Assignment 2: Coding Basics

Shidi Dai

OVERVIEW

This exercise accompanies the lessons 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 Sakai.

Basics Day 1

1. Generate a sequence of numbers from one to 100, increasing by fours. 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 generated a sequence 1-100 increasing by 4 and named it first_sequence.
seq(1, 100, 4)
```

```
##  [1]  1  5  9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89 93 97
```

```
first_sequence <- seq(1, 100, 4)
#2. I computed the mean and median.
mean(first_sequence)
```

```
## [1] 49
```

```
median(first_sequence)
```

```
## [1] 49
```

```
#3. I asked R to determine whether mean is greater than median.
mean(first_sequence) > median(first_sequence)
```

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

Basics Day 2

5. Create a series of vectors, each with four components, consisting of (a) names of students, (b) test scores out of a total 100 points, and (c) whether or not they have passed the test (TRUE or FALSE) with a passing grade of 50.

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.

```
#5,6
vec1 <- c("Alice", "Ben", "Charlie", "Diane") # character vector
vec2 <- c(40, 70, 90, 100) # numeric vector
vec3 <- c(FALSE, TRUE, TRUE, TRUE) # logical vector
#7
df_name_of_students <- as.data.frame(vec1)
df_test_score <- as.data.frame(vec2)
df_pass_or_not <- as.data.frame(vec3)
df_test_results <- cbind(df_name_of_students, df_test_score, df_pass_or_not)
df_test_results
```

```
##      vec1 vec2  vec3
## 1   Alice   40 FALSE
## 2     Ben   70  TRUE
## 3 Charlie   90  TRUE
## 4   Diane  100  TRUE
```

```
#8
names(df_test_results) <- c("Names", "Scores", "Pass")
df_test_results
```

```
##     Names Scores  Pass
## 1   Alice     40 FALSE
## 2     Ben     70  TRUE
## 3 Charlie     90  TRUE
## 4   Diane    100  TRUE
```

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

   Answer: Data frame is more general of a 2 dimensional structure than a matrix. Matrix can only contain elements of the same type (a single class of data), while data frame can contain elements of multiple types (different modes of data, e.g., numeric, factor, logical).

10. Create a function with an if/else statement. Your function should take a **vector** of test scores and print (not return) whether a given test score is a passing grade of 50 or above (TRUE or FALSE). You will need to choose either the `if` and `else` statements or the `ifelse` statement.
11. Apply your function to the vector with test scores that you created in number 5.

```
#10
results1 <- function(x){
  if(x<50){
    print(FALSE)
  }else{
    print(TRUE)
  }
}
results2 <- function(x){
  ifelse(x<50, FALSE, TRUE)
}

#11
#results1(vec2) does not work
```

```
for (i in c(1:length(vec2))){
  results1(vec2[i])
}
```

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

```
# if and else function does not work when I simply plug in vec2 as a vector to the function.
#It works when I plug in numeric values one by one.

results2(vec2)
```

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

```
pass_results <- results2(vec2)
pass_results
```

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

12. QUESTION: Which option of `if` and `else` vs. `ifelse` worked? Why?

    Answer: `ifelse` worked because it checks the condition for every element of a vector. The `if` and `else`statements are not vector operations and only deal with a single value. `if` and `else` function checks conditions of a single value and produces outcome of that value in one line. If I ask the `if` and `else` function to check the test score pass/fail results, I need to ask it to check one by one and it will provide outcome one by one in separate line. For `ifelse` function, it checks conditions of all four values in my vector and provides outcomes in one line.