Assignment 2: Coding Basics

Natalie von Turkovich

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. Finding a sequence from 1 to 100, at intervals of 4. Assigning this function to the name sequence1.
sequence1<-seq(1,100,4)
sequence1
```

```
##  [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. Using the mean and median function to find the mean and median of sequence1.
mean(sequence1)
```

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

```
median(sequence1)
```

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

```
#3. Using a conditional statement to find if the mean is bigger than the median of sequence1. The state
mean(sequence1)>median(sequence1)
```

```
## [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.

```
#5.
names<- c("John", "Paul", "George", "Ringo")
names #character vector
```

```
## [1] "John"   "Paul"   "George" "Ringo"
```

```
test_scores<- c(85, 45, 95, 100)
test_scores #integer vector
```

```
## [1]  85  45  95 100
```

```
pass_fail<- test_scores>50
pass_fail #logical vector
```

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

```
#7
class_grades.df<- data.frame(names, test_scores, pass_fail)
class_grades.df
```

```
##    names test_scores pass_fail
## 1   John          85      TRUE
## 2   Paul          45     FALSE
## 3 George          95      TRUE
## 4  Ringo         100      TRUE
```

```
#8
colnames(class_grades.df) <- c('Student Names','Score','Passed Class')
class_grades.df
```

```
##   Student Names Score Passed Class
## 1          John    85         TRUE
## 2          Paul    45        FALSE
## 3        George    95         TRUE
## 4         Ringo   100         TRUE
```

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

   Answer: The difference between a data frame and a matrix is that a martix can only contain one class of data and a data frame can contain many different classes of data.

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.

```
#10
passing_grade<-function(x){ifelse(x>50, "pass", "fail")}

#11
passing_grade(test_scores)
```

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

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

    Answer: The ifelse function worked well in this case. It is a more concise way to code using a logical expression to decide the output of the function. I believe you could get the same result with the if and else functions used separately but it would not be as effcient, it would use more lines of code.