Assignment 2: Coding Basics

Lu Liu

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. generate the sequence and assign the sequence with #the name lu_sequence,
#print the sequence
lu_sequence <- seq(from = 1, to = 55, by = 5)
lu_sequence
```

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

```
#2. calculate the mean and median of the sequence and #assign name for the result
meanlu <- mean(lu_sequence)
medianlu <- median(lu_sequence)
#3. compare the result. If mean is greater than the median, #then the output is TRUE,
#else the result is FALSE
meanlu > medianlu
```

```
## [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("Bill","Bob","Peter","Diana")
test_scores <- c(90,89,79,83)
on_scholarship <- c(TRUE, TRUE,FALSE,FALSE)
#create the three vectors

#6
class(student_names) # character
```

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

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

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

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

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

```
#use the function class to know the type of the vectors

#7
student_data <- data.frame(student_names, test_scores, on_scholarship)
#combine the vectors into a data frame using data.frame function, then print the
#data.frame out, name it as "student_data"

#8
colnames(student_data) <- c("Name", "Score", "Scholarship_Status")
#rename columns for clarity
#View(student_data)
```

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

   Answer:A matrix can only hold a single data type while a data frame can hold multiple data types

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
eva_score <- function(scores){
    if(scores>50){
    print("Pass")
  }else{
    print("Fail")
  }
  }
#define and create the function, print "Pass" or "Fail" for #each individual score
#(for question 13a)

eva_score(80)
```

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

```
eva_score(50)
```

```
## [1] "Fail"
```

```
#test the function

#11. Create a function using ifelse()
eva_score_ifelse <- function(scores){
  result <- ifelse(scores>50,"Pass","Fail")
  print(result)
}
#define the create the function
eva_score_ifelse(80)
```

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

```
eva_score_ifelse(40)
```

```
## [1] "Fail"
```

```
#test the function

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

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

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

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

```r
#13a. Run the first function with the vector of test scores
#test_scores <- c(90,89,79,83)
#eva_score(test_scores)

#13b. Run the second function with the vector of test #scores
eva_score_ifelse(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()` function worked with the vector of test scores because it is vectorized, # meaning it can handle multiple inputs at once and perform element-wise operations. # The `if...else` construct is not vectorized and only works for a single logical value.

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!)