Assignment 2: Coding Basics

Jalal Bayar

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. I assigned a name to a sequence.

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

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

```
seq_numbers <- seq(1, 55, 5)
seq_numbers
```

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

```
#2. mean and median before the assigned name is the function to measure avg and med.

mean(seq_numbers)
```

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

```r
median(seq_numbers)
```

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

```r
avg <- mean(seq_numbers)
med <- median(seq_numbers)

#3. The statement is FALSE
avg > med
```

```
## [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
std_names <- c("josh", "maria", "sheran", "jhon") #vector type : character
test_scores <- c(75, 78, 72, 82) #vector type: numbers
fellow_ship <- c(FALSE, TRUE, FALSE, TRUE) #vector type : logical
```

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

   Answer: In dataframe, all the variables should be of equal length as in the above case (4 var each), however, this condition is not necessary in the matrix, variables could be of different length. Another difference is matrix have only one data structure such as numbers or text only, but data frame could have more than one data structure.

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...)

```r
#10. Create a function using if...else
# one input function

var_1 <- function(x) {
  if (x > 50) {
    print("Pass")
  }
  else {print
    ("Fail")
  }
}


#11. Create a function using ifelse()

var_2 <- function(x){
ifelse(x > 50, "Pass", "Fail") # log-exp, If TRUE, If FALSE
}


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

var_1(52.5)
```

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

```
#12b. Run the second function with the value 52.5
var_2(52.5)
```

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

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

#var_1 (test_scores)

# The function "if...else" returns an error because the condition has length > 1. This works for single

#13b. Run the second function with the vector of test scores
var_2(test_scores) # this function works
```

```
## [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" function works with many elements in the data frame, however, "if…else" only works with one element.

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!)