# Assignment 2: Coding Basics

Yilin Zhong

## 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. We generate a sequence of numbers from one to 100, increasing by fours and assign this sequence a n
hundred_sequence<-seq(1,100,4)
hundred_sequence
```

```
##  [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. We compute the mean and median of this sequence and assign them a name
means<-mean(hundred_sequence)
means
```

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

```
medians<-median(hundred_sequence)
medians
```

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

```
#3. In here we ask R to determine whether the mean is greater than the median
means > medians
```

```
## [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.

```
name<- c("John", "Vex", "Kat", "Mandy") #character
class(name)
```

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

```
score<- c(45,100,85,79) #numeric
class(score)
```

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

```
pass<- c("FALSE","TRUE","TRUE","TRUE") #character
class(pass)
```

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

```
name.df<-as.data.frame(name)
score.df<-as.data.frame(score)
pass.df<-as.data.frame(pass)

exam.df<-cbind(name.df,score.df,pass.df)
exam.df <- data.frame("student_name"=name,"exam_score"=score, "passing_status"=pass)

exam.df
```

```
##   student_name exam_score passing_status
## 1         John         45          FALSE
## 2          Vex        100           TRUE
## 3          Kat         85           TRUE
## 4        Mandy         79           TRUE
```

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

   Answer:Data frame is different from a matrix by matrix can only contain a single class of data, while data frame can contain different classes of data.

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.

```
status1<-function(x){
  if(x>=50) {
    x=TRUE
  } else{
    x=FALSE
  }
}

#report1<-status1(exam.df$exam_score);report1

status2<-function(x) {
  ifelse(x>=50,TRUE,FALSE)
}
report2<-status2(exam.df$exam_score);report2
```

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

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

    Answer: Only 'ifelse' function works because 'ifelse' function checks each element in a vector one at a time. However, 'if' and 'else' function can only check one element in a vector at one time, but our code try to check every element in the vector at once, so the function can not be performed.