Assignment 2: Coding Basics

Curtis Cha

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. Used function seq() with start = 1, end = 100 and assigned vector of numbers to "seq"
seq <- seq(1,100)

#2. Used functions mean() and median() on "seq" to get statistics. mean = 50.5, median = 50.5
mean(seq)
```

```
## [1] 50.5
```

```
median(seq)
```

```
## [1] 50.5
```

```
#3. Used operator ">" to determine if mean > median: False
mean(seq) > median(seq)
```

```
## [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. created three vectors, each with four items.

name <- c("Bill", "George", "Barack", "Donald")
scores <- runif(4, min = 0, max = 100)
pass <- scores > 50

#6. name vector is character data of names, scores is numerical data randomly generated from runif() fu

class(name)
```

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

```
class(scores)
```

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

```
class(pass)
```

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

```
#7. Use data.frame() to create dataframe with name, scores, and pass as columns

df1 <- data.frame(name, scores, pass)

#8. Assigned new column names to dataframe
names(df1) <- c("Name", "Test_Score", "Pass?")
```

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

   Answer: A matrix can only have data of the same data type. Meaning if one column in the matrix is numerical, the other columns must also be numerical. In contrast, the dataframe can have columns with different data types. The dataframe from 8. has three columns, one for numerical, character, and logical data.

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.

```
#10. Use ifelse() to create user-defined function for if numerical item > 50

grtr_thn_50 <- function(x) {
  ifelse(x > 50, print("Pass"), print("Fail"))
}

#11. Use for-loop to run grtr_thn_50 function on each item in "scores" vector from 5.

for (i in scores) {
  grtr_thn_50(i)
}
```

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

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

    Answer: Both options work but I used the ifelse() format since it's shorter and uses one line of code. It worked because I used the statement x>50 to produce the logical data "True" or "False". Since logical data is binary, there is no other value that can be produced from this statement. For "True" outcomes, then the ifelse will print "Pass". For "False" outcomes, then the ifelse will print "Fail".