Assignment 2: Coding Basics

Melissa Merritt

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.

```
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
```

```
# The sequence function allows you to create a sequence of numbers. The 4 at
# the end shows that I want the sequence to count by 4s.

hundred_sequence <- seq(1, 100, 4)

# With the arrow I am assigning the name hundred_seqeunce to the function. This
# will allow me to preform other codes on this function.

mean(hundred_sequence)
```

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

```
median(hundred_sequence)
```

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

```
# You can see here that I was able to use the object name to run the mean and
# median functions here.

mean(hundred_sequence) > median(hundred_sequence)
```

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

```
# Here, I am asking R if the mean of the sequence is greater than 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.

```
student_names <- c("leslie", "mike", "charlotte", "cameron", "lizzy")  #character vector

# I have created the vector using the c() function. I also names the vector to
# allow me to have an easier time working with the vector in future code. I
# will repeat these steps for the next two vectors.

test_scores <- c(42, 39, 75, 67, 98)  #double vector

passing_score <- c(FALSE, FALSE, TRUE, TRUE, TRUE)  #logical vector

# I was able to check the vector types using the function typeof() with the
# vector name inserted.

Student_Test_Scores <- data.frame(`Student Names` = student_names, `Test Scores (%)` = test_scores,
    `Passed the Test` = passing_score)

# To create the data frame for the different vectors, I used the function
# data.frame() with the vector names included in the parentheses. I was able to
# assign names to the columns which represent the vector values. I assigned a
# name to the data frame that represented what the data is conveying.

Student_Test_Scores
```

```
##   Student.Names Test.Scores.... Passed.the.Test
## 1        leslie              42           FALSE
## 2          mike              39           FALSE
## 3     charlotte              75            TRUE
## 4       cameron              67            TRUE
## 5         lizzy              98            TRUE
```

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

   Answer: The data frame is different from a matrix because it is storing different types of data. This is evident when we look at the different vector classifications. The data in a data frame is tabular. Data in a matrix must all be the same data type (i.e. whether students passed or not).

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.

```r
ifelse(test_scores >= 50, TRUE, FALSE)
```

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

```r
# I was able to use the function ifelse to print the results of test scores
# passed on if they were a passing grade or not.
```

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

    Answer: The option of ifelse() worked because I was able to use the vector in the function. This allowed me to apply the function to the vector directly instead of having to assign the funtion. 'if' and 'else' seems to be better when you are inputing different values, but you would need to put each value from the vector in separately. It does not seem to be compatible with applying the function to the entire vector.