Assignment 2: Coding Basics

Claire Pajka

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, Part 1

1. Generate a sequence of numbers from one to 30, increasing by threes. 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.
seq(1,30,3)
```

```
##  [1]  1  4  7 10 13 16 19 22 25 28
```

```
sequence_basic_A02 <- seq(1,30,3) #assigning the sequence a name.
#2.
mean(sequence_basic_A02) #finding mean of newly named sequence
```

```
## [1] 14.5
```

```
median(sequence_basic_A02) #finding the median of newly named sequence
```

```
## [1] 14.5
```

```
#3.
mean(sequence_basic_A02) > median(sequence_basic_A02) #the mean is not greater than the median.
```

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

Basics, Part 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_name <- c("Jill","Zach", "Sarah", "Joe") #character data
test_score <- c(90, 45, 80, 60) #double data
pass_test <- test_score >= 50 #logical data
data.frame(student_name, test_score, pass_test)
```

```
##   student_name test_score pass_test
## 1         Jill         90      TRUE
## 2         Zach         45     FALSE
## 3        Sarah         80      TRUE
## 4          Joe         60      TRUE
```

```
student_passing <- data.frame(student_name, test_score, pass_test)
colnames(student_passing) <- c("Student", "Score", "Passed")
student_passing
```

```
##   Student Score Passed
## 1    Jill    90   TRUE
## 2    Zach    45  FALSE
## 3   Sarah    80   TRUE
## 4     Joe    60   TRUE
```

```
typeof(student_name)
```

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

```
typeof(test_score)
```

```
## [1] "double"
```

```
typeof(pass_test)
```

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

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

   Answer: A data frame can contain different types of data, like character in column 1, double data in column 2, and logical data in column 3. A matrix can only contain one type of data.

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.

```
passing_score <- function(test_score){
  if(test_score > 50){
    print ("Pass")
  } else {
    print ("Fail")
  }
}
ifelse(test_score > 50, "Pass", "Fail")
```

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

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

    Answer: ifelse worked. This is because it is working with a true/false scenario, and is not dealing with integers or further numerical calculations after the initial logical expression: 'pass' acts as the action preformed/outcome if true, and 'fail' acts as the action preformed/outcome if false.