Assignment 2: Coding Basics

Hanna Bliska

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.
seq_by_four <- seq(1, 100, 4) #from one, to one hundred, by fours
seq_by_four
```

```
##  [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.
mean(seq_by_four) #calculate mean of seq_by_four
```

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

```
median(seq_by_four) #calculate median of seq_by_four
```

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

```
#3.
mean(seq_by_four) > median(seq_by_four) #asking R if the mean > median
```

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

```
#output will return TRUE or 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.

```
student_names <- c("Hanna", "Caroline", "Isaac", "Sam")
#this vector consists of characters
test_scores <- c(48, 90, 38, 95)
#this vector consists of numbers
passing_scores <- c(FALSE, TRUE, FALSE, TRUE)
#this vector consists of logical elements

df_student_names <- as.data.frame(student_names)
#start by turning one vector into a data frame
df_student_scores <- cbind(df_student_names, test_scores, passing_scores)
#add columns to the data frame to create a single data frame with all vectors

colnames(df_student_scores) <- c("student.first.name", "test.scores.numeric", "test.scores.passing")
#renaming columns more informative names
df_student_scores
```

```
##   student.first.name test.scores.numeric test.scores.passing
## 1              Hanna                  48               FALSE
## 2           Caroline                  90                TRUE
## 3              Isaac                  38               FALSE
## 4                Sam                  95                TRUE
```

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

   Answer: This data frame includes elements from different classes (characters, numbers, logical elements). A matrix would include elements from the same class (e.g., only numbers).

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.

```
check_students_pass <- function(test_scores) {
  ifelse(test_scores>=50,TRUE,FALSE) #log_exp, if TRUE, if false
}
#check_students_pass returns TRUE if the test score >= 50, FALSE if < 50

did_my_students_pass <-check_students_pass(test_scores)
#using function check_students_pass on my test_scores
did_my_students_pass #output
```

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

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

    Answer: When I tried to use the `if` and `else` option, I received an error that stated the condition has a length greater than 1. This is because I was asking `if` and `else` to evaluate all of the test scores in my vector, not a single test score. When I used `ifselse` I was able to evaluate all of the test scores in my vector. This is because `ifelse` can evaluate vectors.