553.283 Introduction to R
Homework 4

Note 1: If a question asks you for a numerical answer, your submission for that question must consist of the R command that produces that answer followed immediately by the output.

Note 2: Please label all axes on any plots you create.

Note 3: When writing functions, please indent each nested block of code further away from the margin than the block in which it is nested as in the lecture code. It makes your code much easier for me and the TAs to read.

Note 4: Properly comment your code where ambiguity may arise. Comments in R are preceded by the # sign.

1. Alter the code for the *hello()* function (slide 13 of Lecture 4) such that it takes one argument, say x, and outputs the greeting "hello x". Test it out via the command *hello("kitty")*.

2. Use preexisting functions in R to write a function called *ourhist* that produces a proportion histogram of a numerical vector x such that the number of breaks is determined by the *"Scott"* method and the histogram is overlaid with the density line. Test it on 100 randomly generated observations from the Exp(3) distribution. (To simulate from an Exp(3) distribution, we can use the function *rexp()*.)

3. Without using the *ifelse()* function, write a function that adds 1 to all the positive entries of a vector and subtracts one from all the non-positive entries. Test it on 100 randomly generated observations from the N(0,1) distribution, and use the function from the previous problem to create a histogram of the modified sample.

4. Write the same function as in the previous problem, only this time use the *ifelse()* function.

5. The classic Fibonacci sequence is defined such that the first two numbers are 0 and 1, and each subsequent number in the sequence is the sum of the two numbers preceding it. I.e.,

$$a_1 = 0$$
$$a_2 = 1$$
$$a_k = a_{k-1} + a_{k-2}.$$

   It's easy to see that the first nine numbers in the sequence are $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$. Write a recursive function that returns the nth Fibonnacci number.

6. Write a function called *genericGreeting()* that takes in as user input a character vector defaulted to your name, a character greeting defaulted to "Hello", and a logical value indicating whether it is your birthday. The function should return the greeting, fol.lowed by your name. If it is your birthday, it should also add another sentence saying "Happy Birthday!" For example *genericGreeting("Joshua","Good Day", FALSE)* should return a character that says "Good Day, Joshua!"

7. **Challenge Question (10 bonus points):** Write a function called *genericSort()* that takes in a numeric or integer vector, sorts it, and returns the indices of the sorted values. It should also print an error if the input is a character. For example, *genericSort(c(1,3,7,5))* should return the vector (1,2,4,3).