Introduction to R

R Tutorial

# Objectives of this tutorial

* Introduce R language and R studio
* Describe the structure of RStudio
* Practice Basic R functions
* Define Variables and Vectors
* Discuss Scripts and Executing Commands.

# Introduction

## R

R is a programming language that is popular in both academia and industry for doing statistical analysis and creating graphics from data. Knowing how to program in R (even just a little) is a valuable skill.

You might have some experience working with data in Microsoft Excel, where you point and click to enter data and use buttons, menus, or formulas to perform tasks. In R, you write code to do all your desired calculations and to create graphics. Writing your own code allows for more flexibility than a point-and-click interface like Excel.

If you do not have any programming experience, do not worry about it! It might take a little practice, and patience, to get used to R but we will be here to guide you through all of the code you need to write.

## RStudio

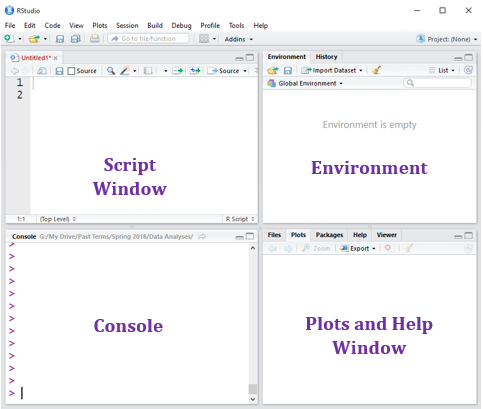
RStudio is a desktop application that makes it easy to use R and create R scripts. All of the tutorials in this course will assume you are using RStudio. It's not required for using R, but the user interface is more intuitive for a novice user. R and RStudio are free to use.

See the document in the “Start Here” module in Canvas called *Installing R and RStudio* for help installing them on your computer.

# Basics of R

## Getting Started with R and R Studio

Open R Studio on your computer. R Studio has four windowpanes:

* **Script Window** – For writing, copying and editing code. Can save scripts with code for later use.
* **Console** – Displays results from commands written and run in the script window. Writing commands in the Console is for basic commands only.
* **Environment** – Imported Datasets and/or named variables will show up here.
* **Plot Window** – Displays and stores plots, can be used for help messages and to search for and install packages.

## The Console

You can use the **Console** as a simple calculator. If you enter R code into the console, it will execute when you press **Enter**. Enter the following lines at the > prompt and see that you get the same output.

1 + 1

## [1] 2

3 \* 12 + 6

## [1] 42

3^2

## [1] 9

10/2

## [1] 5

Note: You will not see the ## next to your output. We will use the ## just to indicate an “answer” the tutorial. The [1] in R output indicates the first object’s placement in an ordered list. This is not part of the answer.

## Statements, Variables, and the Environment

An R program consists of a series of **statements**. Here are two examples of statements:

x <- 10  
y <- x + 2

The <- operator (looks like an arrow) stores the value on the right to the name on the left. The first statement assigns the number 10 to the variable x. You can also assign a name to a function of another assigned variable, like the second statement, which adds 2 to x and stores the result in y.

Enter each of the statements above into the console. Notice that there is no output, but the variables and their values show up in the pane labeled **Environment**. You can see the value of each variable in the **Environment**, or you can run the variable name in the **Console.**

> x

## [1] 10

> y

## [1] 12

## Data Types

A variable contains data of a certain form or type. A common **data type** in R is the vector. A **vector** is just an ordered list of values. The values in a vector can be numbers, text, or logical (TRUE or FALSE). Values in a vector must all be the same type. Create vectors with the c function where each value is separated by a comma. Define z as a vector of three numbers:

z <- c(3, 17, 42)

To see what the variable z contains, look at the environment pane, or enter z by itself in the console. R will output the contents of the variable.

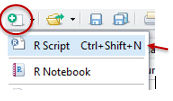
z

## [1] 3 17 42

Notice that in the environment pane, z has num [1:3] before the values. This tells you that z is a numeric vector with 3 values.

## Scripts

**Best Practice** - It will be easier, and reduce the chance of errors, if you type code in the **Script window rather** than individually in the **Console window**. This also allows you to save your code, or your “**script**”, in the **Script window** for later use.

In R Studio, open a blank R script using **File New File R Script** or **Ctrl + Shift + N** (Windows) or **Command + Shift + N** (Mac). This will open the **Editor** pane (if it was not already open), which is where you edit R scripts.

Retype some of the code above now into the **Script window**.

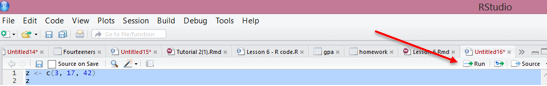
z <- c(3, 17, 42)

z

avg <- mean(z)

avg

To run the script, highlight any number of statements in your R script and click **Run**  at the top of the editor pane or hit **Ctrl + Enter** (Windows) or **Command + Enter** (Mac), and the statements will be executed in the console. If you place your cursor somewhere in the code without anything highlighted and use the same command, it will execute just the line where your cursor is. Try executing your script's statement in the console by highlighting all five lines and clicking on “Run”.



Use **File Save As** to save the file with a name ending in .r or .R in whatever location you like.

**Best Practice -** Start a folder on your home computer or on your student Z drive for your R code and data files for the class**.**