Assignment 2: Coding Basics

Julia Kagiliery

OVERVIEW

This exercise accompanies the lessons/labs in Environmental Data Analytics on coding basics.

Directions

1. Rename this file `<FirstLast>_A02_CodingBasics.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure to **answer the questions** in this assignment document.
5. When you have completed the assignment, **Knit** the text and code into a single PDF file.
6. After Knitting, submit the completed exercise (PDF file) to Canvas.

Basics, Part 1

1. Generate a sequence of numbers from one to 55, increasing by fives. Assign this sequence a name.
2. Compute the mean and median of this sequence.
3. Ask R to determine whether the mean is greater than the median.
4. Insert comments in your code to describe what you are doing.

```r
#1.
FivesSequence <- seq(1, 55, 5) # generate a sequence starting at 1 up to 55 by increasinng by 5 each ti

#2.
mean(FivesSequence) #find the mean of the five sequence above
```

```
## [1] 26
```

```r
median(FivesSequence) #find the median of the five sequence above
```

```
## [1] 26
```

```r
#3.
if (mean(FivesSequence) > median(FivesSequence))
  print("Mean is bigger") #logic text that prints Mean is bigger given the above inequality is true
if (mean(FivesSequence) < median(FivesSequence))
  print("Median is bigger") #logic text that says Median is bigger given the above inequality is true
if (mean(FivesSequence) == median(FivesSequence))
  print("They are the same") #logic text that says the mean and median are the same given the above equ
```

```
## [1] "They are the same"
```

Basics, Part 2

5. Create three vectors, each with four components, consisting of (a) student names, (b) test scores, and (c) whether they are on scholarship or not (TRUE or FALSE).

6. Label each vector with a comment on what type of vector it is.

7. Combine each of the vectors into a data frame. Assign the data frame an informative name.

8. Label the columns of your data frame with informative titles.

```
StudentNames <- c("Ben", "Brian", "William", "Sarah") # character vector of names
TestScores <- c(97, 84, 43, 79) #double (i.e. numeric) vector
Scholarship <- c(FALSE, TRUE, FALSE, FALSE) # logical vector
# type of vectors were evaluated using typeof() function

StudentInformation <- data.frame(StudentNames, TestScores, Scholarship)
names(StudentInformation) <- c("StudentName", "TestScore", "Scholarship")
```

9. QUESTION: How is this data frame different from a matrix?

   Answer: This data frame is different from a matrix because it contains multiple types of data (charcters, doubles, and logic varibles) whereas a matrix only contains one type of data.

10. Create a function with one input. In this function, use `if...else` to evaluate the value of the input: if it is greater than 50, print the word “Pass”; otherwise print the word “Fail”.

11. Create a second function that does the exact same thing as the previous one but uses `ifelse()` instead if `if...else`.

12. Run both functions using the value 52.5 as the input

13. Run both functions using the **vector** of student test scores you created as the input. (Only one will work properly...)

```
#10. Create a function using if...else
PassingIf <- function(x){
  if (x > 50)
  print("Pass")
  else print("Fail")
}

#11. Create a function using ifelse()
PassingIfElse <- function(x){
  ifelse (x > 50, "Pass", "Fail")
}

#12a. Run the first function with the value 52.5
PassingIf(52.5)
```

```
## [1] "Pass"
```

```
#12b. Run the second function with the value 52.5
PassingIfElse(52.5)
```

```
## [1] "Pass"
```

```
#13a. Run the first function with the vector of test scores
#PassingIf(TestScores) this does not work

#13b. Run the second function with the vector of test scores
PassingIfElse(TestScores)
```

```
## [1] "Pass" "Pass" "Fail" "Pass"
```

14. QUESTION: Which option of `if...else` vs. `ifelse` worked? Why? (Hint: search the web for "R vectorization")

    Answer: ifelse() is the function that worked for the vector. This is beacuse a vector has length greater than one (i.e. it is multiple data points) so when we run a function on a vector we are effectively asking the fuction to run x times for x different inputs where x represents the number of objects in a vector. For this example, we are asking the function to run 4 times, with each of the four different scores as an input. if…else() does not accept input with length longer than 1 so it will not accept vectors. We might be able to overcome the funtion's unwillingness to accept a vector with a loop.

NOTE Before knitting, you'll need to comment out the call to the function in Q13 that does not work. (A document can't knit if the code it contains causes an error!)