# Assignment 2: Coding Basics

Sarah Kear

## 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.
# I am using the sequence function to generate a sequence from 1 to 30 by 3. It's function name is sequ
sequence1 <- seq(1, 30, 3)
sequence1
```

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

```
#2.
# I am using the mean() function to determine the mean of sequence1. the function name is mean.
mean <- mean(sequence1)
mean
```

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

```
# I am using the median() function to determine the median of sequence1. the function name is median.
median <- median(sequence1)
median
```

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

```
#3.
# created conditional statements for the R to determine if mean is greater than the median of sequence1
mean > median
```

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

```
mean < median
```

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

```
mean != median
```

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

```
mean == median
```

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

## 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.

```
#5.
students <-list(name = c("Sarah", "Gus", "Mary", "Preston"),
                  test_score = c(100, 49, 82, 24),
                  passed = c("True", "False", "True", "False"))
students
```

```
## $name
## [1] "Sarah"   "Gus"     "Mary"    "Preston"
##
## $test_score
## [1] 100  49  82  24
##
## $passed
## [1] "True"  "False" "True"  "False"
```

```
#6.
class(students$name) # this vector is a character.
```

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

```r
class(students$test_score) # this vector is numeric.
```

```
## [1] "numeric"
```

```r
class(students$passed) # this vector is a character.
```

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

```r
#7.
students_df <- as.data.frame(students) #turned the series of vectors into a dataframe.
students_df
```

```
##      name test_score passed
## 1   Sarah        100   True
## 2     Gus         49  False
## 3    Mary         82   True
## 4 Preston         24  False
```

```r
#8.
colnames(students_df)[colnames(students_df) == "name"] ="Name"
colnames(students_df)[colnames(students_df) == "test_score"] ="Test_Score"
colnames(students_df)[colnames(students_df) == "passed"] ="Passed_Test"
```

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

   Answer: A data frame is different from a matrix because it can hold multiple data classes unlike a matrix which can only hold a singular data class. For example, the students df I created hold both numeric and character data classes. A matrix would only be able to contain either my character or numeric data not both.

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
#10.
passed_test <- function(x){
  ifelse(x>=50, "TRUE", "FALSE")} #log_exp, if TRUE, if FALSE

#11.
passed_test_answer <- passed_test(students$test_score); passed_test_answer
```

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

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

    Answer: `ifelse` worked over the `if` and `else` statement because the `if` and `else` statement does not work with vectors, it needs a singular input.