Assignment 2: Coding Basics

Kelsie Roberton

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
seq(1, 100)
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100
```

```
seq(1,100,4) # from, to, by
```

```
##  [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
```

```
simplesequence <-seq(1,100,4)

#2.
mean(simplesequence)
```

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

```
median(simplesequence)
```

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

```
#3. conditional statement
mean(simplesequence)>median(simplesequence)
```

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

```
mean(simplesequence)==median(simplesequence)
```

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

```
#4 > can tell me if the first value is greater than the second value in the code chunk. == (equality) c
```

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.

```
students <- c( "jerry", "samantha", "cary", "simon", "dan")
total_test_score <- c(100)
jerry_testscore <- c(79) #grade for the first student
samantha_testscore <- c(82) #grade for the second student
cary_testscore <- c(96) #grade for the third student
simon_testscore <- c(71) #grade for the fourth student
dan_testscore <- c(88) #grade for the fifth student
if (simon_testscore > 50) {
  assign_what <- "PASS"
} else {
  assign_what <- "FAIL"
}
assign_what
```

```
## [1] "PASS"
```

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

   Answer: Matrices can only contain a single class of data, while data frames can consist of many different classes of 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.

```
if (jerry_testscore > 50) {
  assign_what <- "PASS"
} else if (samantha_testscore > 50){
  assign_what <-"PASS"
} else if (cary_testscore > 50){
  assign_what <-"PASS"
} else if (simon_testscore> 50){
  assign_what <-"PASS"
}else if (dan_testscore>50){
  assign_what <-"PASS"
} else {
  assign_what <- "FAIL"
}
assign_what
```

```
## [1] "PASS"
```

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

    Answer: An ‘if’ statement is used to execute a block of code if the specified condition is true. An ‘else’ statement is used to execute a block of code if the statement is false. The ifelse() is a conditional statement that allows the application to test a series of conditions in a prescribed order. In this function, I used all three code statements to execute what the assigned answer to the code would be if the test score was more or less than a test score of 50.