


Learning outcomes:

1. Foundation SAS
2. SAS windowing environment.
3. Editing a SAS program
4. Submitting a SAS program
5. Getting program results
6. Structure and components of SAS programs
7. Working with SAS Syntax


SAS components learnt:

1. SAS Editor window
2. SAS Log window
3. SAS Output window
4. DATA step
5. PROC step
6. PROC SETINIT

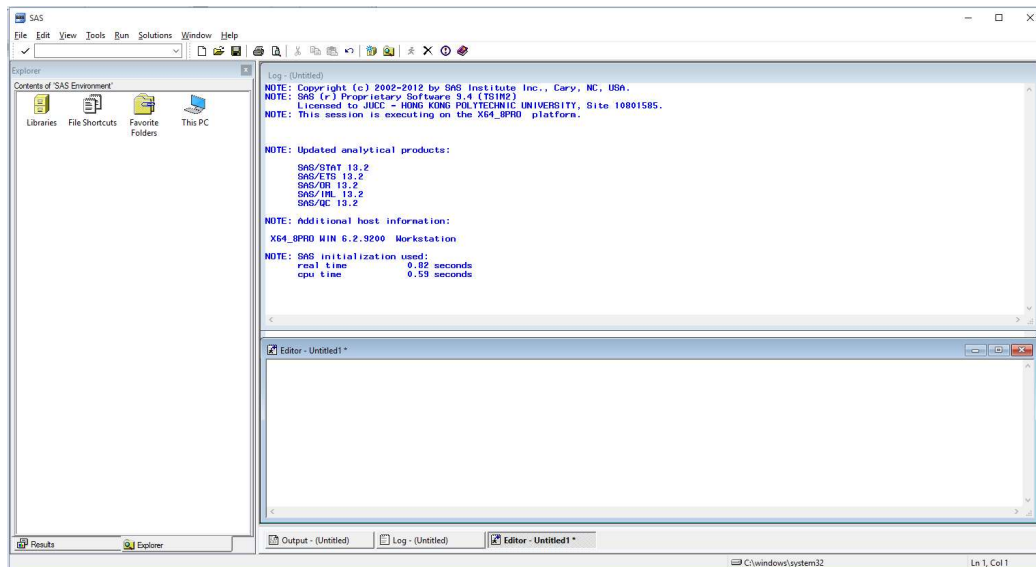
1.1 Foundation SAS




 Foundation SAS is a highly flexible and integrated software environment that can be used in virtually any setting to access, manipulate, manage, store, analyze and report on data.

1.2 SAS Windowing Environment

 Getting Started

➤ Click “SAS”  Run command in the window start menu.



- The SAS windowing environment is used to write, include, and submit SAS programs and examine SAS results.
- Contains “Editor”, “Log” and “Output” windows:
 - (1) Editor window : create SAS programs/ contains SAS program for editing and running (submitting)
 - (2) Log window : contains information about the processing of the SAS program, including any warning and error messages.
 - (3) Output window : contains reports generated by the SAS program.
- SAS can be run in **batch** mode, **non-interactive** mode and **interactive window** mode.
- The batch mode and non-interactive mode are outside SAS windowing environment.

- In batch mode, some operating system control statements are added before the SAS statements.
- In non-interactive mode, SAS are stored in an external file and are executed immediately after the following statement:
 - “SAS *filename*” in directory based system;
 - “SAS INPUT (*filename*)” in z/OS (OS/390) system.

1.3 Running a simple SAS program in SAS windowing environment

- Submitting a SAS program in SAS windowing environment:
 - Open the SAS program, ‘p1_electricity.sas’ in d:\sas_programs.

Method 1	Select File > Open Program > select <u>p1_electricity.sas</u>
Method 2	Type Include ‘d:\sas_programs\p1_electricity.sas’ in the white box adjacent to the tick icon:

- Look at the Editor Window:

```
/*p1_electricity.sas*/
libname DATA1 "D:\SAS_Datasets";

data testing;
set DATA1.electricity;
where average_temperature >50;
run;

Proc Print data=testing;
run;
```

➤ Click  or select **Run** > **Submit**.

- Look at the Log Window:
How many records does the data set “testing” have? 11
Is there any errors detected? No

Line number Partial Log Window output:

```
20 libname DATA1 "D:\SAS_Datasets";
NOTE: Libref DATA1 was successfully assigned as follows:
Engine: V9
Physical Name: D:\SAS_Datasets
21 data testing;
22 set DATA1.electricity;
23 where average_temperature >50;
24 run;
NOTE: There were 11 observations read from the data set DATA1.ELECTRICITY.
WHERE average_temperature>50;
NOTE: The data set WORK.TESTING has 11 observations and 5 variables.
NOTE: DATA statement used (Total process time):
real time 0.02 seconds
cpu time 0.00 seconds
```


- Look at the Output Window:

SAS Base Programming Course
Chapter 1 Introduction, Getting Started to SAS

In Year 1: May to June, how much was the electricity cost? 213.43 dollars
What was the total heating degree days? 479
What was the average daily Temperature? 57°F (The unit is obtained from the data source on paper).

Output window output:

The SAS System				14:32 Friday, May 25, 2012 1	
Obs	Time_Period	Electricity_ Consumed	Cost	Heating_ Degeree_ Days	Average_ Temperature
1	Year 1: May/June	2359	213.43	479	57
2	Year 1: July/Aug	3704	338.16	19	74
3	Year 1: Sept/Oct	3432	299.76	184	66
4	Year 2: May/June	2762	255.85	657	54
5	Year 2: July/Aug	2288	219.72	35	68
6	Year 2: Sept/Oct	2423	256.59	308	62
7	Year 3: May/June	2073	205.16	438	58
8	Year 3: July/Aug	2579	251.07	15	72
9	Year 3: Sept/Oct	2858	279.80	152	67
10	Year 4: May/June	2266	213.09	537	66
11	Year 4: July/Aug	3128	333.49	26	71

 **Ex. 1.1:** Modify the program so that only the data with average temperature ≥ 70 are listed in the output window and save the file as plex2.sas. Write down your output in the box below.

What update is required? Only where statement is required to be updated and the updated version:

```
/*plex2.sas*/  
libname DATA1 "D:\SAS_Datasets";  
  
data testing;  
set DATA1.electricity;  
  
run;  
  
Proc Print data=testing;  
run;
```

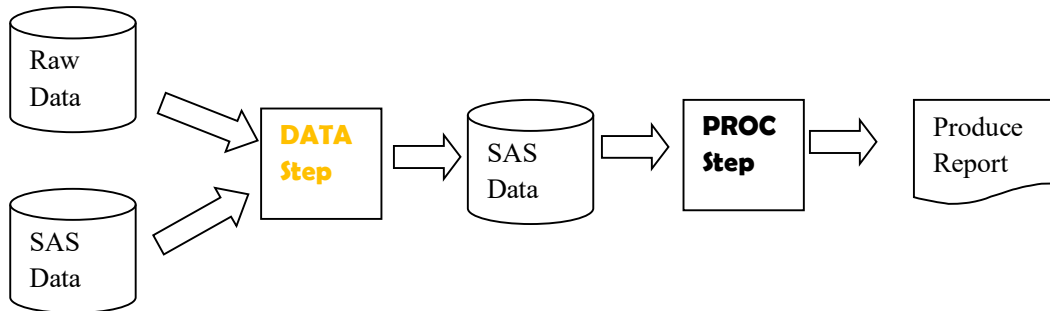
The SAS System				11:59 Monday, June 4, 2012	
1					
Obs	Time_Period	Electricity_ Consumed	Cost	Heating_ Degeree_ Days	Average_ Temperature
1	Year 1: July/Aug	3704	338.16	19	74
2	Year 3: July/Aug	2579	251.07	15	72
3	Year 4: July/Aug	3128	333.49	26	71

1.4 The structure, characteristics and components of SAS program

- Each SAS program consists of any combination of these two components: DATA step and PROC step.
- PROC step can stand alone as a SAS program provided that a valid SAS data is used in PROC step.
- SAS detects the end of a step when it encounters one of the following:
 - a RUN statement (for most steps)
 - a QUIT statement (for some procedures)
 - the beginning of another step (DATA statement or PROC statement)



- Its structure



- DATA step:

- convert raw data into SAS data sets
- compute values
- check for and correct errors in data
- produce new SAS data sets by subsetting, supersetting, merging, and updating existing data sets
- its characteristics:

Statements	Sample Program Code
a DATA statement	data testing;
a SET statement	set DATA1.electricity;
Additional programming statements	where average_temperature>50;
a RUN statement	run;

- PROC (Procedure) steps:

- invoke or call pre-written routines that enable you to analyze and process the data in a SAS data set, for example, PROC REG is used for regression analysis; PROC GLM is used for ANOVA.
- Sometimes create new SAS data sets that contain the results of the procedure (PROC SORT).
- list, sort, and summarize data (PROC TABULATE, PROC SORT, PROC SUMMARY)
- create a report that lists the data (PROC PRINT)
- produce descriptive statistics (PROC FREQ)
- create a summary report (PROC SUMMARY)

- produce plots and charts (PROC GPLOT)
- its characteristics: a PROC statement, optional additional statement(s) and RUN statement. Here is an example of PROC PRINT:

Statements	Sample Program Code
a PROC PRINT statement	proc print data=testing;
another RUN statement	run;

- In the program:

```

/*p1_electricity.sas*/
* a program to list electricity data ;
libname DATA1 "D:\SAS_Datasets";

data testing;
set DATA1.electricity;
where average_temperature >50;
run;

Proc Print data=testing;
run;

```

Diagram labels and arrows:

- Comments:** Points to the first two lines of the program.
- LIBNAME statement:** Points to the `libname DATA1 "D:\SAS_Datasets";` line.
- DATA Step:** Points to the `data testing;`, `set DATA1.electricity;`, `where average_temperature >50;`, and `run;` lines.
- PROC Step:** Points to the `Proc Print data=testing;` and `run;` lines.

- The characteristics of SAS statements
 - Which line is the remark or comment of the program? The first two lines.
What are the characteristic? The first line starts with “/*” and ends with “*/”, while the second line starts with “*” and ends with a semicolon.
 - SAS statements usually begin with **Identifying keywords** (e.g. data and proc) and end with **a semicolon**.



Ex. 1.2: Underline the identifying keywords of the program below.

- (a) Count the number of statements. 8 because of 8 semicolons found.
(b) Count the number of steps in the program. 2 steps.

```

/*p1_electricity.sas*/
* a program to list electricity data ;
libname DATA1 "D:\SAS_Datasets";

data testing;
set DATA1.electricity;
where average_temperature
>50;
run;

Proc Print data=testing;
run;

```

The 2nd line is a comment statement while the first line is not. It is just a part of a statement. See the following example:

```

input @1 name $20. /*last name */
      @200 test 8. /*score test */
      @50 age 3.; /*customer age */

```

1.5 Working with SAS syntax

- The program given in the above quiz is in conventional formatting: structured, consistent spacing makes a SAS program easier to read

SAS Base Programming Course
Chapter 1 Introduction, Getting Started to SAS

- Here is an example of unconventional formatting:

Line No.	SAS Program in Editor Window
1	<code>data testing;</code>
2	<code>length First_Name \$ 12</code>
3	<code> Last_Name \$ 18 Job_Title \$ 50;</code>
4	<code>infile 'newgraduates.csv' dlm=',';</code>
5	<code>input First_Name \$ Last_Name \$ Job_Title \$ Salary; run;</code>
6	

- SAS syntax rules:
 - SAS statements are free-format.
 - One or more blanks or special characters can be used to separate words. (See line 3)
 - They can begin and end in any column. (See line 2, Statement 2)
 - A single statement can span multiple lines. (see lines 2 to 3, i.e. statement 2)
 - Several statements can be on the same line. (See last line)
- Syntax errors occur when program statements do not conform to the rules of the SAS language.
- Common syntax errors:
 - misspelled keywords
 - unmatched quotation marks
 - missing semicolons
 - invalid options
- When SAS encounters a syntax error, SAS prints a warning or an error message to the log. An example output:

```
ERROR 22-322: Syntax error, expecting one of the following:
             a name, a quoted string, (, /, ;, _DATA_, _LAST_, _NULL_.
```

- SAS underlines the error and the following information is written to the SAS log:
 - the word ERROR or WARNING
 - the location of the error
 - an explanation of the error

```
3   ata testing;
   ---
      180


ERROR 180-322: Statement is not valid or it is used out of proper order.

:
:
:

NOTE: The SAS System stopped processing this step because of errors.
```

In this program “ata” is underlined. The correct statement is “Data testing;”.


- Save the program by clicking [FILE>SAVE AS...](#)

 **Ex. 1.3:** Create a new program called plex3.sas so that only the data with electricity consumed less than or equal to 1000 kWh are extracted to a new data set called LOWELECT and listed the new data set in the output window. Save the program file as plex3.sas. Write down your output in the box below.


In the output window:

The SAS System				12:09 Friday, June 8, 2012		5
Obs	Time_Period	Electricity_ Consumed	Cost	Heating_ Degeree_ Days	Average_ Temperature	
1	Year 5: March/Apr	578	77.39	1811	.	

1.6 Identifying SAS components



 Procedure SETINIT is used to produce a list of SAS components licensed at a given site.

```
proc setinit;  
run;
```

 Submit the program and see the following partial output window:

```
NOTE: SAS initialization used:  
      real time      0.82 seconds  
      cpu time       0.59 seconds  
  
1  proc setinit;  
2  run;  
  
NOTE: PROCEDURE SETINIT used (Total process time):  
      real time      0.01 seconds  
      cpu time       0.00 seconds  
  
Original site validation data  
Site name:  'JUCC - HONG KONG POLYTECHNIC UNIVERSITY'.  
Site number: 10801585.  
Expiration: 30DEC2016.  
Grace Period: 0 days (ending 30DEC2016).  
Warning Period: 30 days (ending 29JAN2017).  
System birthday: 22DEC2015.  
Operating System: WX64_WKS.  
Product expiration dates:  
---Base SAS Software  
30DEC2016  
---SAS/STAT  
30DEC2016
```

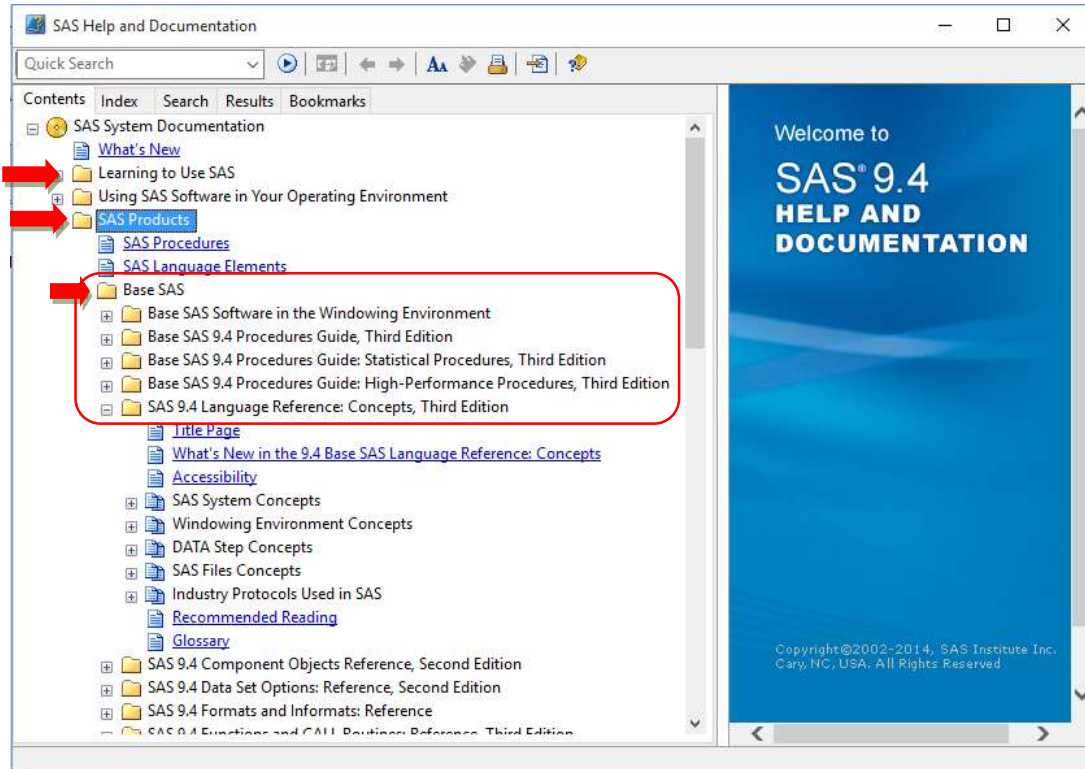
1.7 Using the Help Facility

-  Online web help: <http://support.sas.com/documentation/index.html>
-  Help given in the SAS windowing environment:

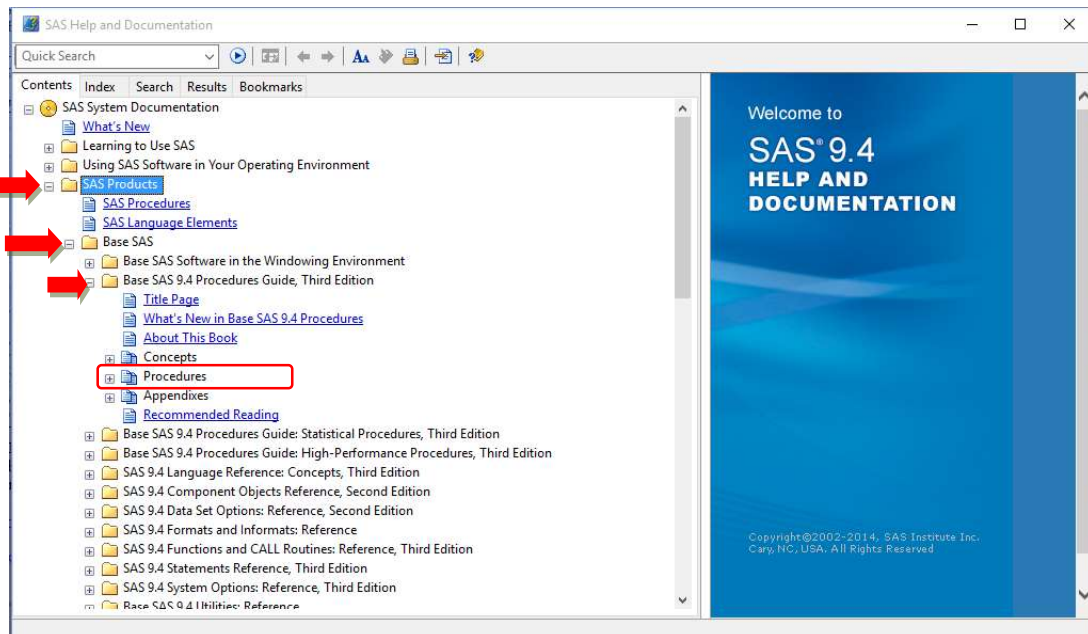
SAS Base Programming Course

Chapter 1 Introduction, Getting Started to SAS

- To open the Help facility, select **Help** > **SAS Help Documentation** or click .
- Select the **Contents** tab.
- From **Contents** tab, select **SAS Products** > **Base SAS**.



⚡ **Ex.1.4:** Find the documentation for PRINT procedure in the Help menu.



1.8 Summary

Components of a SAS program	<ul style="list-style-type: none"> - DATA step - PROC step
Modes for running SAS programs	<ul style="list-style-type: none"> - Batch - Non-interactive - Interactive
Primary windows in SAS windowing environment	<ul style="list-style-type: none"> - Log window (required to check even there is output!) - Editor window - Output window
A SAS statement	<ul style="list-style-type: none"> - Begin with identifying keywords - End with semicolon.
Method of comments	<ul style="list-style-type: none"> - /* this is a comment */ (This comment is a part of a statement) - * this is a comment; (This is a comment statement!)
Syntax errors	<ul style="list-style-type: none"> - misspelled keywords - unmatched quotation marks - missing semicolons - invalid options

1.9 Quiz

1. How many statements are in the DATA step?

```
data testing;
  length First_Name $ 12
         Last_Name $ 18
         Job_Title $ 50;
  infile 'AMAGraduate.csv' dlm=',';
  input First_Name $ Last_Name $
        Job_Title $ Salary;
run;
```

- (a) 5 (b) 6 (c) 7 (d) 8

2. How many syntax error(s) is/are found in the program?

```
daat testing;
set DATA1.electricity;
where average_temperature >70;

Proc Print data=testing
run;
```

- (a) 1 (b) 2 (c) 3 (d) none

3. How many comment(s) is/are found in the program?

```
*-----*
|This is a testing program.      |
*-----*;
data test;
  set DATA1.electricity;
  a=average_temperature+2; /*numeric */
run;

Proc Print data=test;
  *var a;
run;

/*
Proc Print data=test;
run;
*/
```

- (a) 1 (b) 2 (c) 3 (d) 4