RNR 321 • Intro to R

R is a free software environment for computing and graphics that has become a standard in many fields. The goal of this lab is to help you get started working with R and RStudio.

1. We will introduce R and RStudio with the **Basics of R** slides on D2L
2. We will work through the **Basics of R illustration.R**, which is an example of an R 'script'
3. You will work through the **Introduction_to_R.pdf** file on D2L (please pay close attention to the first 7 pages), which is a reference for common operations in R
4. You will install and use the R package **swirl** to learn programing basics of R (see **Swirl** below)
5. You will work on the assignment, which has two parts (see **Assignment** below)

Note: `R code is formatted with this typeface.`

Swirl

Note: Code 'snippets' below are also included near the end of the **Basics of R illustration.R** script.

The swirl package is an interactive tool for learning R. To download and install the package, type this command into the console followed by hitting the Enter key:

```
install.packages("swirl")
```

You only need to install a package on your computer once, but you must load them into memory *each time you start a new R session and wish to use its functions*. To load swirl after you've installed it, type this into the console followed by the Enter key:

```
library("swirl")
```

To start using swirl type this command followed by the Enter key: `swirl()`

We interact with swirl through the console (not through the script editor, as you will not be saving any code). Swirl will start by asking you what name you'd like to be called then explaining how it works.

Swirl will ask you to install a course. We will only use course number `1: R Programming: The basics of programming in R`, so choose '1'. Once it is installed, you'll need to select the course from the list, so again choose '1'.

Swirl will then ask you to choose a lesson from the 15 lessons available in the course:

```
Please choose a lesson, or type 0 to return to course menu.

 1: Basic Building Blocks      2: Workspace and Files        3: Sequences of Numbers
 4: Vectors                    5: Missing Values             6: Subsetting Vectors
 7: Matrices and Data Frames   8: Logic                      9: Functions
10: lapply and sapply         11: vapply and tapply         12: Looking at Data
13: Simulation                14: Dates and Times           15: Base Graphics
```

Work through lesson 1 which includes many of R's important basics. Take your time and make sure that you understand each step so that you'll be set for later work. When you're finished, just choose '2:No' when swirl asks if you want to receive credit on Coursera.

Assignment • Intro to R • Due 27 September 2021

Part A:

Download the R script **Basics of R assignment.R** from D2L and open it into RStudio. Follow the instructions in the script.

Part B:

Note: Code provided below is also included near the end of the **Basics of R assignment.R** script.

Work through the information and problems below. Compose your answers in a MS Word document and enter your R code in the locations specified at the end of the **Basics of R assignment.R** script. When you are finished, upload both your Word file and R script to D2L.

Please be sure you understand the value of saving R commands in a script file, which is described below and in the 'Introduction to R' document. Scripts should include only R commands and no output from the console.

Code to import three datasets into R:

```
plain  <- read.csv("https://cals.arizona.edu/~steidl/plain.csv", header=TRUE)
peanut <- read.csv("https://cals.arizona.edu/~steidl/peanut.csv", header=TRUE)
temp   <- read.csv("https://cals.arizona.edu/~steidl/TucsonTemp.csv", header=TRUE)
```

- To display the contents of an object, enter its name: `plain`
- To see how an object is defined (or 'structured') use the `str()` function: `str(plain)`

You will see that R saved each data set as a data frame, which is like a spreadsheet because it is organized around rows (observations) and columns (variables).

Important: You can refer to individual variables (columns) within a data frame by referring to the name of the data frame followed by $ followed by the name of the variable. For example, to refer to the column of yellow m&ms in the plain data frame you would use `plain$yellow`.

Some R functions simplify applying the same function to all columns (or rows) of a data frame.

For example, `colSums()` computes the sum for all columns and `rowSums()` computes the sum for all rows in a data frame.

Similarly, `colMeans()` computes the mean for all columns and `rowMeans()` computes the mean for all rows in a data frame.

The `apply()` function can apply many R functions to all of the columns or rows of a data frame. For example, you might use `apply(x,2,mean)` to apply the `mean()` function to each column of object `x`. Note, the number '2' tells R to apply the function to columns; using the number '1' instead would tell R to apply the function to rows; e.g., `apply(x,1,mean)`.

Type `help(apply)` or `?apply` into the R console to learn more about `apply()`; learning how this command works will save you time.

1. For both `plain` and `peanut` data frames, use the built-in R functions to compute the total (or sum), average, variance, and standard deviation for the number of m&m's of each color separately.

> Copy the output you've created from the R console and paste it into a Word document – it will be ugly, so please clean it up to make it easy to read.

2. Please read this entire problem before you begin – doing that will save you time.

Create two bar charts, one for `plain` and one for `peanut`, to display the **proportion** of m&ms of each color [the R function to create a bar chart is `barplot()`]. Label both axes and provide a title for each graph. When you are done, cut-and-paste both charts into your Word file. Remember, you can get help with the `barplot` command by typing `help(barplot)` into the console.

Most operations in R require a series of steps. Saving the code for these steps in a script file allows you to reuse the code in the future.

For this problem, you might first create an object to hold the total number of m&m's of each color. You *could* do that (***but don't!***) by creating one object to hold the sum of each color:

```
sumPlainYellow <- sum(plain$yellow)
sumPlainGreen <- sum(plain$green)       # and so on for all the other colors
```

Instead, you can accomplish the same thing with just one command by using either the `colSums()` or `apply()` functions described above, which work on all columns in the data frame at once!

```
totals <- apply(plain,2,sum)
```

Be sure to check to see what is stored in the object `totals` once you've created it.

The next step is to create the proportions of each color that you need for the bar plots. That too can be done with just one command by recognizing that R allows functions to be embedded in other expressions. Try this command to see what happens:

```
totals/sum(totals)
```

Once you understand the result, you will want to save the output as an object so that you can use it to create the bar plots.

3. Create a scatter plot with a trend line to illustrate the relationship between High and Low temperatures in the `temp` data frame. Use Low temperature as the Y-axis and High temperature as the X-axis; be sure to label all axes. Creating the graph is best done in a series of steps:

- First create a scatter plot between High and Low temps [the R function is `plot(); e.g., plot(`*x-axis values, y-axis values*`);` check out `help(plot)` or this site] .
- Next, use the function `abline()` to add a trendline to the plot that you created with the function `lm()` [lm = linear model; simple linear regression is a type of linear model].

When you have made the graph, from the Files/Plots Window in RStudio, choose *Export* then *Copy to Clipboard*…, then open your Word document and *Paste* the graph into your document.