

SAS Intro Manual

Written by Tim Arnold

- SAS: An Introduction
- Getting SAS Started
- Controlling SAS Windows
- Programming with SAS
- How to Edit your SAS Program
- Running (executing) the program
- Saving the program
- Printing SAS Output and Programs
- SAS Programming Tips
- Some SAS Error Messages
- SAS Tutorial

Logging In

When you enter SICL, the screen of your workstation or terminal will probably be blank. The machines are never turned off--just press the RETURN key and the login screen will appear.

If the screen is dark, press the RETURN key and it should light up.

Place the cursor over the login box and click the mouse button so that the login box lights up.

Login:

Type in your username (this will be your first two initials and the first six letters of your last name) and hit return. The password box should then light up. Type your password, which will be your school ID number. (You need to change this as soon as possible.) The password will not appear on the screen as you type. This is for security reasons.

The login window will disappear. The screen should now display two windows: console and xterm. The xterm window is in UNIX and you should see the UNIX system prompt: % . (This is the window where you add your software, your class and start the package that you have added.)

Note: You can change your password using the command "passwd". Choose something you can remember, but preferably not something easily guessed. For instance: names, phone numbers and words appearing in a dictionary are bad choices.

How to start SAS

After you log in to your SICL account you will see two windows sitting in a grey background. In the window called "xterm" type the following sequence of commands:

```
add sas
add classname_info (where classname is the course you are taking.)
sas
```

The files for your class will be stored in /ncsu/classname_info. (For example, if you are doing work for st512, you would type:

```
add sas
add st512_info
sas
```

The files for st512 are stored in /ncsu/st512_info/)

How to end SAS

To quit your SAS session type:

bye

on the command line of any SAS window and press RETURN. If you are using pulldown menus (see below), select "Exit" from the "File" menu.

What you see when you start SAS

A few seconds after you type `sas` in the xterm window, three additional windows will appear on the screen. You may want to move them around so you can see them all. (You move windows by sliding the mouse over the mouse pad until the on-screen cursor is on the titlebar of the window you want to move. Then, while holding down the first button slide the mouse again. The cursor should drag a ghost outline of the window until you stop. Release the mouse button when the ghost is positioned where you want the window to be.) Notice that you can make a different windows "current" by placing the on-screen cursor in the window you want to use and clicking the mouse button. The three main SAS windows are:

Program Editor window

You write and edit SAS programs in this window.

Log window

This window collects information from the SAS system as your programs execute. Warnings and error messages from SAS are written here.

Output window

The final results requested by your program will be printed in this window.

All of these windows will have a command line at the top of the window (just below the title bar). You may either use the command line to "communicate" with SAS or you can use pulldown menus. Hit the key sequence "Control-a" to switch back and forth between the menus and the command line. To use the menus, position the cursor above the menu you want to see and press the left mouse button. Holding the mouse button, scroll down the menu by sliding the mouse towards you. Release the mouse button when the desired option is highlighted.

What you will usually do in SAS

1. Log into your SICL account. Your login name is made up of your first initial, middle initial and the first six letters of your last name. Type your login name in the login box and hit return. The password box should light up. Your password is your social security number or your school ID number. (You should change this as soon as possible- use the "passwd" command.) Type in your password and hit return. (Wait for the xterm and console windows to appear.)
2. Type: "add sas" in the xterm window and hit the return key
3. Add the class you are taking by typing: "add classname_info" in the xterm window and hit the return key. Remember classname refers to the class you are taking.
4. Type "sas" in the xterm window and hit the return key. In a moment the SAS windows will appear.
5. (A) If you are writing your own program, press RETURN. This puts the cursor into the programming area of the Program window, so you can type in your program.

(B) If you want to use a SAS program which is in a file (either a file your instructor has provided or one you wrote in an earlier SAS session) type:
include 'filename'
on the Program window's command line and press RETURN. Or choose "Open-> Read file" from the "File" pulldown menu. The filename is the name of the file containing the program.
6. Hold down the Control key and press the letter E. This key sequence (Control-E) will execute your program (first making sure the Log and Output windows are clear). Your program will disappear, but don't worry, you will be able to get it back.
7. Use the mouse to go to the top of the Log window Check all the messages for errors or warnings. If you have errors:
Go back to the Program window

Retrieve your program (Control-R)

Fix the error and execute again (step 6).

8. When your program executes and no error messages or warnings are listed in the Log window, use the mouse to go to the top of the Output window.
9. If the output produced by your program looks like it should, you may want to print or save it for later use. Make sure you are in the Output window.
To print the output, type "print" on the command line or choose the print option from the "file" pulldown window.
To save the output in a file, type file 'filename' and press RETURN or choose "Save as-> Write to file" in the "File" menu. Here, filename is the name you want to give the file which will contain your output (see sidebar for one type of naming scheme).
10. the program window and hit Control-R, type "recall", or choose "Recall text" from the "Locals" menu to recall your program.
To print the program, type "print" on the command line or choose the print option from the "file" pulldown window.
To save the program in a file, type file 'filename' and press RETURN or choose "Save as -> Write to file" from the "File" menu. Here, filename is the name you want to give the file which will contain the program (this name should be different from the name you specified for output).
11. If you want to use SAS some more, go to the Program window, use Control-W, type "clear", or chose "Clear text" in the "Edit" menu to erase the window (make sure you saved it as in step 7 if you will need it later). Continue as in the previous steps.
If you are finished with your work, (make sure you saved your program and output if you will need them later), type
bye
on the command line of any window and press RETURN. If you are using pulldown menus select "Exit" from the "File" menu. The SAS windows will disappear and the xterm window will once again have the UNIX ready prompt (%).
12. To log out, go to the background window of the screen and hold down the right mouse button. The main system menu will appear. Drag down to the logout option on the menu and release the mouse button. After a few seconds, the login window should reappear.

USING SAS ON THE SICL SYSTEM

Entering SAS: Most of the time, unless you wish to do some "housecleaning," the first thing you will do after logging in will be to enter SAS. At the % prompt in the xterm window, type

```
add sas (hit the RETURN key)
add classname_info (hit the RETURN key)
sas (hit the RETURN key)
```

("classname" refers to class you are taking that requires SAS; for example if you are taking st512 you would type: add st512_info.)

After you follow the above procedure, the SAS windows will appear on the screen. There are three windows, you will probably want to move them around (see below) so that you can see all three simultaneously. The three windows are the program editor, the log window and the output window. The window names will appear in the title boxes of the corresponding windows.

PROGRAM EDITOR window: When you start SAS, the cursor will be on the command line of the PROGRAM EDITOR. This is where you will enter your programs and submit them for execution (i.e., to run). In a moment we will discuss how to type in and edit a program in this window.

LOG window: The LOG window will contain a copy of your program when it is through running, annotated with comments from the system and error messages, if any.

OUTPUT window: The OUTPUT window will contain any output generated by your program.

Moving among windows : Note that each window has a command line. You may switch back and forth from the command line to pull down menus by holding the control key down and pressing a. (This key sequence is denoted by Control-a.) You may move the windows by sliding the mouse over the mouse pad until the on-screen cursor is on the titlebar of the window you want to move and then while holding down the first button slide the mouse again. The cursor should drag a ghost outline of the window until you stop. Release the mouse button when the ghost is positioned where you want the window to be.

In SAS you will have move back and forth between the windows. To do this you just slide the mouse until the on-screen cursor is inside the window you want to use and click the mouse button. The frame of the window should darken letting you know that it is the "current" window.

Editing a program: To type in a program, go to the PROGRAM EDITOR window. In this window, you will type the statements that make up your SAS program.

To begin typing, simply use the arrow keys or the mouse to move down to the first line of the window and start typing the statements. RETURN will take you to the next line. You can move around freely in the window with either the arrow keys or the mouse. If you make a mistake, you can simply backspace over it as if you were using a word processing program.

There is an insert mode and a typeover mode -- in the first, you can insert text in front of other text on a given line by simply moving the cursor (with the arrow keys) to the spot you want to insert and type. In the second, you type over things that are already there. When you start SAS, you are in typeover mode. To get into insert mode, type the key sequence Control-x (or Control-n for remote access). To get delete mode back, press that same key sequence again. Notice that the cursor changes shape depending on which mode is being used. It is a block cursor in typeover mode and an underline cursor in insert mode. Try using both modes so that you understand the difference.

Saving a program file: Once you have typed in your program and have it the way you want it, you will want to save it in a file so that you can access it again. In particular, you may run the program and find it has errors, but you need to go to class and wish to come back later to work on it some more. To save a program in a file, type the following command at the command line of the PROGRAM EDITOR window:
file 'filename'

and hit RETURN. Here, filename is the name you have chosen for the file. The name must be in single quotes as above. Notice the single quotes -- you must enclose the name in them. If you are using pulldown menus, go to the "file" menu and select "Save as -> Write to file" . You will then be prompted for a filename. The file is now saved under this name in your directory.

You can also save the contents of the LOG and OUTPUT windows to files if you desire with the same command in the same way.

Including a file: While we're on the subject, suppose you come back from your class, login, enter SAS, and wish to work on the program some more. To get the contents of the file back into the PROGRAM EDITOR window, type at the command line of the

PROGRAM EDITOR window:

```
include 'filename'
```

and hit return. Here, filename is the name of the file whose contents you wish to appear in the window. Again, the name must be enclosed in single quotes. If you are using pulldown menus select "Open -> Read file" in the "File" menu. In fact, even if the window already contains some program statements, including a file will cause the statements in that file to be added to the end of those already there. This is useful if you want to combine 2 programs into 1.

The include command is only valid in the PROGRAM EDITOR window.

Running a program:

Suppose you have entered a program and wish to run it, that is, submit it to SAS to obtain the analysis you desire. Go to the command line (use arrow keys, the mouse, or the keystroke sequence Control-f). Type submit OR use Control-E. If you are using pulldown menus select "Submit" from the "Locals" menu. The program will disappear from the window, as will the cursor. Note that if use "submit" (either by typing it on the command line or by using pulldown menus) rather than control-e you must "clear" both the output window and the log window before submitting another program (see below for more details). Examining the results of running a program:

LOG window: When the cursor reappears on the command line, you know that the program has run. Move to the LOG window. By placing the mouse on the scroll bar in the frame of the LOG window, you can "scroll" through the window. If there were errors, you can try to determine their nature by looking at the log on the screen.

OUTPUT window: Once you have examined the contents of the LOG window, you may wish to view the results of the analysis. Depending on the severity of any errors in the program, the OUTPUT window may or may not contain anything. An error-free program (or one with only minor errors) will, of course, generate output. To view it, move to the OUTPUT window. You may scroll through this window the same way as you scroll through the LOG window. If you are satisfied with the output, and wish to obtain a "hard copy", you will wish to print out the contents of the OUTPUT window. Again, only print essential results. In the interests of conservation, don't print the results of every run, but only your final run (to turn in). See below for how to print the contents of a window and how to "clear" the window.

The recall command: Suppose you need to go back to the program to correct errors you have observed in the LOG. Thus, you need to move back to the PROGRAM EDITOR window and get the program to reappear in this window. You have 2 choices on how to do this:

provided you had saved the program in a file, you can use the include command as described above.
you can use the recall command. You can either type
recall
and hit RETURN on the command line or use the Control-R keystroke. (If you are using pulldown menus, use "Locals -> Recall text".) The effect of this command is to recall into the window the last set of statements that was there.

The clear command:

Whether or not you print the contents of a window, you should type clear at the command line (or choose "Clear text" from the "Edit" menu if using pulldown menus) when you are finished viewing its contents on the screen. If you do not, and you run subsequent programs, the logs for all of them will continue to "pile up" in the LOG window, and their output will "pile up" in the OUTPUT window, making it difficult to determine which is the relevant part you wish to look at in each case. Further, if you print the contents of the window, everything will be printed, wasting paper and tying up the printer. The clear command deletes everything from a window. It is thus a good idea to remember to clear anything with which you are finished as you work.

Printing the contents of a window:

On the command line of the window you want to be printed, type the command "print". If you are using menus you can also choose "print" from the file menu. This prints the entire contents of the window, not just the part of the window you see on the screen.

Exiting SAS: You may exit SAS from any window by simply typing from any command line

bye

and then hitting RETURN or by choosing "Exit" from the "File" pulldown window. This will return you to the % prompt (at which point you may logout, if desired).

SAS Window Commands

Each SAS window has a command line. You can control your SAS session by entering special keywords on this line. Commonly used commands include: SAS COMMAND Line Keywords

| | |
|----------------------|--|
| **Submit | execute program |
| Clear | clears (erases) window |
| **Recall | retrieve window contents |
| bye | end your SAS session |
| **include 'filename' | retrieves file called filename into the window |
| file 'filename' | saves window contents into a file called filename. |
| reshow | redraws the screen |
| top | goes to the top line |
| bot | goes to the bottom line |
| down | scrolls down one page |
| down n | scrolls down n lines |
| up | scrolls up one page |
| up n | scrolls up one page |

Function Keys and Pulldown Menus

Some keys have been defined as 'short-cuts' to SAS commands. Just pressing the key combination will execute the command just as if you had typed the command on the command line. You also have the option of using "pulldown menus". These are used just like all of the other menus in the windows system. Position the mouse so that the on-screen cursor is on the name of the menu you want to read and then press and hold the left button. Scroll down the menu until you reach the desired option and then release the mouse button.

| SAS Function | Function Keys | Pulldown Menu Commands |
|--|---------------|----------------------------------|
| Switch between a command line and pulldown menus | Control-A | Globals-> Command-> Command line |
| **Execute | Control-E | Locals-> Submit |
| **Recall program | Control-R | Locals-> Recall text |
| Scroll up a page | Control-V | |
| Scroll back a page | Control-D | |
| Go to Bottom | Control-B | |
| Go to Top | Control-T | |
| **Insert mode | Control-X | |
| Help Facility | Help | |

| | | |
|-------------------------|-----------|--------------------------------|
| Cursor Home | Control-F | |
| Quit SAS | Bye | File-> Exit |
| **Include 'filename' | | File-> Open-> Read file |
| file 'filename' | | File-> Save as-> Write to file |
| Clear | | Edit-> Clear text |
| Recall previous command | ? | |

Control-keyname means that you should hold down the CONTROL key and press the key called keyname.

**Only applicable in the Program Editor Window.

SAS Programming

There are two components to SAS programs: the data step and the proc step. The data step reads data into a SAS dataset, the proc step analyzes the SAS dataset. A note about SAS names:

You will make up your own names for your SAS datasets and variables. These names must conform to these rules:

- no longer than 8 characters,
- start with a letter, and
- contain only letters, numbers, or underscores (_).

SAS is not case-sensitive. You can use capital or lowercase letters in your SAS variables. However, when you specify filenames (as you do with the include and file SAS commands), you must type it exactly as it exists in UNIX.

The DATA step

The data step is used to describe and modify your data. Within the data step you tell SAS how to read the data and generate or delete variables and observations. The data step transforms your raw data into a SAS dataset.

There are four statements that are commonly used in the DATA Step

- DATA statement names the dataset
- INPUT statement lists names of the variables
- CARDS statement indicates that data lines immediately follow.
- INFILE statement indicates that data is in a file and the name of the file.

Generally, the data step portion of your program will either look like this:

```
DATA dataname; INPUT varname1 varname2 (etc); . . . In this part of the program,
you can create . . . new variables, use if statements, do loops, . . . or other data
manipulation statements. CARDS; lines of data . . .
```

or like this:

```
DATA dataname;
INFILE 'filename';
INPUT varname1 varname2 (etc);
```

```
. . . In this part of the program, you can create
. . . new variables, use if statements, do loops,
. . . or other data manipulation statements.
```

Each style uses the DATA and INPUT statements. However, the first style has the data lines inside the program, so it uses the CARDS statement. The second style uses

data from another file: it uses the INFILE statement to let SAS know where to get the file.

The DATA statement

This statement must begin your DATA step. It is used to name your SAS dataset. All data statements must end with a semicolon.

Example:

```
data hwk1;
```

This tells SAS to create a new dataset and call it hwk1. The name you choose is up to you, but it must conform to SAS naming conventions.

The INPUT statement

The INPUT statement comes after the data statement. It designates the names of the variables in your dataset. The variable names must follow the SAS naming rules, and a space separates the variable names in an input list.

Example:

```
input age weight height;
```

This is an example of using free format in naming the input variables. It is also possible to specify the columns that the variables occupy.

Example:

```
input age 1-2 weight 3-5 height 7-8;
```

This statement tells SAS that the value of the variable age is found in the first two columns of each line, weight occupies columns 3-5 and height is in columns 7 and 8. Notice that column 6 is not used in this example.

SAS reads a dataset one line at a time, reading in each value and putting it into the next variable in the input statement list. When it has filled out the list, SAS moves on to the next line of data. However, sometimes you may want to put several observations for the variables on each line. This is illustrated in the next example.

The dollar sign (\$) after a variable name tells SAS that the variable has character values (not numbers). If your data contains character variables, you must let SAS know by following the variable name in the INPUT statement with the dollar sign:

Example:

```
input age sex $ salary @@;
```

A line of data might look like this:

```
32 m 150 20 f 108 22 m 200
```

The double 'at' character (@@) is used in an input statement when information for more than one observation will be located on each line.

The CARDS statement

The CARDS statement tells SAS that the data immediately follows on the next line. The keyword is named CARDS because years ago data fed into a computer came on

real cards with holes punched to represent different characters or numbers.

Use this style of input when you want to enter the data directly into your program (i.e., not when reading in an external file using the INFILE statement).

Example:

```
cards;
27 118 63 24 170 70 25 173 73 23 183 68 19 203 78
```

The INFILE statement

The INFILE statement precedes the INPUT statement in the data step. It tells SAS two things: first, that the data will be coming from an external file, and second, the name of that file.

Example:

```
infile 'rebound.dat';
```

Note that the filename must be enclosed in single quotes and must be spelled exactly as it exists in the UNIX system (i.e., capitalization matters).

EXAMPLES:

With data included with the program statements:

```
data hwk7;
input age sex $ weight salary;
cards;
32 m 150 20
27 f 108 22
45 m 200 48
... more data lines ...
```

With data from another (external) file:

```
data hwk12;
infile 'age.data';
input age sex $ weight salary;
```

The PROC step

The proc steps tell SAS what analyses need to be performed on the data, such as regression, analysis of variance, computation of means, etc.

All of the proc steps in SAS begin with a line containing a PROC statement.

A PROC statement will generally have the following form:

```
PROC procedurename DATA = datasetname options;
```

where procedurename is the name of the procedure to be used, datasetname is the name of the dataset on which analyses are going to be performed, and options are keywords which modify the procedure.

Example:

```
PROC MEANS DATA = pitcher MEAN VAR;
```

```
VAR weight height;
```

This proc step will compute the mean and variance for the two numeric variables weight and height in the dataset pitcher.

Editing in the Program window

Unlike UNIX, SAS is not case-sensitive. You can use capital or lowercase letters in your program statements or SAS commands. However, when you specify filenames (as you do with the include and file SAS commands), you must type it exactly as it exists in UNIX. This is because SAS needs to know the exact name in order to retrieve the file from UNIX.

Moving the cursor:

You can move the cursor around on the screen by using the arrow keys or by using the mouse. The arrow keys on the SUN workstations are on the number pad. To use the mouse slide it over the mouse pad until the on-screen cursor (the large arrow on the screen) is where you want to be (in one of the SAS windows) and press the left mouse button. The SAS cursor (a small rectangular box) will move to where the on-screen cursor is positioned when you clicked the button.

Inserting Text:

When you start SAS, you are in typeover mode. This means that if you type characters in the middle of a word, the characters will type over the word. You may switch to insert mode, so characters are inserted into the word instead. To do this press the 'INS' key on the number pad.

This will switch back and forth between insert mode and typeover mode. In insert mode, the cursor is an underline, in typeover mode it is a block cursor.

Deleting Characters:

To delete a character, move the cursor to the character just after the one you want to delete and press the delete key.

Line Number Area Commands:

The area in the Program window which shows line numbers may be used in editing to manipulate the lines of your program.

To execute line commands, move the SAS cursor into the line number area (right on top of the numbers) and type the line command over the line numbers. Use only one line command at a time. Press the RETURN key to execute the command.

Commonly Used Line Number Area Commands

```
d deletes the line
i or in inserts a line or inserts n lines after the current line
m . . . . . a
moves line marked with m
after line marked with a.
c . . . . . a
copies line marked with c
after line marked with b.
```

Example:

Before the command:

```
0024    I want this line to stay where it is
0025    Some SAS statement line
0026    This line to move after line 24
```

Issue the command:

```
0a24      I want this line to stay where it is
0025      Some SAS statement line
0m26      This line to move after line 24
```

The 'm' marks the line which should be moved.
The 'a' says to move the line marked 'm' after this line.
After the command:

```
0024      I want this line to stay where it is
0025      This line to move after line 24
0026      Some SAS statement line
```

Retrieving Programs

You can retrieve a file, bringing it into the Program window, by typing
include 'filename'
on the Program window command line, where filename is the name of the file you want
to bring into your SAS session. If you are using pulldown menus, chose "Open ->
Read file" from the "File" menu. This may be a file you wrote yourself (and saved in an
earlier SAS session with the file command) or a file your instructor provided.

If you cannot remember what you named a program in a previous session, use the
UNIX command (in the xterm window):

```
ls
to list the names of the files in your directory.
```

To list the contents of a text file type:

```
more filename
```

(where filename is the name of the file). In viewing a file this way, you can press the
spacebar to see the next screen, or type q to quit and return to the system prompt.

To remove (or erase) a file type:

```
rm filename
```

(where filename is the name of the file). Note that there is no unerase command so be
careful removing files.

Executing Your Program

Control-E or the submit command?

Either method will execute the program statements in your Program window.
However, it is better to use the Control-E key sequence (by holding down the Control
key and pressing the letter E). This sequence first checks to make sure your Log and
Output windows are empty before executing your program.

If you use the submit command instead (either by typing "submit" on the command line
or by choosing "submit" from the "Locals" menu), you must remember to clear the
windows yourself (by giving the clear command on the command line of each window
or choosing "Clear text" from the "Edit" menu in each window) each time before you
execute a program.

Why you should clear your windows

Unlike the Program window, which clears itself each time you execute a program, both
the Output and Log windows accumulate information. Consequently, they can grow
very large. It is confusing to look at log and output which resulted from several
programs.

Besides the possibility of becoming confused when looking at old log notes or output, you might print an output window that has a lot of old output. This wastes a lot of paper.

By clearing your windows before executing a program, you will have only the fresh log-notes and output that pertain to the program you just executed. There is an easy way to do it: the Control-E key sequence.

When you execute your program

When you submit your program for execution, the program statements will disappear. Don't let this worry you: Your statements in the Program window can be recalled with the Control-R key sequence. If you use the sequence twice, the previous two program window contents will be recalled, and so forth.

While your program is executing, the cursor also disappears. When it reappears, execution is finished.

The next step is to look at the contents of the Log window. Position the on-screen cursor in the log window with the mouse and press the left button to make the log window current. Then either hit Control-t or use the mouse (by placing it on the arrow at the top of the scroll bar and pressing the left mouse button) to go to the top of the Log window.

Spend a little time in this window and you will see how useful it really is. Error messages, warnings and other bits of information found here will help you correct problems in your program.

Saving Your Work

You can save the contents of any of the three SAS windows into a file by typing: file 'filename' on the command line, where filename is the name you want the file to have. Don't forget the single quotes surrounding the filename. If you are using pulldown menus choose "Save as -> Write to File" from the "File" pulldown menu.

You should figure out some naming scheme for the files you save. One method that has worked well in the past is to give each file a name indicating which homework assignment the file is for, as well as whether the file contains output or programming statements.

For example, for your first homework assignment, you would save your program as hw1.pgm, and your output file as hw1.out. For the second assignment, hw2.pgm and hw2.out. Using this system, you will always know what is in a file just by looking at its name.

Printing

The printer is located in the corner of the lab. Every user will be assigned a print quota of fifty pages unless otherwise instructed by the professor. Students may add to their quota by going to WolfCopy.

Printing Commands (for UNIX: done in a xterm window):

- lpquota : gives you a summary of your printing charges. The option -l added to this command will give you a complete account of your printing.
- lpq : lists the files (and their corresponding job numbers) waiting to be printed.
- lprm jobnumber: removes the corresponding jobnumber from the printer. The jobnumber is found using the lpq command.
- lprm : removes all of your printer jobs from the queue.

You can print your SAS output in two ways:

1. During a SAS session (recommended): On the command line of the window you want to be printed, simply type the command print and press the return key.

If you are using pulldown menus, choose "print" from the "File" menu.

Printing will print the entire contents of the window, not just the part of the window you see on the screen.

2. After a SAS session, from the xterm window at the UNIX prompt use the UNIX command:
sprint filename
where filename is the name of the file. This assumes that you have saved the contents of a window using the SAS file command.

Programming Tips

Like other programming environments, SAS has rules of syntax to follow. Here are a few guidelines:

All SAS program statements must end with a semicolon ";"

Most SAS statements will begin with a SAS keyword (i.e. DATA, PROC, BY).

When reading data into a SAS dataset, SAS will recognize a period "." by itself as representing a missing value.

Comments are used to make your program easier to read and edit. SAS does not execute comment statements; in other words, they exist for the programmer only. They may also be used to "turn off" sections of your program. When SAS sees a comment in a program, it ignores it. All comments must be terminated.

There are two ways to write comments into your program:

1. Put an asterisk "*" at the beginning of a line, write the comment, and terminate the comment with a semicolon:
*This is a one-line comment;
2. Start a block of comment lines with /*. Write as many comment lines as you want. Terminate the comment with */.
/* This is
a three line
comment */

When you create names for the variables and SAS datasets to be used in your programs, you must follow the SAS naming rules:

No name can be more than eight characters long.

All names must begin with either a letter or underscore.

The only characters that can be used in names are letters, numbers, and underscores.

You must always end each SAS program with the RUN; statement.

SAS Error Messages

sas: command not found

Either you forgot to add sas before trying to start the package or perhaps the system doesn't know what terminal type you are using. You must log out and log back in. When you log in this time, be sure you specify the correct terminal type at the question:

What DISPLAY are you using? [| none] ?

This should only happen when connecting to SICL from the outside. If that's the case, try using vt100 as the DISPLAY type.

If you see the prompt:

1?

2?

.

.
.

You have started SAS in interactive mode. Type "endsas;" and hit return. (Don't forget the ";".) This will get you back to unix. Log out and then start again.

statement not valid or used out of proper order
SAS error usually caused by a missing semicolon or the misspelling of a SAS keyword.

unable to open file or
cannot find dataset

Usually caused by misspelling a file name, not enclosing a file name in single quotes, or referencing a nonexistent file.

syntax error: expecting one of the following . . .

A SAS procedure option is misspelled or invalid, or may be caused by a missing semicolon.

Title has become more than 200 characters long

If you see this message from SAS, or if you notice that your Log window has no NOTES or WARNINGS, but only the exact copy of your program statements, you have unmatched quotes, or an unterminated comment.

Look for a single quote (') that has no match or a comment that is not terminated. SAS thinks you are trying to create one BIG title. It finally decides enough is enough when the alleged title becomes over 200 characters long.

To take care of the problem, the best thing is to save your file (with the file 'filename' command), and quit SAS (with the bye command). Restart SAS, retrieve the file again (include 'filename'), and delete the quote or terminate the comment that caused the problem.

file space full or
unable to save file

You have used up all of your allocated space in your account. If you get this message from SAS, the file is actually created, but it will be empty. Remove it as well as old files you don't need to get more space.

invalid SAS name

A dataset or variable name is longer than 8 characters, starts with a number, or contains characters other than letters, numbers, and underscores.

Other SAS problems

dataset is created, but procedures don't execute

Usually caused by not putting the necessary run; statement as the last line of the program.

titles come out in the wrong places

When you create a title, SAS will put that title on each procedure until you make a new title. To get rid of a title, write an empty title statement:

TITLE;

Printed output broken across pages

Caused by using the wrong page size. You must not change the SAS pagesize to be greater than 56 lines. Plots may be increased to take up one sheet of paper with the program statement:
options ps=55;

SAS Tutorial

If you are new to SAS, you may want to practice writing some simple SAS programs. This tutorial is designed to help you take the first steps toward learning SAS.

You will generally follow these steps:

- log into your SICL account
- type add sas and hit the return key
- type add classname_info and hit the return key. (where classname is the class you are taking)
- type the word sas and press the RETURN key.
- do the work you need to do in SAS: programming, saving files, printing files.
- on any window command line, type BYE to end your SAS session or choose "exit" from the "file" menu if you are using pulldown menus.

logout by typing logout in the xterm window or by selecting the logout option in the main system menu (press the right mouse button in the grey background to see the menu.)

When you start SAS, you are in 'menu mode'. This tutorial is written as if you are in 'command-line mode'. To switch from having the pull-down menus at the top of the window to having a command line instead, use the CONTROL-A key sequence. Hold down the control key and press the letter a. See the command line come up?

Let's write a typical data step for a SAS program. Type the program below.

```
DATA ht_wt;
INPUT name $ sex $ age height weight;
wtkilo=weight*.45;
CARDS;
ALFRED      M      14      69      112
BARBARA     F      13      62      102
JAMES       M      12      57      083
JANE        F      12      59      084
JOHN        M      12      59      099
JUDY        F      14      64      090
LOUISE      F      12      56      077
MARY        F      15      66      112
RONALD      M      15      67      133
WILLIAM     M      15      66      112
PROC PRINT;
RUN;
```

The DATA statement creates the SAS dataset ht_wt. We could use any name here as long as it is no longer than 8 characters and begins with a letter.

The INPUT statement creates variable names for each column of data in the dataset. The \$ following the variables name and sex is necessary to read in their values: it tells SAS that the variable contains character data (not numeric). Notice that every statement in the data step is followed by a semicolon.

The statement

```
wtkilo = weight*.45;
```

creates a new variable called wtkilo for the dataset by transforming each individual's weight from pounds to kilograms.

The CARDS statement tells SAS that you are finished with the instructions on how to create the dataset ht_wk and the very next line will be the start of the data itself. The keyword is named CARDS because years ago, data fed into a computer came on real cards with holes punched to represent different characters or numbers.

The data must correspond exactly with the ordering of the INPUT statement. SAS knows lines in your program are commands and not data by whether the line contains a semicolon: Your data lines should not have any semicolons.

The PROC PRINT statement marks the beginning of the PROC step. This particular procedure will print out the contents of the dataset in a nice format, with each variable having the name it was given in the INPUT statement.

The RUN statement is included as the last line of the program to tell SAS to execute every statement in your program. You must always end each SAS program with the RUN; statement.

To execute the program use the Control-E key sequence, or type the SAS command "submit" on the command line and press the RETURN key. (If you are using pulldown

windows you may select "Submit" from the "Locals" menu.) There is an advantage in using the key sequence instead of the SAS command: When you use Control-E, your Log and Output windows are automatically cleared for you, so you don't have to remember to do it yourself before executing the program. You use a Control sequence by holding down the Control key and pressing a letter key.

You will see your cursor jump down to the bottom right corner of the screen-- when it returns to the command line, execution is finished. This will happen very fast--you may not even notice it.

Don't worry when you see your program vanish. SAS automatically clears this window when you execute a program. You will be able to retrieve the program.

Go to the Log window by placing the on-screen cursor in the Log window and clicking on the left mouse button. The Log window provides an explanation of what happened when SAS executed your program. It will give error messages or warnings for any mistakes which might be in the program.

If you have error messages shown in the Log window, you probably made a typing mistake. The error messages will show you the problem.

If you have errors, you can move back and forth between the Log and Program windows, noting the errors in the Log window and fixing them in the Program window.

Go to the Program window by placing the on-screen cursor in the Program window and clicking on the left mouse button.

Recall your program (remember SAS clears this window when you execute a program) with Control-R or type recall on the command line and press RETURN. If you are using pull down windows choose the "recall text" option in the "locals" menu.

Usually, the first error is the most important one. Sometimes SAS sees an error and decides not to execute any more of the program, but just checks for more errors. If that happens, you may see error messages around some of your program statements which really have no errors. The problems will usually clear up after you fix the first error.

Use the Control-E key sequence to execute your program.

Once the program executes with no errors, go to the Output window by placing the on-screen cursor in the Output window and clicking on the left mouse button.

The output window will contain the output your program requested. The results of the PROC PRINT statement were sent to the OUTPUT Window.

Return to the PROGRAM EDITOR window.

You can do further analyses on the dataset even though the original DATA step and data are no longer visible: they are still in the SAS's memory.

Let's do a few more procedures before we quit.

The PROC MEANS statement is often used to obtain descriptive statistics for your data. The VAR sub-statement specifies which variables you want analyzed.

Let's subdivide the dataset into classes (by sex) to compare the descriptive statistics between groups. To do this, we must sort the dataset by sex first:

To split a dataset by a variable, you have to sort it by the variable first.

Type in these statements to sort the data and calculate means for the two groups. You don't need to recall your previous statements: SAS remembers them.

```
PROC SORT;
```

```

        BY sex;
PROC MEANS;
        VAR height weight wtkilo;
        BY sex;
TITLE 'Calculation of Means for Males and Females';
RUN;

```

The SORT procedure sorts the dataset by the values for sex.

The MEANS statement tells SAS that you want means calculated for some of the variables.

You name the variables for which you want means in the VAR statement. Since you can only calculate a mean of a numeric variable, the VAR statement can contain only numeric variables.

The BY statement in the means procedure tells SAS to split analyses by the different values of the variable indicated in the BY statement (in this case, sex).

Make sure that variables in the BY statement of an any PROC classify the data into categories or groups. This is because the BY statement tells the procedure to run the analysis for each value of the BY variable. If this variable has fifty different values (as it might if it was a continuous variable, like height or weight for example), fifty different analyses will be run.

Execute the code with Control-E and check the Log window for errors or warnings. If all is well, go to the top of the Output window and look at your output.

The average height and weight for males is greater than the average weight and height for females. However, sometimes one number can be misleading in comparing the relationship between two groups.

Let's look at a comparison between the two groups by plotting height and weight by sex. This can be done by using the PROC PLOT statement.

Return to the Program window.

The following statements will produce a scatterplot with height on the vertical axis, weight on the horizontal axis, and the value of sex will be the plotting symbol:

```

PROC PLOT;
        PLOT height*weight=sex;
        TITLE 'Plot of Height vs. Weight by Sex';
RUN;

```

There is no need to recall the previous program statements: SAS still remembers the dataset ht_wt, so just type in these statements and execute. Check the Log window for errors and if there are none, go to the top of the Output window to see the plot.

Now go back to the Program window and type the following program statements to find out more about the relationship between height and weight. The output from these statements will compute the correlation between the two variables, as well as produce a simple statistical summary of the data:

```

PROC CORR;
        VAR weight height;
        TITLE 'Correlation for Height and Weight';
RUN;

```

To leave SAS, give the command

bye

on any SAS command line. If you are using pulldown menus, choose the "exit" option

from the "file" menu.

The SAS windows should disappear and you should now see the UNIX ready prompt (%) in the xterm window. Go to the background window of the screen and hold down the right mouse button. The system menu will appear. Drag down to the logout option on the menu and release the mouse button. After a few seconds, the login window should reappear.