# LeonardoRueda_Assignment 2: Coding Basics

Leonardo Rueda

## 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. Answer:

seq_increasing_by_four <- seq(1, 100, 4) # Sequence from 1, to 100, increasing by 4

seq_increasing_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. Answer:

mean_seq <- mean(seq_increasing_by_four) # Mean of the sequence

mean_seq
```

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

```r
median_seq <- median(seq_increasing_by_four) # Median of the sequence

median_seq
```

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

```r
#3. Answer:

mean_seq > median_seq # Is the mean greater than the median?
```

```
## [1] 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.

```r
students <- c("Nicole", "Sam", "Manuel", "Joe") # 5.a. Vector 1. Names of students.
# This vector contains characters

students
```

```
## [1] "Nicole" "Sam"    "Manuel" "Joe"
```

```r
set.seed(123) # I set this seed to guarantee replicability of results regarding
# the vector of scores, which I generate randomly following a uniform distribution

test_scores <- round(runif(4, min = 0, max = 100),0) # 5.b. Vector 2. Test scores for each
# student. This vector contains numbers

test_scores
```

```
## [1] 29 79 41 88
```

```r
past_or_fail <- test_scores>50 # 5.c. Vector 3. Whether students pass or fail the test
# with a grade greater than 50. This is a logical vector

past_or_fail
```

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

```r
students_scores <- data.frame(students, test_scores, past_or_fail) # Data frame with the
# students' names, test scores and approval information

names(students_scores) <- c("Student","Test score","Passed?") # Renaming the data frame
# columns with the labels "Student", "Test Score" and "Passed?"

View(students_scores)
```

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

   **Answer:** This data frame has a more general structure compared to a matrix. In the data frame, columns can have a different nature (they can be numbers, strings, logical operators, etc.), while a matrix contains elements of the same type.

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
Passing_score <- function(x) {
ifelse(x>50, "TRUE", "FALSE") #x greater than 50, if yes then "TRUE", else then "FALSE"
}

Passing_score(test_scores) # Applying the function to the vector test_scores
```

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

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

    **Answer:** The option that worked was "ifelse" because the input of the function is a vector and "ifelse" only requires a line of code to test whether the values inside the vector comply with a specific condition. On the other hand, the option "if" and "else" would require more than one line of code, because we would need to specify that the function performs the validation for every item of the vector with a loop.