Lab 7: for-loops and the apply() family

The following worksheet is due by 8pm one day after this lab. You can find the submission dropbox in Brightspace by clicking on Content − > Lab Content.

0. Open a new R Markdown file.

   Note: Your worksheet is to be submitted as the output of an R Markdown file (you can knit it to HTML and then convert it to PDF, or you can knit it to PDF if you have LaTeX on your computer, or you can knit it to Word and then convert that to a PDF).

1. Generate a sequence of random integers between 20 and 30 without replacement, and you want to stop the sequence once a value of 27 is generated. Please use a while loop and an if statement to accomplish this.

2. Write a program that reads in a list of numbers (6, 7, 4, 3, 1, 6, 7, 4, 9).

   (a) calculate the sum of the even numbers using a “for” loop and “if” condition.

   (b) print out "The sum of even numbers is" with the sum of even numbers.

3. Download the data set boombust.csv and save it to whatever directory you are using for this course. The goal is to write a for-loop to create a new column at the end of the matrix that contains the sum of each row.

   (a) Create a matrix that contains only the numerical values and name it nums

   (b) Create a new column of zeros at the end of nums by using: nums = cbind(nums, rep(0, length(nums[,1]))).

   (c) Write a for-loop that calculates the sum (for each row) of the first through eighth columns of nums and saves the sum in the nineth column of nums. ie.

   (d) Print out the nums matrix.

4. Loops in R are notoriously slow. While loops are incredibly important to master from a theoretical sense, when working with large data sets we should always try to use the apply family of functions to increase efficiency.

   You have learned about sapply and lapply in class, but until you learn how to write your own functions, sapply and lapply can be fairly limited. Today we will take a quick look at the power of the apply()

function, which allows us to perform functions on 2 dimensional objects like matrices and dataframes. The apply() function has 3 main parameters: apply(X = , MARGIN = , FUN =). The only difference between apply() and sapply() is the MARGIN parameter which tells R whether you want to calculate something on the rows (MARGIN = 1) or the columns (MARGIN = 2)

(a) Create a matrix that contains only the numerical values of boombust.csv and name it names.

(b) Create a new column at the end of names (similarly to how you were shown in 3c).

(c) Use apply() to fill this new column with the sum of each row for columns 1 − 9. Hints: in the apply() function you should set X = names[, 1:9] and FUN = sum. You cab set MARGIN equal to one.

(d) Print out names