

# Assignment 2: Coding Basics

Sujay Dhanagare

## OVERVIEW

This exercise accompanies the lessons/labs 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 Canvas.

## Basics, Part 1

1. Generate a sequence of numbers from one to 55, increasing by fives. 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.
sequence <- (0:10)*5 + 1

#2.
mean_sequence <- mean(sequence)
median_sequence <- median(sequence)

#3.
mean_sequence > median_sequence #If the mean is greater then R should return TRUE. If it is smaller or

## [1] FALSE
```

## Basics, Part 2

5. Create three vectors, each with four components, consisting of (a) student names, (b) test scores, and (c) whether they are on scholarship or not (TRUE or FALSE).
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("Harry", "Ron", "Hermione", "Draco") # Character
score <- c(59,38,87,65) # Numeric
scholarship <- c(FALSE, FALSE, TRUE, FALSE) # Logical

student_info <- data.frame(
  "Student_Name" = student_name,
  "Test_Score" = score,
  "Scholarship" = scholarship
)
```

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

Answer: Data frames can contain columns of different data types (e.g., numeric, character, logical)  
Matrices are homogeneous and all elements must be of the same data type.

10. Create a function with one input. In this function, use `if...else` to evaluate the value of the input: if it is greater than 50, print the word "Pass"; otherwise print the word "Fail".
11. Create a second function that does the exact same thing as the previous one but uses `ifelse()` instead of `if...else`.
12. Run both functions using the value 52.5 as the input
13. Run both functions using the **vector** of student test scores you created as the input. (Only one will work properly...)

*#10. Create a function using if...else*

```
function1 <- function(x) {
  if (x > 50)
  { "pass"
  }
  else {
    "fail"
  }
}
```

*#11. Create a function using ifelse()*

```
function2 <- function(x){
  ifelse(x > 50 , "pass", "fail")
}
```

*#12a. Run the first function with the value 52.5*

```
function1(52.5)
```

```
## [1] "pass"
```

*#12b. Run the second function with the value 52.5*

```
function2(52.5)
```

```
## [1] "pass"
```

```
#13a. Run the first function with the vector of test scores  
#function1(score)  
#13b. Run the second function with the vector of test scores  
function2(score)
```

```
## [1] "pass" "fail" "pass" "pass"
```

14. QUESTION: Which option of `if...else` vs. `ifelse` worked? Why? (Hint: search the web for “R vectorization”)

Answer: The second function, `ifelse` worked with the vector of test scores, whereas `if...else` did not work. After looking it up, I found out that Vectorization is the ability to perform operations on entire vectors or arrays at once, rather than using explicit loops. The ‘`if...else`’ is not vectorized and is designed to work with only single logical conditions. When it is given a vector, it only evaluates the first element and ignores the rest. On the other hand, `ifelse` is vectorized and works on each element of the vector.

**NOTE** Before knitting, you’ll need to comment out the call to the function in Q13 that does not work. (A document can’t knit if the code it contains causes an error!)