Assignment 2: Coding Basics

Abby Liu

OVERVIEW

This exercise accompanies the lessons 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 Sakai.

Basics Day 1

1. Generate a sequence of numbers from one to 100, increasing by fours. 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.

```
#1.

sequence <- seq(1, 10, 4) #generate the required sequence with seq() function
sequence #return the sequence generated
```

```
## [1] 1 5 9
```

```
#2.

mean(sequence) #use mean() function to calculate the mean of the sequence
```

```
## [1] 5
```

```
median(sequence) #use median() function to calculate the median of the sequence
```

```
## [1] 5
```

```
#3.

mean(sequence) > median(sequence) #use conditional statement to determine whether the mean is greater t
```

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

Basics Day 2

5. Create a series of vectors, each with four components, consisting of (a) names of students, (b) test scores out of a total 100 points, and (c) whether or not they have passed the test (TRUE or FALSE) with a passing grade of 50.
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.

```
name <- c("Mario", "Toad", "Peach", "Bowser") #vector type: character
score <- c(80, 47, 96, 68) #vector type: numeric
pass <- c(TRUE, FALSE, TRUE, TRUE) #vector type: logical

test_result_df <- data.frame("Name"=name, "Score"=score, "Pass"=pass)
test_result_df
```

```
##     Name Score  Pass
## 1  Mario    80  TRUE
## 2   Toad    47 FALSE
## 3  Peach    96  TRUE
## 4 Bowser    68  TRUE
```

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

   Answer: A matrix can only contain a single class of data while dataframe can contain different types of data.

10. Create a function with an if/else statement. Your function should take a **vector** of test scores and print (not return) whether a given test score is a passing grade of 50 or above (TRUE or FALSE). You will need to choose either the `if` and `else` statements or the `ifelse` statement.
11. Apply your function to the vector with test scores that you created in number 5.

```
#Option1: 'if' and 'else' statements
pass_or_not1 <- function(score) {
  if(score > 50) {
    TRUE
  }
  else {
    FALSE
  }
}
#pass_or_not1(score) #This doesn't work with an error message of the condition has length > 1. The 'if'
pass1 <- sapply(score, FUN=pass_or_not1) #This error can be solved if we use sapply() to apply this fun
pass1
```

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

```
#Option2: 'ifelse' statement
pass_or_not2 <- function(score) {
  ifelse(score > 50, TRUE, FALSE)
}
pass_or_not2(score)
```

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

12. QUESTION: Which option of `if` and `else` vs. `ifelse` worked? Why?

    Answer: Both options work because 'ifelse' statement can be seen as a combination of the 'if' and 'else' statements with the same function. But 'if' and 'else' statements can only work if we use sapply() to apply this function on every single component of the vetcor series one at a time. 'ifelse' statement doesn't have this issue and can be successfully applied to vector series.