Multivariate Analysis Using SAS

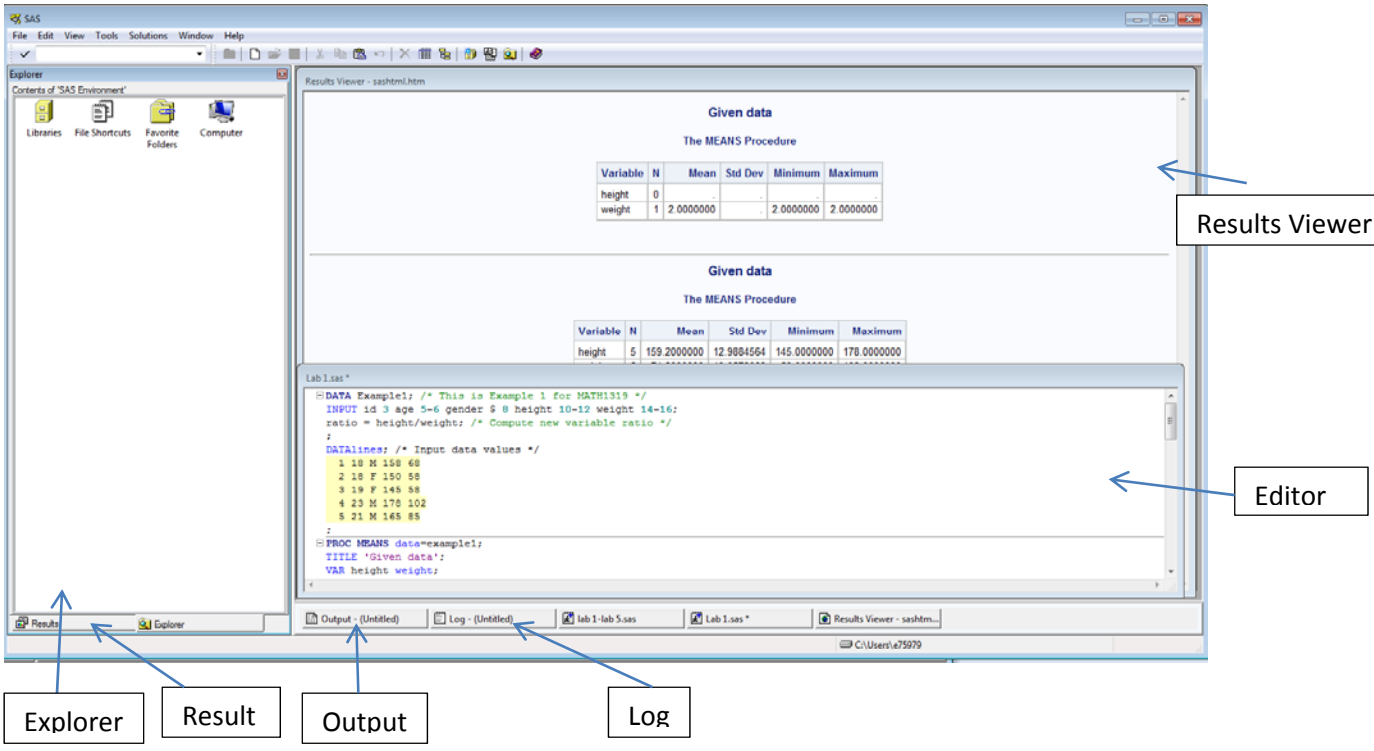
A certain degree of familiarity with SAS is the prerequisite for this course. Here is just a quick introduction of getting started with SAS if you have not contacted SAS before.

SAS programs A SAS program is a sequence of statements executed in order. SAS programs are constructed from two basic building blocks: DATA steps and PROC steps. A typical program starts with a DATA step to create a SAS data set and then passes to the data to a PROC step for processing. DATA steps read and modify data while PROC steps analyse data, preform utility functions, or print reports. A step ends when SAS encounters a new step (marked by a DATA or PROC statement), a RUN statement, or the end of the program.

SAS statements **EVERY SAS statement ends with a semicolon.**

Comments There are two styles of comments you can use: one starts with an asterisk (*) and ends with a semicolon; The other starts with a slash asterisk (/*) and ends with an asterisk slash (*/).

Base SAS Windows



Editor This window is a text editor. You can use it to type in, edit, and submit SAS programs as well as edit other text files such as raw data files. In Windows operating environment, the default editor is the Enhanced Editor.

Log The log window contains notes about your SAS session, and after you submit a SAS program, any nots, errors, or warnings associated with your program as well as the program statements themselves will appear in the Log window.

Output If your program generates any printable results, then they will spear in the Output window.

Results The results window is like a tale of contents for your Output window; the results tree lists each part of your results in an outline form.

Explorer The explorer window gives you easy access to your SAS files and libraries.

If you want to find more information about the basics of SAS, you can refer to the book: ***The Little SAS Book by L.D. Delwiche and S. J. Slaughter (2008)***. Or you can find plenty of support online at SAS.com

Submitting a program in the SAS Windowing Environment

Getting your program into the editor The first thing you need to do is get your program into the Editor window. You can either type your program into the editor, or your can bring the program into the Editor window from a file (select Open from the File pull-down menu).

Submitting your program Once your program appears in the editor, you execute it using the SUBMIT command.

- Use the Submit icon on the toolbar. Or make the editor window active and select Submit from the RUN pull-down menu.

Viewing the SAS Log and Output After you submit your program, the program remains in the Enhanced Editor window and the results for your program go into eh Log and Output windows. If your program produced any output, then you will also get new entries in the Results window.

Reading the SAS Log Many programmers ignore the SAS log and go straight to the output. That is understandable, but dangerous. It runs doesn't mean it's right!

Getting your data into SAS

If you type raw data directly in SAS program, then the data are internal to your program. You may want to do this when you have small amounts of data, or when you are testing a program with a small test data set. Use the DATALINES statement to indicate internal data. All lines in the SAS program following the DATALINES statement are considered data until SAS encounters a semicolon.

Usually you will want to keep data in external files. Use the INFILE statement to tell SAS the filename and path, if appropriate, of the external file containing the data. The INFILE

statement follows the DATA statement and must precede the INPUT statement. After the INFILE keyword, the file path and name are enclosed in quotation marks, for example: INFILE 'c:\mydir\example.dat';.

Reading raw data separated by spaces The INPUT statement, which is part of the DATA step, tell SAS how to read your raw data. To write an INPUT statement using list input, simple list the variable names after the INPUT keyword in the order they appear in the data file. If the values are character (not numeric), then place a dollar sign ($) after the variable name. Leave at least one space between names, and remember to place a semicolon at the end of the statement. Example:

INPUT Name $ Age Height;

Reading raw data arranged in columns After the name of each variable (if the variable is character, leave a $), list the column or range of columns for that variable. The columns are positions of the characters or number in the data line. Example

INPUT Name $ 1-10 Age 11-13 Height 14-18;

Reading raw data not in standard format There are three general types of informats: character, numeric, and date. The three have the following general forms:

Character	**Numeric**	**Date**
$informat*w.*	informat*w.d*	informat*w.*

INFORMAT is the name of the informat, w is the total width, and d is the number of decimal places (numeric informats only).

Example:
INPUT Name $10. Age 3. Height 5.1 BirthdDate MMDDYY10.;

The data value for the 1st variable, Name, which has an informat of $10., are in columns 1 through to 10. Now the starting point for the second variable is column 11, and SAS reads variable Age in columns 11 through 13. The values for the 3rd variable, Height, are in columns 14 through 18. The five columns include the decimal place and the decimal point itself (150.3 for example). The values for the last variable, BirthDate, start in column 19 and are in a date form.

Reading delimited files with the DATA step Delimited files are raw data files that have a special character separating data values. Many programs can save data as delimited files, often with commas or tab characters for delimiters. SAS gives you the options for the INFILE statement that make it easy to read delimited data files: the DLM=option option. Example with comma-delimited file:

```
Data Reading;
        INFILE 'c:\mydata\books.dat' DLM=',' ;
        INPUT Name $ Week1 Week2 Week3 Week 4;
        Run;
```

If the data had tab character between values instead of commas, then you could use the DLM='09'X option.