Assignment 2: Coding Basics

Jiyeong Pyo

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.

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

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

2. Compute the mean and median of this sequence.

```
mean(increase_seq)
```

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

```
median(increase_seq)
```

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

3. Ask R to determine whether the mean is greater than the median.

```
mean(increase_seq)>median(increase_seq)
```

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

4. Insert comments in your code to describe what you are doing.

```
#1. assign sequence 1 to 55 with increasing by fives named increase_seq

#2. compute mean and median of the increase_seq

#3. comparing mean and median of the increase_seq, the result will be false or true
```

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).

```
student_names <-c('Jiyeong','Nina','Joy','Bella')
student_names
```

```
## [1] "Jiyeong" "Nina"    "Joy"     "Bella"
```

```
test_scores <-c(40,100,95,92)
test_scores
```

```
## [1]  40 100  95  92
```

```
scholarship <- c(TRUE,FALSE,FALSE,TRUE)
scholarship
```

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

6. Label each vector with a comment on what type of vector it is.

```
class(student_names) #character
```

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

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

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

```
class(scholarship) #logical
```

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

7. Combine each of the vectors into a data frame. Assign the data frame an informative name.

```
df_student_data <- data.frame(student_names,test_scores,scholarship)
df_student_data
```

```
##   student_names test_scores scholarship
## 1       Jiyeong          40        TRUE
## 2          Nina         100       FALSE
## 3           Joy          95       FALSE
## 4         Bella          92        TRUE
```

```r
class(df_student_data)
```

```
## [1] "data.frame"
```

8. Label the columns of your data frame with informative titles.

```r
df_student_data <- data.frame("Name"=student_names,"Score"=test_scores,"Scholarship"=scholarship)
df_student_data
```

```
##      Name Score Scholarship
## 1 Jiyeong    40        TRUE
## 2    Nina   100       FALSE
## 3     Joy    95       FALSE
## 4   Bella    92        TRUE
```

```r
class(df_student_data)
```

```
## [1] "data.frame"
```

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

   Answer: Matrix always requires same class of data such as only numeric data. However, this data frame's column has different type of data, including character, numeric and logical.

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".

```r
Diagnose_test <- function(x){
  score <- x
  if(x>50){
    "Pass"
  }
  else
    "Fail"

}
```

11. Create a second function that does the exact same thing as the previous one but uses `ifelse()` instead if `if...else`.

```r
Diagnose_test_2 <- function(x){
  score <- x
  ifelse(x>50,"Pass","Fail")
}
```

12. Run both functions using the value 52.5 as the input

```
score=52.5
Diagnose_test(score)
```

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

```
Diagnose_test_2(score)
```

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

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

#11. Create a function using ifelse()

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

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

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

#13b. Run the second function with the vector of test scores
```

```
#Exam_1 <- function(x){
#  if(x>50){
#    "Pass"
#  }
#  else
#    "Fail"
#}
#Exam_1(52.5)
#Exam_1(test_scores)
```

```
Exam_2 <- function(x){
  ifelse(x>50,"Pass","Fail")
}

Exam_2(52.5)
```

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

```
Exam_2(test_scores)
```

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

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

    Answer: Only 'ifelse' worked because it is vectorized which means that it is determined by each element of the vector, however `if...else` determine by the first 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!)