R Workshop Exercises
Fall 2011
Dave Allen and Doug Jackson

Exercise 1: If you've ever been new car shopping, you might have noticed that new cars have city, highway, and combined fuel economy ratings posted on their windows. The combined fuel economy is calculated based on the city and highway fuel economies using the following equation:

$$FE_{combined} = \frac{1}{\frac{0.55}{FE_{city}} + \frac{0.45}{FE_{highway}}}$$

Imagine that you're designing a car, and that there's a tradeoff between highway and city fuel economy such that the sum of the two is always 50 miles per gallon. For example, if you increase the city fuel economy from 18 to 20 mpg, the highway fuel economy would decrease from 32 to 30 mpg.

What combination of city and highway fuel economies will give you the best combined fuel economy?

a) Assign a value to a city fuel economy variable, `fe_city`
b) Calculate the highway fuel economy, `fe_highway`, based on `fe_city`
c) Using the equation above, calculate the combined fuel economy, `fe_combined`. Note: use the variable names, `fe_city` and `fe_highway`, in the equation.
d) Using the up arrow to access previous commands, repeat steps a-c with different values of `fe_city` to determine the best combination of fuel economies.
e) When you're finished, list your environment variables and then clear all of them.
f) While you're waiting for others to finish, use the help command or the help browser to look up some of the following commands:
   1. `log, exp, round, ceiling, floor, ls, rm`

Relevant commands and operators:

1. *`variable <- value`*
2. `ls()`
3. `rm()`
4. `rm(list=ls())`
5. `+, -, *, /`
6. `?, help()`

Exercise 2:

a) Using your operating system's file browser, create a new folder somewhere on your computer (e.g., on your desktop). Call the folder `Exercise_2`.
b) Using either the command line or menu command, set your working directory to `Exercise_2`.
c) Confirm that you have successfully changed your working directory.
d) Create a new source code file.
e) Copy the commands that you used in Exercise 1 into the source code file. Remember that you can use the up arrow or the history browser to access previous commands.
f) Save the new file in your working directory. It's best to use a name without spaces or special characters and to include the ".R" suffix. For example, `Exercise_2.R` would be a valid name.
g) Execute (run) the commands in the source code file (the script) using one of the three options we've covered: the `source` command or either of the two keyboard shortcuts. Note: if you're using the `source` command, try hitting the Tab key after you've typed the first few letters of the file name.
h) In preparation for future exercises, create a folder on your desktop called R_workshop_folder. Download the data files at http://tiny.cc/gwzft and move them into this new folder. Change your working directory to R_workshop_folder.

Relevant commands and operators:

1. ⌘-enter or Edit -> Execute (Mac)
2. ⌘-E or Edit -> Source Document (Mac)
3. Ctrl-R or Edit -> Run line or selection (Windows)
4. Edit -> Run all (Windows)
5. `source('filename.R', echo=TRUE)`
6. `setwd('path_to_directory')`
7. `getwd()`

Exercise 3:

Let's return to Doug's fuel efficiency example.

a) Create a vector of city fuel efficiencies going from 20 to 40 mpg by increments of 0.25 mpg.
b) Now assume that highway fuel efficiency is generally 25% higher than city fuel efficiency. Create a vector of corresponding highway fuel efficiencies.
c) Use Doug's formula to create a vector of combined fuel efficiencies.

Relevant commands:
1. `seq`

Exercise 4:

Use this command to load in some data for exercise 4:

```
load(file="exercise4_data.RData")
```

This file contains four vectors documenting the history of Australian prime ministers:

`pm_name` is the name of the prime minister
`party` is the political party of that prime minister
`days` is the number of days he or she spent in office
`state_rep` is the state that prime minister represented in parliament

a) How many PMs were there who represented the state of Tasmania?
b) The Liberal and Labor parties are the two major parties in Australian politics, what fraction of the time (the data go from 1/1/1901 to 9/1/2010) was each party in power?
c) Which prime minister served the longest term? How long was that term in years.

Relevant commands and operators:
1. `length, max, sum`
2. Vector indexing using logical operators.

Exercise 5:

```
load(file="exercise5_data.RData")
```

This file contains a single matrix called `MagSqu`. A magic square is a matrix whose rows, columns and two major diagonals add up to the same number.

a) Is a `MagSqu` magic square? If not can you make it one?

Exercise 6.

Find the excel file `tadpole_mortality.xls`. Open it up and look at both sheets. There are some problems with the sheets that will 'choke' R, making it impossible or problematic to load this data into R. They are common problems that you might have with your data.

a) Ultimately you want to save each sheet as a separate .csv file and read those files into R so you have two data frames, one for green frogs and one for wood frogs. In order to do this you will have to modify the Excel file to address the problems alluded to above. Make sure that the first and fourth columns are of mode numeric, not character, once they are in R.
b) Find the mean survival rate across the trials of green frogs raised with predators. Do the same for wood frogs. (Hint: it will be easier if you use `attach` and `detach`).

Relevant commands and operators:
`1.read.csv, attach, detach, search, mean, names`

Exercise 7.

Read in `bwdata.csv`. Attach it. Each row represents a tree censused in 2003 with the x and y position of the tree (in meters), its species, its girth at breast height (GBH) in 2003 and its GBH in 2008 (in cm). If the three died between 2003 and 2008 its GBH in 2008 was recorded as 0.

a) Calculate the fraction of Red Maples which were censused in 2003 that died before 2008.
b) What was the average growth rate (amount of growth divided by 2003 size) of the surviving Red Maples between 2003 and 2008.
c) Now load in `big_woods_mortality.csv`. This has the mortality and average growth rate of the 15 most common species in the Big Woods plot. Fit a linear regression to the relationship between these two values. How well does the line fit the relationship? Plot the 15 points and add the line even if it is a weak fit.
d) Now return to the overall Big Woods data. Assume individuals with a GBH in 2008 greater than 100cm are in the canopy and those with a GBH under 100cm are the subcanopy. Black Oak, White Oak and Pignut Hickory are the three most common species in the canopy. What percentage of canopy trees do these three species represent?
e) Black Cherry, Red Maple and Witch Hazel are the three most common species in the subcanopy. What percentage of subcanopy trees doe these three species represent?
f) Make a plot showing the location of the three most common canopy trees with different labels. Include a legend.
g) Do the same for the subcanopy.

Exercise 8:

Load the magic square from Exercise 5 (exercise5_data.RData). Write a *for* loop to sum the values in the first column. Then, modify the *for* loop to sum the values in the first row. Finally, modify the loop to sum the values in the upper-left to lower-right diagonal.

Relevant commands and operators:
`1. for(i in 1:x){}`

Exercise 9:

Imagine that you want to model the growth of a tree. Assume that it grows a random amount per year, with an average growth of 0.5 m, i.e., growth in each year can be drawn from a random distribution: `runif(1, min=0, max=1).` (Note that we're using a uniform distribution for simplicity. A normal distribution would probably be more realistic.) How many years will it take for a tree to reach a height of 25 m? Start from the following code sketch, filling in the missing parts marked with **...**:

```
height <- 0
year <- 0

while(...)
{
     year <- year + 1
     height <- ...
}

print(year)
print(height)
```

Run your code a few times to get a feel for the range of possible outcomes.

Exercise 10:

The file exercise_10.csv contains deer count data that your longtime research assistant has collected for you over the past decade. Except for the occasional sick day, marked with a placeholder of -999, your assistant has been extremely accurate. Accurate, that is, excluding Sundays, when her Saturday nights at Ashley's leave her predictably bleary-eyed.

Luckily, you can quickly fix these data in R. Read the file into R and use a *for* loop to step through each line of data. If the count for the day is -999, replace the placeholder with the previous day's data; else, if it's a Sunday (day == 1), divide the count data by 2 to account for her double vision.

As a sanity check, plot the count data before and after you fix them.

Hint: When assigning a new value to an element in a vector, you can use the element itself in the assignment equation. For example, to double the value of the first element, you could execute the following:

```
count[1] <- count[1]*2
```

Relevant commands and operators:

1. `if(){}`
2. `else if(){}`
3. `else{}`
4. `for(i in 1:x){}`
5. `<, >, <=, >=, ==, &&, ||, !`
6. `length()`

Exercise 11:

Using the code that you wrote for Exercise 9, create a function to calculate how many years it will take for a tree growing an arbitrary amount per year to grow to an arbitrary height. In other words, your function will take three values as inputs – let's call them `runif_min`, `runif_max`, and `end_height` – and return one output (`year`).

Call your function a few times with different inputs to verify that it returns reasonable results.