INTRODUCTION TO SAS MACRO LANGUAGE

* The **SAS macro language** is generally used to shorten SAS codes.
* Macro language is composed of **macro variables, macro functions** and **macro programs**.
* Macro variables are either **user-defined** or **system-defined** (or **automatic**).
* User-defined macro variables are either **global** or **local**. A **global variable** can be used in any SAS procedure or data step, whereas a **local variable** can be used only inside the macro program in which it was defined.

USER-DEFINED MACRO VARIABLES

* A user-defined macro variable is a string variable that substitutes some text in a SAS code.
* The macro variable can be created by using the **%let** statement.
* When it is referenced, it is preceded by the ampersand sign **&**.
* When the code is submitted, SAS processes macro variables first, substituting them with the text string they were defined to be and then processes the code as a standard SAS code.

Example. The data grades contain scores on several assignments. Below is a regular SAS code that computes means of these scores.

data grades;

input id$ quiz1 hw1 exam1 quiz2 hw2 exam2;

cards;

0495 94 97 95 94 100 97

8612 89 92 82 97 100 88

6236 94 89 85 91 94 86

;

proc means data=grades;

var quiz1 hw1 exam1 quiz2 hw2 exam2;

run;

The output is

The MEANS Procedure

Variable N Mean Std Dev Minimum Maximum

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ

quiz1 3 92.3333333 2.8867513 89.0000000 94.0000000

hw1 3 92.6666667 4.0414519 89.0000000 97.0000000

exam1 3 87.3333333 6.8068593 82.0000000 95.0000000

quiz2 3 94.0000000 3.0000000 91.0000000 97.0000000

hw2 3 98.0000000 3.4641016 94.0000000 100.0000000

exam2 3 90.3333333 5.8594653 86.0000000 97.0000000

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ

The code can be simplified by creating a macro variable that consists of the variable names listed in the var statement.

data grades;

input id$ quiz1 hw1 exam1 quiz2 hw2 exam2;

cards;

0495 94 97 95 94 100 97

8612 89 92 82 97 100 88

6236 94 89 85 91 94 86

;

%let varlist= quiz1 hw1 exam1 quiz2 hw2 exam2;

proc means data=grades;

var &varlist;

run;

* To display macro variable value as text in the log window, the **%put** statement is used.

Example. In the previous example, the line of code

%put &varlist;

produces the following lines in the log window:

%put &varlist;

quiz1 hw1 exam1 quiz2 hw2 exam2

* To display the values of all user-defined macro variables in the log window, the statement %put \_user\_; is used. The output from the log window is

%put \_user\_;

GLOBAL VARLIST quiz1 hw1 exam1 quiz2 hw2 exam2

* The first column indicates the type of the macro variable.
* The second column indicates the name of the macro variable
* The third column contains the value of the macro variable

SYSTEM-DEFINED MACRO VARIABLES

* System-defined macro variables are created automatically when SAS is started.
* System-defined macro variables are sometimes called automatic macro variables.
* There are many system-defined macro variables.
* System-defined macro variables can be used in the same way as user-defined ones.
* To display the values of all system-defined macro variables, %put \_automatic\_; statement is used. A partial output from the log window is

%put \_automatic\_;

AUTOMATIC AFDSID 0

AUTOMATIC AFDSNAME

AUTOMATIC AFLIB

AUTOMATIC AFSTR1

AUTOMATIC AFSTR2

AUTOMATIC FSPBDV

AUTOMATIC SYSBUFFR

AUTOMATIC SYSCC 3000

AUTOMATIC SYSCHARWIDTH 1

AUTOMATIC SYSCMD

AUTOMATIC SYSDATE 04DEC10

AUTOMATIC SYSDATE9 04DEC2010

AUTOMATIC SYSDAY Saturday

* The first column indicates the type of the macro variable
* The second column indicates the name of the macro variable
* The third column contains the value of the macro variable

Example. The code below uses in the title statement two system-defined macro variables SYSDATE (today’s date) and SYSDAY (day of the week).

title "Today is &sysday, &sysdate";

options nodate nonumber;

proc means data=grades;

var &varlist;

run;

The title of the output is

Today is Monday, 14NOV11

* Note that **double quotation marks** must be used when macro variables are embedded in the title statement. Using just single quotation marks

title 'Today is &sysday, &sysdate';

produces the title

Today is &sysday, &sysdate

MACRO FUNCTIONS

There are many macro functions that are related to macro variables. The most useful ones are **%scan**, **%eval**, and **%sysevalf**.

* The function **%scan** takes a string and an integer *n* as arguments and returns the *n*th word in the string.

Example. In our example, the code below identifies the second word in the string defined by the macro variable &varlist.

%let word2=%scan(&varlist,2);

%put &word2;

The output printed in the log window is

%let word2=%scan(&varlist,2);

%put &word2;

hw1

* The function **%eval** evaluates expressions using arithmetic of integers.

Example. The code below computes 2+2.

%let i=2;

%let count=%eval(&i+2);

%put &count;

The log window output is

%let i=2;

%let count=%eval(&i+2);

%put &count;

4

* Note that if %eval is omitted in this example, then the result is computed symbolically, not numerically.

%let i=2;

%let count=&i+2;

%put &count;

2+2

* The function **%sysevalf** evaluates expressions with real numbers, not necessarily integers.

Example. The code below computes 2.17+3.14.

%let e=2.17;

%let sum=%sysevalf(&e+3.14);

%put &sum;

The log window output is

%let e=2.17;

%let sum=%sysevalf(&e+3.14);

%put &sum;

5.31

* Note that if %sysevalf is omitted in this example, then the result is computed symbolically, not numerically.

%let e=2.17;

%let sum=&e+3.14;

%put &sum;

2.17+3.14

MACRO PROGRAMS

* A macro program is a stored collection of macro language statements.
* Macro programs always start with the **%macro** statement. The syntax is

%macro *macroname*;

* Macro programs always end with the **%mend** statement. The syntax is

%mend;

* To call a macro program, use syntax

%*macroname*

Note that this statement doesn’t end with a semicolon.

Example. The code below prints the grades data set.

%let dataset=grades;

%macro printmacro;

title "Data set &dataset";

options nodate nonumber;

proc print data=&dataset;

run;

%mend;

%printmacro

* When a macro program takes arguments, the names of the arguments are listed in parentheses after the name of the program in the %macro statement.

Example. The code below prints the grades data set. The data set name is now an argument in the printmacro macro program.

%macro printmacro (dataset);

title "Data set &dataset";

options nodate nonumber;

proc print data=&dataset;

run;

%mend;

%printmacro(grades)