We recommend anyone new to R and RStudio to try **Swirl** (statistics with interactive R learning). Swirl is a software package for the R statistical programming language. Its purpose is to teach statistics and R commands interactively.

Type the following commands in the **Console**, pressing Enter or Return after each line:

install.**packages**("swirl")

**library**(swirl)

**swirl**()

Note that the > symbol at the beginning of the line is R's prompt for you type something into the console. We include it here so you know that the above commands are to be typed into the console and not elsewhere. The part you type begins after >.

# The goal of files, like this one, is to introduce you to the

# R programming language. Let's start with by unraveling a

# little mystery!

# 1. Run the code below to create the vector 'udacious'.

# You need to highlight all of the lines of the code and then

# run it. You should see "udacious" appear in the workspace.

udacious <- c("Chris Saden", "Lauren Castellano",

"Sarah Spikes","Dean Eckles",

"Andy Brown", "Moira Burke",

"Kunal Chawla")

# You should see something like "chr[1:7]" in the 'Environment'

# or 'Workspace' tab. This is because you created a vector with

# 7 names that have a type of character (strings). The arrow-like

# '<-' symbol is the assignment operator in R, similar to the

# equal sign '=' in other programming languages. The c() is a

# generic function that combines arguments, in this case the

# names of people, to form a vector.

# A vector is one of the data types in R. Vectors must contain

# the same type of data, that is the entries must all be of the

# same type: character, logical (TRUE or FALSE), or numeric.

# HELPFUL TIP: You can add values to a vector.

# Run each line of code one at a time below to see what is happening.

numbers <- c(1:10)

numbers

numbers <- c(numbers, 11:20)

numbers

# 2. Replace YOUR\_NAME with your actual name in the vector

# 'udacious' and run the code. Be sure to use quotes around it.

udacious <- c("Chris Saden", "Lauren Castellano",

"Sarah Spikes","Dean Eckles",

"Andy Brown", "Moira Burke",

"Kunal Chawla", YOUR\_NAME)

# Notice how R updates 'udacious' in the workspace.

# It should now say something like 'chr[1:8]'.

# 3. Run the following two lines of code. You can highlight both lines

# of code and run them.

mystery = nchar(udacious)

mystery

# You just created a new vector called mystery. What do you

# think is in this vector? (scroll down for the answer)

# Mystery is a vector that contains the number of characters

# for each of the names in udacious, including your name.

# 4. Run this next line of code.

mystery == 11

# Here we get a logical (or boolean) vector that tells us

# at what locations or indices in the vector where the number

# of characters for the name is equal to 11.

# 5. Let's use this boolean vector, mystery, to subset our

# udacious vector. What do you think the result will be when

# running the line of code below?

# Think about the output before you run this next line of code.

udacious[mystery == 11]

# Scroll down for the answer

# It's your Udacious Instructors for the course!

# (and you may be in the output if you're lucky enough

# to have 11 characters in YOUR\_NAME) Either way, we

# think you're pretty udacious for taking this course.

# 6. Alright, all mystery aside...let's dive into some data! Run

# this next command to load the mtcars data.

data(mtcars)

# You should see mtcars appear in the 'Environment' tab.

# R has stored the mtcars data into a spreadsheet-like object

# called a data frame. Run the next command to see what variables

# are in the data set.

names(mtcars)

# names(mtcars) should output all the variable

# names in the data set. You might notice that the car names

# are not a variable in the data set. The car names have been saved

# as row names. More on this later.

# You should also see how many observations (obs.) are in the

# the data frame and the number of variables on each observation.

# 7. To get more information on the data set and the variables

# run the this next line of code.

?mtcars

# You can type a '?' before any command or a data set to learn

# more about it. The details and documentation will appear in

# the 'Help' tab.

# 8. To print out the data, run this next line as code.

mtcars

# Scroll up and down in the console to check out the data.

# This is the entire data frame printed out.

# 9. Run these next two functions, one at a time,

# and see if you can figure out what they do.

str(mtcars)

dim(mtcars)

# Scroll down for the answer.

# The first command, str(mtcars), gives us the structure of the

# data frame. It lists the variable names, the type of each variable

# (all of these variables are numerics) and some values for each

# variable.

# The second command, dim(mtcars), should output '[1] 32 11'

# to the console. Don't worry about the [1].

# I'll explain that later.

# 10. Read the documentation for row.names if you're want to know more.

?row.names

# Run this code to change the row names of the cars to numbers.

row.names(mtcars) <- c(1:32)

# Now print out the data frame by running the code below.

mtcars

# It's tedious to relabel our data frame with the right car names

# so let's reload the data set and print out the first ten rows.

data(mtcars)

head(mtcars, 10)

# The head() fucntion prints out the first six rows of a data frame

# by default. Run the code below to see.

head(mtcars)

# I think you'll know what this does.

tail(mtcars, 3)

# 11. We've run nine commands so far:

# c, nchar, data, str, dim, names, row.names, head, and tail.

# All of these commands took some inputs or arguments.

# To determine if a command takes more arguments or to learn

# about any default settings, you can look up the documentation

# using '?' before the command, much like you did to learn about

# the mtcars data set and the row.names

# 12. Let's examine our car data more closely. We can access an

# an individual variable (or column) from the data frame using

# the '$' sign. Run the code below to print out the variable

# miles per gallon. This is the mpg column in the data frame.

mtcars$mpg

# Print out any two other variables to the console.

# This is a vector containing the mpg (miles per gallon) of

# the 32 cars. Run this next line of code to get the average mpg for

# for all the cars. What is it?

# Enter this number for the quiz on the Udacity website.

# https://www.udacity.com/course/viewer#!/c-ud651/l-729069797/e-804129314/m-830829287

mean(mtcars$mpg)

[**QuickBook for R**](https://www.datacamp.com/doc/r/category/r-documentation)