Assignment 2: Coding Basics

Reed Leon-Hinton

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.

#creating a simple sequence of numbers from 1 to 100 which increases by 4 each time
#called problem_sequence
problem_sequence <- seq(1, 100, 4)

#testing to make certain it actually works
print(problem_sequence)
```

```
##  [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.

#generating the mean and median
summary(problem_sequence)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1      25      49      49      73      97
```

```
summary_sequence <- summary(problem_sequence)
sequence_mean <- summary_sequence[4] #extracting the mean
sequence_median <- summary_sequence[3] #extracting the median

#3.
```

```r
#writing an if statement with multiple statements to cover every scenario for
#greater than, equal to, or less than.
if(sequence_mean > sequence_median){
  print("The mean of the sequence is greater than the median of the sequence")
} else if(sequence_mean == sequence_median){
  print("The mean of the sequence is equal to the median of the sequence")
} else{
  print("The mean of the sequence is less than the median of the sequence")
}
```

```
## [1] "The mean of the sequence is equal to the median of the sequence"
```

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.

```r
#creating a vector of student names
names <- c("Robb Stark", "Sansa Stark", "Arya Stark", "Jon Snow") #character vector
print(names)
```

```
## [1] "Robb Stark"  "Sansa Stark" "Arya Stark"  "Jon Snow"
```

```r
#creating the vector of their test scores out of 100 points
scores <- c(73, 94, 42, 81) #numeric vector
print(scores)
```

```
## [1] 73 94 42 81
```

```r
#creating the vector telling whether or not the student passed
passing <- ifelse(scores >= 50, TRUE, FALSE) #logical vector
print(passing)
```

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

```r
#creating a data frame with these vectors inside
grade_registry <- data.frame(names, scores, passing)
names(grade_registry) <- c("Student Name", "Test Score", "Passing Status"); print(grade_registry)
```

```
##   Student Name Test Score Passing Status
## 1   Robb Stark         73           TRUE
## 2  Sansa Stark         94           TRUE
## 3   Arya Stark         42          FALSE
## 4     Jon Snow         81           TRUE
```

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

   Answer: A matrix is only capable of containing a single type of data, such as numbers or strings. However, a data frame is able to contain multiple elements of different data types. Like this one which contains string, integer, and logical values.

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.

```
#creating the function for checking grades
grade_check <- function(x) {
  if(x >= 50){
    print("This test score passes the minimum grade requirement")
  }
  else{
    print("This test score is below the minimum grade requirement: FAILURE")
  }
}

#checking each score in the vector. You have to use this method due to the nature
#of the if statement only handling a single input.
grade_check(scores[1])
```

```
## [1] "This test score passes the minimum grade requirement"
```

```
grade_check(scores[2])
```

```
## [1] "This test score passes the minimum grade requirement"
```

```
grade_check(scores[3])
```

```
## [1] "This test score is below the minimum grade requirement: FAILURE"
```

```
grade_check(scores[4])
```

```
## [1] "This test score passes the minimum grade requirement"
```

```
#This seems too labor intensive, let us check the other alternative
alt_grade_check <- function(x) {
  ifelse(x >= 50, "This score passed",
       "This score failed the requirement.")
}

#checking the score with this method
checked_grades <- alt_grade_check(scores)
print(checked_grades)
```

```
## [1] "This score passed"                  "This score passed"
## [3] "This score failed the requirement." "This score passed"
```

```
#this second option would be much better with a longer list
```

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

    Answer: I utilized both to try and make them work, since I had also used both before this question in the rest of the assignment. However, I found that, in this case, using the ifelse worked much better due to the potential to expand into larger datasets if necessary.