Assignment 2: Coding Basics

Andreana Chou

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, Part 1

1. Generate a sequence of numbers from one to 30, increasing by threes. 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_30 <- seq(1, 30, 3)
sequence_30
```

```
##  [1]  1  4  7 10 13 16 19 22 25 28
```

```
#used sequence function, assigned to object

#2.
mean_sequence_30 <- mean(sequence_30)
mean_sequence_30
```

```
## [1] 14.5
```

```
median_sequence_30 <- median(sequence_30)
median_sequence_30
```

```
## [1] 14.5
```

```
#assigned mean and median functions to objects

#3.
answer_3 <- mean_sequence_30 > median_sequence_30
answer_3
```

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

```
#used conditional statement, assigned to object
```

Basics, Part 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.

```
#5.and #6.

a_vector <- c("Anna", "Steve", "Emily", "Jason")   #a_vector is a character vector
b_vector <- c(93, 49, 86, 52)    #b_vector is a number vector
c_vector <- c(b_vector >=50)    #c_vector is a logical vector

#7.

test_scores <- data.frame(a_vector, b_vector, c_vector)
test_scores
```

```
##   a_vector b_vector c_vector
## 1     Anna       93     TRUE
## 2    Steve       49    FALSE
## 3    Emily       86     TRUE
## 4    Jason       52     TRUE
```

```
#8.

test_scores <- data.frame("names"=a_vector, "score"=b_vector, "pass"=c_vector)
test_scores
```

```
##   names score  pass
## 1  Anna    93  TRUE
## 2 Steve    49 FALSE
## 3 Emily    86  TRUE
## 4 Jason    52  TRUE
```

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

   Answer: The data frame is different from a matrix because the columns have different modes (one is number, the others are character).

Basics, Part 3

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.

```
#ifelse: created a function with ifelse statement using vector of the test scores
#assigned to object that could act as a function with numerical inputs

passing_grade1 <- function(b_vector) {print(ifelse(b_vector >= 50, TRUE, FALSE))}
passing_grade1_vector <- passing_grade1(b_vector)   #using a vector input into ifelse
```

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

```
passing_grade1_40 <- passing_grade1(40)             #using a numerical input into ifelse
```

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

```
#if/else: created a function with if/else statement using a vector of the test score
#assigned to object that could act as a function with numerical inputs

passing_grade2 <- function(b_vector) {print(if(b_vector >= 50) {TRUE} else {FALSE})}
                                                    #a vector input does not work
passing_grade2_40 <- passing_grade2(40)             #using a numerical input into if/else
```

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

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

    Answer: While both if/else and ifelse worked with numerical inputs, ifelse was the option that worked with vector inputs. This is because the if/else function can only interpret one value at a time. When if/else is given a vector, or multiple entries, it is unable to provide an output as to whether the student passed or failed.