Assignment 2: Coding Basics

Brynn Rotbart

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.
vector1 <- c(1:55) #named valued 1-55 vector 1
vector1
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
## [51] 51 52 53 54 55
```

```
#2.
mean(vector1) #used the mean function = 28
```

```
## [1] 28
```

```
median(vector1) #used the median function = 28
```

```
## [1] 28
```

```r
#3.
mean(vector1)>median(vector1) #used the logical expression function to give me a TRUE/FALSE for this eq
```

```
## [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.

```r
student_names <-c("Sophie", "Aidan", "Vig", "Ky", "Justin") #Character vector
student_names
```

```
## [1] "Sophie" "Aidan"  "Vig"    "Ky"     "Justin"
```

```r
student_test_scores <-c(98, 80, 72, 82, 35) #numerical vector
student_test_scores
```

```
## [1] 98 80 72 82 35
```

```r
student_scholarship <-c(TRUE, FALSE, FALSE, TRUE, FALSE) #logical expression vector
student_scholarship
```

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

```r
Student_Information <- data.frame(student_names,
                                  student_test_scores,
                                student_scholarship)
Student_Information
```

```
##   student_names student_test_scores student_scholarship
## 1        Sophie                  98                TRUE
## 2         Aidan                  80               FALSE
## 3           Vig                  72               FALSE
## 4            Ky                  82                TRUE
## 5        Justin                  35               FALSE
```

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

   Answer: Matrices can only store one class of data wheras data frames can store many. As we see here, the data fram can store numerical, charachter, and logical expression vectors all at once.

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

#11. Create a function using ifelse()
student_scores_if_else <- function(x) {
  ifelse((x>50),{print("Pass")},{print("Fail")})
  }

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

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

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

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

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

```
#13a. Run the first function with the vector of test scores
#passing_result_dataset <- function(Student_Information) { if(Student_Information >50) {print("Pass")}

#passing_result_dataset(Student_Information)

#13b. Run the second function with the vector of test scores
passing_result_dataset_if_else <- function(Student_Information) {
  ifelse((Student_Information>50),{print("Pass")},{print("Fail")})
}
passing_result_dataset_if_else(Student_Information)
```

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

```
##      student_names student_test_scores student_scholarship
## [1,] "Pass"        "Pass"              "Fail"
## [2,] "Pass"        "Pass"              "Fail"
## [3,] "Pass"        "Pass"              "Fail"
## [4,] "Pass"        "Pass"              "Fail"
## [5,] "Pass"        "Fail"              "Fail"
```

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

    Answer:if... else can only assess one thing at a time but ifelse is vectorized allowing it to assess an entire function across the whole vector. Seems kind of similar to the differences between matrices and vectors

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!)