You might think that our job is done, but there is still one very important thing to do, and that is to label certain values. If you open the data editor, you can see that the variable id takes on values such as 1, 2, 3, 4, etc… This makes sense. You can also see that the variable *gpa* takes on a variety of values which we can easily interpret. But what does it mean when gender is 2? Scrolling through the dataset, it becomes clear that this variable is either a 1 or a 2. Actually, a 1 means a female and a 2 means a male. But if this is the case, why did the person who entered the data not type "female" or "male", instead of typing "1" or "2"? The reason is that it is simply more efficient to type in the numbers. It takes up less space and is faster, especially when you are entering a dataset that might end up containing thousands of observations. An even more important reason is that Stata, like all computer programs, is better at dealing with numbers. Because Stata likes numbers while we like text, Stata allows us to use both at the same time. Data can be entered as numbers but can be visualized as text. This is where value labelling comes in. We can create value labels that tell Stata what a "1" means and what a "2" means. This is a two-step process. In the first step, we need to create the label, and in the second step we need to assign it to a variable. In order to create the label, we need to use the **label define** command:

help label

Looking at the help file, we can see that to define a label, we need to use the command **label define**, follow it by the name of the label, and then specify the label for each individual value. This is best understood using an example. Let us create a label that will define the value "1" as being "female" and the value "2" as being "male":

label define gender 1 "Female" 2 "Male"

We have defined a label that we named *gender*. Note that Stata allows us to use the same name for variables and for labels. We cannot have two variables with the same name, but we can have a variable and a label share the same name. Looking at the command again, we see that in the newly created gender label, we are instructing Stata that a "1" means "Female" while a "2" means "Male". Creating the label however is only half the story. We now need to tell Stata to apply it to a variable. This is accomplished using the "label values" command. Looking at the help file again, we see that we need to type **label values** followed by the *varlist* followed by the label name. Here, *varlist* stands for a list of variables. This is because Stata allows us to label more than one variable with the same label. Assume that you distributed a survey and that several questions were in the form yes/no. Assume that you then enter the yes responses as 0 and the no responses as 1. Stata will allow you to apply the same label to more than one variable. In our case, we just have one variable to which we plan to apply the label, so let's do that:

label values gender gender

The first gender is the name of the variable to which we want to apply the label and the second variable is the name of the label which we wish to apply. Open the data editor now and look at the values under the gender column. We see that they have all been labelled appropriately. One interesting thing that you notice is that the text is blue, unlike the text for other string variables. This is because Stata is telling you that although you see string values, these values are only the

labels and not the actual data. So when you see blue text in the editor, you should know that a label has been assigned to this variable. When you see red text, you should know that the actual values stored in the variable are text.

Run the command **describe** now. You will see in the output that Stata is telling you that the variable gender has a label applied to it, and that the name of the label is also gender. If you want more information about this label, run the following command:

label list gender

This command tells Stata to list information about the label which is named gender. Looking at the output, we can clearly see the code used in this label.

Now that we have a well labelled dataset, it would make sense that we save our work. If you want to save the dataset by overwriting the original version, just type:

save, replace

If however you want to save the dataset under a new name in order to retain the older version, then you can type:

save datasetnew