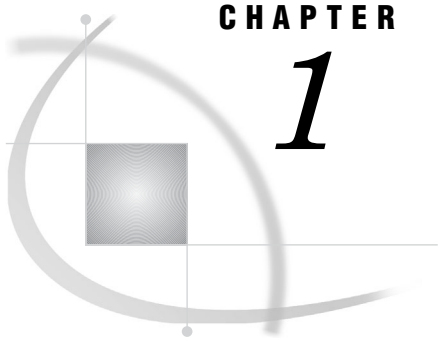


CHAPTER

1

What Is the SAS System?

Introduction to the SAS System

SAS is an integrated system of software solutions that enables you to perform the following tasks:

- data entry, retrieval, and management
- report writing and graphics design
- statistical and mathematical analysis
- business forecasting and decision support
- operations research and project management
- applications development

How you use SAS depends on what you want to accomplish. Some people use many of the capabilities of the SAS System, and others use only a few.

At the core of the SAS System is Base SAS software which is the software product that you will learn to use in this documentation. This section presents an overview of Base SAS. It introduces the capabilities of Base SAS, addresses methods of running SAS, and outlines various types of output.

Components of Base SAS Software

Overview of Base SAS Software

Base SAS software contains the following:

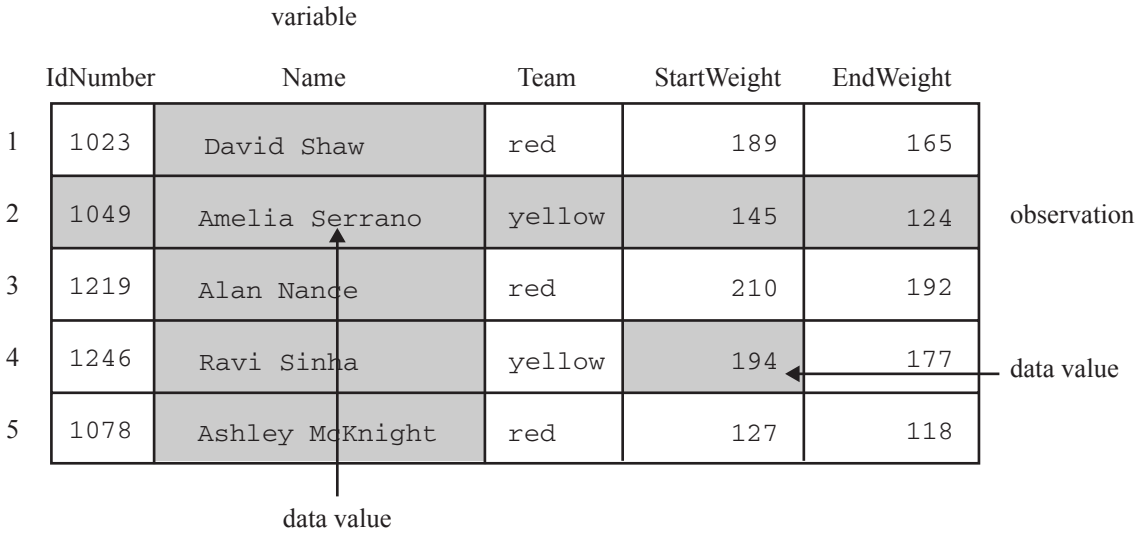
- a data management facility
- a programming language
- data analysis and reporting utilities

Learning to use Base SAS enables you to work with these features of SAS. It also prepares you to learn other SAS products, because all SAS products follow the same basic rules.

Data Management Facility

SAS organizes data into a rectangular form or table that is called a *SAS data set*. The following figure shows a SAS data set. The data describes participants in a 16-week weight program at a health and fitness club. The data for each participant includes an identification number, name, team name, and weight (in U.S. pounds) at the beginning and end of the program.

Figure 1.1 Rectangular Form of a SAS Data Set



In a SAS data set, each row represents information about an individual entity and is called an *observation*. Each column represents the same type of information and is called a *variable*. Each separate piece of information is a *data value*. In a SAS data set,

an observation contains all the data values for an entity; a variable contains the same type of data value for all entities.

To build a SAS data set with Base SAS, you write a program that uses statements in the SAS programming language. A SAS program that begins with a DATA statement and typically creates a SAS data set or a report is called a *DATA step*.

The following SAS program creates a SAS data set named WEIGHT_CLUB from the health club data:

```
data weight_club; ❶
   input IdNumber 1-4 Name $ 6-24 Team $ StartWeight EndWeight; ❷
   Loss=StartWeight-EndWeight; ❸
   datalines; ❹
1023 David Shaw          red 189 165 ❺
1049 Amelia Serrano      yellow 145 124 ❺
1219 Alan Nance          red 210 192 ❺
1246 Ravi Sinha          yellow 194 177 ❺
1078 Ashley McKnight     red 127 118 ❺
; ❻

run;
```

The following list corresponds to the numbered items in the preceding program:

❶ The DATA statement tells SAS to begin building a SAS data set named WEIGHT_CLUB.

❷ The INPUT statement identifies the fields to be read from the input data and names the SAS variables to be created from them (IdNumber, Name, Team, StartWeight, and EndWeight).

❸ The third statement is an assignment statement. It calculates the weight each person lost and assigns the result to a new variable, Loss.

❹ The DATALINES statement indicates that data lines follow.

❺ The data lines follow the DATALINES statement. This approach to processing raw data is useful when you have only a few lines of data. (Later sections show ways to access larger amounts of data that are stored in files.)

❻ The semicolon signals the end of the raw data, and is a step boundary. It tells SAS that the preceding statements are ready for execution.

Note: By default, the data set WEIGHT_CLUB is temporary; that is, it exists only for the current job or session. For information about how to create a permanent SAS data set, see Chapter 2, "Introduction to DATA Step Processing," on page 19. △

Programming Language

Elements of the SAS Language

The statements that created the data set WEIGHT_CLUB are part of the SAS programming language. The SAS language contains statements, expressions, functions and CALL routines, options, formats, and informats – elements that many programming languages share. However, the way you use the elements of the SAS language depends on certain programming rules. The most important rules are listed in the next two sections.

Rules for SAS Statements

The conventions that are shown in the programs in this documentation, such as indenting of subordinate statements, extra spacing, and blank lines, are for the purpose of clarity and ease of use. They are not required by SAS. There are only a few rules for writing SAS statements:

- SAS statements end with a semicolon.
- You can enter SAS statements in lowercase, uppercase, or a mixture of the two.
- You can begin SAS statements in any column of a line and write several statements on the same line.
- You can begin a statement on one line and continue it on another line, but you cannot split a word between two lines.
- Words in SAS statements are separated by blanks or by special characters (such as the equal sign and the minus sign in the calculation of the Loss variable in the WEIGHT_CLUB example).

Rules for Most SAS Names

SAS names are used for SAS data set names, variable names, and other items. The following rules apply:

- A SAS name can contain from one to 32 characters.
- The first character must be a letter or an underscore (_).
- Subsequent characters must be letters, numbers, or underscores.
- Blanks cannot appear in SAS names.

Special Rules for Variable Names

For variable names only, SAS remembers the combination of uppercase and lowercase letters that you use when you create the variable name. Internally, the case of letters does not matter. “CAT,” “cat,” and “Cat” all represent the same variable. But for presentation purposes, SAS remembers the initial case of each letter and uses it to represent the variable name when printing it.

Data Analysis and Reporting Utilities

The SAS programming language is both powerful and flexible. You can program any number of analyses and reports with it. SAS can also simplify programming for you with its library of built-in programs known as *SAS procedures*. SAS procedures use data values from SAS data sets to produce preprogrammed reports, requiring minimal effort from you.

For example, the following SAS program produces a report that displays the values of the variables in the SAS data set WEIGHT_CLUB. Weight values are presented in U.S. pounds.

```
options linesize=80 pagesize=60 pageno=1 nodate;

proc print data=weight_club;
   title 'Health Club Data';
run;
```

This procedure, known as the PRINT procedure, displays the variables in a simple, organized form. The following output shows the results:

Output 1.1 Displaying the Values in a SAS Data Set

Health Club Data 1

Obs	Id Number	Name	Team	Start Weight	End Weight	Loss
1	1023	David Shaw	red	189	165	24
2	1049	Amelia Serrano	yellow	145	124	21
3	1219	Alan Nance	red	210	192	18
4	1246	Ravi Sinha	yellow	194	177	17
5	1078	Ashley McKnight	red	127	118	9

To produce a table showing mean starting weight, ending weight, and weight loss for each team, use the TABULATE procedure.

```
options linesize=80 pagesize=60 pageno=1 nodate;

proc tabulate data=weight_club;
   class team;
   var StartWeight EndWeight Loss;
   table team, mean*(StartWeight EndWeight Loss);
   title 'Mean Starting Weight, Ending Weight,';
   title2 'and Weight Loss';
run;
```

The following output shows the results:

Output 1.2 Table of Mean Values for Each Team

Mean Starting Weight, Ending Weight, and Weight Loss 1

	Mean		
	StartWeight	EndWeight	Loss
Team			
red	175.33	158.33	17.00
yellow	169.50	150.50	19.00

A portion of a SAS program that begins with a PROC (procedure) statement and ends with a RUN statement (or is ended by another PROC or DATA statement) is called a *PROC step*. Both of the PROC steps that create the previous two outputs comprise the following elements:

- a PROC statement, which includes the word PROC, the name of the procedure you want to use, and the name of the SAS data set that contains the values. (If you omit the DATA= option and data set name, the procedure uses the SAS data set that was most recently created in the program.)
- additional statements that give SAS more information about what you want to do, for example, the CLASS, VAR, TABLE, and TITLE statements.

- a RUN statement, which indicates that the preceding group of statements is ready to be executed.

Output Produced by the SAS System

Traditional Output

A SAS program can produce some or all of the following kinds of output:

a SAS data set
: contains data values that are stored as a table of observations and variables. It also stores descriptive information about the data set, such as the names and arrangement of variables, the number of observations, and the creation date of the data set. A SAS data set can be temporary or permanent. The examples in this section create the temporary data set WEIGHT_CLUB.

the SAS log
: is a record of the SAS statements that you entered and of messages from SAS about the execution of your program. It can appear as a file on disk, a display on your monitor, or a hardcopy listing. The exact appearance of the SAS log varies according to your operating environment and your site. The output in Output 1.3 shows a typical SAS log for the program in this section.

a report or simple listing
: ranges from a simple listing of data values to a subset of a large data set or a complex summary report that groups and summarizes data and displays statistics. The appearance of procedure output varies according to your site and the options that you specify in the program, but the output in Output 1.1 and Output 1.2 illustrate typical procedure output. You can also use a DATA step to produce a completely customized report (see “Creating Customized Reports” on page 391).

other SAS files such as catalogs
: contain information that cannot be represented as tables of data values. Examples of items that can be stored in SAS catalogs include function key settings, letters that are produced by SAS/FSP software, and displays that are produced by SAS/GRAPH software.

external files or entries in other databases
: can be created and updated by SAS programs. SAS/ACCESS software enables you to create and update files that are stored in databases such as Oracle.

Output 1.3 Traditional Output: A SAS Log

```
NOTE: PROCEDURE PRINTTO used:
      real time           0.02 seconds
      cpu time            0.01 seconds

22
23   options pagesize=60 linesize=80 pageno=1 nodate;
24
25   data weight_club;
26      input IdNumber 1-4 Name $ 6-24 Team $ StartWeight EndWeight;
27      Loss=StartWeight-EndWeight;
28      datalines;
NOTE: The data set WORK.WEIGHT_CLUB has 5 observations and 6 variables.
NOTE: DATA statement used:
      real time           0.14 seconds
      cpu time            0.07 seconds

34   ;
35
36
37   proc tabulate data=weight_club;
38      class team;
39      var StartWeight EndWeight Loss;
40      table team, mean*(StartWeight EndWeight Loss);
41      title 'Mean Starting Weight, Ending Weight,';
42      title2 'and Weight Loss';
43   run;
NOTE: There were 5 observations read from the data set WORK.WEIGHT_CLUB.
NOTE: PROCEDURE TABULATE used:
      real time           0.18 seconds
      cpu time            0.09 seconds

44   proc printto; run;
```

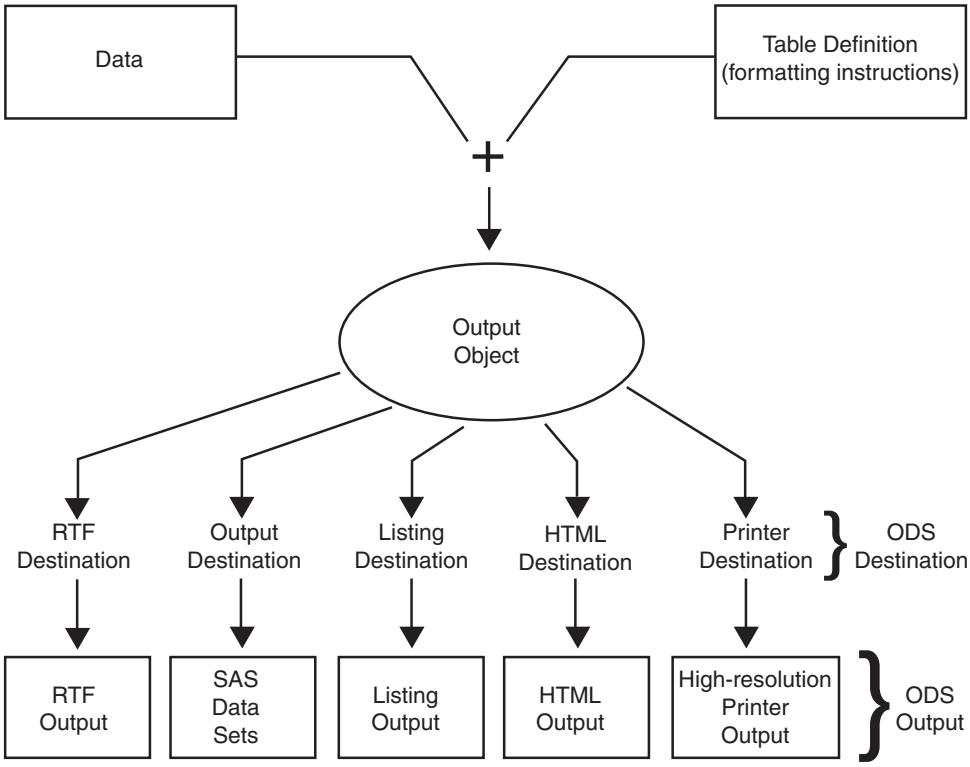
Output from the Output Delivery System (ODS)

The Output Delivery System (ODS) enables you to produce output in a variety of formats, such as

- an HTML file
- a traditional SAS Listing (monospace)
- a PostScript file
- an RTF file (for use with Microsoft Word)
- an output data set

The following figure illustrates the concept of output for SAS Version 8.

Figure 1.2 Model of the Production of ODS Output



The following definitions describe the terms in the preceding figure:

data
Each procedure that supports ODS and each DATA step produces data, which contains the results (numbers and characters) of the step in a form similar to a SAS data set.

table definition
The table definition is a set of instructions that describes how to format the data. This description includes but is not limited to

- the order of the columns
- text and order of column headings
- formats for data
- font sizes and font faces

output object
ODS combines formatting instructions with the data to produce an output object. The output object, therefore, contains both the results of the procedure or DATA step and information about how to format the results. An output object has a name, a label, and a path.

Note: Although many output objects include formatting instructions, not all do. In some cases the output object consists of only the data. △

ODS destinations
An ODS destination specifies a specific type of output. ODS supports a number of destinations, which include the following:

RTF
produces output that is formatted for use with Microsoft Word.

Output
produces a SAS data set.

Listing
produces traditional SAS output (monospace format).

HTML
produces output that is formatted in Hyper Text Markup Language (HTML). You can access the output on the web with your web browser.

Printer
produces output that is formatted for a high-resolution printer. An example of this type of output is a PostScript file.

ODS output
ODS output consists of formatted output from any of the ODS destinations.

For more information about ODS output, see Chapter 23, "Directing SAS Output and the SAS Log," on page 351 and Chapter 32, "Understanding and Customizing SAS Output: The Output Delivery System (ODS)," on page 565.

For complete information about ODS, see *SAS Output Delivery System: User's Guide*.

Ways to Run SAS Programs

Selecting an Approach

There are several ways to run SAS programs. They differ in the speed with which they run, the amount of computer resources that are required, and the amount of interaction that you have with the program (that is, the kinds of changes you can make while the program is running).

The examples in this documentation produce the same results, regardless of the way you run the programs. However, in a few cases, the way that you run a program determines the appearance of output. The following sections briefly introduce different ways to run SAS programs.

SAS Windowing Environment

The SAS windowing environment enables you to interact with SAS directly through a series of windows. You can use these windows to perform common tasks, such as locating and organizing files, entering and editing programs, reviewing log information, viewing procedure output, setting options, and more. If needed, you can issue operating system commands from within this environment. Or, you can suspend the current SAS windowing environment session, enter operating system commands, and then resume the SAS windowing environment session at a later time.

Using the SAS windowing environment is a quick and convenient way to program in SAS. It is especially useful for learning SAS and developing programs on small test files. Although it uses more computer resources than other techniques, using the SAS windowing environment can save a lot of program development time.

For more information about the SAS windowing environment, see Chapter 39, "Using the SAS Windowing Environment," on page 655.

SAS/ASSIST Software

One important feature of SAS is the availability of SAS/ASSIST software. SAS/ASSIST provides a point-and-click interface that enables you to select the tasks that you want to perform. SAS then submits the SAS statements to accomplish those tasks. You do not need to know how to program in the SAS language in order to use SAS/ASSIST.

SAS/ASSIST works by submitting SAS statements just like the ones shown earlier in this section. In that way, it provides a number of features, but it does not represent the total functionality of SAS software. If you want to perform tasks other than those that are available in SAS/ASSIST, you need to learn to program in SAS as described in this documentation.

Noninteractive Mode

In noninteractive mode, you prepare a file that contains SAS statements and any system statements that are required by your operating environment, and submit the program. The program runs immediately and occupies your current workstation session. You cannot continue to work in that session while the program is running,* and you usually cannot interact with the program.** The log and procedure output go to prespecified destinations, and you usually do not see them until the program ends. To modify the program or correct errors, you must edit and resubmit the program.

Noninteractive execution may be faster than batch execution because the computer system runs the program immediately rather than waiting to schedule your program among other programs.

Batch Mode

To run a program in batch mode, you prepare a file that contains SAS statements and any system statements that are required by your operating environment, and then you submit the program.

You can then work on another task at your workstation. While you are working, the operating environment schedules your job for execution (along with jobs submitted by other people) and runs it. When execution is complete, you can look at the log and the procedure output.

The central feature of batch execution is that it is completely separate from other activities at your workstation. You do not see the program while it is running, and you cannot correct errors at the time they occur. The log and procedure output go to prespecified destinations; you can look at them only after the program has finished running. To modify the SAS program, you edit the program with the editor that is supported by your operating environment and submit a new batch job.

When sites charge for computer resources, batch processing is a relatively inexpensive way to execute programs. It is particularly useful for large programs or when you need to use your workstation for other tasks while the program is executing. However, for learning SAS or developing and testing new programs, using batch mode might not be efficient.

---

* In a workstation environment, you can switch to another window and continue working.

** Limited ways of interaction are available. You can, for example, use the asterisk (*) option in a %INCLUDE statement in your program.

Interactive Line Mode

In an interactive line-mode session, you enter one line of a SAS program at a time, and SAS executes each DATA or PROC step automatically as soon as it recognizes the end of the step. You usually see procedure output immediately on your display monitor. Depending on your site's computer system and on your workstation, you may be able to scroll backward and forward to see different parts of your log and procedure output, or you may lose them when they scroll off the top of your screen. There are limited facilities for modifying programs and correcting errors.

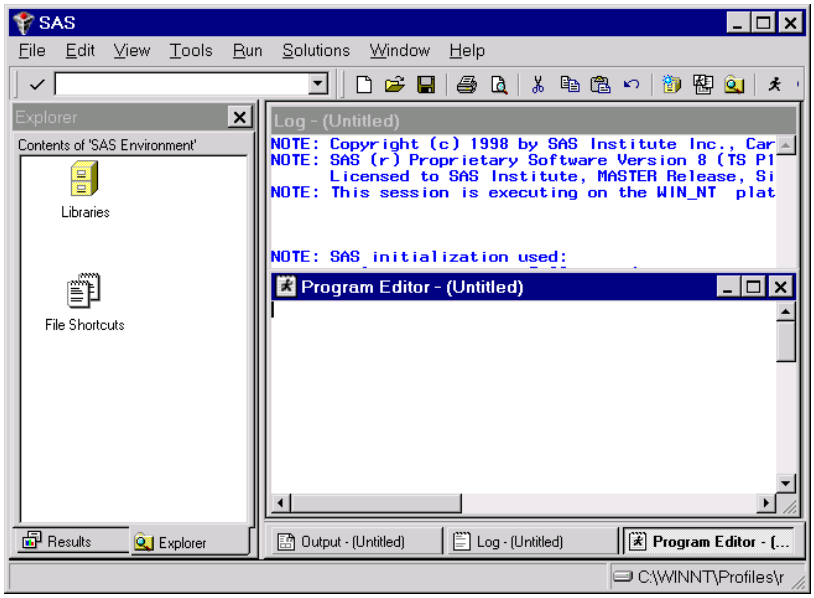
Interactive line-mode sessions use fewer computer resources than a windowing environment. If you use line mode, you should familiarize yourself with the %INCLUDE, %LIST, and RUN statements in *SAS Language Reference: Dictionary*.

Running Programs in the SAS Windowing Environment

You can run most programs in this documentation by using any of the methods that are described in the previous sections. This documentation uses the SAS windowing environment (as it appears on Windows and UNIX operating environments) when it is necessary to show programming within a SAS session. The SAS windowing environment appears differently depending on the operating environment that you use. For more information about the SAS windowing environment, see Chapter 39, "Using the SAS Windowing Environment," on page 655.

The following example gives a brief overview of a SAS session that uses the SAS windowing environment. When you invoke SAS, the following windows appear.

Display 1.1 SAS Windowing Environment

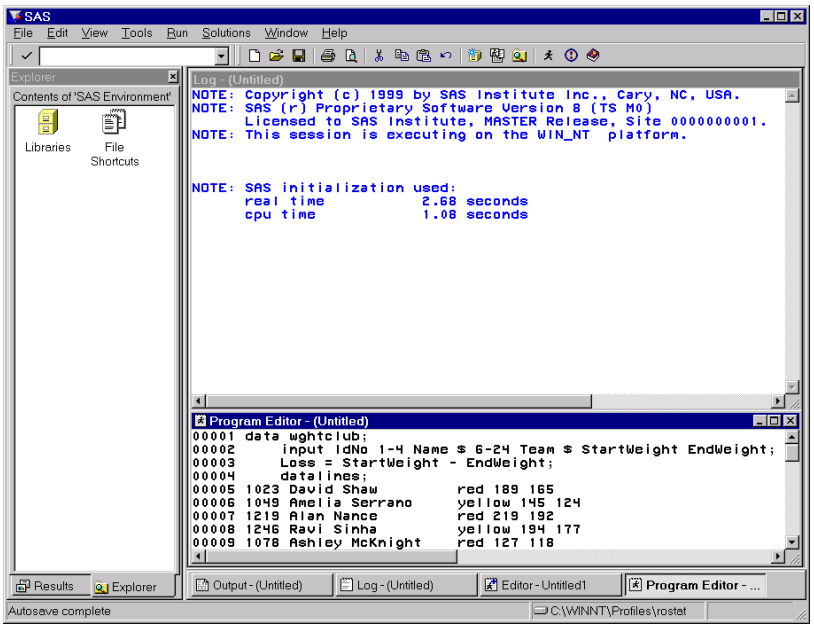


The specific window placement, display colors, messages, and some other details vary according to your site, your monitor, and your operating environment. The window on the left side of the display is the SAS Explorer window, which you can use to assign and locate SAS libraries, files, and other items. The window at the top right is the Log

window; it contains the SAS log for the session. The window at the bottom right is the Program Editor window. This window provides an editor in which you edit your SAS programs.

To create the program for the health and fitness club, type the statements in the Program Editor window. You can turn line numbers on or off to facilitate program creation. The following display shows the beginning of the program.

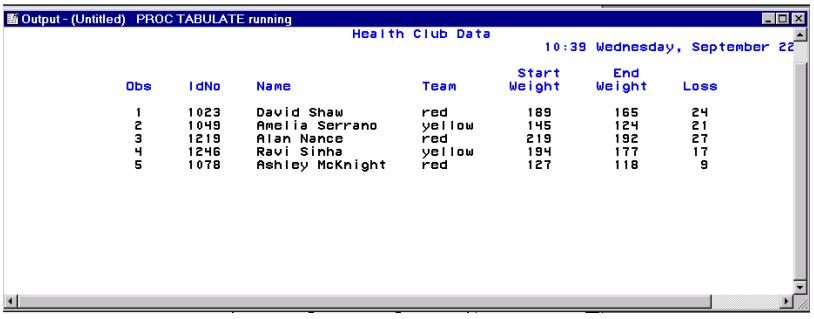
Display 1.2 Editing a Program in the Program Editor Window



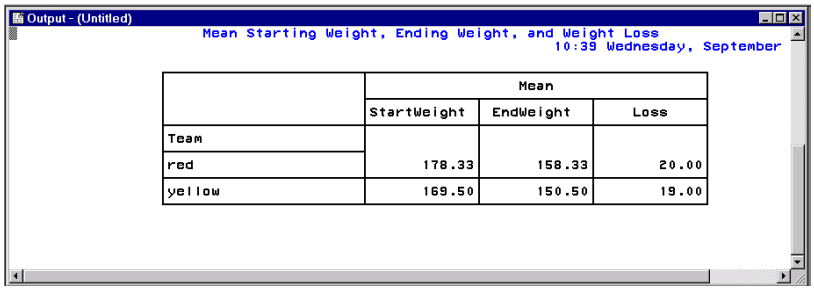
When you fill the Program Editor window, scroll down to continue typing the program. When you finish editing the program, submit it to SAS and view the output. (If SAS does not create output, check the SAS log for error messages.)

The following displays show the first and second pages of the Output window.

Display 1.3 The First Page of Output in the Output Window



Display 1.4 The Second Page of Output in the Output Window


Output - (Untitled)

Mean Starting Weight, Ending Weight, and Weight Loss
10:39 Wednesday, September

	Mean		
	StartWeight	EndWeight	Loss
Team			
red	178.33	158.33	20.00
yellow	169.50	150.50	19.00

After you finish viewing the output, you can return to the Program Editor window to begin creating a new program.

By default, the output from all submissions remains in the Output window, and all statements that you submit remain in memory until the end of your session. You can view the output at any time, and you can recall previously submitted statements for editing and resubmitting. You can also clear a window of its contents.

All the commands that you use to move through the SAS windowing environment can be executed as words or as function keys. You can also customize the SAS windowing environment by determining which windows appear, as well as by assigning commands to function keys. For more information about customizing the SAS windowing environment, see Chapter 40, "Customizing the SAS Environment," on page 695.

Review of SAS Tools

Statements

DATA *SAS-data-set*;
begins a DATA step and tells SAS to begin creating a SAS data set. *SAS-data-set* names the data set that is being created.

%INCLUDE *source(s)* </<SOURCE2> <S2=*length*> <*host-options*>>;
brings SAS programming statements, data lines, or both into a current SAS program.

RUN;
tells SAS to begin executing the preceding group of SAS statements.

For more information, see Statements in *SAS Language Reference: Dictionary*.

Procedures

PROC *procedure* <DATA=*SAS-data-set*>;
begins a PROC step and tells SAS to invoke a particular SAS procedure to process the SAS data set that is specified in the DATA= option. If you omit the DATA= option, then the procedure processes the most recently created SAS data set in the program.

For more information about using procedures, see the *Base SAS Procedures Guide*.

Learning More

Basic SAS usage
: For an entry-level introduction to basic SAS programming language, see *The Little SAS Book: A Primer, Second Edition*.

DATA step
: For more information about how to create SAS data sets, see Chapter 2, "Introduction to DATA Step Processing," on page 19.

DATA step processing
: For more information about DATA step processing, see Chapter 6, "Understanding DATA Step Processing," on page 97.

  For information about how to easily use the SAS environment, see *Getting Started with the SAS System*.