Chapter 1

An introduction to SAS

SAS, written by The SAS Institute is a computing package for performing advanced statistical analyses. The current PC version on our machines is 9.1 (a UNIX version is also available).

To start SAS on a PC, click on the Start button, select Programs, SAS and then SAS V9.1 (depending on the machine, these names may change a little). Figure 1.1 displays a screenshot of SAS on the PC.

Figure 1.1: Screenshot from a previous version of SAS on the PC.
1.1 Understanding the windows

There are a number of windows in SAS that have different purposes:

- **Editor** window: This is where you enter and modify your SAS programs.
- **Log** window: Informs you of your current status. Messages appear here to tell you how things are going. Error messages also appear here.
- **Output** window: Any output you print from a SAS program goes to this window. Think of it like pages of output that would come from a printer.
- **Results** window: Stores objects created by SAS.
- **Explorer** window: Provides a view of the SAS files.

You can select a specific window by clicking on the tab at the bottom of the program window. You can also use the View menu commands (useful if you closed a window by accident).

You can print a particular window using the printer icon. You can save the contents of a window using the File → Save menu command.

1.2 Using SAS

There are two ways to use SAS:

1. Use the Solutions menu (we will not cover this in the notes).
2. We write programs in the editor. We then run the programs looking at the results in the output and log windows.

1.3 SAS programs

SAS programs are sequences of statements.

Each statement **must** end with a semi-colon ;

You can have many statements on a line (as long as you separate each command with a semi-colon), and you can have statements that go across multiple lines (as long as you do not split the command names).
Unlike R, SAS does not care about the case of names (upper case and lower case names are the same).

**Convention**: I will use capital letters to denote SAS commands and lower case letters to denote the variable names.

### 1.3.1 An example

The price of fish varies by species and time. We will look at data showing the price in cents per pound for eight different species in 1970 and 1980.

Enter the following program in the **Editor** window.

```sas
DATA fish_prices;
   INPUT species $ price_1970 price_1980;
DATALINES;
COD 13.1 27.3
FLOUNDER 15.3 42.4
HADDOCK 25.8 38.7
MENHADEN 1.8 4.5
OCEAN_PERCH 4.9 23
SALMON_CHINOOK 55.4 166.3
SALMON_COHO 39.3 109.7
TUNA_ALBACORE 26.7 80.1
; RUN;
```

Notes:

- SAS highlights the program to help you understand the syntax.
- The dollar sign, \$, after `species` lets SAS know that `species` is a character variable.
- All the other variables are numeric variables.

You can execute (submit) the program in many ways. Make sure you have selected the **editor** window. Either:

- Press the F8 key.
- Use the menu command **Run → Submit**.
• Click on the running person icon.

• Right click the mouse and select submit all.

Now check the Log window. If all goes well you should get something like:

```
DATA fish_prices;
   INPUT species $ price_1970 price_1980;
DATALINES;
NOTE: The data set WORK.FISH_PRICES has 8 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time 0.45 seconds
      cpu time 0.04 seconds
12 ;
13 RUN;
```

Now check the Log window. If all goes well you should get something like:

```
1 DATA fish_prices;
2     INPUT species $ price_1970 price_1980;
3     DATALINES;
4
NOTE: The data set WORK.FISH_PRICES has 8 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time 0.45 seconds
      cpu time 0.04 seconds
12 ;
13 RUN;
```

Now check the Log window. If all goes well you should get something like:

```
1 DATA fish_prices;
2     INPUT species $ price_1970 price_1980;
3     DATALINES;
4
NOTE: The data set WORK.FISH_PRICES has 8 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time 0.45 seconds
      cpu time 0.04 seconds
12 ;
13 RUN;
```

Now check the Log window. If all goes well you should get something like:

```
1 DATA fish_prices;
2     INPUT species $ price_1970 price_1980;
3     DATALINES;
4
NOTE: The data set WORK.FISH_PRICES has 8 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time 0.45 seconds
      cpu time 0.04 seconds
12 ;
13 RUN;
```

Note how the Output window is empty. We did not tell SAS to print the data. We can use the explorer window to check that the data is correct. In the explorer double click on libraries, then work and finally fish_prices. Check the data is correct.

• Note how the names have been cut down to eight characters long (we will fix this later).

Here is the full program to read in the data and then print the data (to the screen, not a printer!).

```
DATA fish_prices;
   INPUT species $ price_1970 price_1980;
DATALINES;
COD         13.1 27.3
FLOUNDER    15.3 42.4
HADDOCK    25.8 38.7
MENHADEN     1.8  4.5
OCEAN_PERCH  4.9  23
SALMON_CHINOOK 55.4 166.3
SALMON_CHIP  39.3 109.7
TUNA_ALBACORE 26.7  80.1
;
PROC PRINT data=fish_prices;
   title "Fish Prices for different species in 1970 and 1980.";
RUN;
```

After submitting the program (e.g., press F8), examine the log window to make sure everything ran okay:
DATA fish_prices;
   INPUT species $ price_1970 price_1980;
DATALINES;

NOTE: The data set WORK.FISH_PRICES has 8 observations and 3 variables.
NOTE: DATA statement used (Total process time):
   real time 0.03 seconds
   cpu time 0.02 seconds

; PROC PRINT data=fish_prices;
   title "Fish Prices for different species in 1970 and 1980."
RUN;

NOTE: There were 8 observations read from the data set WORK.FISH_PRICES.
NOTE: PROCEDURE PRINT used (Total process time):
   real time 0.25 seconds
   cpu time 0.02 seconds

Now check the Output window

Fish Prices for different species in 1970 and 1980.
20:34 Friday, October 17, 2008

<table>
<thead>
<tr>
<th>Obs</th>
<th>species</th>
<th>price_1970</th>
<th>price_1980</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>COD</td>
<td>13.1</td>
<td>27.3</td>
</tr>
<tr>
<td>2</td>
<td>FLOUNDER</td>
<td>15.3</td>
<td>42.4</td>
</tr>
<tr>
<td>3</td>
<td>HADDOCK</td>
<td>25.8</td>
<td>38.7</td>
</tr>
<tr>
<td>4</td>
<td>MENHADEN</td>
<td>1.8</td>
<td>4.5</td>
</tr>
<tr>
<td>5</td>
<td>OCEAN_PE</td>
<td>4.9</td>
<td>23.0</td>
</tr>
<tr>
<td>6</td>
<td>SALMON_C</td>
<td>55.4</td>
<td>166.3</td>
</tr>
<tr>
<td>7</td>
<td>SALMON_C</td>
<td>39.3</td>
<td>109.7</td>
</tr>
<tr>
<td>8</td>
<td>TUNA_ALB</td>
<td>26.7</td>
<td>80.1</td>
</tr>
</tbody>
</table>

You can also look at the Results window. It will summarize what you have done so far.

**Exercise:** Try changing the PRINT command to:

PROC PRINT data=fish_prices NOOBS;
   title "Fish Prices for different species in 1970 and 1980."
RUN;

What is the effect of NOOBS?
1.4 Reading in data from a text file

Go to the class website and save the file `fish_prices.txt` to the folder `My Computer → Drive C: → My SAS files`.

Here is the SAS DATA statement to read in the text file.

```sas
/* set up the DATA variable "fish_prices" */
DATA fish_prices;
   /* First tell SAS where to get the data from */
   INFILE "C:\My SAS Files\fish_prices.txt";
   * Declare the 3 variables ;
   INPUT species $ price_1970 price_1980;
RUN;
```

Notes:

- The `INFILE` command goes before the `INPUT` command.
- The `DATALINES` command goes after the `INPUT` command.
- Comments in SAS are of the form `/* your comments */` or `* your comments ;`

1.4.1 Reading data arranged in columns

Let’s look at the `fish_prices.txt` file in some detail. The first line is a ruler that counts the number of characters (the column) (you don’t need to add this line to the file – I have included here just to help you visualize the structure of the data).

```
----+----1----+----2----+----3
COD  13.1  27.3
FLOUNDER  15.3  42.4
HADDOCK  25.8  38.7
MENHADEN  1.8   4.5
OCEAN_PERCH  4.9  23
SALMON_CHINOOK  55.4 166.3
SALMON_COHO  39.3  109.7
TUNA_ALBACORE 26.7   80.1
```

Note that
- **Species** goes from column 1-15.
- **price_1970** goes from column 16-19.
- **price_1980** goes from column 24-28.

Try the following code:

```sas
/* set up the DATA variable "fish_prices" */
DATA fish_prices;
    /* where is the data? */
    INFILE "C:\My SAS Files\fish_prices.txt";
    /* Declare 3 variables and the column positions. */
RUN;
```

**Remarks**

1. You do not have to declare the columns for all the variables.

2. Numeric data which is read in as columns must be right justified (look at this and the next example).

**Exercise**: PRINT this dataset to the output window. What do you notice about the species variable?

### 1.4.2 Skipping input

Go to the class website and save the file `sewage.txt` to the folder `My Computer → Drive C: → My SAS files`. Here is the contents of that file (with a ruler to help view the data)

```
---------1---------2
sample  MSI    SIB
  1  0.39   0.36
  2  0.84   1.35
  3  1.76   2.56
  4  3.35   3.92
  5  4.69   5.35
  6  7.70   8.33
  7 10.52  10.70
  8 10.92  10.91
```

We do not want to read in the first line of the file, or the sample number. Here is the SAS code:
DATA sewage;
   /* where is the data? */
   INFILE "C:\My SAS Files\sewage.txt" FIRSTOBS=2;
   /* Declare 3 variables and the column positions. */
   INPUT msi 5-9 sib 12-16;
RUN;

FIRSTOBS=2 tells SAS to start reading from line 2 of the textfile. Using the column positions we only read in msi and sib.

1.5 Variables and missing values

1.5.1 Choice of variable names

- Each name must start with a character or _ symbol. It can then be followed by any character, digit or _ symbol.
- Variables names are not case sensitive e.g. radius and Radius are the same names;
- Choose your variable names so that they are representative of what you are storing; e.g., blood_pressures could represent a dataset of blood pressure values.

1.5.2 Missing values

Missing values in SAS are denoted by a period '.

Example: In the fish prices example, suppose the 1970 price is missing for haddock. Part of the text file would look like

<table>
<thead>
<tr>
<th>FLOUNDER</th>
<th>15.3</th>
<th>42.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>HADDOCK</td>
<td>.</td>
<td>38.7</td>
</tr>
<tr>
<td>MENHADEN</td>
<td>1.8</td>
<td>4.5</td>
</tr>
</tbody>
</table>