Assignment 2: Coding Basics

Natasha Jacob

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on coding basics.

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your first and last name into the file name (e.g., "FirstLast_A02_CodingBasics.Rmd") prior to submission.

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. Sequence of numbers from 1 to 100 increasing by fours
Seq_one_four <- seq(1, 100, 4)
Seq_one_four
```

```
##  [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
```

```
# 2. Finding Mean and Median of Seq_one_four
mean(Seq_one_four)
```

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

```
median(Seq_one_four)
```

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

```
# 3.Using a conditional statement to determine whether the
# mean is greater than the median
mean(Seq_one_four) > mean(Seq_one_four)
```

```
## [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.

```
# Creating a series of vectors
student_names <- c("Natasha", "Tori", "Christie", "Molly")  #Vector type is character
test_scores <- c(95, 88, 40, 50)  #Vector type is numeric
pass_fail <- c("PASS", "PASS", "FAIL", "PASS")  #Vector type is character

# Class function is used to understand the type of object
class(student_names)
```

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

```
class(test_scores)
```

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

```
class(pass_fail)
```

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

```
# Combining the vectors into a data frame
Test_Results <- data.frame(Names_of_students = student_names,
    Test_Scores = test_scores, Result = pass_fail)
Test_Results
```

```
##   Names_of_students Test_Scores Result
## 1           Natasha          95   PASS
## 2              Tori          88   PASS
## 3          Christie          40   FAIL
## 4             Molly          50   PASS
```

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

   Answer: While matrix is a m*n array with similar data type, a data frame can contain multiple data types in multiple columns (it is a list of vector of equal length). This dataframe is a generalized form of a matrix.

10. Create a function with an if/else statement. Your function should determine whether a 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. Hint: Use `print`, not `return`. The name of your function should be informative.
11. Apply your function to the vector with test scores that you created in number 5.

```r
# Using the ifelse statement to create a function to
# determine whether a test score is a passing grade of 50
# or above
PassOrFailFunction <- function(x) {
    ifelse(x >= 50, print("pass"), print("fail"))
}

# Transforming test_scores from numeric to a dataframe to
# execute the next step
df_testscores <- as.data.frame(test_scores)
df_testscores
```

```
##   test_scores
## 1          95
## 2          88
## 3          40
## 4          50
```

```r
# Applying the function to the test scores data
Results_output <- PassOrFailFunction(df_testscores)
```

```
## [1] "pass"
## [1] "fail"
```

```r
Results_output
```

```
##      test_scores
## [1,] "pass"
## [2,] "pass"
## [3,] "fail"
## [4,] "pass"
```

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

    Answer: I tried both statements - if and else and ifelse but ifelse was the statement that worked. This is because the function with the if and else statement had a length greater than 1 and only the first element in the dataframe (95) was recognised and used. The function could not recongise the other elements in the data frame. Whereas, the function with the ifelse statement recognised the other elements in the dataframe.