Assignment 2: Coding Basics

Shiyi Zheng

OVERVIEW

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

Directions

1. Rename this file `<ShiyiZheng>_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.
7. Initial here to acknowledge that you did not use AI at all in completing this assignment: ShiyiZheng_____

Basics, Part 1

1. Use R's `seq()` function to create a sequence of numbers from 100 to 333, increasing by threes. Assign this sequence a variable name.
2. Compute the *mean* of this sequence, assigning this values its own variable name.
3. Compute the *standard deviation* (`sd()`) of this sequence, assigning this values its own variable name.
4. Display the the mean minus the standard deviation as well as the mean plus the standard deviation.
5. Insert comments in your code to describe what you are doing.

```
#1.
sequence_a <- seq(100,333,3) #sequence_a is the sequence of the number from 100 to 333 with an incracem
#2.
value_a <- mean(sequence_a) #value_a is the mean of sequence_a
#3.
value_b <- sd(sequence_a) #value_b is the standard deviation of sequence_a
#4.
c(value_c <- value_a-value_b) #value_c is the result of value_a(mean of sequence_a) minus value_b(stand
```

```
## [1] 147.5184
```

```r
c(value_d <- value_a+value_b) #value_d is the result of value_a(mean of sequence_a) plus value_b(standa
```

```
## [1] 283.4816
```

Basics, Part 2

6. 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).
7. Label each vector with a comment on what type of vector it is.
8. Combine each of the vectors into a data frame. Assign the data frame an informative name.
9. Label the columns of your data frame with informative titles.

```r
#6 & 7
vector_a <- c("Name1","Name2","Name3","Name4") #Character Vector
vector_b <- c(98,97,99,100) #Numeric Vector
vector_c <- c(TRUE,TRUE,FALSE,FALSE) #Logical Vector

#8
dataframe_student_record <- data.frame(vector_a,vector_b,vector_c)

#9
names(dataframe_student_record) <- c("Student_Name","Test_Scores","Scholarship_Status");
```

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

    Answer: This data frame contains 3 different types of vectors(data), which I have defined privously as character, numeric and logical. While a martix can only contains 1 type of vectors(data), which is numeric data for mathmatic calcultation mostly.Thus, data frame might be more flexible than matrix.

Basics, Part 3

11. 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, it returns the word “Pass”; otherwise it returns the word “Fail”.
12. Create a second function that does the exact same thing as the previous one but uses `ifelse()` instead if `if...else`.
13. Run both functions using the value 54 as the input
14. Run both functions using the **vector** of student test scores you created as the input. (Only one will work properly...)

```r
#11. Create a function using if...else
result_if_else <- function(x){
  if(x>50){return("Pass")}
  else{return("Fail")}
}


#12. Create a function using ifelse()
result_ifelse <-function(y){
  ifelse(y>50,"Pass","Fail")
  }

#13a. Run the first function with the value 54
result_if_else(54)
```

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

```r
#13b. Run the second function with the value 54
result_ifelse(54)
```

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

```r
#14a. Run the first function with the vector of test scores
#result_if_else(vector_b) #Error in if (x > 50) { : the condition has length > 1

#14b. Run the second function with the vector of test scores
result_ifelse(vector_b)
```

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

15. QUESTION: Which option of `if...else` vs. `ifelse` worked? Why? (Hint: search the web for "R vectorization"; it's ok here if an AI response is presented in the search response.)

    Answer:The option of "ifelse" worked.In R, the `if ... else` statement is not vectorized. It requires the result of the conditional expression to be a logical value of length 1 (i.e., a single TRUE or FALSE). When the input is a vector, conditional checks (like `x > 50`) return a logical vector containing multiple TRUE/FALSE values. The `if ... else` statement cannot process such results individually and thus fails to execute correctly.In contrast, `ifelse()` is a vectorized function. This means it can evaluate conditional expressions for each element in a vector individually and return a result vector matching the input vector's length. Thus, when a vector of student grades is input, `ifelse()` can check each grade individually to determine if it exceeds 50 and correctly return the corresponding "Pass" or "Fail".

NOTE Before knitting, you'll need to comment out the call to the function in Q14 that does not work. (A document can't knit if the code it contains causes an error!)