Enter key, the string expands to a longer character string.

For example, you could define the abbreviation myv7sasfiles, which would expand to `'c:\winnt\profiles\myid\personal\mysasfiles\v7';`. Abbreviations are actually macros that insert one or more lines of text.

To create an abbreviation,

1. Select **Tools** ⇨ **Add Abbreviation**.
2. For **Abbreviation**, type the name of the abbreviation.
3. For **Text to insert for abbreviation**, type the text that the abbreviation will expand into.
4. Click **OK**.

To use an abbreviation, type the abbreviation. When an abbreviation is recognized, a tooltip displays the expanded text. Press the Tab key or Enter key to accept the abbreviation.

You can modify or delete abbreviations that you create.

Opening Multiple Views of a File

When you use the Enhanced Editor, you can see different parts of the same file simultaneously by opening multiple views of the same file. While you are working with multiple views, you are working with only *one file*, not multiple copies of the same file.

To open multiple views of the same file,

1. make the file the active window.
2. select **Window** ⇨ **New Window**.

The filename in the title bar is appended with a colon and a view number. Changes that you make to a file in one view, such as changing text or bookmarking a line, occur in all views simultaneously. Actions such as scroll bar movement, text selection, and expanding or contracting a section of code occur only in the active window.

Setting Enhanced Editor Options

You can customize the Enhanced Editor by setting Enhanced Editor options. To open the Enhanced Editor Options window, activate an Editor window and select **Tools** ⇨ **Options** ⇨ **Enhanced Editor**.

Click the tabs that are located along the top of the window to navigate to the settings that you want to change, and then select the options that you want.

For example, **Show line numbers** specifies whether to display line numbers in the margin. When line numbers are displayed, the current line number is red. **Insert spaces for tabs** specifies whether to insert the space character or the tab character when you press the Tab key. If it is selected, the space character is used. If it is not selected, the tab character is used.

When you are finished, click **OK**.

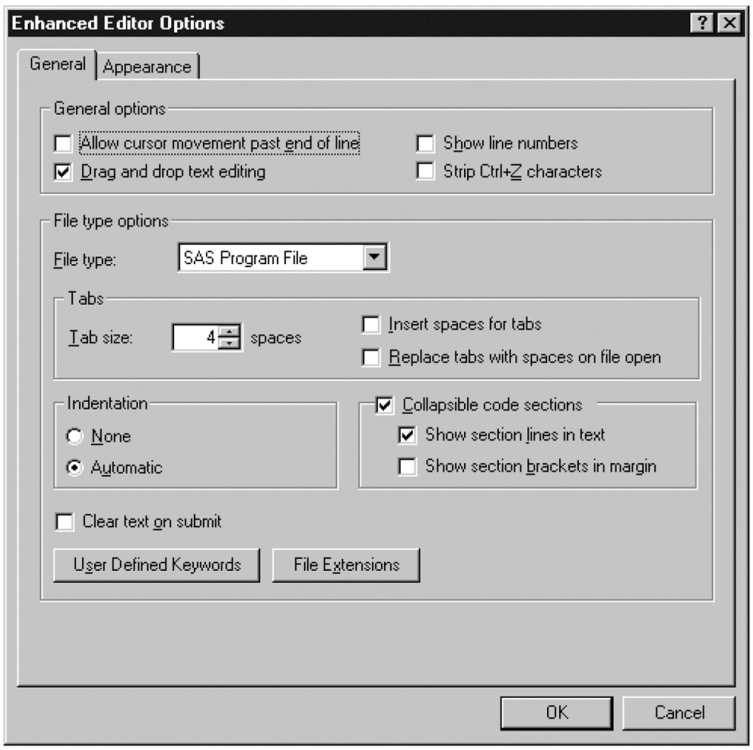

Figure 3.8: Enhanced Editor Options Window

Additional Note For more information about setting options in the Enhanced Editor, select **Help** ⇨ **Using This Window** from an Enhanced Editor window.

Program Editor Features

You use the Program Editor window to submit SAS programs. You can also enter and edit SAS programs in this window, or you can open existing SAS programs that you created in another text editor.

To edit SAS programs in the Program Editor window, you can display line numbers. Then you can use text editor line commands and block text editor line commands within the line numbers to insert, delete, move, and copy lines within the Program Editor window.

Additional Note In all operating environments, you can use line numbers, text editor line commands, and block text editor line commands. Depending on your operating environment, you can also use features such as dragging and dropping text, inserting lines with the Enter key, or marking and deleting text. For information about these features, see the SAS

documentation for your operating environment.

Line Numbers

In some operating environments, line numbers appear in the Program Editor window by default. In other systems, the use of line numbers is optional. Activating line numbers can make it easier for you to edit your program regardless of your operating environment.

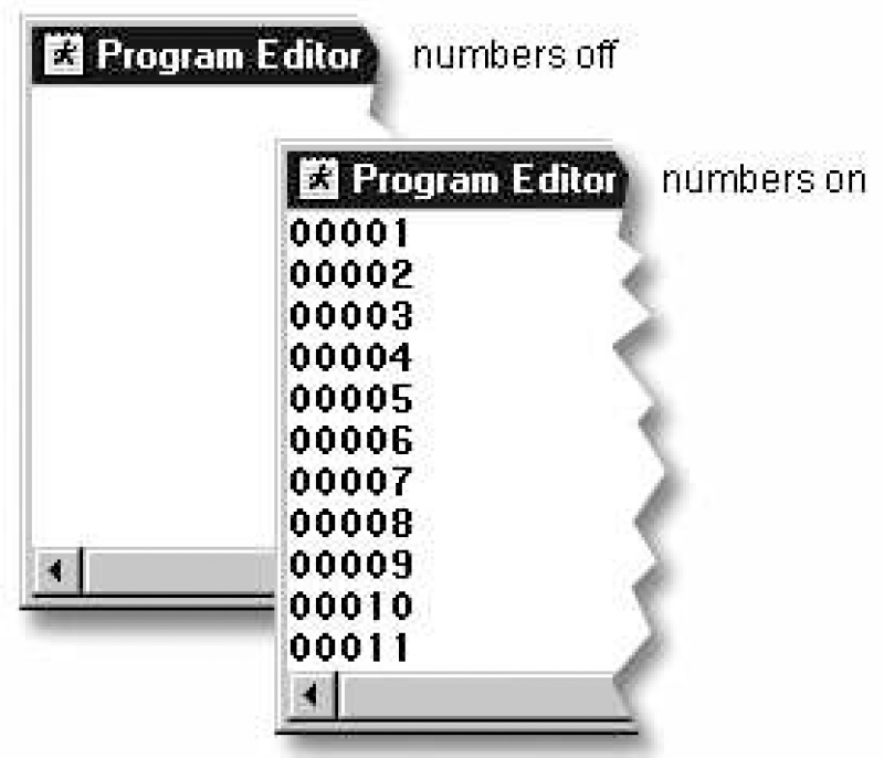

Figure 3.9: Line Numbers

To turn on line numbers, use the NUMS command. Type nums on the command line or in the command box and press Enter. To turn line numbers off, enter the NUMS command again. Line numbers activated using the NUMS command will remain in effect for the duration of your SAS session.

Additional Note To permanently display line numbers, select **Tools** ⇨ **Options** ⇨ **Program Editor**. Select the **Editing** tab. Select the **Display line numbers** option. Click **OK**.

Text Editor Line Commands

Text editor line commands enable you to delete, insert, move, copy, and repeat text. You enter these commands in the line number area in the Program Editor window.

The table below summarizes the basic text editing line commands.

Table 3.2: Text Editor Line Commands

Command	Action
C*n*	copies *n* lines (where *n* = a number up to 9999)
D*n*	deletes *n* lines
I*n*	inserts *n* blank lines
M*n*	moves *n* lines
R*n*	repeats current line *n* times
A	after (used with C, I, and M)

B	before (used with C, I, and M)

You can use text editor line commands to perform actions like these:

Table 3.3: Results from Using Text Editor Line Commands

Command	Action
00001 i3002 00003	inserts 3 lines after line 00002
0ib01 00002 00003	inserts 1 line before line 00001
0ib41 00002 00003	inserts 4 lines before line 00001
000c2 00002 0a003	copies 2 lines (00001 and 00002) after line 00003
00001 0d302 00003	deletes 3 lines (00002, 00003, and 00004)
00b01 00002 00m03	moves 1 line (00003) before line 00001

Example 1

In the example below, a PROC PRINT statement and a RUN statement need to be inserted after line 00004.

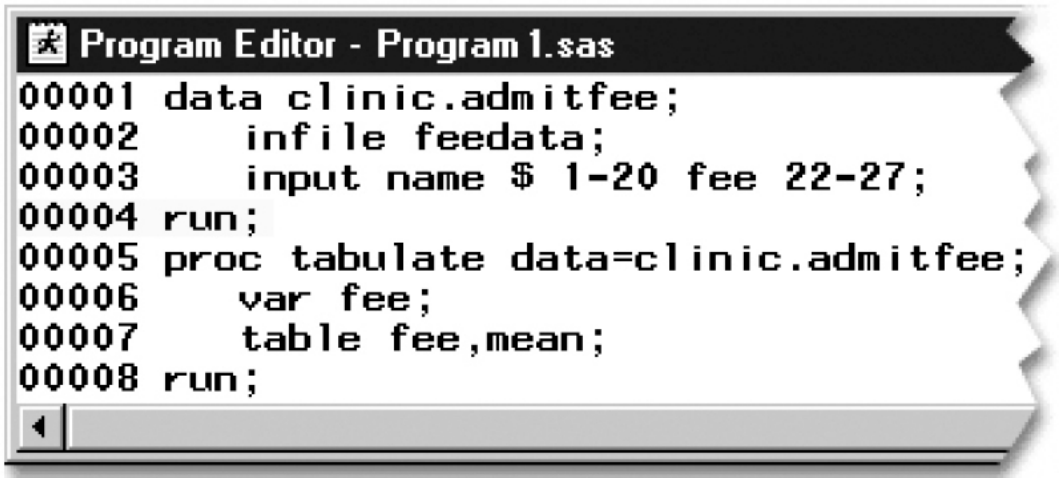

Figure 3.10: Incomplete Program

To insert the PROC PRINT and RUN statements in the program,

1. Type **i2** anywhere in the line number area for line 00004 in the Program Editor window. Two blank lines are inserted, and the cursor is positioned on the first new line.
2. Type the PROC PRINT statement on line 00005.
3. Type a RUN statement on line 00006.

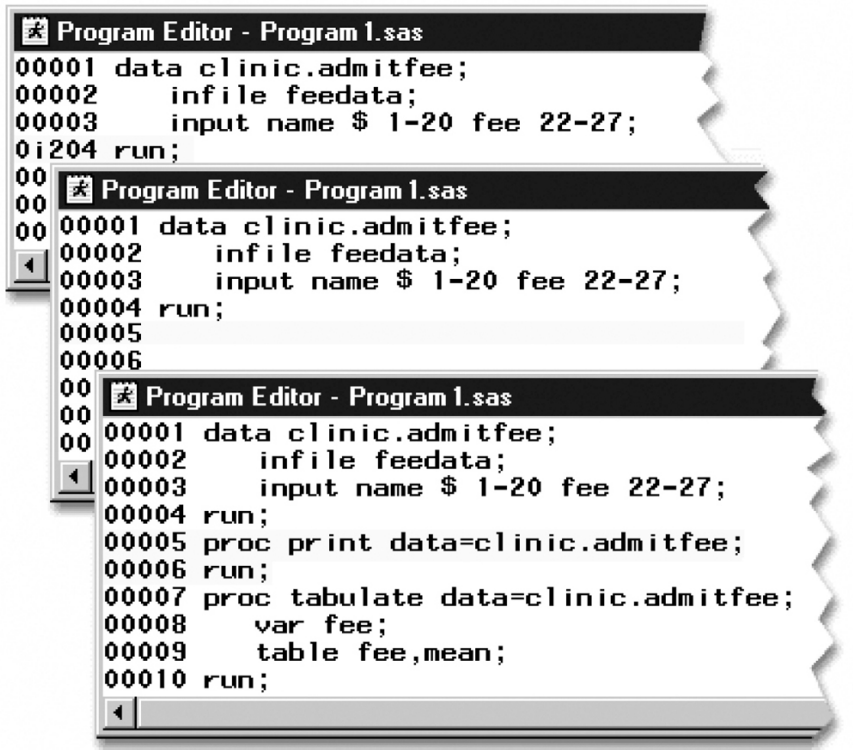

Figure 3.11: Complete Program

Block Text Editor Line Commands

Block text editor line commands enable you to delete, repeat, copy, and move multiple lines in the Program Editor window.

The block text editor line commands include the following:

Table 3.4: Block Text Editor Line Commands

Command	Action
DD	deletes a block of lines
CC	copies a block of lines
MM	moves a block of lines
RR	repeats multiple lines
A	after (used with CC and MM commands)
B	before (used with CC and MM commands)

To use a block command, specify the command on the first line affected and on the final line affected, and then press Enter.

Example 1

In the program below, the PROC TABULATE step needs to be deleted.

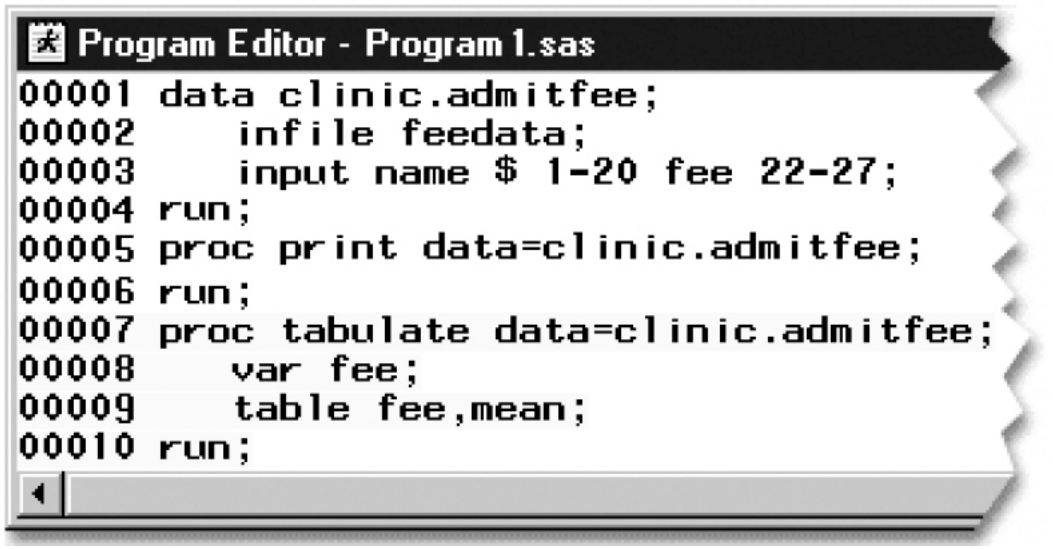

Figure 3.12: Original Program

You can use the DD block command to remove the step. To use the DD command:

1. Type **DD** anywhere in the line number area for line 00007 to mark the first line affected.
2. Type **DD** anywhere in the line number area for line 00010 to mark the final line affected.
3. Press Enter to delete lines 00007-00010.

Figure 3.13: Edited Program

Example 2

In the example below, the PROC TABULATE step needs to be moved to the line after the PROC PRINT step.

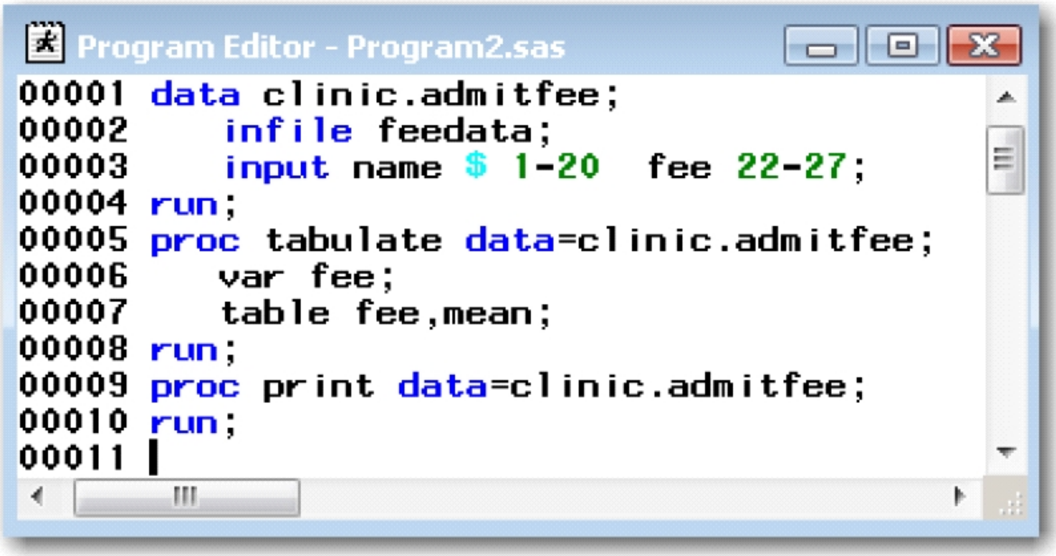

Figure 3.14: Original Program

You can use the MM block command with the A or B command to move a block of code to a specific location. The MM is used to mark the block to be moved and the A or B is used to mark where you want the block to go. You use the A if you want to move the block after the line, and you use b if you want to move the block before the line.

To use the MM command:

1. Type **MM** anywhere in the line number area for line 00005 to mark the first line affected.
2. Type **MM** anywhere in the line number area for line 00008 to mark the last line affected.
3. Type **A** anywhere in the line number area for line 00010 to move the block after line 00010.
4. Press Enter to move lines 00005-00008 to line 00010.

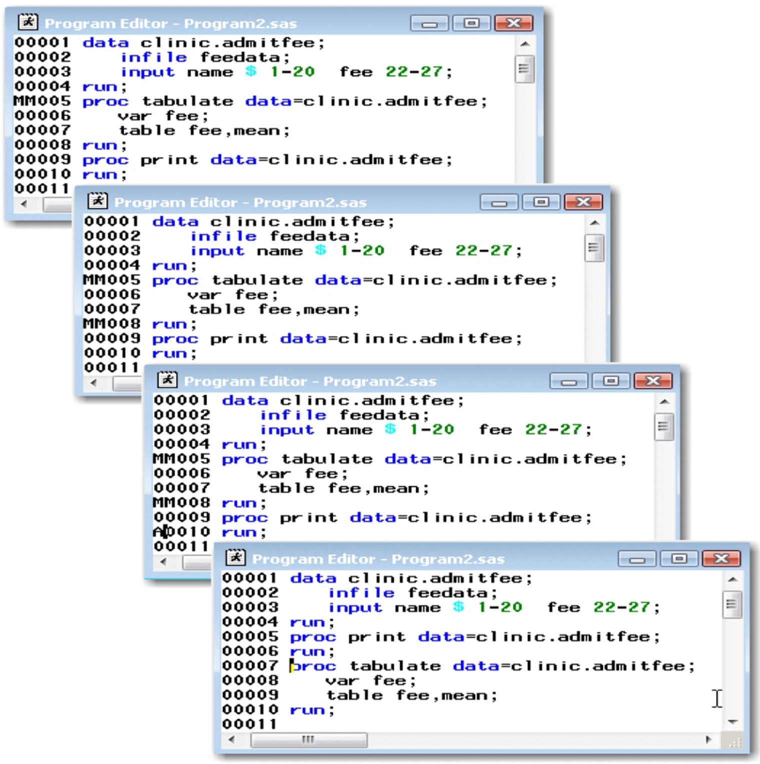

Figure 3.15: Edited Program

Recalling SAS Programs

SAS statements disappear from the Program Editor window when you submit them. However, you can recall a program to the Program Editor by selecting **Run** ⇨ **Recall Last Submit**.

The area in which submitted SAS code is stored is known as the recall buffer. Program statements accumulate in the recall buffer each time you submit a program. The recall buffer is last-in/first-out (the most recently submitted statements are recalled first).

For example, if you submit two programs, you will need to select **Run** ⇨ **Recall Last Submit** two times to recall the first program to the Program Editor window. You can recall statements any time during your SAS session.

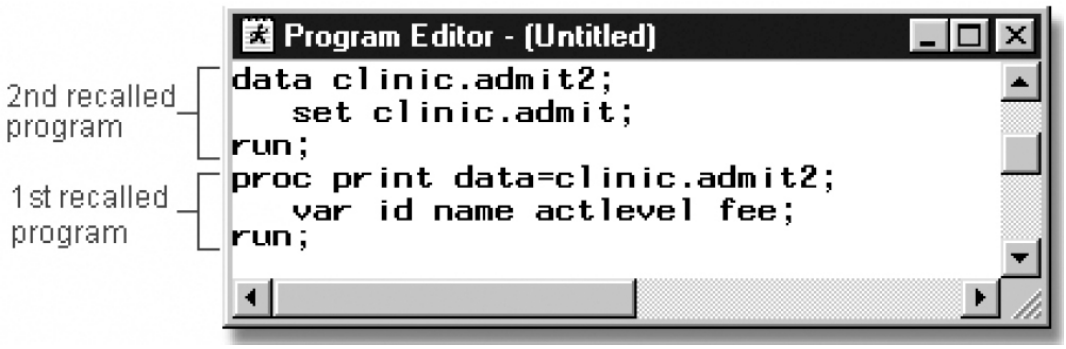

Figure 3.16: Recalled Programs

Saving SAS Programs

To save your SAS program to an external file, first activate the code editing window and select **File** ⇨ **Save As**. Then specify the name of your program in the Save As window. It's good practice to assign the extension .SAS to your SAS programs to distinguish them from other types of SAS files (such as .LOG for log files and .DAT for raw data files).

Additional Note To save a SAS program as an external file in the z/OS operating environment, select **File** ⇨ **Save As** ⇨ **Write to File**, and then specify the name of your program in the Save As dialog box. You can specify SAS as the last level of the filename.

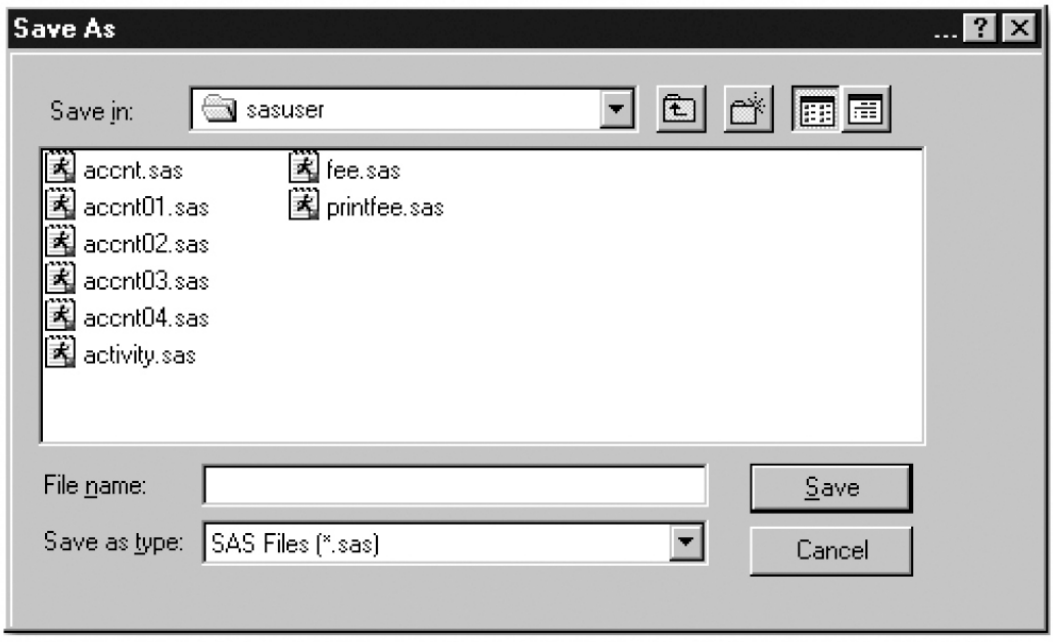

Figure 3.17: Save as WIndow

You can also save a SAS program by issuing a FILE command.

General form, basic FILE command:

FILE *'file-specification'*

where *file-specification* is the name of the file to be saved.

Example

Suppose you want to save a program as **`D:\Programs\Sas\Newprog.Sas`** in the Windows operating environment. To do so, you can issue the following FILE command:

```
file 'd:\programs\sas\newprog.sas'
```

Clearing SAS Programming Windows

When you run SAS programs, text accumulates in the Output window and in the Log window.

You might find it helpful to clear the contents of your SAS programming windows. To clear the Output window, Editor window, Program Editor window, or Log window, activate each window individually and select **Edit** ⇨ **Clear All**.

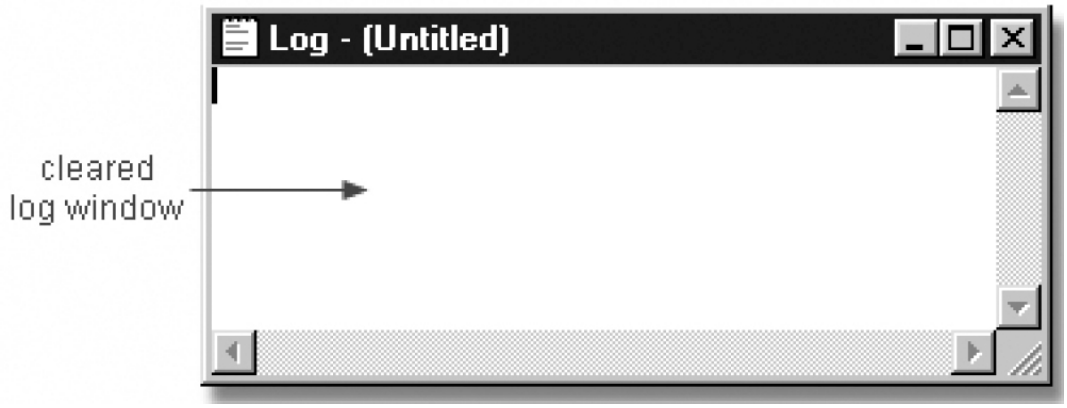

Figure 3.18: Cleared Log Window

Additional Note You can also clear the contents of a window by activating the window and then issuing the CLEAR command.

Caution Text that has been cleared from windows cannot be recalled. However, in the Editor and Program Editor windows, you can select **Edit** ⇨ **Undo** to redisplay the text.

Interpreting Error Messages

Error Types

So far, the programs that are shown in this chapter have been error-free, but programming errors do occur. SAS can detect several types of errors. The most common are

- syntax errors that occur when program statements do not conform to the rules of the SAS language
- data errors that occur when some data values are not appropriate for the SAS statements that are specified in a program.

This chapter focuses on identifying and correcting syntax errors.

Syntax Errors

When you submit a program, SAS scans each statement for syntax errors, and then executes the step (if no syntax errors are found). SAS then goes to the next step and repeats the process. Syntax errors, such as misspelled keywords,

generally prevent SAS from executing the step in which the error occurred.

You already know that notes are written to the Log window at the conclusion of execution of the program. When a program that contains an error is submitted, messages regarding the problem also appear in the Log window. When a syntax error is detected, the Log window

- displays the word ERROR
- identifies the possible location of the error
- gives an explanation of the error.

Example

The program below contains a syntax error. The DATA step copies the SAS data set Clinic.Admit into a new data set named Clinic.Admitfee. The PROC step should print the values for the variables ID, Name, Actlevel, and Fee in the new data set. However, print is misspelled in the PROC PRINT statement.

```
data clinic.admitfee;
   set clinic.admit;
run;
proc prin data=clinic.admitfee;
   var id name actlevel fee;
run;
```

When the program is submitted, messages in the Log window indicate that the procedure PRIN was not found and that SAS stopped processing the PRINT step due to errors. No output is produced by the PRINT procedure, because the second step fails to execute.

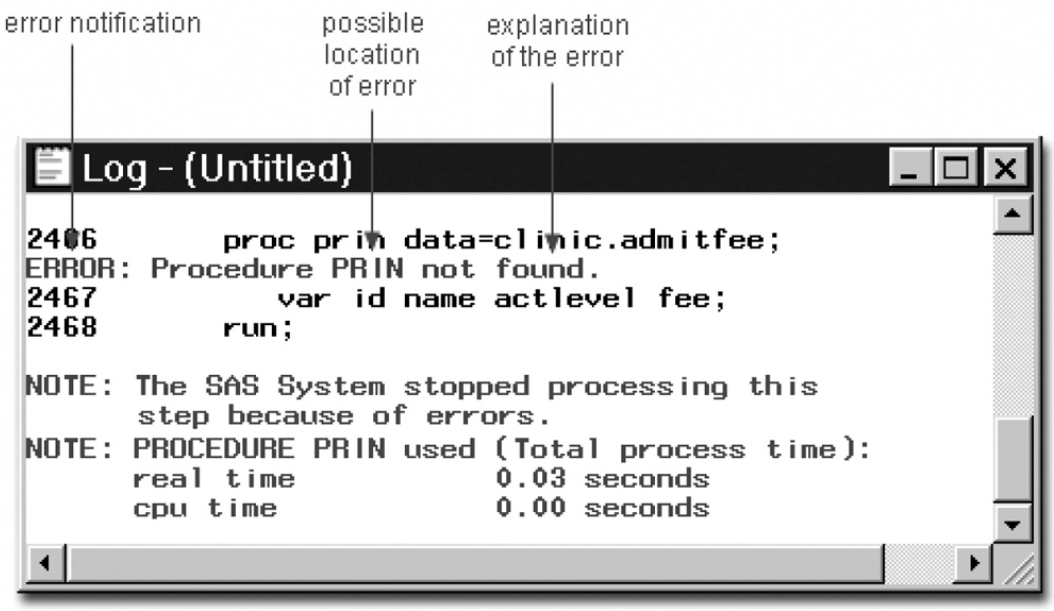

Figure 3.19: Log Window Displaying Error Message

Caution Problems with your statements or data might not be evident when you look at results in the Output window. Therefore, it is important to review the messages in the Log window each time you submit a SAS program.

Correcting Errors

Overview

To modify programs that contain errors, you can edit them in the code editing window. You can correct simple errors, such as the spelling error in the following program, by typing over the incorrect text, deleting text, or inserting text.

```
data clinic.admitfee;
   set clinic.admit;
```

```
run;
proc prin data=clinic.admitfee;
   var id name actlevel fee;
run;
```

Additional Note If you use the Program Editor window, you usually need to recall the submitted statements from the recall buffer to the Program Editor window, where you can correct the problems. Remember that you can recall submitted statements by issuing the RECALL command or by selecting **Run** ⇨ **Recall Last Submit**.

In the program below, the missing t has been inserted into the PRINT keyword that is specified in the PROC PRINT statement.

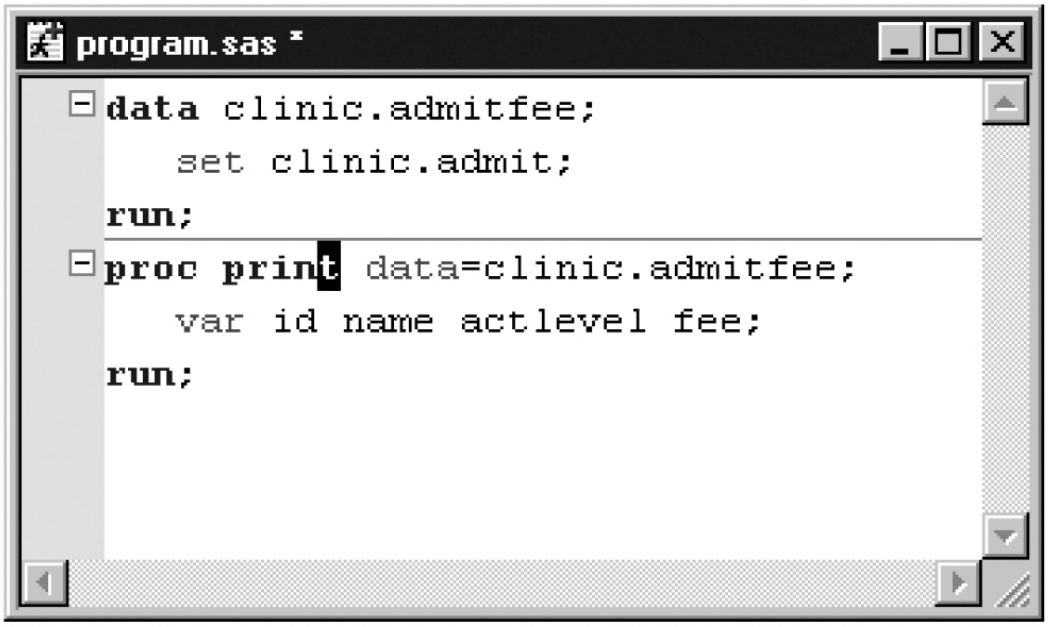

Figure 3.20: Corrected Program

Additional Note Some problems are relatively easy to diagnose and correct. But sometimes you might not know right away how to correct errors. The online Help provides information about individual procedures as well as help that is specific to your operating environment. From the **Help** menu, you can also select **SAS on the Web** for links to Technical Support and Frequently Asked Questions, if you have Internet access.

Resubmitting a Revised Program

After correcting your program, you can submit it again. However, before doing so, it is a good idea to clear the messages from the Log window so that you don't confuse the old error messages with the new messages. Then you can resubmit the program and view any resulting output.

The SAS System				
Obs	ID	Name	ActLeuel	Fee
1	2458	Murray, W	HIGH	85.20
2	2462	Almers, C	HIGH	124.80
3	2501	Bonaventure, T	LOW	149.75
4	2523	Johnson, R	MOD	149.75
5	2539	LaMance, K	LOW	124.80
6	2544	Jones, M	HIGH	124.80

7	2552	Reberson, P	MOD	149.75
8	2555	King, E	MOD	149.75
9	2563	Pitts, D	LOW	124.80
10	2568	Eberhardt, S	LOW	124.80
11	2571	Nunnelly, A	HIGH	149.75
12	2572	Oberon, M	LOW	85.20
13	2574	Peterson, V	MOD	149.75
14	2575	Quigley, M	HIGH	124.80
15	2578	Cameron, L	MOD	124.80
16	2579	Underwood, K	LOW	149.75
17	2584	Takahashi, Y	MOD	124.80
18	2586	Derber, B	HIGH	85.20
19	2588	Ivan, H	LOW	85.20
20	2589	Wilcox, E	HIGH	149.75
21	2595	Warren, C	MOD	149.75

Figure 3.21: Correct PROC PRINT Output

Remember to check the Log window again to verify that your program ran correctly.

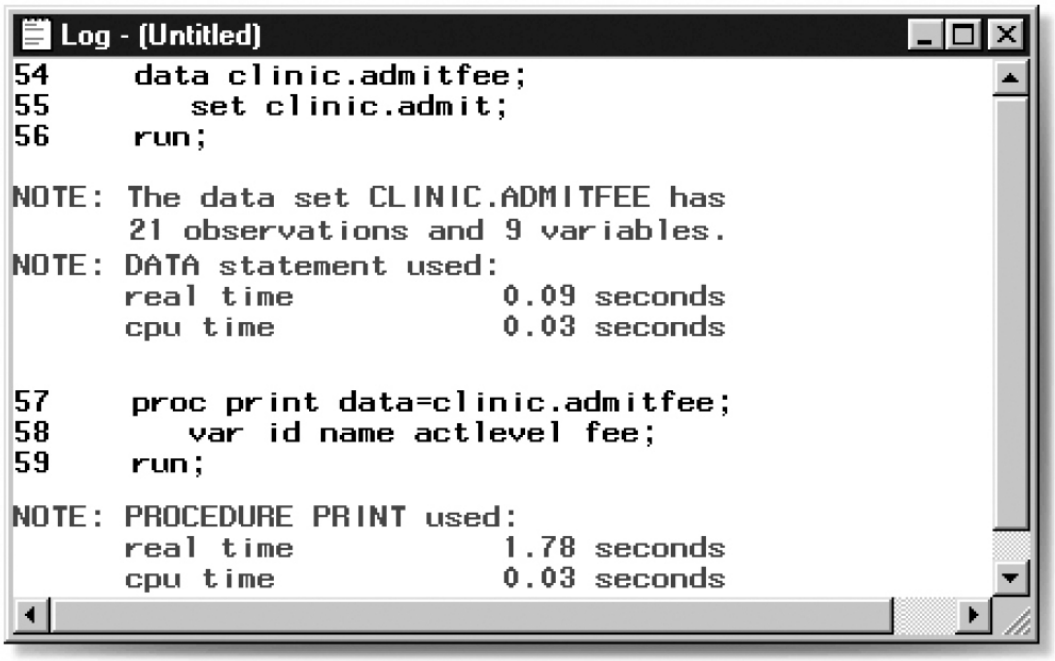

Figure 3.22: Log Message with No Errors

Additional Note Because submitted steps remain in the recall buffer, resubmitting error-free steps wastes system resources. You can place SAS comment symbols before and after your error-free code until you debug the rest of your program. SAS ignores text between comment symbols during processing. When your entire program is error-free, remove the comment symbols and your entire SAS program is intact. See "Additional Features" on page 102 for instructions about creating a SAS comment statement.

Additional Note To resubmit a section of code in the Windows operating environment, highlight the selected code in the code editing window. Then select **Run** ⇨ **Submit**.

Caution This and other chapters show you the Enhanced Editor window only. If you are not using the Enhanced Editor as a code editing window, be sure to adapt the directions for the Program Editor window. For example, you might need to recall programs.

Resolving Common Problems

Overview

In addition to correcting spelling mistakes, you might need to resolve several other types of common syntax errors. These errors include

- omitting semicolons
- leaving quotation marks unbalanced
- specifying invalid options.

Another common problem is omitting a RUN statement at the end of a program. Although this is not technically an error, it can produce unexpected results. For the sake of convenience, we'll consider it together with syntax errors.

The table below lists these problems and their symptoms.

Table 3.5: Identifying Problems in a SAS Program

Problem	Symptom
missing RUN statement	"PROC (or DATA) step running" at top of active window
missing semicolon	log message indicating an error in a statement that seems to be valid
unbalanced quotation marks	log message indicating that a quoted string has become too long or that a statement is ambiguous
invalid option	log message indicating that an option is invalid or not recognized

Missing RUN Statement

Each step in a SAS program is compiled and executed independently from every other step. As a step is compiled, SAS recognizes the end of the current step when it encounters

- a DATA or PROC statement, which indicates the beginning of a new step
- a RUN or QUIT statement, which indicates the end of the current step.

When the program below is submitted, the DATA step executes, but the PROC step does not. The PROC step does not execute because there is no following DATA or PROC step to indicate the beginning of a new step, nor is there a following RUN statement to indicate the end of the step.

```
data clinic.admitfee;
   set clinic.admit;
run;
proc print data=clinic.admitfee;
   var id name actlevel fee;
```

Because there is nothing to indicate the end of the PROC step, the PRINT procedure waits before executing, and a "PROC PRINT running" message appears at the top of the active window.

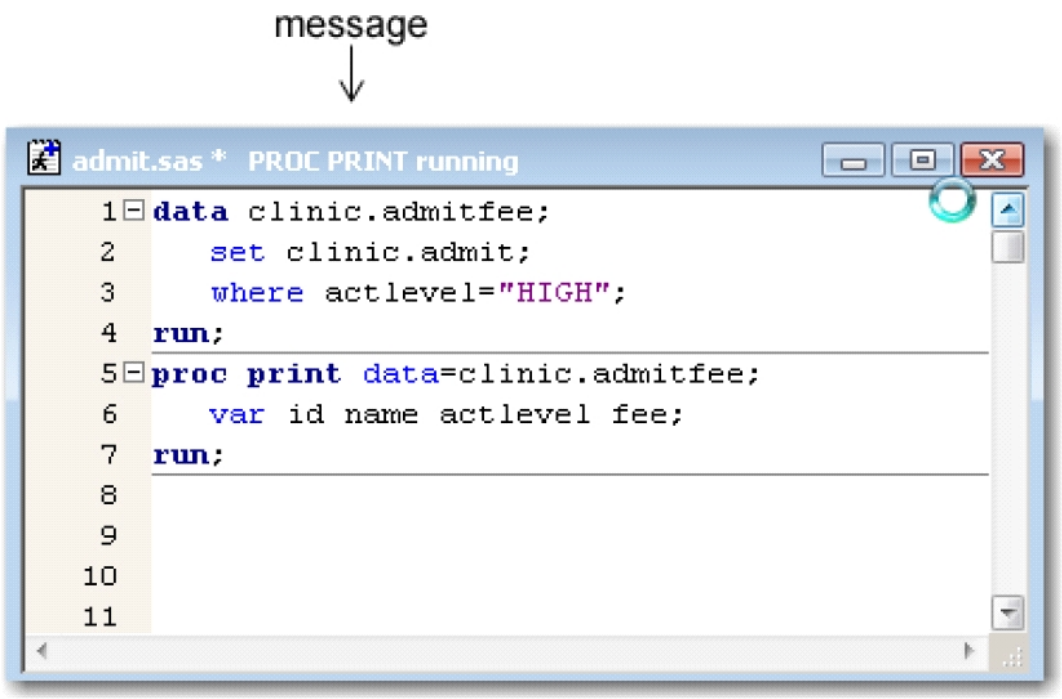

Figure 3.23: Program Window with Message

Resolving the Problem

To correct the error, submit a RUN statement to complete the PROC step.

```
run;
```

If you are using the Program Editor window, you do *not* need to recall the program to the Program Editor window.

Missing Semicolon

One of the most common errors is the omission of a semicolon at the end of a statement. The program below is missing a semicolon at the end of the PROC PRINT statement.

```
data clinic.admitfee;
  set clinic.admit;
run;
proc print data=clinic.admitfee
   var id name actlevel fee;
run;
```

When you omit a semicolon, SAS reads the statement that lacks the semicolon, plus the following statement, as one long statement. The SAS log then lists errors that relate to the combined statement, not the actual mistake (the missing semicolon).

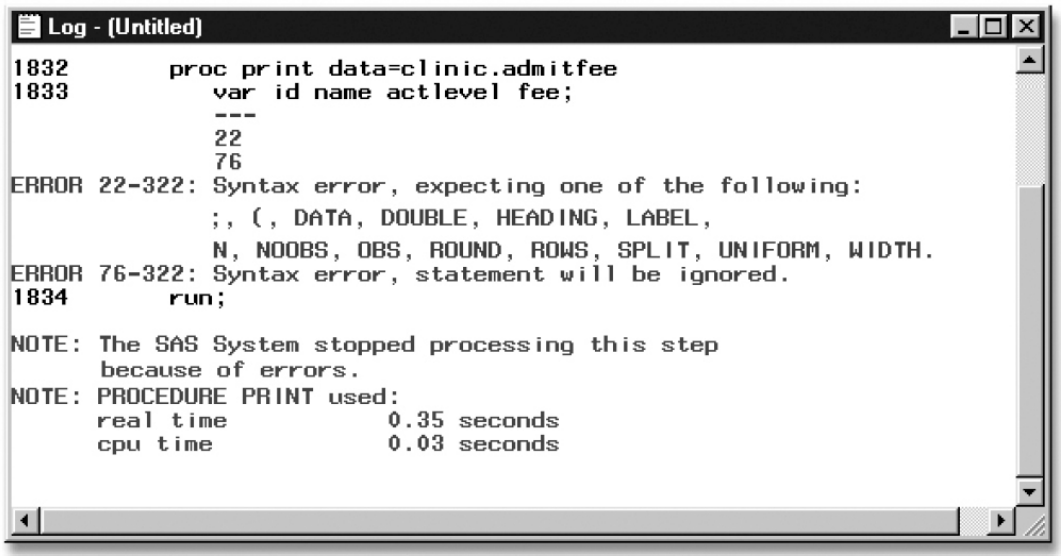

Figure 3.24: Log Window with Error Message

Resolving the Problem

To correct the error, do the following:

1. Find the statement that lacks a semicolon. You can usually locate the statement that lacks the semicolon by looking at the underscored keywords in the error message and working backwards.
2. Add a semicolon in the appropriate location.
3. Resubmit the corrected program.
4. Check the Log window again to make sure there are no other errors.

Unbalanced Quotation Marks

Some syntax errors, such as the missing quotation mark after HIGH in the program below, cause SAS to misinterpret the statements in your program.

```
data clinic.admitfee;
   set clinic.admit;
   where actlevel='HIGH;
run;
proc print data=clinic.admitfee;
   var id name actlevel fee;
run;
```

When the program is submitted, SAS is unable to resolve the DATA step, and a "DATA STEP running" message appears at the top of the active window.

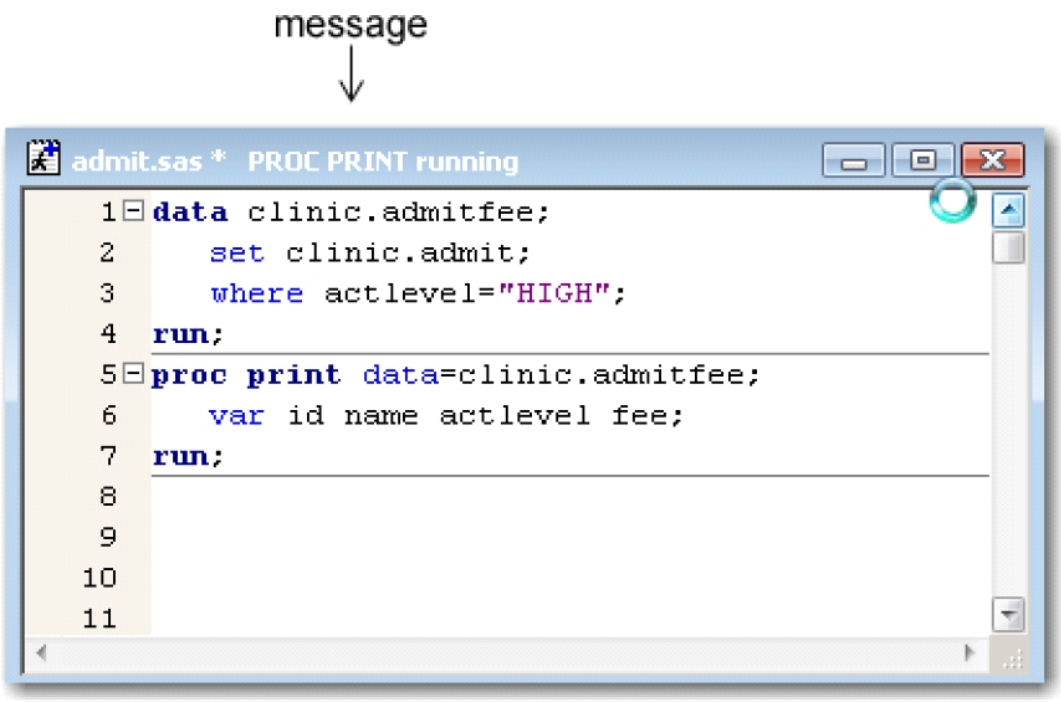

Figure 3.25: Program Editor with Message

Sometimes a warning appears in the SAS log which indicates that

- a quoted string has become too long
- a statement that contains quotation marks (such as a TITLE or FOOTNOTE statement) is ambiguous due to invalid options or unquoted text.

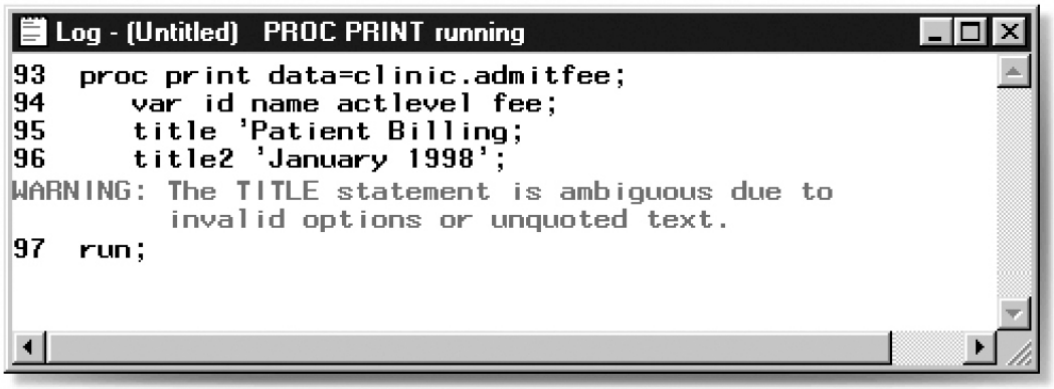

Figure 3.26: Log with Warning Message

When you have unbalanced quotation marks, SAS is often unable to detect the end of the statement in which it occurs. Simply adding a quotation mark and resubmitting your program usually does *not* solve the problem. SAS still considers the quotation marks to be unbalanced.

Therefore, you need to resolve the unbalanced quotation mark by canceling the submitted statements (in the Windows and UNIX operating environments) or by submitting a line of SAS code (in the z/OS operating environment) before you recall, correct, and resubmit the program.

Additional Note If you do not resolve this problem when it occurs, it is likely that any subsequent programs that you submit in the current SAS session will generate errors.

Resolving the Error in the Windows Operating Environment

To resolve the error in the Windows operating environment:

1. Press the Ctrl and Break keys or click the Break Icon on  the toolbar.
2. Select **1. Cancel Submitted Statements**, and then click **OK**.

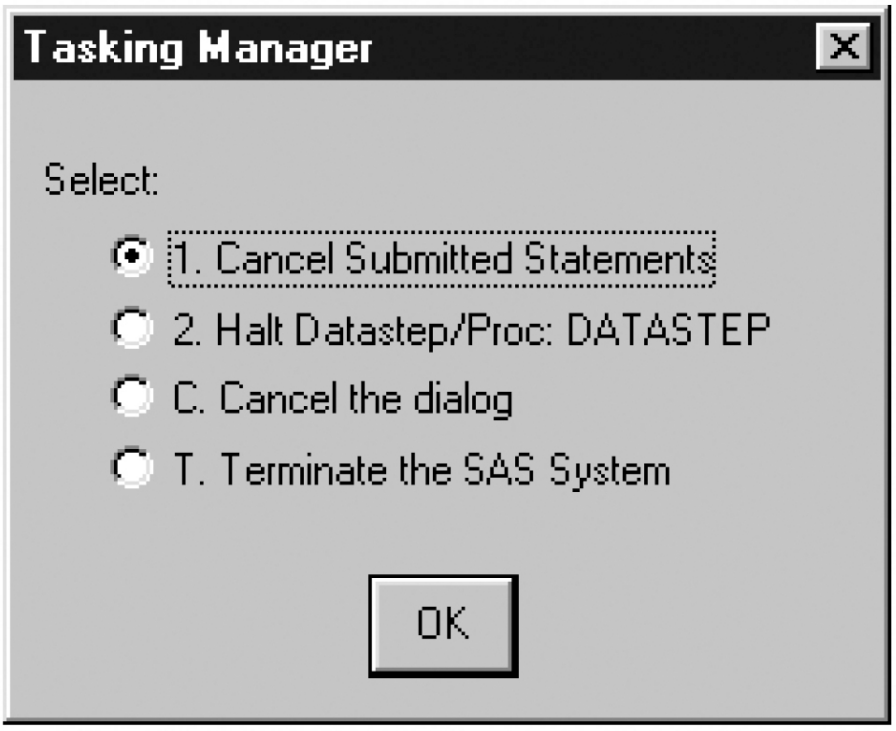

Figure 3.27: Tasking Manager Window

3. Select **Y to cancel submitted statements**, and then click **OK**.

Figure 3.28: Break Window

4. Correct the error and resubmit the program.

Resolving the Error in the UNIX Operating Environment

To resolve the error in the UNIX operating environment:

1. Open the Session Management window and click **Interrupt**.

Figure 3.29: Session Management Window

2. Select **1. Cancel Submitted Statements**, and then click **Y**.

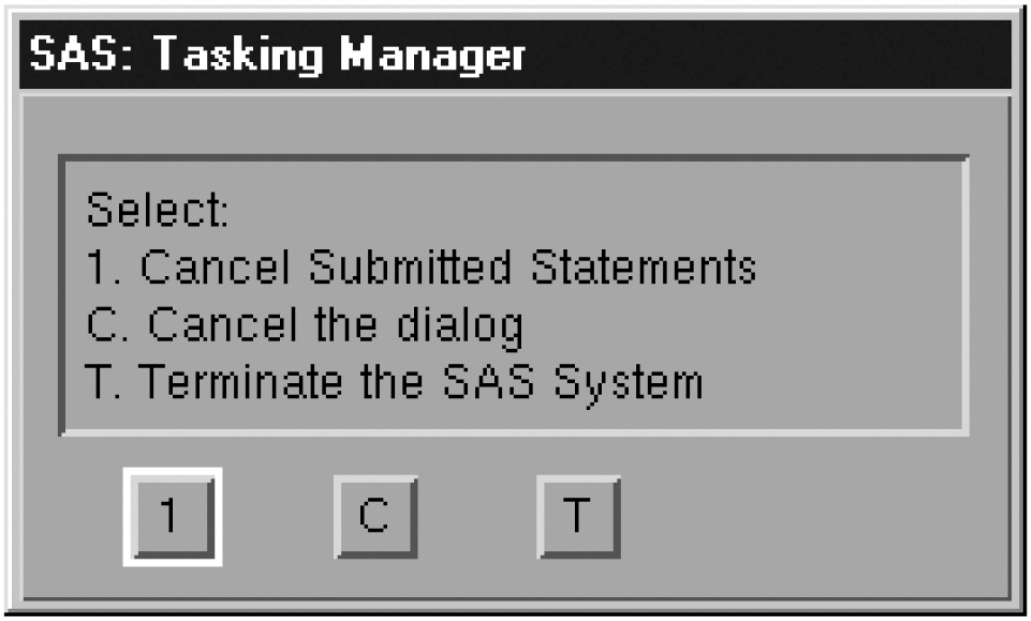

Figure 3.30: Tasking Management Window

3. Correct the error and resubmit the program.

Resolving the Error in the z/OS Operating Environment

To resolve the error in the z/OS operating environment:

1. Submit an asterisk followed by a quotation mark, a semicolon, and a RUN statement.

```
*'; run;
```

2. Delete the line that contains the asterisk followed by the quotation mark, the semicolon, and the RUN statement.
3. Insert the missing quotation mark in the appropriate place.

4. Submit the corrected program.

Additional Note You can also use the above method in the Windows and UNIX operating environments.

Invalid Option

An invalid option error occurs when you specify an option that is not valid in a particular statement. In the program below, the KEYLABEL option is not valid when used with the PROC PRINT statement.

```
data sasuser.admitfee;
   set clinic.admit;
   where weight>180 and (actlevel="MOD" or actlevel="LOW");
run;
proc print data=clinic.admitfee keylabel;
   label actlevel='Activity Level';
run;
```

When a SAS statement that contains an invalid option is submitted, a message appears in the Log window indicating that the option is not valid or not recognized.

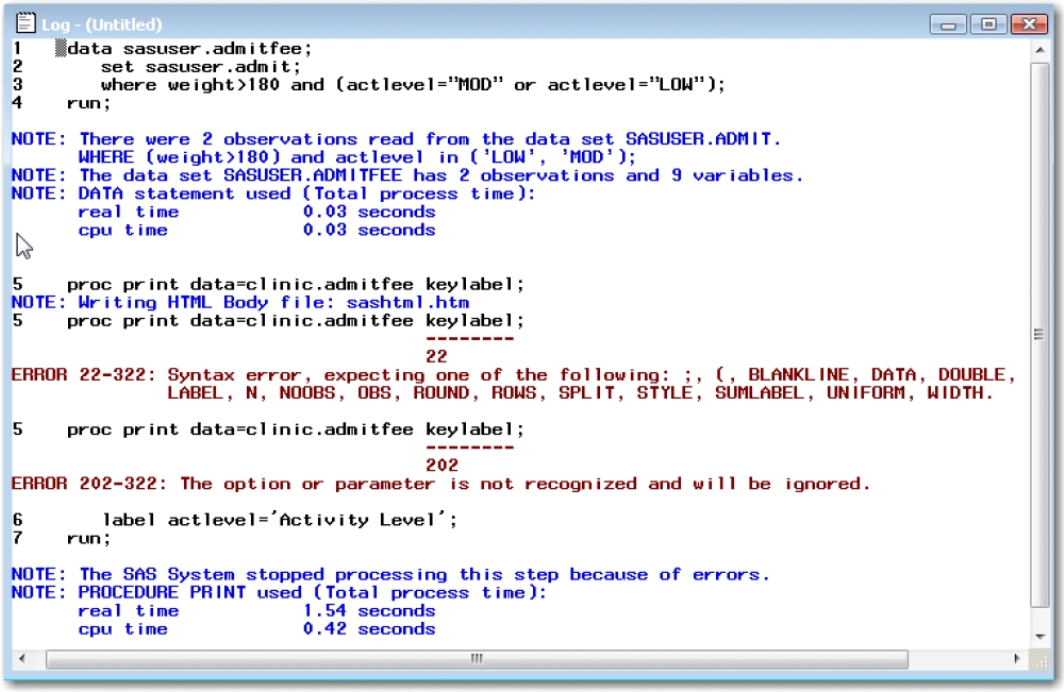

Figure 3.31: Log Window with Syntax Error Message

Resolving the Problem

To correct the error:

1. Remove or replace the invalid option, and check your statement syntax as needed.
2. Resubmit the corrected program.
3. Check the Log window again to make sure there are no other errors.

Additional Features

Comments in SAS Programs

You can insert comments into a SAS program to document the purpose of the program, to explain segments of the program, or to describe the steps in a complex program or calculation. A comment statement begins and ends with a comment symbol. There are two forms of comment statements:

```
*text;
```

or

```
 /*text*/
```

SAS ignores text between comment symbols during processing.

The following program shows some of the ways comments can be used to describe a SAS program.

```
   /* Read national sales data for vans */
   /* from an external raw data file */
data perm.vansales;
   infile vandata;
   input @1 Region $9.
         @13 Quarter 1. /* Values are 1, 2, 3, or 4 */
         @16 TotalSales comma11.;
   /* Print the entire data set */
proc print data=perm.vansales;
run;
```

Caution Avoid placing the /* comment symbols in columns 1 and 2. On some host operating systems, SAS might interpret a /* in columns 1 and 2 as a request to end the SAS job or session. For more information, see the SAS documentation for your operating environment.

System Options

SAS includes several system options that enable you to control error handling and Log window messages. The table shown below contains brief descriptions of some of these options. You can use either the OPTIONS statement or the SAS System Options window to specify these options.

Table 3.6: Options for Controlling Error Handling and Log Window Messages

Option	Descriptions
ERRORS=*n*	Specifies the maximum number of observations for which complete data error messages are printed.
FMTERR | NOFMTERR	Controls whether SAS generates an error message when a format of a variable cannot be found. NOFMTERR results in a warning instead of an error. FMTERR is the default.
SOURCE | NOSOURCE	Controls whether SAS writes source statements to the SAS log. SOURCE is the default.

Error Checking In the Enhanced Editor

If you are using the Enhanced Editor, you can use its color-coding for program elements, quoted strings, and comments to help you find coding errors.

You can also search for

- ending brackets or parentheses by pressing Ctrl+].
- matching DO-END pairs by pressing Alt+[. (You can learn about DO-END pairs in .)

See the following table for suggestions about finding syntax errors using the Enhanced Editor.

Table 3.7: Finding Syntax Errors Using the Enhanced Editor

To find…	Do this…
Undefined or misspelled keywords	In the Appearance tab of the Enhanced Editor Options dialog box, set the file elements **Defined keyword, User defined keyword**, and the **Undefined keyword** to unique color combinations. When SAS recognizes a keyword, the keyword changes to the defined colors. You'll be able to easily spot undefined keywords by looking for the color that you selected for undefined keywords.
Unmatched quoted strings	Look for one or more lines of the program that are the same color. Text following a quotation mark remains the same color until the string is closed with a matching quotation mark.
Unmatched comments	Look for one or more lines of the program that are the same color. Text that follows an open comment symbol (/*) remains the same color until the comment is closed with a

	closing comment symbol (*/).
Matching DO-END pairs	Place the cursor within a DO-END block and press Alt+ [. The cursor moves first to the DO keyword. If one of the keywords is not found, the cursor remains as positioned. When both of the keywords exist, pressing Alt+[moves the cursor between the DO-END keywords.
Matching parentheses or brackets	Place the cursor on either side of the parenthesis or bracket. Press Ctrl+]. The cursor moves to the matching parentheses or bracket. If one is not found, the cursor remains as positioned.
Missing semi-colons (;)	Look for keywords that appear in normal text.

Debugging with the DATA Step Debugger

Unlike most syntax errors, logic errors do not stop a program from running. Instead, they cause the program to produce unexpected results.

You can debug logic errors in DATA steps by using the DATA step debugger. This tool enables you to issue commands to execute DATA step statements one by one, and then to pause to display the resulting variables' values in a window. By observing the results that are displayed, you can determine where the logic error lies.

The debugger can be used only in interactive mode. Because the debugger is interactive, you can repeat the process of issuing commands and observing results as many times as needed in a single debugging session. To invoke the debugger, add the DEBUG option to the DATA statement, and submit the program.

```
data perm.publish / debug;
   infile pubdata;
   input BookID $ Publisher & $22. Year;
run;
proc print data=perm.publish;
run;
```

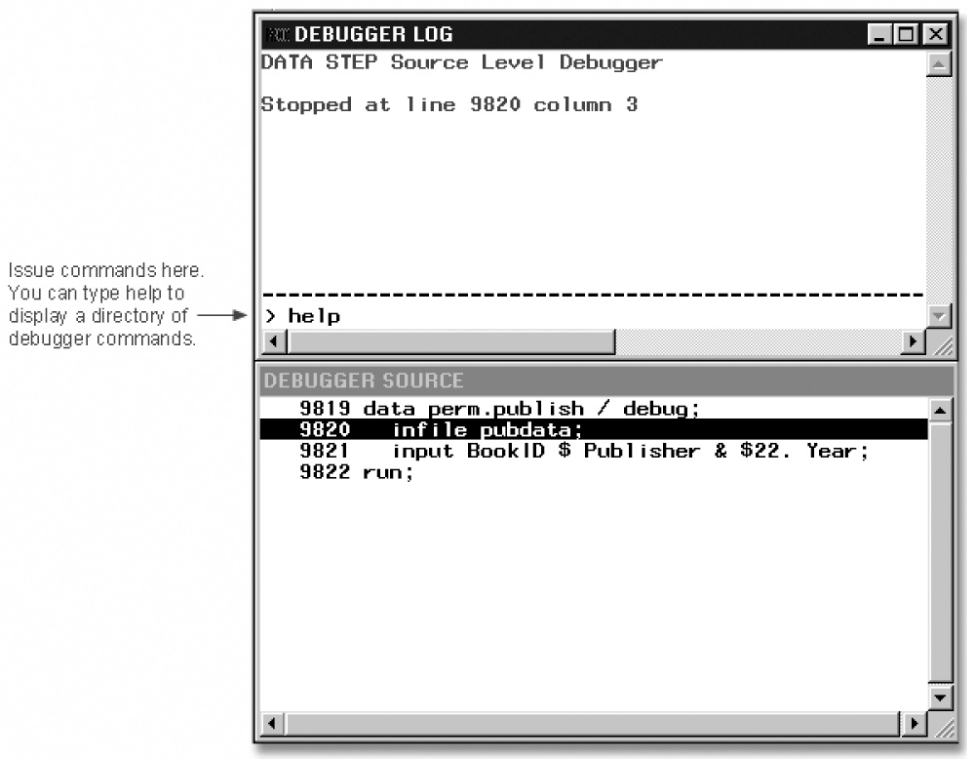

Figure 3.32: Debugger Log Window

Additional Note For detailed information about how to use the debugger, see the SAS documentation.

Chapter Summary

Text Summary

Opening a Stored SAS Program

A SAS program that is stored in an external file can be opened in the code editing window using

- file shortcuts
- My Favorite Folders
- the Open window
- the INCLUDE command.

Editing SAS Programs

SAS programs consist of SAS statements. Although you can write SAS statements in almost any format, a consistent layout enhances readability and enables you to understand the program's purpose.

In the Windows operating environment, the Enhanced Editor enables you to enter and view text and

- select one or more lines of text
- use color-coding to identify code elements
- automatically indent the next line when you press the Enter key
- collapse and expand sections of SAS procedures, DATA steps, and macros
- bookmark lines of code for easy access to different sections of your program.

Using the Editor window, you can also find and replace text, use abbreviations, open multiple views of a file, and set Enhanced Editor options.

The Program Editor window enables you to edit your programs just as you would with a word processing program. You can also use text editor commands and block text editor commands to edit SAS programs. Activating line numbers can make it easier for you to edit your program regardless of your operating environment.

Remember that SAS statements disappear from the Program Editor window when they are submitted. However, you can recall a program to the Program Editor window.

To save your SAS program to an external file, first activate the Program Editor window and select **File** ⇨ **Save As**. Then specify the name of your program in the Save As window.

Clearing SAS Programming Windows

Text and output accumulate in the Editor, Program Editor, Log, and Output windows throughout your SAS session. You can clear a window by selecting **Edit** ⇨ **Clear All**.

Interpreting Error Messages

When a SAS program that contains errors is submitted, error messages appear in the Log window. SAS can detect three types of errors: syntax, execution-time, and data. This chapter focuses on identifying and resolving common syntax errors.

Correcting Errors

To modify programs that contain errors, you can correct the errors in the code editing window. In the Program Editor window, you need to recall the submitted statements before editing them.

Before resubmitting a revised program, it is a good idea to clear the messages from the Log window so that you don't confuse old messages with the new. You can delete any error-free steps from a revised program before resubmitting it.

Resolving Common Problems

You might need to resolve several types of common problems: missing RUN statements, missing semicolons, unbalanced quotation marks, and invalid options.

Points to Remember

- It's a good idea to begin DATA steps, PROC steps, and RUN statements on the left and to indent statements within a step.
- End each step with a RUN statement.
- Review the messages in the Log window each time you submit a SAS program.
- You can delete any error-free steps from a revised program before resubmitting it, or you can submit only the revised steps in a program.

Chapter Quiz

Select the best answer for each question. After completing the quiz, check your answers using the answer key in the appendix.

1. As you write and edit SAS programs, it's a good idea to ?

a. begin DATA and PROC steps in column one.

b. indent statements within a step.

c. begin RUN statements in column one.

d. do all of the above.

2. Suppose you have submitted a SAS program that contains spelling errors. Which set of steps should you perform, in the order shown, to revise and resubmit the program? ?

a.
- Correct the errors.
- Clear the Log window.
- Resubmit the program.
- Check the Log window.

b.
- Correct the errors.
- Resubmit the program.
- Check the Output window.
- Check the Log window.

c.
- Correct the errors.
- Clear the Log window.
- Resubmit the program.
- Check the Output window.

d.
- Correct the errors.
- Clear the Output window.
- Resubmit the program.
- Check the Output window.

3. What happens if you submit the following program? ?

```
proc sort data=clinic.stress out=maxrates;
    by maxhr;
run;
proc print data=maxrates label double noobs;
   label rechr='Recovery Heart Rate;
   var resthr maxhr rechr date;
   where toler='I' and resthr>90;
   sum fee;
run;
```

a. Log messages indicate that the program ran successfully.

b. A "PROC SORT running" message appears at the top of the active window, and a log message might indicate an error in a statement that seems to be valid.

c. A log message indicates that an option is not valid or not recognized.

d. A "PROC PRINT running" message appears at the top of the active window, and a log message might indicate that a quoted string has become too long or that the statement is ambiguous.

4. What generally happens when a syntax error is detected? ?

a. SAS continues processing the step.

b. SAS continues to process the step, and the Log window displays messages about the error.

c. SAS stops processing the step in which the error occurred, and the Log window displays messages about the error.

d. SAS stops processing the step in which the error occurred, and the Output window displays messages about the error.

5. A syntax error occurs when ?

a. some data values are not appropriate for the SAS statements that are specified in a program.

b. the form of the elements in a SAS statement is correct, but the elements are not valid for that usage.

c. program statements do not conform to the rules of the SAS language.

d. none of the above

6. How can you tell whether you have specified an invalid option in a SAS program? ?

a. A log message indicates an error in a statement that seems to be valid.

b. A log message indicates that an option is not valid or not recognized.

c. The message "PROC running" or "DATA step running" appears at the top of the active window.

d. You can't tell until you view the output from the program.

7. Which of the following programs contains a syntax error? ?

a.
```
proc sort data=sasuser.mysales;
   by region;
run;
```

b.
```
dat sasuser.mysales;
   set mydata.sales99;
   where sales<5000;
run;
```

c.
```
proc print data=sasuser.mysales label;
   label region='Sales Region';
run;
```

d. none of the above

8. What should you do after submitting the following program in the Windows or UNIX operating environment? ?

```
proc print data=mysales;
   where state='NC;
run;
```

a. Submit a RUN statement to complete the PROC step.

b. Recall the program. Then add a quotation mark and resubmit the corrected program.

c. Cancel the submitted statements. Then recall the program, add a quotation mark, and resubmit the corrected program.

d. Recall the program. Then replace the invalid option and resubmit the corrected program.

9. Which of the following commands opens a file in the code editing window? ?

a. `file 'd:\programs\sas\newprog.sas'`

b. `include 'd:\programs\sas\newprog.sas'`

c. `open 'd:\programs\sas\newprog.sas'`

d. all of the above

10. Suppose you submit a short, simple DATA step. If the active window displays the message "DATA step running" for a long time, what probably happened? ?

a. You misspelled a keyword.

b. You forgot to end the DATA step with a RUN statement.

c. You specified an invalid data set option.

d. Some data values weren't appropriate for the SAS statements that you specified.

Answers

1. Correct answer: d

Although you can write SAS statements in almost any format, a consistent layout enhances readability and enables you to understand the program's purpose. It's a good idea to begin DATA and PROC steps in column one, to indent statements within a step, to begin RUN statements in column one, and to include a RUN statement after every DATA step or PROC step.

2. Correct answer: a

To modify programs that contain errors, if you use the Program Editor window, you usually need to recall the submitted statements from the recall buffer to the Program Editor window, where you can correct the problems. After correcting the errors, you can resubmit the revised program. However, before doing so, it's a good idea to clear the messages from the Log window so that you don't confuse the old error messages with the new messages. Remember to check the Log window again to verify that your program ran correctly.

3. Correct answer: d

The missing quotation mark in the LABEL statement causes SAS to misinterpret the statements in the program. When you submit the program, SAS is unable to resolve the PROC step, and a "PROC PRINT running" message appears at the top of the active window.

4. Correct answer: c

Syntax errors generally cause SAS to stop processing the step in which the error occurred. When a program that

contains an error is submitted, messages regarding the problem also appear in the Log window. When a syntax error is detected, the Log window displays the word ERROR, identifies the possible location of the error, and gives an explanation of the error.

5. Correct answer: c

Syntax errors are common types of errors. Some SAS system options and features of the code editing window can help you identify syntax errors. Other types of errors include data errors and logic errors.

6. Correct answer: b

When you submit a SAS statement that contains an invalid option, a log message notifies you that the option is not valid or not recognized. You should recall the program, remove or replace the invalid option, check your statement syntax as needed, and resubmit the corrected program.

7. Correct answer: b

The DATA step contains a misspelled keyword (dat instead of data). However, this is such a common (and easily interpretable) error that SAS produces only a warning message, not an error.

8. Correct answer: c

This program contains an unbalanced quotation mark. When you have an unbalanced quotation mark, SAS is often unable to detect the end of the statement in which it occurs. Simply adding a quotation mark and resubmitting your program usually does not solve the problem. SAS still considers the quotation marks to be unbalanced. To correct the error, you need to resolve the unbalanced quotation mark before you recall, correct, and resubmit the program.

9. Correct answer: b

One way of opening a file in the code editing window is by using the INCLUDE command. Using the INCLUDE command enables you to open a single program or combine stored programs in a single window. To save a SAS program, you can use the FILE command.

10. Correct answer: b

Without a RUN statement (or a following DATA or PROC step), the DATA step doesn't execute, so it continues to run. Unbalanced quotation marks can also cause the "DATA step running" message if relatively little code follows the unbalanced quotation mark. The other three problems above generate errors in the Log window.