Assignment 2: Coding Basics

Megan McClaugherty

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. The seq function creates a sequence in the order of from, to, and by. The
# arrow indicates I named the object onehundred_sequence. The
# onehundred_sequence code shows that sequence below the code chunk.
onehundred_sequence <- seq(1, 100, 4)
onehundred_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. Here I use the mean and median functions with the onehundred_sequence I
# created.
mean(onehundred_sequence)
```

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

```
median(onehundred_sequence)
```

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

```
# 3. I used a the greater than symbol to ask whether the mean of the
# onehundred_sequence dataset is larger than the median.
mean(onehundred_sequence) > median(onehundred_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.

```
# 6
student_names <- c("Dorothy", "Blanch", "Rose", "Sophia")  #student_names is a character vector
student_names
```

```
## [1] "Dorothy" "Blanch"  "Rose"    "Sophia"
```

```
test_scores <- c(98, 0, 77, 80)  #test_scores is a numeric vector
test_scores
```

```
## [1] 98  0 77 80
```

```
passed_gradeabovefifty <- c(TRUE, FALSE, TRUE, TRUE)  #passed_gradeabovefifty is a logical vector
passed_gradeabovefifty
```

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

```
# 7
golden_student_grades <- data.frame(student_names, test_scores, passed_gradeabovefifty)
golden_student_grades
```

```
##   student_names test_scores passed_gradeabovefifty
## 1       Dorothy          98                   TRUE
## 2        Blanch           0                  FALSE
## 3          Rose          77                   TRUE
## 4        Sophia          80                   TRUE
```

```
# 8
colnames(golden_student_grades) <- c("Golden Student", "Test Score", "Passed")
golden_student_grades
```

```
##   Golden Student Test Score Passed
## 1        Dorothy         98   TRUE
## 2         Blanch          0  FALSE
## 3           Rose         77   TRUE
## 4         Sophia         80   TRUE
```

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

   Answer: This data frame is different from a matrix because it can have columns containing different types of vectors, while matrix columns must have all the same type of vector.

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.

```
passed <- ifelse(test_scores > 50, "TRUE", "FALSE")
passed
```

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

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

    Answer: ifelse worked. It worked because the function allows you to create logical condition, then runs the following codes for each element of a vector. If and else only works for vectors with a single element.