Assignment 2: Coding Basics

Alex Lopez

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.

```
#1.

#sequence function seq(from, to, by)
seq(1, 55, 5)
```

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

```
#name sequence function
five_sequence <- seq(1, 55, 5)

#call sequence function
five_sequence
```

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

```
#2.

#compute mean of sequence
mean(five_sequence)
```

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

```
#compute median of sequence
median(five_sequence)
```

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

```
#3.

#output will be TRUE if statement is true, FALSE if statement is false
mean(five_sequence) > median(five_sequence)
```

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

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.

```
#5.
student_names <- c("Alex", "Ben", "Carolina", "Damien")
test_scores <- c(100, 95, 90, 85)
on_scholarship <- c(TRUE, TRUE, FALSE, FALSE)

#6.
class(student_names)  #character vector
```

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

```
class(test_scores)  #numeric vector
```

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

```
class(on_scholarship)  #logical vector
```

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

```
#7.
df_students <- data.frame(student_names, test_scores, on_scholarship)

#8.
names(df_students) <- c("Names", "Test Scores", "Scholarship Holder")
```

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

Answer: This data frame contains elements of different types, whereas a matrix can only contain elements of the same type.

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

function_one <- function(x) {
  if(x > 50) {
    print("Pass")
  }
  else {
    print("Fail")
  }
}

#11. Create a function using ifelse()

function_two <- function(x){
  result <- ifelse(x > 50, "Pass", "Fail")
  print(result)
}

#12a. Run the first function with the value 52.5

function_one(52.5)
```

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

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

function_two(52.5)
```

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

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

#function_one(test_scores) did not work for reason stated under #14.

#13b. Run the second function with the vector of test scores

function_two(test_scores)
```

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

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

    Answer: The 'ifelse' option worked, because using 'if' ... 'else' evaluates only one condition, whereas 'ifelse' evaluates the condition for each element contained in the vector. The 'if' ... 'else' option will try to evaluate the entire vector against the logical condition in one swwop, which then creates an error.

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!)