

# Assignment 2: Coding Basics

Diane Sanchez

## 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.  
# this is a sequence of Green Devil responses to survey.  
response_rate_GD <- seq( 1, 100, 4) # from, to, by  
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  
  
#2.  
  
mean(response_rate_GD)  
  
## [1] 49  
  
median(response_rate_GD)  
  
## [1] 49  
  
#I am running summary stats data on response rates  
  
#3.  
mean(response_rate_GD)> median(response_rate_GD)  
  
## [1] FALSE  
  
#this is a simple comparison of summary stats.
```

## 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.

```
#data_frame
#student_test_score
student <- c('Sam', 'LD', 'Phil', 'Abbey', 'Jarius', 'Geroldine')
Score <- c( 100 , 90 , 80 , 70 , 60 , 40 )
passed_test <- c( TRUE, TRUE, TRUE, TRUE, TRUE, FALSE)

student_test_score <- data.frame("Names"=student,"TestScores"= Score,"Test Results"=passed_test)
student_test_score
```

```
##      Names TestScores Test.Results
## 1      Sam         100          TRUE
## 2       LD          90          TRUE
## 3      Phil          80          TRUE
## 4     Abbey          70          TRUE
## 5     Jarius          60          TRUE
## 6 Geroldine          40         FALSE
```

```
#this is a data frame of studnet test results and whether they passed or not.
```

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

Answer: A data frame allows for multiple forms of data to be analyzed, while a matrix is used to analyze same type of data. For example question 8 would only work with a data frame because it is a combination of different vectors (names, test scores, and if students passed), the matrix could not do that.

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.

```
Pass_failure <- function(x) {
  if (x>50) {
    return (TRUE)
  }
  else {
    return (FALSE)
  }
}
Pass_failure (student_test_score$TestScores)
```

```
## Warning in if (x > 50) {: the condition has length > 1 and only the first
## element will be used
## [1] TRUE
```

```
Pass_fail <- function(x) {  
  ifelse (x >= 50, TRUE, FALSE)  
}  
Pass_fail (student_test_score$TestScores)
```

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

```
#this is a function of whether the student passed their test or not.
```

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

Answer: the `if else` function can only process one value at a time. As explained in the error message the `if` statement will only run on the first element, so in my case since the first value on my data frame was large than 50, the result was true. `ifelse` worked because it goes through all the values in the vector and will assign either true or false based on what the data values are assigned.