Instructions for Using PC SAS

To open the SAS software...

Locate the software under: **All Programs → SAS → SAS 9.2.**

Maximize the SAS software to take up the whole screen.

SAS User Windows

Upon opening SAS, several windows will appear. The most important windows are PROGRAM EDITOR, OUTPUT, and LOG. The OUTPUT window is probably hidden, but you can click on the **Output-(Untitled)** button toward the bottom of the screen to see it. Similarly, you can move back and forth between the windows by clicking on the respective buttons.

Uses of windows:

- PROGRAM EDITOR - Contains the SAS statements (i.e. SAS program) to perform the requested statistical analysis. Programs are submitted from this window.
- OUTPUT - Contains the output generated by the statistical procedures.
- LOG - Contains running commentary that the computer generates while running your SAS program. When things run correctly, general notes appear. When there is an error, commentary will appear in the color red. When you have trouble, you should look in the LOG window for information. Since the LOG window keeps a running history, you may want to clear it out occasionally. You can do this by having the LOG window active, right-clicking on the mouse, then choosing **edit, ..., clear all.**

THE MOST COMMON ERROR... you forgot to put a semi-colon at the end of a line.

Programming in SAS

The following is a brief summary for opening existing program files, saving them, and submitting them for execution.

Opening an Existing Program:
From the PROGRAM EDITOR window, click on **File,...,Open** and then choose your program file. SAS program files have a '.sas' ending.

Entering a New Program:
To enter a new program, click on the PROGRAM EDITOR window and enter the program lines.

Editing a Program:
Editing a SAS program on PC SAS is similar to word processing.

Saving a Program File:
From the PROGRAM EDITOR window, click on **File**, and then **Save as ...**. You may then enter the file name or click on the existing name if it is there.

Submitting a Program:
From the PROGRAM EDITOR, click on **Run** and drag the mouse down to **Submit**. To run a program, you can also just click on the 'running person' icon at the top of the screen. Any error messages will appear in the LOG window (in red). It is helpful to clear the LOG and the OUTPUT windows before submitting a program by having the respective window active, right-clicking on the mouse, then choosing **edit**, and **clear all**.

Switching Between Windows:
To switch between windows either click on the desired window if it is visible, or go to **View** from the pull-down menu and click on the desired window. You can also click on the respective buttons toward the bottom of the screen.

Printing a SAS window:
When editing a program, it's often convenient to have a hardcopy. To print a program file, have the PROGRAM EDITOR window active, then choose **file**, **print...**, and choose the desired printer location. To print in Snedecor, you need to have an account and password. You can also print from the LOG and OUTPUT windows in the same way.

Quitting PC SAS:
To quit, click on **File, . . . , Exit** from the PROGRAM EDITOR window.

An Example:

Below is a SAS program for analyzing 24 measurements of ozone concentration (ppb) in Denver, CO. Comments regarding these SAS commands are shown following the code on the next page.

```
data denver;
   input ozone @@;
   cards;
    65 21 44 47 53 26
    47 30 49 47 16 50
    49 40 34 56 47 37
    24 17 32 52 53 47
;

proc print data=denver;
run;

proc univariate data=denver plot;
   var ozone;
run;

proc sort data=denver;
   by ozone;
run;

proc print data=denver;
run;

proc chart data=denver;
   vbar ozone/midpoints=15 to 75 by 10;
run;

/*This line is a comment and will not be executed.*/

proc gchart data=denver;
   vbar ozone/midpoints=15 to 75 by 10;
run;
```

Comments about the above program:

- The data set is titled 'denver'.
- The variables in a SAS data set are listed after the *input* statement. Here, we have only one variable called 'ozone'. The '@@' tells SAS to read through a full row of data and then continue to the next line, rather than going down a column first.
- The *cards* statement tells SAS the data will be coming next.
- '*Proc*' is short for procedure and SAS has many different procedures available.
- *Proc Print* will print your SAS data set in the OUTPUT window.
- *Proc Univariate* provides summary statistics for a single variable at a time specified as var _____ . Including the *Plot* option generates a number of graphs including a histogram and a normal probability plot of the named variable.
- You can sort the data by a variable using *Proc Sort*.
  Here, the data is sorted in ascending order by the value of 'ozone'.
- *Proc chart* will produce a bar chart with the given x-axis scaling.
- You can add comments to your program by starting a line with '/*', writing the desired comments, then ending the line with '*/'.
- More aesthetically pleasing graphs can usually be created by starting the graphing command with a 'g', like *gchart* or *gplot*.

Help for SAS software

Help from the SAS Institute:

- Each SAS package contains a 'Help' library. You can open this library by clicking on the button with a '?' or by pulling down the **Help** option from the menu and choosing **Sas System Help**. This library is *very* useful when you're looking for documentation on SAS syntax.
- You can also get a lot of useful information from the by *googling* any SAS commands you're curious about.

Example of SAS Analysis

of Bear Measurement Data

See home website to down load SAS files and data.

"bear.sas"

```
/* This text is a comment.  SAS ignores such text.   */
/* The comments are here to help you understand the  */
/* program.  The next three lines read the data from */
/* a file called bear.txt located in the listed      */
/* pathway.                                          */

data bear;
  infile 'Y://Users/rdecook/Desktop/bear.txt';
  input age length sex weight chest headlth headwth month neck;
run;

/* The next line prints the data set.                */

proc print;

/* The next set of commands computes the mean of the */
/* variable neck along with other statistics.  The   */
/* output statement creates a new data set called    */
/* cidata that will contain the sample size n, the   */
/* mean of the neck circumference observations       */
/* (labeled xbar), and the standard error of the     */
/* mean (labeled soversqrtn for s divided by the     */
/* square root of n).  The names on the left are SAS */
/* names for these quantities.  The names on the     */
/* right can be anything you choose.                 */

proc means;
  var neck;
  output out=cidata
    n=n
    mean=xbar
    stderr=soversqrtn;

/* The next line prints the contents of the most     */
/* recently created data set, cidata.                */

proc print;

/* The next lines add three new variables to the     */
/* data set cidata.  The first is t which equals     */
/* our book calls t with a 0.05 subscript.  It is    */
/* the t value that has 97.5% of the observations    */
/* from a t-distribution with n-1 degrees of freedom */
/* to its left.                                      */

data cidata; set cidata;
  t=tinv(0.975,n-1);
  lcl=xbar-t*soversqrtn;
  ucl=xbar+t*soversqrtn;

proc print;

proc print;
  title '95% Confidence Interval for the Mean Neck Circumference';
  var lcl ucl;
run;


/* Unless you name the data set SAS should be looking  */
/* in, it will always assume you want to analyze the   */
/* the last one used.  Below, I want to return to the  */
/* original data set 'bear' to do some regression.     */
/* Also, a new data set called 'diagnost' is created   */
/* containing the residuals and predicted values from  */
/* the regression model.                               */

proc gplot data=bear;
  plot chest*age;
run;
proc reg data=bear;
  model chest=age;
  output out=diagnost r=resids p=preds;
run;
proc print;
run;
proc gplot data=diagnost;
  plot resids*preds;
run;
proc capability;
  var resids;
  qqplot;
  title 'QQ Plot ';
run;
```

"Bears.txt"

19	53.0	1	80	26	11.0	5.5	7 16.0
55	67.5	1	344	45	16.5	9.0	7 28.0
81	72.0	1	416	54	15.5	8.0	9 31.0
115	72.0	1	348	49	17.0	10.0	7 31.5
104	62.0	2	166	35	15.5	6.5	8 22.0
100	70.0	2	220	41	13.0	7.0	4 21.0
56	73.5	1	262	41	15.0	7.5	7 26.5
51	68.5	1	360	49	13.5	8.0	4 27.0
57	64.0	2	204	38	13.5	7.0	9 20.0
53	58.0	2	144	31	12.5	6.0	5 18.0
68	73.0	1	332	44	16.0	9.0	8 29.0
8	37.0	1	34	19	9.0	4.5	8 13.0
44	63.0	2	140	32	12.5	4.5	8 10.5
32	67.0	1	180	37	14.0	5.0	8 21.5
20	52.0	2	105	29	11.5	5.0	8 17.5
32	59.0	1	166	33	13 0	8.0	8 21.5
45	64.0	1	204	39	13 5	7.0	9 24.0
9	36.0	2	26	19	9 0	4.5	9 12.0
21	59.0	1	120	30	13.0	6.0	9 19.0
177	72.0	1	436	48	16.0	9.5	9 30.0
57	57.5	2	125	32	12 5	5.0	9 19.02
81	61.0	2	132	33	13.0	5.0	9 20.0
21	54.0	1	90	28	13 0	5.0	9 17.0
9	40.0	1	40	23	10.0	4.0	9 13.0
45	63.0	1	220	42	16 0	6.0	9 24.0
9	43.0	1	46	23.0	10.0	4.0	9 13.5
33	66.5	1	154	34.0	13.5	6.0	9 22.0
57	60.5	2	116	31.0	13.0	5.5	9 17.5
45	60.0	2	182	34.5	13.0	6.5	9 21.0
21	61.0	1	150	34.0	14.5	5.5	9 20.0
10	40.0	1	65	26.0	9.5	4.5	10 16.0
82	64.0	2	356	48.0	13.5	6.5	10 28.0
70	65.0	2	316	48.0	14.5	6.5	10 26.0
10	49.0	1	94	29.0	11.0	5.0	10 17.0
10	47.0	1	86	29.5	11.5	5.0	10 17.0
34	59.0	1	150	35.0	13.0	7.0	10 21.0
34	72.0	1	270	44.5	16.5	6.5	10 27.0
34	65.0	1	202	39.0	14.0	5.5	10 24.0
58	63.0	2	202	40.0	13.5	6.5	10 21.5
58	70.5	1	365	50.0	15.5	7.0	10 28.0
11	48.0	1	79	31.0	11.5	6.0	11 16.5
23	50.0	1	148	38.0	12.0	6.5	11 19.0
70	76.5	1	446	55.0	15.5	7.0	10 28.0
11	46.0	2	62	27.0	9.0	5.0	11 15.0
83	61.5	2	236	44.0	14.5	7.0	11 23.0
35	63.5	1	212	44.0	13.5	8.5	11 23.0
16	48.0	1	60	26.0	10.0	4.0	4 15.5
16	41.0	1	64	26.0	10.0	5.0	4 15.0
17	53.0	1	114	30.5	11.5	5.0	5 17.0
17	52.5	2	76	28.0	11.5	5.0	5 15.0
17	46.0	2	48	23.0	11.0	4.5	5 13.0
8	43.5	2	29	24.0	10.0	4.5	8 10.0
83	75.0	1	514	54.0	15.5	8.0	11 30.5
18	57.3	1	140	32.8	12.5	8.5	6 18.0

"data_and_graphics.sas"

```
/*** This SAS code gives some examples of
     how to use SAS to manipulate data
     and create sub-sets of data.

       It also give you some examples of graphing.***/


/*Import data which is an Excel Spreadsheet with variable names at the top.*/
PROC IMPORT OUT= salary_data
            DATAFILE= "Y://Users/rdecook/Desktop/salary_SAS_intro.xls"
            REPLACE;
RUN;
proc print data=salary_data (obs=10);
run;



/*Create a new variable which is the log transformation of salary and
add it to the present data set called 'salary_data'.*/
data salary_data; set salary_data;
      logSAL=log(Salary_9_mo);
run;
proc print data=salary_data (obs=10);
run;
proc univariate data=salary_data plot;
var logSAL;
run;

/*Get a frequency table for the number of faculty in each department.*/
proc freq data=salary_data;
table department;
run;

/*Because there are a lot of departments, I chose to put the departments into
  5 departmental categories.  This is done with the 'if' and 'then' statements.
  The quotations are needed because Department is a class or nominal variable.*/
data salary_data; set salary_data;
  if Department="AGRON" or Department="AN S" or Department="NREM" or
               Department="FSHNA" or Department="FSHNF" or Department="PL P" or
               Department="HORT" or Department="ENT" then deptcat="AgSci";
   if Department="BBMBA" or Department="BBMBS" or Department="GDCBA" or
               Department="GDCBS" or Department="EEOBA" or Department="EEOBS"  then deptcat="BioSci";
   if Department="PHYSA" or Department="CHEM" or Department="GE AT" then deptcat="PhysSci";
   if Department="STAT" or Department="MATH" or Department="COM S" then deptcat="MathSci";
   if Department="CH E" or Department="E CPE" or Department="IMSE" or
        Department="M S E" or Department="A B E" or Department="CCE E" or
        Department="M E" or Department="AER E" then deptcat="Eng";
proc freq data=salary_data;
table deptcat;
run;

proc print data=salary_data (obs=30);
run;


/*Create separate data sets for the different ranks of professors.
R1 are full, R2 are associate, and R3 and assistant professors.*/
data R1; set salary_data;
   if rank_code=1;
data R2; set salary_data;
   if rank_code=2;
data R3; set salary_data;
   if rank_code=3;
run;


/*Look at relationship between salary and 'Average contracts and grants' for each rank,
  use a different symbol for men and women (gender).*/
proc gplot data=R1;
   plot Salary_9_mo*Avg_Cont_Grants=gender;
   run;

/*A couple outliers are making that plot not that useful.*/
data R1_noouts; set R1;
   if Avg_Cont_Grants<= 3000000;
   run;
proc gplot data=R1_noouts;
plot Salary_9_mo*Avg_Cont_Grants=gender;
run;


/************    Choose and define the symbols used  **************/
symbol1 v=square i=none color=black;
symbol2 v=star i=none color=red;
symbol3 v=circle i=none color=green;
symbol4 v=plus i=none color=black;

proc gplot data=R1_noouts;
   plot Salary_9_mo*Avg_Cont_Grants=gender;
   title 'Full Professors';
   run;
proc gplot data=R2;
   plot Salary_9_mo*Avg_Cont_Grants=gender;
```

```
proc gplot data=R3;
   plot Salary_9_mo*Avg_Cont_Grants=gender;
   title 'Assistant Professors';
   run;



/*********************************************************************
***                General Linear Model:                          ***
***   Fit a model to predict salary from other variables  ***
*********************************************************************\

/*Put R1_noouts, R2 and R3 together as a single data set for analysis.*/
data finalset; set R1_noouts R2 R3;
run;

/*Run the analysis using the full model.*/
proc glm data=finalset;
   class gender deptcat Rank_Code;
   model Salary_9_mo = gender deptcat Rank_Code Avg_Cont_Grants/solution;
run;


/*Run the analysis using the full model and keep your output for
  checking diagnostics.*/
proc glm data=finalset;
   class gender deptcat Rank_Code;
   model Salary_9_mo = gender deptcat Rank_Code Avg_Cont_Grants/solution;
   output out=diagaut p=preds r=resids;
run;
/*Diagnostics: Checking for constant variance.
  The '=4' here means I want the 4th symbol used.*/
proc gplot data=diagout;
   plot resids*preds=4;
   run;
prac gplot data=diagout;
   plot resids*preds=deptcat;
   run;
/*Looks like there's some violation of the assumption.*/


/*Diagnostics: Normality*/
proc univariate data=diagout normal plots;
       var resids;
       run;
/*Which observations are the outliers?*/
data outliers1; set diagout;
   if resids<=-55000;
   keep Department Rank_Code Avg_Cont_Grants resids;
   run;
proc print data=outliers1;
   run;
data outliers2; set diagout;
   if resids>=55000;
   keep Department Rank_Code Avg_Cont_Grants resids;
   run;
proc print data=outliers2;
   run;


/* Exploring how salaries relate to gender,deptcat,rank?*/
proc sort data=finalset;
   by gender rank_code deptcat;
proc means data=finalset;
   var Salary_9_mo;
   by gender;
   run;

proc means data=finalset noprint;
   var Salary_9_mo;
   by gender rank_code deptcat;
   output out=salarymeans2 mean=m;
   run;
proc sort data=salarymeans2;
   by deptcat rank_code;
proc print data=salarymeans2;
run;

proc freq data=finalset;
   table rank_code*gender;
   run;
```