1. a) In class we discussed using the `ls()` function to list objects that we have created in our workspace. We also created a number of objects during the course of the lecture. Use the `ls()` command to create a list of the objects in your R workspace.
   Solutions:

   ```
   > ls()
   [1] "x" "y" "z"
   ```

   b) We also discussed how to use the `rm()` command to remove objects from your workspace. In part a) above, you should have found a variable named `z`. Try removing `z`.
   Solutions:

   ```
   >rm(z)
   ```

   c) What happens now if you try to look at `z` by typing z in the R console? What does R print?
   Solutions:

   ```
   > z
   Error: object "z" not found
   ```

   d) We discussed different ways of looking through previous commands to reuse them. Using one of these methods find the the command that you initially used to create `z`, and then re-create it. Try doing the same thing, but modify the variable name to call the object `newZ`
   Solutions:

   Note: There is more than one possible solution to this part of the question, although only one solution is presented.

   ```
   > history()
   ```

   This opens a text file where you will find the command `z=10`. You can then edit the text file such that `newZ=10`, and then copy it into the R console.

2. This question will require you to open a script file that I have written for you called 'Exercises1.R'. Open this script file to complete the following exercises.

   a) Using either copy and paste, or highlighting and pressing ctrl-r, run the lines of code one-by-one to see what they do. Eventually you should see both of the graphs that were demonstrated earlier in the presentation.

   Solutions:
   This will either work or it won't...

   b) I have used a new command in this script file called `attach()`. Using R help, try to discover what the `attach()` function does. There is more than one way to access the help, so use whichever method appeals to you. Don't worry if you find the R help description a little cryptic, this is normal!
   Solutions:

   ```
   > ?attach
   or
   >help.search("attach")
   ```

   These commands should eventually result in finding a help page that begins with *The database is attached to the R search path. This means that the database is searched by R when evaluating a variable, so objects in the database can be accessed by simply giving their names.*

   c) Now run the command `detach()`. After doing this, try to draw histogram again without running the `attach()` function (It shouldn't work!). What does this suggest to you about the use of `attach()` and `detach()`?

Solutions:

The command `attach()` tells R which data set you are currently interested in using. When a data set is attached, we can access the variables in that data object simply by referring to them. Conversely, `detach()` is essentially a function that tells R that you are no longer working with the data object that you attached earlier. This means when you try to draw a histogram of the weight data, R does not know which data you are interested in working with, and so draws nothing (and sends you an error message).

d) We are going to make some changes to the script file. Before we do this, save the script file under a different name of your choosing. This is so you can go back to the original script file in case any of the changes don't work out.
Solutions:
I saved mine under the name MyExercise1.R.

e) Before doing this question, make sure that the `ChickWeight` data set has been attached (see part b). Look at the command for drawing the second more complicated histogram. Recall our discussion on arguments for functions. The items `breaks=,freq=,col=,main=` are all optional arguments for the histogram command. Choose one of these arguments, and see if you can change it. How does changing this argument change the histogram?

Solutions:
`breaks` - This argument controls the number of bars in our histogram. If we increase the number we will have more breaks/bars in our histogram, and if we decrease the value there will be fewer bars in our histogram.

`freq` - This argument tells R what kind of histogram you want to draw. If the argument is set to `TRUE`, a frequency(counts) histogram is drawn. If the argument is set to `FALSE` then a density histogram is drawn.

`col` - Tells R what colour you want filling in the bars of the histogram. There are lots of ways to set the colours, including typing the colour as I have shown, such as 'blue', 'red', 'purple', 'black'. Note the quotations around the colours!

An alternative is to set the argument to a number such as 1, 2, 3, ... This cycles through a list of about 10 or 15 colours that R uses as a default.

There are also much more complicated colour selections that can be made that are beyond the scope of this course. If you are interested in some of these there are some help files listed under `?colors`.

`main` - Changing this changes the title of our histogram. We can change the title to anything we want, but it should be between quotation marks or R will think it is looking for a data object rather than printing words! More complicated titles, including symbols or other fancy stuff can be achieved using the `paste()` function. You will have to look up how to use it if you are interested.

There are also arguments to boxplot called `xlab`, and `ylab` that set the labels on the x and y axes respectively.

3. Refer to the information on arithmetic expressions and the order of operations, mathematical expressions, and logical values to answer the following question.

a) Create your own arithmetic formula for R to evaluate. Try to include at least one out of each of the following: brackets, exponents, (multiplication or division), (addition or subtraction). Hint: Type your formula into the script file you saved earlier so that you can access it in b).
Solutions:
Note: There is more than one possible solution to this part of the question, although only example formula is presented.

```
> (x+7)^2/2+5
[1] 77
```

b) We also talked about mathematical expressions, and I provided a table on some common mathematical expressions. Choose one of these expressions, and the argument for it, and substitute it into the arithmetic formula you made above. How does R evaluate this? Does including a mathematical function as part of

an equation change the order of operations?

Solutions:
Note: There is more than one possible solution to this part of the question, although only example formula is presented.

```
> (sqrt(x)+7)^2/2+5
[1] 47.65248
```

Mathematical expressions are evaluated first, followed by brackets, exponents, etc.

c) Do the same thing as above, only use a logical expression rather than a mathematical expression. For example, rather than writing `log(x)...`, substitute something like `(x==9)`. Does R evaluate this expression? What does this suggest to you about the nature of logical values in R? Hint: Don't forget the brackets!
Solutions:
Note: There is more than one possible solution to this part of the question, although only example formula is presented.

```
> ((x>10)+7)^2/2+5
[1] 47.65248
```

Even though the logical expression evaluates as TRUE or FALSE, R treats a TRUE as a 1 when evaluating an arithmetic expression, and a FALSE as a 0.