Biostat 512 Computing Workshop: Introduction to Stata 10

NOTE: This document has been modified from an original document entitled by Carolyn Hutter, Lixuan Qin, and Jia Yin Wan, entitled "Biost 536 Computing Workshop : Introduction to Stata 8.2".

Outline of Topics

Part A

Part A

0.0 Exploring Stata

0.1 Accessing Stata

In the Health Sciences 3rd floor microcomputer lab, Stata 10.x should be installed on all the computers. To access Stata 10.x click on the following buttons:

START --> Programs --> Stata 10 --> StataIC 10

0.2 Stata Windows

After following the above instructions, the Stata program should be up and running. You will notice that Stata is made up of the following components:

Toolbar: Provides buttons for common tasks
Stata Command: Where the commands are typed
Results: Shows all the output from typed in commands
Review: Shows previous commands
Variables: Shows all variables in the dataset
Data Browser: Shows the data *without* allowing changes
Data Editor: Shows the data, but allows for changes
Graph: Appears when graphs are created
Do-file Editor: Allows editing of do-files

Please note that the some windows cannot be accessed simultaneously. For example, when the "Data Editor" window is opened, you cannot input commands into the "Stata Command" window.

0.3 Getting Help

The most important skill to gain is to understand the help features that are already installed in Stata. Being able to search for functions, access the help pages for these functions, and being able to understand the examples and the format of the commands will be invaluable.

Stata comes with the following documentation (available at the Microlab).

Getting Started [GSW]
User's Guide [U]
Base Reference Manual (4 vols) [R]
Graphics Reference Manual [G]

0.3.1 Help from the menu bar:

Stata also has an on-line help system that can be accessed by choosing Help from the menu bar. When you click on Help you get the following submenu

Contents	See the help table of contents

Search…	Search for help on a particular topic
Stata Command…	Get help for a particular Stata command

:
:

If you are a new user it is useful to browse through the table of Contents just to get an idea of what the package can do. Search is useful if you have a general topic you are interested in but don't know the actual command.

Help pages are shown using the Viewer. Each command's help page uses boldface font for stata commands, ordinary type for documentation/explanation of the command, and underlined blue for cross-references and hyperlinks. When you click on a word in blue you get either the help screen for that word or its dialog box, depending upon the context.

There are Stata tutorials linked from http://www.stata.com/links/resources1.html; the most useful for you will discuss Stata 10.x.

0.3.2 Help via Stata commands:

Alternatively, instead of accessing help from the menu bar, you can also type in :

```
search word [word ...] [, [ local | net | all ] author entry exact faq
            historical or manual sj ]

help [command or topic name]
```

Note: The underlined portion of the command is the minimal abbreviation that Stata will recognize. The [...] indicate optional parts of the command. The *italicized words* describe the input that can be substituted. **Bold keywords** are defined in the help file. There are also useful examples in the help file.

Note: The “Page Up” and “Page Down” keys will automatically take you to the previous or the latter command in the Stata Command window.

Exercise 0.3.2:

```
help search
help help
search inputting data
```

Now make sure your Stata Command window is active. Press the “Page Up” key to go to a previous command. Press the “Page Down” key to go to a latter command.

0.4 Version control

```
version
version #[, born(ddMONyyyy) missing]
version #[, born(ddMONyyyy) missing]:  command
```

Stata is continuously being updated; therefore, in order to be sure that you are using Stata version 10.x, get into the habit of specifying the version before you begin your Stata session.

Exercise 0.4:

```
      version 10.x
```

1.0 Creating, Saving, and Printing Log files

1.1 Creating a log file: Echo copy of session to file or device

```
log using filename [, append replace [ text | smcl ] ]
cmdlog using filename [, append replace ]

log { on | off | close }
cmdlog { on | off | close }
```

The "**log**" command allows you to make a full record of your Stata session. A log is a file containing what you type and Stata's output. The "**cmdlog**" command allows you to make a record of only the commands you type during your Stata session. You can make full logs and command logs simultaneously, one or the other, or neither. Neither is produced until you tell Stata to start logging.

Full logs are recorded in one of two formats: SMCL (Stata Markup and Control Language) or text (meaning ASCII). The default is SMCL, but you can specify an option to state the format. The SMCL format preserves graphs and Stata formats, but the ASCII format is easily copied into Word or other text editor.

Exercise 1.1:

Create a log file called test.smcl in the C:\ directory:
```
. log using "C:\test.smcl"
```

Check the status of logging:
```
. log
```

Type in some made-up numeric variable called testvar1:
```
. input testvar1
1. 12
2. 23
3. 43
4. 523
5. end
```

Suspend the recording of your commands in the test.smcl log file:
```
. log off
```

Check the status of logging:
```
. log
```

Type in some more commands that will not be in the test.smcl log file:
```
. input str15 testvar2
1. "nothing"
2. "much"
3. "here"
```

```
4. "lol"
```

Turn the recording back on:

```
. log on
```

Ending the recording session; You should get into the habit of closing your logs:

```
. log close
```

Creating a new log file that is text:

```
log using "C:\test.txt", text
```

1.2 Viewing the log file

```
      view [file] ["]filename["] [, asis]
```

The log file can be viewed by typing in (**view** "*filename*" **)**. The **asis** option specifies that the file should be in ASCII format.

Exercise 1.2:

To view the log:

```
.view "C:\test.smcl"
```

Alternatively, go to the toolbar: File → Log → View…

Note that the creation of "testvar2" is not in `"C:\test.smcl"` because the log was suspended with the **log off** command.

1.3 Saving/Translating the log file into a txt file

```
      translate input_filename output_filename [, translator(tname)
                override_options replace ]
```

Use this command to produce printable versions of SMCL logs by converting the SMCL format files to ACSII format files. Alternatively, you can print or view SMCL logs by clicking on File → Log. SMCL logs can be viewed in the Viewer.

Exercise 1.3:

Convert the *sml log file to an ASCII format.

```
     . translate "C:\test.smcl" "C:\mylog.log"
```

Another cute use of translate is to recover a log and save it to a log-file (*i.e.*, `mylog.txt)` when you have forgotten to start one:
(Note a location directory does not always have to be specified when specifying the log-file, but you must be consistent throughout. In other words, if you decide to specify the log-file with the location then you can't refer to the same log-file without the location specification.)

```
      . translate @Results mylog.txt
```

View the contents by going to File → View…
Type in `“C:\mylog.log” and then “mylog.txt”`

2.0 Getting Data into Stata

2.1 Enter data from keyboard

```
        input [varlist] [, automatic label ]
```

The command “`input`” should be followed by variable names `[varlist]` that are user-specified. Every name is separated by a space, and character or string-valued variables should be preceded by “`str#`” where # is the max number of characters to read in for that character variable. If “`str#`” is not specified then the variable is numeric by default.

Alternatively, you can go to the toolbar: Data --> Data Editor. The variable values can be typed. Double-click on the column heading and the variable name, label, and format can be specified. To save the data, click on “Preserve”. To restore the previously saved data, click on “Restore”. Variables or certain values can also be deleted by clicking the “Delete...” button and then specifying what variable(s) or highlighted values are to be deleted.

Exercise 2.1:

Clear existing data
```
. clear
```

From the Stata command window:

```
      . input str20 TAname heightcm str1 sex
      1.  “Lixuan” . “F”
      2.  “Angel” 165 “F”
      3.  “Carolyn” . “F”
      4.  end
```

Now go to the toolbar: Data --> Data Editor

```
I.  Change Carolyn’s height to 160
II. Input a new variable called “ateLunch” which is a “yes” or “no” character
variable.  Assume that everyone has eaten lunch.
```

2.2 Read ASCII (text) data created by a spreadsheet

```
        insheet [varlist] using filename [, [no]double [no]names
                [ comma | tab | delimiter("char") ]
                clear ]
```

This command is useful when reading in data that is either comma-delimited, tab-delimited or character-delimited data. The *`filename`* is the path or location of the file. It could either be a url or a directory location. The “`clear`” option replaces all other data from the working memory with the one being currently read. When `[varlist]` is not specified, the variable names from the original file are used.

Exercise 2.2:

```
. insheet using "http://staff.washington.edu/jyw/TA/height_weight.csv", clear
```

You could also download the dataset from http://staff.washington.edu/jyw/TA/height_weight.csv and then use the following command:

```
. insheet using "PATH\height_weight.csv", clear
```

Where PATH is the directory location of the file
(ie, `C:\Documents and Settings\c-16\Desktop`
or something similar)

2.3 Read unformatted ASCII (text) data

```
infile varlist [_skip[(#)] [varlist [_skip[(#)] ...]]] using filename
          [if exp] [in range] [, automatic byvariable(#) clear ]
```

This command reads in a data file that has no column variable labels and is in ANSI text format. Therefore, *varlist*, the list of names for the column variables has to be defined. This command allows the data to be subset based on an expression `[if exp]` or by observation positions `[in range]`. The "**byvariable(#)**" option is used when the data has rows as variables and columns as observations. In this case, the # would denote the number of row variables.

Exercise 2.3:

To subset only the males and then view:

```
. infile ht wt str1 sex case using
http://staff.washington.edu/jyw/TA/height_weight.raw if sex=="M", clear

. list ht wt sex case
```

To obtain the first 10 observations and then view:

```
. infile ht wt str1 sex case using
http://staff.washington.edu/jyw/TA/height_weight.raw in 1/10, clear

. list ht wt sex case
```

Note, you can copy a previous command into the command window by clicking on it in the review window. You can also scroll through previous commands using the Page Up key.

To read in the whole dataset and then view:

```
. infile ht wt str1 sex case using
http://staff.washington.edu/jyw/TA/height_weight.raw, clear

. list ht wt sex case
```

To view individuals with a weight > 160:
```
. list ht wt sex case  if wt > 160
```

To view the last 10 observations:
```
. list ht wt sex case in 10/20
```

You could also download the dataset from
http://staff.washington.edu/jyw/TA/height_weight.raw and then use the following command:

```
. infile ht wt str1 sex case using "PATH\height_weight.raw", clear
```

Where PATH is the directory location of the file
(ie, `C:\Documents and Settings\c-16\Desktop`
or something similar)

2.4 Importing the Dataset using the menu

A short-cut would be to use the toolbar: File --> Import --> (select the type of data to import).

3.0 Viewing the Data

3.1 List the values of variables

```
        list [varlist] [if exp] [in range] [, options]
```
where options are

Options affecting style of list:
```
        table [clean]
        display
```

In the command prompt, the variables can be displayed in a table format. Certain displays of subsets of the data can be used with `[if exp]` and certain observations can be accessed by using `[in range]` options. See the above exercise for examples.

3.2 Viewing the Data in the Data browser window

You can also view the complete data from the browser window which can be accessed via the toolbar: Data --> Data browser (read-only editor)

4.0 Keeping or Dropping data

4.1.0 Eliminate variables from the working memory

```
drop varlist
```

4.1.1 Eliminate observations from the working memory

```
drop if exp

drop in range [if exp]
```

```
clear
```

4.2.0 Keep certain variables in the working memory

```
keep varlist
```

4.2.1 Subset the Data observations in the working memory

```
keep if exp

keep in range [if exp]
```

Stata works with data that is stored in its current memory. A list of these variables can be seen in the Variables window. You may decide to drop some variables or keep a subset of variables that may be defined by an expression: (**if** *exp*) and/or by observation id's: (**in** *range*).

5.0 Save and use datasets

To save a dataset as a Stata *.dta file:

```
save [filename] [, nolabel replace all orphans emptyok intercooled ]
```

To Write ASCII-format dataset:

```
outfile [varlist] using filename [if exp] [in range] [, comma
          dictionary nolabel noquote replace wide runtogether rjs fjs
          missing ]
```

To Write spreadsheet-style dataset:

```
outsheet [varlist] using filename [if exp] [in range] [, nonames
          nolabel noquote comma replace ]
```

To read in Stata datasets: *.dta files

```
use   filename  [, clear nolabel ]
```

Loading a specific subset of the data into memory for use:

```
use  [varlist]  [if exp] [in range] using filename [, clear nolabel ]
```

Exercise 5.0:

Save the complete data:

```
. save "C:\completedata.dta"
```

Output the data as an ASCII dataset:

```
. outfile ht wt sex case using "C:\completedata.txt"
```

Output the data as a spreadsheet-style comma-separated file:

```
. outsheet ht wt sex case using "C:\completedata.csv", comma
```

Drop the case variable for all observations:

```
. drop case
```

Keep all observations except the first one:
```
. keep in 2/20
```

View the current data:
```
. list ht wt sex
```

Options for other ways to view the current data:
```
.  list, table clean
.  list, display
```

Subset data based on height:
```
. keep if ht > 70
```

View the subset data:
```
. list wt ht sex
```

Save the subset data:
```
. save "C:\newsubsetdata.dta"
```

Drop all the variables: Note **_all** is a global keyword:
```
. drop _all
```

Read in the subset data and further using only the Male data:
```
. use wt sex ht if sex=="M" using "C:\newsubsetdata.dta"
```

Replace the old dataset with this male-only data:
```
. save "C:\newsubsetdata.dta", replace
```

Clear out the workspace:
```
. clear
```

Read in the complete data:
```
. use "C:\completedata.dta"
```

6.0 Documenting the variables

Labels allow you to document the variables with more detailed descriptions. Alternatively, the toolbar: Data → Labels & notes , could also have been used.

6.1 Label manipulation

To document a variable *(varname)* with a character string descriptive label `["label"]` use:

```
        label variable varname ["label"]
```

To define a descriptive label `(lblname )` and attach descriptions `("label")` to the categorical, numeric values (#) :

```
        label define   lblname # "label" [# "label" ...] [, add modify nofix]
```

To label the categorical, numeric values of a variable *(varname)* with a pre-defined descriptive label **(***lblname* **)**

```
        label values   varname [lblname] [, nofix ]
```

To view the labeling and frequency information for variable(s) [*varlist*]:

```
codebook [varlist] [, all notes mv tabulate(#) header problems detail]
```

Exercise 6.1:

Label the variables with a description :

```
.label variable ht "height (inches)"
.label variable wt "weight (lbs)"
```

Table an unlabeled version of the "case" variable to see how it looks before labeling

```
.  table case
```

Label the case variable with a description

```
.label variable case "Disease status (0=control, 1=case)"
```

Construct a label that describes disease status categories:

```
. label define status 1 "Case" 0 "Control"
```

Apply this labeling to the values of the variable, case:

```
. label values case status
```

Table the labeled version of the "case" variable to see how it looks after labeling

```
. table case
```

To view the labels on specific variables:

```
. codebook case ht wt
```

To describe contents of data in current memory:

```
. describe
```

To view value labels

```
. labelbook
```

6.2 Place notes in data

To add a note or documentation *(text)* for an entire dataset use :

```
        notes: text
```

To add a note or documentation *(text)* for any variable [*varname*] in a dataset :

```
        notes [varname]: text
```

Exercise 6.2:

Add a note to this entire dataset

```
. note: "This data shows the heights, weights, gender, and disease status of
20 people."
```

Add a more complete description to the ht variable:

```
. note ht : "It cannot be assumed that these height observations were made
for individuals of the same age."
```

To view all the notes (_dta refers to the working dataset):

```
. note
```

6.3 Setting missing values for Stata recognition

Change contents of variable

```
      replace oldvar = exp [if exp] [in range] [, nopromote ]
```

Exercise 6.3:

Changing the old code of -99 for height into a missing value (.) which is recognized by Stata

```
. replace ht  =  .  if ht = =-99
```

7.0 Cutting and Pasting results

Copy or highlight the item and left click the mouse to choose the "copy" option. Go to your Word document and paste the item by left clicking and clicking on the "paste" option.

From the Stata menu, the graphs can be copied over as an image or saved and then can be inserted into a document.

NOTE: It is highly discouraged to cut and paste ALL of your output into your homeworks. In your homework, you should present your results in the form of graphs and tables. We do not want to see *log or *do files as solutions to homework questions.

8.0 Just .do it

A do-file can be created by any text editor. Make sure you save the file as with a (.do) extension. In a do-file, each line beginning with an * or lines between /* ... */ are ignored by Stata. Therefore you can add comments within these asterisks.

8.1 Generating a do file from scratch

Stata has a built-in do file editor. From the toolbar: Window → Do-file Editor.

8.2 Generating a do-file from the Review window

Let's say you forgot about making a do-file. Left-click on the top part of the Review window. Then go to "Save Review Contents…". This approach will save all your commands that you typed into a do-file. However, don't always depend on this approach. There is a limit as to how far back in comments the Results window will display and thus save into the do-file.

Exercise 8.2:

Left-click on the top of the Review Window and save the commands into a do-file. Then go to the dofile editor and open your saved do-file.

8.3 Running commands in a do-file

Either **run** or **do** the do-file. The **run** method does not echo the commands; therefore, **do** is suggested. These commands can either be typed in the command window, or, alternatively from the menu of the Stata Do-file editor, highlight appropriate sections of the do-file and then press one of the two most right icons on the right to either **do** or **run** the highlighted sections of the file. If nothing is highlighted, Stata will **do** or **run** the whole file.

9.0 Exploring the Data and Descriptive Statistics

To make tables of summary statistics:

```
table rowvar [colvar [supercolvar]] [if exp] [in range] …
```

A related command is **tabulate**, which produces one- and two-way tables of frequency counts and row and column percentages, along with various measures of association, including the Pearson chi-squared, and Fisher's exact test.

```
tabulate varname1 varname2 [if exp] [in range] …
```

Tabulate and table work well for categorical variables. For continuous variables we will want to generate summary statistics, such as mean, standard deviation, minimum, maximum, etc. STATA will provide this information with the **summarize** command.

```
summarize [varlist] [weight] [if exp] [in range] [, [detail|meanonly]
```

If you are interested in a specific list of summary statistics, or if you want summary statistics for numeric variables conditioned on another variable you can use the **contents** option with **table**, or you can use the **tabstat** command.

```
tabstat varlist ,statistics(statname[...]) by(varname)
```

A list of allowable statistics can be found in the tabstat help window.

Exercise 9.0:

Look in the help menu to familiarize yourself with the options for table, tabulate, summarize and tabstat.

Try different commands included in the do file October5.do

```
/* specify the version */
version 10.x

* Comments
* can also be typed in
* this manner.
```

```
/* This do file can be found at
http://staff.washington.edu/jyw/TA/October5.do */
/* First do file for Biost/Epi 536 Intro to Stata, Session 2 */
/* created by Carolyn Hutter, October 2, 2004 */

/* close any open log file and open a new one.  Note, the capture command
allows the do file to proceed even if the line would produce an error.  In
this situation, it will close an open log if there is one, but will proceed
to the next step if no log was open */
capture log close
log using tables.log

/* In this do file we will explore the data using table, tabulate, summarize
and tabstat */

table case
table sex case
table ht sex case

tabulate sex
tabulate sex case
tabulate sex case, row column chi2

summarize wt
summarize wt, detail

table sex case, content(mean ht)

tabstat ht
tabstat ht, stat(mean sd p25 p50 p75)
tabstat ht, by(sex)
tabstat ht, by(sex) stat(mean sd min max n)
tabstat ht wt, by(sex) stat(mean sd min max n) columns(stat)

** Here we will promote good-bye habits by :
** (1) closing the log
log close
** (2) check the status of the log
log

** (3) saving the working data (ie, _dta) into the C:\ directory:
save "C:\day2data.dta", replace
```

You can run the entire file, or you can highlight specific commands and then press the **do** button in the editor. Alternatively press the **run** button.
Save the do-file in a location of your choice.

10.0 Graphics

Stata upgraded their graphics starting with Version 8. The best way to fully explore the new graphics options is to use the drop down menus. After using the menus to specify the graph, titles, symbols, etc. Stata then prints the command for the graph in the Results window. You can then copy that command into your do-file for future use.

10.1 Using graphics commands

Basic graphs can be generated from the command line. The **`plot`** command can be used to generate scatterplots, but **`graph`** is a more sophisticated command. Here we will focus on the **`graph`** command and leave the plot command for you to investigate further. The **`graph`** command is of the form:

```
        graph <graphtype> varlist, <options>
```

Although certain graph types (including histograms, do not start with the graph command). We will generate some basic graphs in the following exercise:

Exercise 10.1:

Create a scatter plot of ht and wt

```
. graph twoway scatter ht wt
```

Histogram of height

```
. hist ht, bin(5)
```

Bar chart of mean wt over case status

```
. graph bar wt, over(case)
```

Bar chart of median wt over case status with titles

```
. graph bar (median)wt , over(case) b1title (case status) ytitle (Median
weight in lbs) title (Median weight by case status)
```

Box plot of ht,with males and females on same graph

```
. graph box ht, over(sex)
```

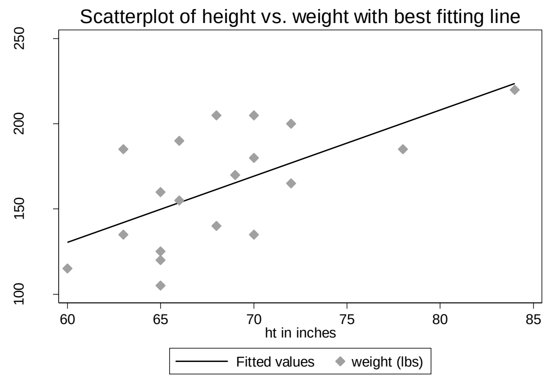
Box plot of ht, with males and females in separate panels

```
. graph box ht, by(sex)
```

Black and white scatter plot of height vs. weight with superimposed best fitting line, titles and x and y labels:

```
. graph twoway lfit wt ht || scatter wt ht, scheme(s1mono) title(Scatterplot
of height vs. weight with best fitting line) xtitle(ht in inches) ytitle(wt
in lbs)
```

10.2 Putting graphs in homeworks



There are two ways to put graphs in a word document.
1) Right click on the graph, copy, and then paste in the document
2) Right click on the graph, save (as a *.wmf), and then insert as a picture.

It is recommended that you use the "scheme(s1mono)" option to create black-and-white graphs for homeworks.
It is also recommended that you resize the graph,

and then double-click on the graph in word, select "layout" and then choose square, to fit the text around the graph.

Part B

11.1 Review of Part A

11.1 Accessing Stata

In the Health Sciences 3rd floor microcomputer lab, Stata 10.x should be installed on all the computers. To access Stata 10.x click on the following buttons:

START --> Programs --> Stata 10 --> StataIC 10

11.2 Getting Help from the menu bar

Stata also has an on-line help system that can be accessed by choosing Help from the menu bar. When you click on Help you get the following submenu

Contents	See the help table of contents
Search…	Search for help on a particular topic
Stata Command…	Get help for a particular Stata command

11.3 Version control

Stata is continuously being updated; therefore, in order to be sure that you are using Stata version 10.x, get into the habit of specifying the version before you begin your Stata session.

Exercise 11.3:

```
version 10.x
```

11.4 Creating a log file: Echo copy of session to file or device

Exercise 11.4:

Create a log file called test.smcl in the C:\ directory:
```
log using "C:\test.smcl"
```

Check the status of logging:
```
log
```

Ending the recording session; You should get into the habit of closing your logs:
```
log close
```

Creating a new log file that is text:
```
log using "C:\Intro_to_stata_day2.txt", text
```

11.5 Opening a STATA dataset

.

Exercise 11.5:

```
  use “http://staff.washington.edu/jyw/TA/height_weight_week2.dta”, clear
```

11.6 Viewing the data

Exercise 11.6:

View the current data:

```
  list
```

View the labels in the current dataset

```
  codebook
  describe
  labelbook
```

See if there are any notes in the current dataset

```
  note
```

11.7 Using do-files

Stata has a built-in do file editor. From the toolbar : Window → Do-file Editor.

Exercise 11.7:

Download the do-file http://staff.washington.edu/jyw/TA/October12_1.do and open the file in the do-file editor.

12.0 Subsets and relational opperators

12.1 Using the by command

The **by** *varlist*: option may be used with a number of commands. This option repeats the command groups of observations specified by the variable list.

The dataset must be sorted before you can use the by command. However, recent versions of STATA allow you to use **bysort** (abbreviated bys), which will both sort and subset the data.

If you want to sort in descending (rather than ascending order) use the **gsort** command with a – in front of the variable.

Exercise 12.1:

Read in the data

```
   use “http://staff.washington.edu/jyw/TA/height_weight_week2.dta”
```

Summarize height separately for males and females

```
  sort sex
  by sex: summarize ht
```

Find the mean, median and 90th percentile of weight of men and women separated by cases/control status

```
  bys case: tabstat wt, stat(mean, p50, p90) by(sex)
```

Sort height in descending order (from tallest to shortest)

```
  gsort -ht
  list ht
```

12.2 Using the if and in commands

We introduced the **if** and **in** options last week when creating subsets of the data. We can also use these options to run the command on a subset of the data.

12.2.1 ***Relational Operators***

The **if** option uses logical expressions which are created with the following relational operators:

```
== is equal to (double equal sign)
~= not equal to (can also use !=)
>  is greater than
<  is less than
>= is greater than or equal to

<= is less than or equal to
&  specifies AND
|  specifies OR
~  specifies NOT
```

You can create simple logical expressions, such as

```
ht < 72
```

or more complex expressions such as

```
(ht < 72 & sex==”M”) | (ht<68 & sex==”F”)
```

For complicated expressions it is good to use parentheses to make sure expressions are evaluated in the order that you want. Also, we have to put the M and F in quotes because sex is a string variable

12.2.2 ***Missing values in logical expressions***

It is important to consider missing values when using relational operators! STATA codes missing values as a . which is equivalent to a very large number.

(an aside on missing values: Starting with Version 8 of Stata, you can specify up to 27 different types of missing values. They are: ".", ".a", ".b", ... ,".z". (During data entry, you can use these to differentiate among Refused, Not Applicable, Don't Know, and other possible reasons for missing values.) These are the largest values allowed by the data type, so you can use "<." to exclude all 27 missing values for a variable.)

(a second aside on missing values: For string variables, indicate missing values with empty quotations`“”,` rather than .)

12.2.3 *The in option*

The **in** qualifier specifies a range of values for the command. The **in** option is based on the order of the variables in storage, and is sensitive to the way in which the data is sorted.

Exercise 12.2.3:

Summarize height for males
```
  summarize ht if sex==”M”
```

Determine what percentage of cases are female/male.
```
  tabulate sex if case==1
```

Summarize weight for individuals 66 inches or taller
```
  summarize wt if ht >=66 & ht~=.
```

Summarize weight of the 5 shortest people
```
  sort ht
  summarize wt in 1/5
```

13.0 Generating and Changing Variables

13.1 Generate

Generate creates a new variable. The values of the variable are specified by = exp. You can find out more about allowable expressions by looking up “functions” in the help menu.

```
        generate [type] newvar[:lblname] = exp [if exp] [in range]...
```

13.2 Replace

Replace changes the contents of an existing variable.

```
        replace oldvar = exp [if exp] [in range]...
```

13.3 Recode

Recode changes the values of a numeric variable and is often used to code variables into catergories

```
       recode varlist (rule) [(rule) ...] [, generate(newvarlist) ]
```

13.4 Encode

Encode changes a string variable into a numeric variable

```
        encode varname [if exp] [in range], generate(newvar)...
```

13.5 Egen

Egen is an extension of generate, and allows for more complicated functions

Exercise 13.5:

Look in the help menu to familiarize yourself with the options for generate, replace, recode, encode and egen

Try different commands included in the do-file October12_1.do

```
/* Specify the correct version */
version 10.x

/* This do file can be found at
http://staff.washington.edu/jyw/TA/October12_1.do */
/* First do file for Biost/Epi 536 Intro to Stata, Session 2 */
/* created by Carolyn Hutter, October 2, 2004 */

/*  This do file should be used with the dataset height_weight_week2.dta */
use "http://staff.washington.edu/jyw/TA/height_weight_week2.dta"

/* In this do file we will generate new variables for
metric versions of height and weight and for bmi.  We will also create
variables for height, weight and bmi categories and we will recode the
sex variable */

/* open new log file*/
capture log close
log using newvariables.log

/* first we will convert weight in pounds to weight in kg. */

capture drop wtmetric
/* Capture intercepts an error message and allow the do-file
to continue. In this case we will drop wtmetric if it is already in the
data set, but will proceed without interruption if it is not.  This is a
good trick to use if you are generating variables in a dofile */

generate wtmetric = wt*0.45
label variable wtmetric "weight (kg)"
summarize wtmetric wt

/* next we will convert height in inches to height in meters, for the sake
of example I have abbreviated the STATA commands */

cap drop htmetric
g htmetric = ht*0.0254
la variable htmetric "height (meters)"
su htmetric ht

/* finally we will generate a variable for bmi */

capture drop bmi
```

```
gen bmi= wtm/(htm^2)
lab variable bmi "bmi"
sum bmi

/* Silly example of creating a variable "tall"*/

capture drop tall
gen tall=0
replace tall=1 if (sex=="F" & ht >=68) | (sex=="M" & ht >=72)

/*Note: we had to put "F" and "M" in quotes, since sex is a string variable*/

/* Question:  What happened to the individual who was missing data for ht? */
/* Two options to deal with the missing value
1) we can replace the person now */

replace tall=. if ht==.

/* or 2) we can pay attention to missing when generating our variables */

capture drop tall
gen tall=0
replace tall=1 if (sex=="F" & ht >=68 & ht <.) | (sex=="M" & ht >=72 & ht <.)
capture label drop tall
label define tall 0 short 1 tall
label values tall tall
bys sex: tabstat ht, by(tall) stat(min max n)
note tall: this is an indicator variable that arbitrarily sets males as
"tall" if they are 72 inches or taller, and women as "tall" if they are 68
inches or taller

/* Generating bmi categories */
capture drop bmicategory
generate bmicategory=bmi
recode bmicategory (min/18.5=0) (18.5/25=1) (25/30=2) (30/max=3)
label variable bmicategory "BMI categories"
capture label drop bmicat
label define bmicat 0 "underweight" 1 "normal" 2 "overweight" 3 "obese"
label values bmicategory bmicat
tabulate bmi bmicategory, missing

/* Next we will categorize weight into quartiles */
tabstat wtmetric, stats(p25, p50, p75)
capture drop wtcategory
gen wtcategory=(wtmet>58.5)+(wtmet>73.125)+(wtmet>84.375)
/* note, the above is a clever way to create categories, but be
careful if you have missing values!*/
bys wtcategory: summarize wt

/* A second "quick and dirty" way to do this is to just sort the variable
and then create four groups.  This method is not recommended if a lot of
people have the same value.  Why? */
sort wtmetric
capture drop wtgroup
gen wtgroup=group(4)
tabulate wtcat wtgroup, missing

/* An example of encode, creating a numeric variable for sex */
capture drop gender
```

```
encode sex, generate(gender)
tab sex gender
sum sex gender
tab gender if gender==1

/* An example of an egen command.  Creating groups formed by case, sex and
bmicategory */

capture drop groups
egen groups = group(case sex bmicat)
bys groups: list case sex bmicat

log close
save "C:\day2data.dta", replace
```

14.0 Epidemiology Tables

Since this is a class on epidemiology methods, we will often be interested in evaluating the association between and exposure and an outcome. Stata has a set of commands specifically designed to do this. They are located under "epitab" (short for epidemiology tables)

14.1 epitab commands

There are several epitab commands. We will focus on the following three commands:

cs (short for cohort study): gives relative risks, risk differences, attributable risk (percent) and population attributable risk (percent).
cc (short for case-control): gives odds ratios
mhodds (short for Mantel-Haenszel estimate of the odds ratio): gives odds ratio adjusting for a confounder

These commands all require that the exposure and outcome be coded as 0, 1 where 0= not exposed/not diseased and 1= exposed/diseased. The confounding variable does not have to be coded 0, 1, but it does need to be categorical.

Exercise 14.0:

Look in the help menu to familiarize yourself with the options for epitab
Try different commands included in the do file October12_2.do

```
/* specify the version */
version 10.x

/* This do file can be found at
http://staff.washington.edu/jyw/TA/October12_2.do */
/* Second do file for Biost/Epi 536 Intro to Stata, Session 2 */
/* created by Carolyn Hutter, October 2, 2004 */

/* This do-file uses variables generated in October12_1.do */

/* In this do file we will expore the "epitab" functions */
/* make sure there is not an open log file */
capture log close

/* open a log file */
log using epitab.log

/* We will explore if sex is associated with case status in our data set */
/* For epitab tables we need to have both exposure and outcome coded as
indicator variables */
/* To see what happens when variables are not coded properly */
cs case gender

/* let's make our variable for gender have 0=male, 1=female) */
/* first I regenerate the gender variable */
capture drop gender
encode sex, gen(gender)

/* next I double check how gender is coded */
sum gender
/* I now see it is coded as 1s and 2s */
tab gender if gender==1
/* I now see that female is coded 1 */
recode gender 1=1 2=0
capture label drop female
label define female 0 male 1 female
label value gender female
tab gender sex

/* explore cs, cc and mhodds commands*/

cs case gender
cc case gender
cc case gender, by(bmicategory)
mhodds case gender bmicategory

/* note the cc case gender, by(bmicategory) and mhodds did not work well
because the categories were too sparce, I will create a new dichotomous
variable for bmi */

capture drop bmicategory2
gen bmicategory2=bmicategory
recode bmicategory2 0/1=0 2/3=1
capture label drop overweight
label define overweight 0 "underweight or normal" 1 "overweight or obsese"
label value bmicategory2 overweight
tab bmicategory bmicategory2

/* now look at options for "adjusting" for bmi: */
cs case gender, by(bmicategory2)
cc case gender, by(bmicategory2)
mhodds case gender bmicategory2


** Here we will promote good-bye habits by :
** (1) closing the log
log close
** (2) check the status of the log
log
```

```
** (3) saving the working data (ie, _dta) into the C:\ directory:
save "C:\day2data.dta", replace
```

14.2 intermediate commands

The **cc** and **cs** commands also have "intermediate" versions **cci** and **csi**. This is what you would use if you didn't have a dataset, but wanted to evaluate data presented in a table:

		Exposure	
		Yes	No
Disease	Yes	a	b
	No	c	d

		Exposure	
		Yes	No
Disease	Yes	65	25
	No	75	75

It is important to pay attention to the orientation of the table. Stata makes tables with exposure status in columns and disease status in rows (whereas a number of epidemiology text books, including Koepsell and Weiss put exposure status in rows and disease status in columns).

Example 14.2:

To find the odds ratio from the above table:
```
csi 65 25 35 75
```

To find the relative risk from the above table:
```
cci 65 25 35 75
```

Another note on homeworks:
We want to stress that simply giving us Stata output is NOT acceptable "good faith" effort for a homework assignment. For example, if your homework assignment was:

Q: Assume the above table is from a cohort study. What is the relationship between exposure and disease?

An unacceptable homework answer would be:
```
. csi 65 25 35 75

                 |   Exposed   Unexposed  |      Total
-----------------+------------------------+-----------
           Cases |        65          25  |         90
        Noncases |        35          75  |        110
-----------------+------------------------+-----------
           Total |       100         100  |        200
                 |                        |
            Risk |       .65         .25  |        .45
                 |                        |
                 |      Point estimate    |    [95% Conf. Interval]
                 |------------------------+------------------------
 Risk difference |                    .4  |   .2737382    .5262618
      Risk ratio |                   2.6  |   1.798273    3.759163
 Attr. frac. ex. |              .6153846  |   .4439108    .7339833
 Attr. frac. pop |              .4444444  |
                 +-----------------------------------------------
                             chi2(1) =    32.32  Pr>chi2 = 0.0000
```

Another unacceptable homework answer would be:

```
      Risk ratio = 2.6
```

An acceptable homework answer would be:

Based on the above table, the relative risk for an exposed person, compared to an unexposed person, is estimated as 2.6 (95% CI: 1.80-3.76).

There are three situations where Stata output is acceptable in a homework:
1) The homework specifically asks for Stata output.
2) You have formatted the Stata output to give you the answer (for example with the **tabstat** or **ci** commands)
3) You were not sure how to do a problem, so you are including ***some*** Stata output to demonstrate your thought process.
Example:
To investigate the relationship between exposure and disease, I used the command:

```
csi 65 25 35 75
```

I thought about using the cc command, but then realized that is not appropriate since this is a cohort study. Stata gave several measures of the association between exposure and disease:

```
 Risk difference |               .4       |  .2737382    .5262618
      Risk ratio |              2.6       |  1.798273    3.759163
 Attr. frac. ex. |         .6153846       |  .4439108    .7339833
 Attr. frac. pop |         .4444444       |
```

I am pretty sure that the risk ratio of 2.6 can be used as an estimate of relative risk, as discussed in class on October 5th, but I am not sure if that is the risk measure that you were looking for in this question.

15.0 One and two sample binomial inference, and confidence intervals

Stata can be used to do one and two sample binomial inference, and to calculate confidence intervals. Similar to the epitab commands, there are "intermediate" forms of each of these tasks.

15.1 One sample binomial inference

bitest performs *exact hypothesis tests* for binomial random variables. The null hypothesis is that the probability of a success on a single trial is #p.

bitest *varname* == *#p* [*weight*] [**if** *exp*] [**in** *range*] [, **detail**]
bitesti *#N #succ #p* [, **detail**]

15.2 Two sample binomial inference

There are several methods for doing two sample binomial inference in Stata. I tend to prefer to use the **chi2** and/or **exact** options for the **tabulate** command.

15.3 Confidence intervals

ci computes standard errors and confidence intervals for a variable. You need to specify if the variable is binomially distributed.

Example 15.3:

Is our dataset likely to have been selected from a population that is 75% female?

```
  bitest gender=.75
```

You flip a coin 100 times and observe 75 tails. Do you think the coin is a fair coin?

```
  bitesti 100 75 .5
```

Based on our dataset, does the proportion of females who are cases differ from the proportion of males who are cases?

```
  tabulate case gender, exact chi2
```

(Silly example): If our dataset represents a random sample from the population, does the proportion of people who are cases differ from the proportion of people who are female?

```
  prtest case=gender
```

What is the estimate and 95% confidence interval for the frequency of cases?

```
  ci case, binomial
```

What is the estimate and 95% confidence interval for mean height (in inches)?

```
  ci ht
```

You flip a coin 100 times and observe 75 tails. What is the 90% CI for the probability that the coin will be tails?

```
  cii 100 75, level(90)
```

16.0 Dummy Variables

In this class we will often want to use dummy variables (indicator variables for each value of a categorical variable). STATA has a number of ways to create and use dummy variables.

16.1 Generating dummy variables from scratch

If you have a small number of categories, you can just generate a new series of indicator variables, but this can be time consuming and is prone to mistakes.

16.2 Using tabulate, gen

A better way to generate dummy variables from an existing categorical variable is to use the generate option after tabulate. This will create an indicator variable for each row of the table.

Exercise 16.2:

Use tabulate to generate a new categorical variable

```
 tabulate bmicategory, generate (bmicat)
```

View the new variables that were generated by the above command (note what bmicat* and bmicat$ give you. "*****" and "**$**" are recognized as "wildcards" by Stata)

```
  sort bmicategory
  list bmicategory bmicat$,  clean
  describe bmicat*
```

16.3 Using xi

You can also use **xi** as a prefix for a number of commands (including **logistic**) to expand a categorical variable into dummy variables within the command.

For the xi command, the default in STATA is to use the lowest value of the categorical variable as the reference group. You can change what value is selected as the reference using the **char** option.

Exercise 16.3:

Use xi to do logistic regression for case and dummy variables for bmi category

```
  xi: logistic case i.bmicategory
```

Change so "normal weight" is reference for bmi category (note you need to use hard parentheses [] around omit)

```
  char bmicategory[omit] 1
```

Repeat logistic regression for case and dummy variables for bmi category

```
  xi: logistic case i.bmicategory
```

17.0 Links for other (more detailed) STATA tutorials

There are a number of resources for STATA on the web. Below are links for resources that we find particularly useful. We should note that we used these resources as references in putting together the materials for this workshop.

Stata tutorials
http://www.stata.com/links/resources1.html

On-line tutorial on University of North Carolina Web Site ** Highly Recommended **
http://www.cpc.unc.edu/services/computer/presentations/statatutorial

Notes from UCLA Academic Technology services
http://www.ats.ucla.edu/stat/stata/notes3/