# Lab 0: Introduction to RStudio

## Overview

The goal of this lab is to introduce you to R and RStudio, which you will be using throughout the course both to learn the statistical concepts discussed in the class and also to analyze actual data and come to informed conclusions. R is the programming language's name, and RStudio is a convenient interface.

As you progress through the labs, you are encouraged to explore beyond what the labs dictate; a willingness to experiment will make you a better programmer. However, before we get to that stage, you need to build some basic fluency in R. Today, we begin with understanding the components of RStudio and executing code from an R script file.

The first thing you need to do before beginning ANY work in RStudio is open an "R Script." You can do this by navigating to your RStudio toolbar and clicking `File -> New File -> R Script.` It will create a new blank document in your RStudio environment. You should write all of your code into this document, including careful comments explaining what your code is doing. A well-written and organized R Script will help you immensely complete assignments for this course.

Once you have written a line of code, there are two ways to "submit it to the console" (which means run it). You highlight the line you want, and then either click `Run` or type `Ctrl+Enter` on a PC or `Cmd+Enter` on a Mac.

Once code has been submitted, you should see the command appear in the R console window in the lower-left panel. Note that the console has a prompt (>) - this means that R is ready to receive a command.

Some basic functions of R are that it can be used as a calculator to generate numbers, create and work with objects, make plots, and access help files.

Try typing the following commands into your R Script, and see what they do!

```r
2+2
1:20
```

As you can see, R can function as a calculator! However, this is not very exciting. For most of this class, we will be using "objects" to store various things: numbers, datasets, lists, and more. Keeping these items as "objects" within RStudio allows us to use them repeatedly without retyping them!

For example, try the code below:

```r
# Store the list of numbers into "x"
x<-c(2,3,5,7)

# View x
x

# Multiply every number in the "x" list by 5
x2 <- x*5

# View x2
x2
```

If you run this code, you should see a few new items listed in your `Environment` panel in the top right section of RStudio. Each object you create will show up here.

Finally, we can use our "objects" in functions. Functions are small programs inside R that perform tasks for us. For example, a "plot" function will take our numbers and create a plot:

```r
# Plot X by X2
plot(x=x,y=x2)
```

Each function has "arguments." It is how we tell the `plot` function what we want it to use as the X-axis and the Y-axis. Fancier arguments can say what color we want to make the plot, what kind of plot we want, if we wish to include titles, and more. You can learn more about any function by typing `?` followed by the function's name. For example, to see all of the options for plots, we type `?plot`.

Note that this is just a basic overview. More example code is provided in an R Script on Canvas–try opening and running the commands inside. There is also a "Glossary of Terms" included at the end of this document that you are encouraged to read and save for review.

## Importing a Dataset

For almost everything in this class, we will be using a dataset. A dataset is a table, where each row is an observation, and each column is a variable. For example, if I asked 1400 adults their height, weight, and favorite color, my dataset would have 1400 rows (1 for each adult) and three columns (one for each question I asked).

We have provided all of the datasets you will need on Canvas. However, to use them in R, you will need to "import" them. There are two main ways. You can either use point-and-click or write code to import the file. While we provide instructions for both methods below, you are HIGHLY encouraged to learn the code method. This method is faster (in the long run) and will help you learn more about coding logic.

### To use point and click:

1) Click `File` –> `Import Dataset` –> `From Text (base)`
2) Select the `.csv` or `.txt` file you want to open and click `Open.`
3) Change the `Heading` option to `Yes.`
4) Click `Import.` The dataset should pop up on your Environment panel (much like the other objects from earlier in this section). If you click the small arrow next to its name, you should see a list of variable names.

### To use code:

To use code, you need to know your file path. A "file path" means where your file is located on your computer. If you find it on your Finder or Files on your computer, you can right-click on the file and see its file path. Once you know the file path, you type: `setwd("Your/File/Path/").` This tells R where to look for your file. You then type `dataname <- read.csv("FileName.csv", header=T)` if the file ends in ".csv". If it ends in .txt, you type `dataname <- read.txt("FileName.txt", header=T)`. You can replace "dataname" with anything you'd like–it sets the object's name that will hold your dataset. We recommend something short, as you will likely be typing this name a lot!

## Once you have imported your dataset

Once you have imported the dataset, you can use it for many analyses and functions. To access a particular variable (a column of your data), you use the symbol `$`: `dataname$variablename.` You can then tell R to give you the mean of this variable, the median, or use it in a plot. For example: `mean(dataname$variablename)` will provide you with the average of your variable (provided, of course, that it contains numbers).

Below, see an example with the "yrbss2013" dataset. You can download this data from Canvas and run the following code.

```r
# Summary statistics for respondent height
# (Assumes you have imported the dataset - see instructions on importing a dataset)
summary(yrbss2013$height_m)

#basic plot
plot(x = yrbss2013$height_m,y = yrbss2013$weight_kg)

#custom plot - add labels
plot(x = yrbss2013$height_m,
     y = yrbss2013$weight_kg,
     type = "p",
     xlab = "Height (m)",
     ylab = "Weight (kg)")
```

## Glossary of Terms

### General Terms

| Term | Meaning |
|---|---|
| `Argument` | Parameter/settings for each function, which control its behavior |
| `Concatenate` | Combine many individual numbers or words into a list, denoted using c() |
| `Console` | Where your commands run, and all results are printed out |
| `Environment` | Where all of your objects are stored |
| `Filepath` | The location of a file (which computer folder contains it) |
| `Function` | A program that performs a given task. You can learn more about any function by typing `?function.` |
| `Package` | Code written by an external party that you can install and use. One package will contain many functions. |
| `Object` | An item stored in the R environment that you can use again |
| `R Script` | The file you should write ALL code into to save it and reuse it |

### Data Structures

| Data Type | Description |
|---|---|
| `Boolean` | True/False values |
| `Character` | A letter, number, or symbol, indicated in code by `" "` or ' ' (essentially the same as `string` in R, but this is not always true in other programming languages) |
| `Dataframe` | A specially formatted table, where rows are observations and columns are variables |
| `Factor` | A set of defined levels (like "Yes," "No," "Maybe") for a given list or variable |
| `Integer` | Numbers |
| `String` | A sequence of characters e.g., a word or sentence, indicated in code by `" "` or ' ' |