Assignment 2: Coding Basics

Zhaoxin Zhang

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.

```
Sequence_1_to_55 <- seq(1, 55, 5)
#from 1, to 55, by 5
Sequence_1_to_55
```

```
##  [1]  1  6 11 16 21 26 31 36 41 46 51
```

2. Compute the mean and median of this sequence.

```
SequenceMean <- mean(Sequence_1_to_55)
SequenceMedian <- median(Sequence_1_to_55)
SequenceMean
```

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

```
SequenceMedian
```

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

3. Ask R to determine whether the mean is greater than the median.

```
SequenceMean>SequenceMedian
```

```
## [1] FALSE
```

4. Insert comments in your code to describe what you are doing.

```
#1. Use the seq() to build a sequence from one to 55.
#The three numbers seperated by commas are "from", "to" and "by".
#So seq(1,55,5) means from 1, to 55, by 5.
#Print the Sequence_1_to_55, R will show what the sequence looks like.

#2. Create two vectors named "SequenceMean" and "SequenceMedian"
#to represent the mean and median of the sequence.
#Print the vector and get the mean and median.

#3. Compare the mean and median to see if the mean is greater than the median.
#If it is greater, the printed result will be TRUE, if not, the printed result will be FALSE.
#In the Console we can see a FALSE, which means that the mean is not greater than the median.
```

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).

```
a <- c("Alex","Betty","Carol","Dave")
a
```

```
## [1] "Alex"  "Betty" "Carol" "Dave"
```

```
b <- c(88, 98, 90, 86)
b
```

```
## [1] 88 98 90 86
```

```
c <- c(FALSE, TRUE, FALSE, FALSE)
c
```

```
## [1] FALSE  TRUE FALSE FALSE
```

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

```
class(a)
```

```
## [1] "character"
```

```
#The vector "a" is classified as "character"
class(b)
```

```
## [1] "numeric"
```

```
#The vector "b" is classified as "numeric"
class(c)
```

```
## [1] "logical"
```

```
#The vector "c" is classified as "logical"
```

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

```
#None of them is data frame
#Combine these vectors into a data frame
StudentList <- data.frame(a,b,c)
StudentList
```

```
##       a  b     c
## 1  Alex 88 FALSE
## 2 Betty 98  TRUE
## 3 Carol 90 FALSE
## 4  Dave 86 FALSE
```

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

```
names(StudentList) <- c("Names","Scores","Scholarship"); print(StudentList)
```

```
##   Names Scores Scholarship
## 1  Alex     88       FALSE
## 2 Betty     98        TRUE
## 3 Carol     90       FALSE
## 4  Dave     86       FALSE
```

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

   Answer: A matrix is a 2 dimensional structure that can only contains elements of the same type (e.g., numeric, characters, etc). A data frame is also a 2 dimensional structure, but its columns can have different modes (e.g., numeric, factor, characters, etc).

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...)

```r
#10. Create a function using if...else
FunctionQ10 <- function(x) {
  if (x > 50) {
    print("Pass")
  } else {
    print("Fail")
  }
}

#11. Create a function using ifelse()
FunctionQ11 <- function(x) {
  ifelse(x > 50, "Pass", "Fail")

}

#12a. Run the first function with the value 52.5
FunctionQ10(52.5)
```

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

```r
#12b. Run the second function with the value 52.5
FunctionQ11(52.5)
```

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

```r
#13a. Run the first function with the vector of test scores

#This one does not work

#13b. Run the second function with the vector of test scores
FunctionQ11(b)
```

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

```r
#This one works
```

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

    Answer: The function using`ifelse` worked while the function using`if...else` did not work. The reason is that for FunctionQ10 which using `if...else`, there are more than one vector conditions. The x > 50 is a vector, while the "else" give another vector. Thus, the function() cannot run two vector conditions at one time. If we vectorized it by using `ifelse`, the new function will have one vector condition which's length equals to 1. As a result, the `ifelse` function works.

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!)