Use the following symbols: < or lt, =, > or gt, ge, le, ne

```
Data libraryname.newdatasetname ;  set
libraryname.datasetname;
   If variable1 = value then variable2 = value; else variable2 = value;
  newvariable = log(variable);            (create a new variable that is the natural logarithm of a variable)
  newvariable = lag(variable);            (create a new variable that is the lagged value of a variable)
  newvariable = lag2(variable); etc.      (create a new variable that is the value of a variable lagged two periods)
  newvariabel = variable**2;              (creates the square of a variable)
  newvariable = variable**0.5;            (create the square root of a variable)
  newvariable = exp(variable);            (create a variable that is exp^variable)
  newvariable = variable1*variable2;      (creates an interaction between variables)
run;                                      (always end SAS statements including formulas with a semicolon)
```

Example of a data step program:

```
Data newroses; set resec312.roses;
 if quarter = 2  then q2 = 2; else q2 = 0;
 proseq2 = prose*q2;           * create an interaction variable;
 lnsales = log(sales);         * create natural log variables to estimate a log-log model;
 lnprose = log(prose);
 lnpcarn = log(pcarn);
 lndinc  = log(dinc);
 lnproseq2 = lnprose*q2;       * create an interaction variable using natural logs;
run;
```

The following example shows creating a new data set using the **set** and **where** statements:

```
data resec312.cps2011civlfnesa;
  set resec312.cps2011civil;     * copy the resec312.cps2011civil SAS data set;
  where gediv = 1 or gediv = 6; * selects only data for New England and South
                                   Atlantic regions;
run;
```

Procedures: Procedures in SAS perform an operation on data. We're mostly in statistical procedures, but there are several other procedures that are very useful. Example programs are on the next page.

Procedure	**Description**
`proc contents`	Lists the variables in your data set, identifies the size of the data set and the types of variables.
`proc sort`	Sorts the data in ascending or descending order. Ascending is the default (eg. smallest to largest).
`proc means`	Lists the means, st. dev., min. and max. for all your variables as a default, but you can ask for more stats, like the median, quartiles, etc.
`proc freq`	Determines the frequencies for your variables. Use this for categorical variables where the number of different values for each variable is limited.
`proc reg`	This is a biggie for econometrics. It provides the OLS estimates for a regression model.
`proc glm`	Estimates a regression model – OLS is the default, but you can use proc GLM to obtain feasible generalized least squares estimates and correct for heteroskedasticity.
`proc autoreg`	Provides feasible generalized least squares estimates to correct for autocorrelation problems.

Exploring the contents of a data set:

```
proc contents data=resec312.cps2011; * provide contents for resec312.cps2011;
run;
```

Sorting data:

```
proc sort data=new; by f;     * sort the data in ascending order by the variable f;
run;
* it’s then easy to then estimate different models for the different values for f;
proc reg data=new;  where f=1;      * use only the data when f=1;
  model wage = hsi bsi msi proi exp expsq ;
run;
proc reg data=new; where f=0;       * use only the data when f=1;
  model wage = hsi bsi msi proi exp expsq ;
run;
```

Descriptive statistics:

These are options that define which descriptive stats are provide in the table.

```
proc means data =
private2011NE n mean std q1                                        median q3
min max ;
  var hsi bsi msi proi lths;  * var statement chooses a subset of variables;
run;

proc freq data = private2011NE;  * provides frequencies and relative frequencies;
  tables hsi bsi msi proi lths;  * tables statement chooses a subset of variables;
run;

proc corr data = private2011NE ; * provides table of correlation coefficients;
  var hsi bsi msi proi lths;     * var statement chooses a subset of variables;
run;


proc reg data= private2011NE;    * estimates regression models – our workhorse proc;
  model wage = hsi bsi msi proi exp expsq f fexp fbsi fmsi /
            noint dw dwprob spec covb;
                                       * model specification and options after the /;
   test f = fexp = fbsi = fmsi = 0;    * F-test of the joint hypothesis that the
                                         parameters for these variables are all zero;
   restrict fbsi - fmsi = 0;           * restricts parameter estimates – in this
                                           case to be equal;
   output out=ests p=wagehat r=errors;     * creates data set from the results named
                                            “ests” which goes into the WORK library
and includes all data plus the predicted wages (wagehat) and the residuals (errors);
run;

proc glm data=new2; * used to correct for heteroskedasticity – feasible GLS;
  model wage = hsi bsi msi proi exp expsq f fexp fbsi fmsi; * model specification
   weight wt;       * weights used to correct for heteroskedasticity must be created
                    in a data step;
run; quit;

proc autoreg data= new;  * corrects for autocorrelation;
  model y = x1 x2 x3 / method=yw nlag=3 dw=3 dwprob backstep ;  * method Yule-Walker,
check for 3 lags, eliminate the insignificant effects (backstep);
run;
```