Assignment 2: Coding Basics

Sam Tolbert

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.
onetofiftyfive<- c(seq(1,55,5)) #creating and naming sequence

#2.
mean(onetofiftyfive)
```

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

```
median(onetofiftyfive) #finding mean and median
```

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

```
#3.
mean(onetofiftyfive)!=median(onetofiftyfive)#comparison
```

```
## [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.
vector_1<- c("adam" ,"bob", "cindi", "dave") #nominal vector
vector_2<- c(100, 90, 80, 70) #numerical vector
vector_3<- c(TRUE, TRUE, FALSE, FALSE) #logical vector

student_data<- data.frame(
  Name= vector_1,
  Grade= vector_2,
  Scholarship= vector_3
  )
```

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

   Answer: I think of data frames of Excel spreadsheets operating inside of R. They have rows and columns with different data sets of different types that I manipulate– through different commands-than matrices. Matrices are 2D arrays of data of the same type. They are most usable for linear algebra and things that I might need to get watch some math youtube videos in order to properly utilize, whereas data frames I can conceptualize more readily. I can multuply matrices together, they are mathetimcal instruments, I cannot do that for data frames, they are instruments for data visualization and comparison.

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
#Disclosure: asked R Wizard GPT for an example of if...else structure.
#The individual parts make sense
#but GPT helps with structure.

check_score<-function(score){
  if (score>50) {
    return("PASS")
  }else{
    return ("FAIL")
  }
}
```

```r
#11. Create a function using ifelse() Disclosure: Consulted
#with R Wizard GPT to debug (I had parentheses issues).
#It also helped explain the uses of "return as variable"
#and return(result)" which it explains is a little
#extraneous but will help when the logic gets
#more complicated, so I am using it.


check_score_fifty <- function(score) {
result <- (ifelse(score>50, "PASS", "FAIL"))
return(result)
}

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

check_score(52.5)
```

```
## [1] "PASS"
```

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

check_score_fifty(52.5)
```

```
## [1] "PASS"
```

```r
#13a. Run the first function with the vector of test scores
#check_score(vector_2)

#13b. Run the second function with the vector of test scores
check_score_fifty(vector_2)
```

```
## [1] "PASS" "PASS" "PASS" "PASS"
```

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

    Answer: ifelse worked because ifelse allows for vectorization. if...else will allow for one answer for one data, but ifelse will return results for an entire vector.

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!)