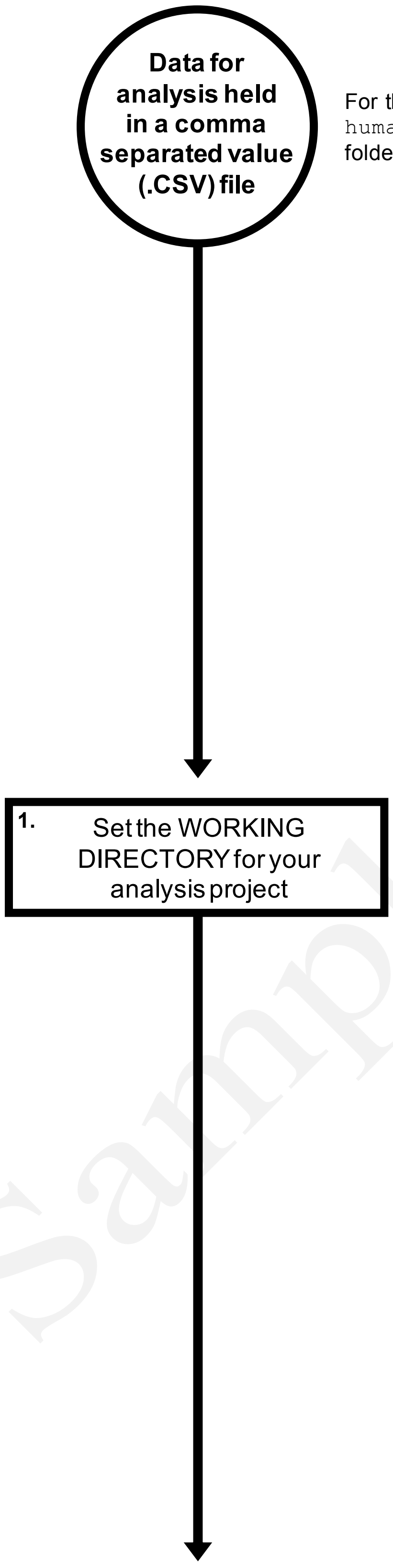
EXERCISE 3.1: HOW TO ASSESS WHETHER A BIOLOGICAL DATA SET HAS A NORMAL DISTRBUTION USING R:

One of the main factors that will determine which statistical tests you can apply to a specific data set is whether or not the data being tested have a normal distribution. While you can get an idea of whether the data in a particular data set is likely to have a non-normal distribution by plotting a frequency distribution histogram (see Exercise 2.1) such assessments are subjective. As a result, it is important that you also apply an objective test to assess whether there really is a significant difference from normality.

There are three tests that are commonly used by biologists for doing this. These are the Shapiro-Wilk test, the Anderson-Darling test and the Kolmogorov-Smirnov test. Of these, the Shapiro-Wilk test is generally considered the most powerful, but it is also the one that is most sensitive to the proportion of tied or non-unique values within the data being examined. The Kolmogorov-Smirnov test is generally considered to be the least powerful, and is no longer recommended by many statisticians as a normality test for biological data. It does, however, allow you compare your data not just to a normal distribution, but to any distribution you wish to specify, and as a result is still retains enough usefulness for biologists to be included here.

Regardless of the test you wish to use, you need to have your data arranged in a spreadsheet or table where each row contains data from a single record in your data set. In this table, there also needs to be a column which contains values for the continuous variables you wish to test for normality. For this exercise, you will test whether or not two variables in a data set from a small sample of human males have distributions that differ significantly from normal. You will start by using a Shapiro-Wilk test to assess whether data on the body mass from these men have a normal distribution. To do this, work through the flow diagram that starts on the next page.


Data for analysis held in a comma separated value (.CSV) file
1.
Set the WORKING DIRECTORY for your analysis project

For this example, the data set you will use is stored in a file called `human_data.csv` that is located in the WORKING DIRECTORY folder you created during the introduction to this chapter.

Before you start any analysis in R, you first need to set the WORKING DIRECTORY. To do this, enter the text `setwd("` and then type the address of your WORKING DIRECTORY, using slashes (/) as the folder separators, before entering a second quotation mark followed by a closing bracket, like this `")`. For example, if your WORKING DIRECTORY has the address C:\STATS_FOR_BIOLOGISTS_ONE, your `setwd` command should look like this:

```
setwd("C:/STATS_FOR_BIOLOGISTS_ONE")
```

If you are using RGUI, enter your `setwd` command in the R CONSOLE window (remembering to use the address of your own WORKING DIRECTORY folder in it) and then press the ENTER key on your keyboard. If you are using RStudio, enter your `setwd` command into the SCRIPT EDITOR window. To run it, select it and then click on the RUN button at the top of this window. You will enter all the remaining commands for this exercise in a similar manner, depending on the user interface you are using.

To check that your WORKING DIRECTORY has been set properly, enter the command `getwd()` and carefully check that the address it returns is the same as the one for the STATS_FOR_BIOLOGISTS_ONE folder you created at the start of this chapter.

Before you move on to step 2, make sure that all the data you wish to use in your analysis project are located in this WORKING DIRECTORY folder. In this case, this is a file called `human_data.csv`. **NOTE:** If the data you are going to import into R in step 2 are not located in the WORKING DIRECTORY you set in this step, the import code provided in the next step will not work.

The `read.table` command provides the easiest way to import data held in a .CSV file into R so you can analyse it. To do this, you will use the following command:

```
human_data <- read.table(file="human_
  data.csv", sep=",", header=TRUE)
```

This code has to be entered exactly as it is written here or it will not work. If you wish to use the copy-and-paste approach for entering this command, copy the text directly below CODE BLOCK 55 in the document R_CODE_BASIC_STATS_WORKBOOK.DOC and paste it into R.

2. Load your data into R using the `read.table` command

This command will create a new object in R called `human_data` which will contain the data from the specified .CSV file. To load a different .CSV file into R, all you need to do is change the file name in the `file` argument to the name of the one you wish to import. In addition, you can use whatever name you wish for the object which will be created by this command. To do this, simply replace `human_data` at the start of the first line of the above code with the name you wish to use for it. **<u>NOTE</u>:** If your .CSV data set uses a semicolon as the decimal separator, you would need to replace the code `sep=","` with the code `sep=";"`.

Whenever you import any data into R, you need to check that they have loaded correctly. First, you need to check that all the required columns are present in the R object you just created. To do this, enter the following command into R:

```
names(human_data)
```

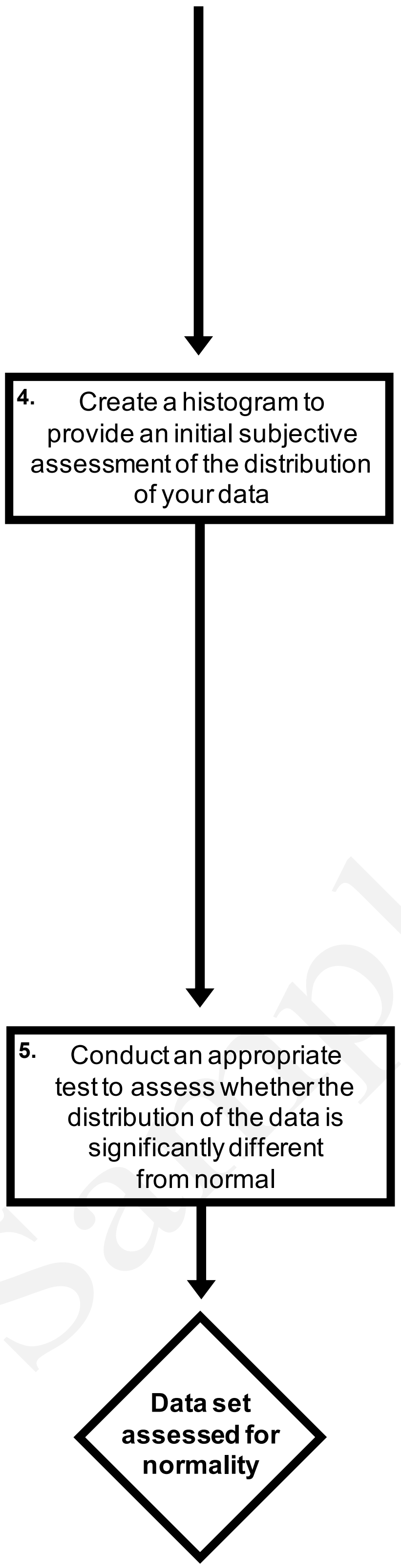
This is CODE BLOCK 56 in the document R_CODE_BASIC_STATS_WORKBOOK.DOC. This command will return the names used for each column in the R object created in step 2. For this example, the names should be `id`, `mass`, `height` and `no_offspring`.

3. Check the data have loaded into R correctly by checking the names of the columns and by viewing it

Next, you should view the contents of the whole table using the `View` command. This is done by entering the following code into R:

```
View(human_data)
```

This is CODE BLOCK 57 in the document R_CODE_BASIC_STATS_WORKBOOK.DOC. This command will open a DATA VIEWER window where you can examine your data set and check that the correct data have been loaded into R.


4. Create a histogram to provide an initial subjective assessment of the distribution of your data
5. Conduct an appropriate test to assess whether the distribution of the data is significantly different from normal
Data set assessed for normality

Once your data have been successfully imported into R, you are ready to start assessing whether or not it has a distribution that differs significantly from normal. The first step in this process is to plot a frequency distribution histogram that you can examine to provide an initial subjective assessment of the distribution of your data. To do this, enter the following command into R:

```
hist(human_data$mass, nclass=7)
```

This is CODE BLOCK 58 in the document R_CODE_ BASIC_STATS_WORKBOOK.DOC. This `hist` command creates a frequency distribution histogram from the data in the column called `mass` in the R object called `human_data` created in step 2 of this exercise. The term `nclass=7` in this command means that the data will be divided into approximately seven equal-sized classes to make this histogram (see Exercise 2.1 for more details). For your own data sets, you may wish to use a different value for the `nclass` argument.

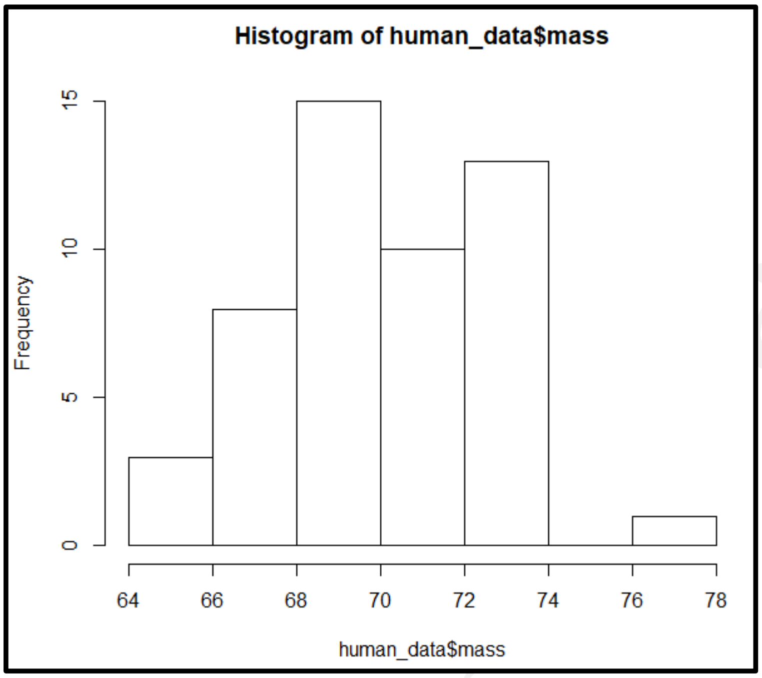
Once you have created your frequency distribution histogram, you can examine it to see whether it looks like it may be normal, or if it is clearly non-normal (see figure 1). This histogram can also be used to assess whether your data violate the assumptions of any particular normality test (such as the proportion of data points with identical values). If, based on your histogram, you suspect that your data may violate the assumptions of a particular normality test, you can use the `View` command from step 3 to allow you to explore your data in more detail.

Once you have conduced a subjective assessment of the distribution of your data, and you have ensured that they do not violate the requirements of the normality test you wish to use, you are ready to apply it to your data set. In this example, you will apply a Shapiro-Wilk test to assess whether or not the data on body mass from a sample of human males differs significantly from normal. To do this, enter the following command into R:

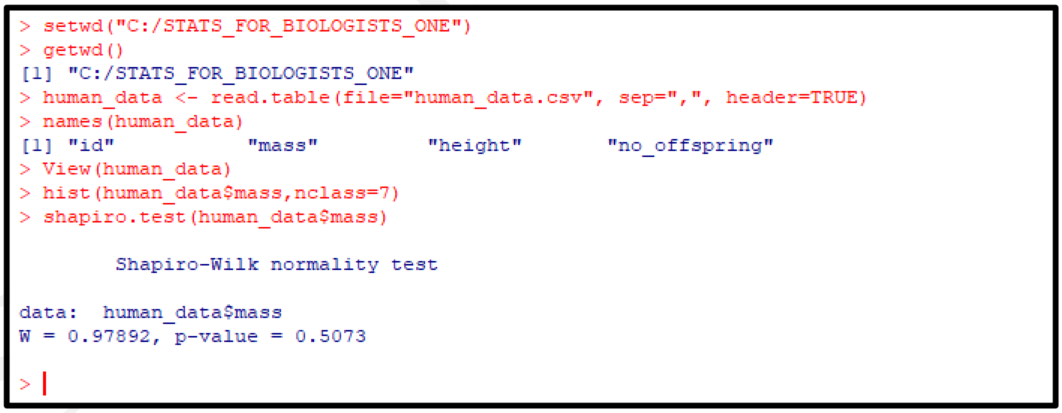
```
shapiro.test(human_data$mass)
```

This is CODE BLOCK 59 in the document R_CODE_ BASIC_STATS_WORKBOOK.DOC. This code applies the Shapiro-Wilk test to data contained in the column called `mass` in the R object called `human_data` created in step 2 of this exercise.

At the end of the first part of this exercise, the frequency distribution histogram created in step 4 should look like this:



While the contents of your R CONSOLE window should look like this:


```
> setwd("C:/STATS_FOR_BIOLOGISTS_ONE")
> getwd()
[1] "C:/STATS_FOR_BIOLOGISTS_ONE"
> human_data <- read.table(file="human_data.csv", sep=",", header=TRUE)
> names(human_data)
[1] "id"           "mass"          "height"        "no_offspring"
> View(human_data)
> hist(human_data$mass,nclass=7)
> shapiro.test(human_data$mass)

        Shapiro-Wilk normality test

data:  human_data$mass
W = 0.97892, p-value = 0.5073

> 
```

If you examine the results of this Shapiro-Wilk test, you will see that the p-value for this test is 0.5073. This means there is no significant difference between the distribution of the body mass data from this sample of human males and a normal distribution. As a result, it would be appropriate to apply a parametric test, such as a t-test, to these data. To report the results of a Shapiro-Wilk test in a manuscript, you need to provide the value of the test statistic (W), the associated p-value (p), and the sample size (n). For the above example, you could report it as follows (with all values rounded to an appropriate number of figures – see Appendix III for details):

There was no significant difference between the distribution of the data on body mass from a sample of male humans and a normal distribution (Shapiro-Wilk Test: W=0.98; p=0.507; n=50).

When assessing the distribution of your data for normality using the above workflow, you can use any suitable normality test in step 5 as long as your data do not violate its assumptions. Information on the three normality tests most commonly used by biologists, along with the commands you can use to run them in R, are provided in the table below.

Normality Test	**How To Conduct It In R**
Shapiro-Wilk Test	To conduct a Shapiro-Wilk test, you can use the `shapiro.test` command. For this command, you need to define the column and the R object that contains the data for the continuous variable you wish to test for normality. For example, to run this test on the contents of a column called `mass` in an R object called `human_data`, the command would be: `shapiro.test(human_data$mass)` If you need to refine your Shapiro-Wilk test, there are a number of additional arguments that can be used with this command. Details of these additional arguments can be found at *www.rdocumentation.org/ packages/stats/versions/3.6.1/topics/shapiro.test*.
Anderson-Darling Test	To conduct an Anderson-Darling test, you can use the `ad.test` command. This command is contained in the `goftest` package, and you will need to install this package and library before you can use it (see page 114 for more details). To use this command, you need to define the column and the R object that contains the data for the continuous variable you wish to test for normality. For example, to run this test on the contents of a column called `mass` in an R object called `human_data`, the full command would be: `ad.test(human_data$mass)` If you need to refine your Anderson-Darling test, there are a number of additional arguments that can be used with this command. Details of these additional arguments can be found at *www.rdocumentation.org/packages/goftest/versions/1.2-2/topics/ad.test*.

Normality Test	**How To Conduct It In R**
Kolmogorov-Smirnov (K.S.) Test	To conduct a Kolmogorov-Smirnov test, you can use the `ks.test` command. To use this command, you need to define the column and the R object that contains the data for the continuous variable you wish to test for normality. For example, to test whether contents of a column called `mass` in an R object called `human_data` differs significantly from a normal distribution (set by the argument `pnorm`), the full command would be: `ks.test(human_data$mass, pnorm)` If you need to refine your Kolmogorov-Smirnov test, there are a number of additional arguments that can be used with this command. Details of these additional arguments can be found at *www.rdocumentation.org/packages/stats/versions/3.6.1/ topics/ks.test*.

For the next part of this exercise, you will customise the above approach for conducting normality tests in a number of different ways. Firstly, you will run the same Shapiro-Wilk test on a different column of data in the same `human_data` R object. This column is called `no_offspring` and it contains data on the number of children that each male fathered in their lifetime. To run the second version of this test, you will first need to modify the code used in step 4 to create a frequency distribution histogram based on a different column. The modified code should look like this (required modifications are highlighted in **bold**):

```
hist(human_data$no_offspring, nclass=7)
```

You can modify this code either by editing it in the R CONSOLE window of RGUI or through the SCRIPT EDITOR window of RStudio (depending on which interface you are using). If you are entering commands directly into the R CONSOLE window, you can use the UP arrow on your keyboard to bring commands you have previously run during the same session back on to the command line of this window, and then use the LEFT and RIGHT arrows to scroll through and edit them. In this case, use the UP arrow to bring the previous version of the `hist` command back onto the command line and edit it so that it looks like the one above. Once you have finished modifying your `hist` command, you can run it by pressing the ENTER key on your keyboard. If you are using RStudio, you can copy and paste the original `hist` command in the SCRIPT EDITOR window before editing the new version to include the required modifications. Once you have done this, you can select the modified version of the command and click on the RUN button to run it in the R CONSOLE window.