Assignment 2: Coding Basics

Meghan Seyler

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.
increasingbyfours<-seq(1,100,4) #here I created a sequence that increases by fours and assigned it the
increasingbyfours
```

```
##  [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_increasingbyfours<-mean(increasingbyfours) #taking the mean of increasingbyfours and assigning the
mean_increasingbyfours
```

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

```
median_increasingbyfours<-median(increasingbyfours) #taking the median of increasingbyfours and assigni
median_increasingbyfours
```

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

```
#3.
#testing statements that will generate a true or false output. The mean is NOT greater than the median
mean_increasingbyfours>median_increasingbyfours
```

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

```
mean_increasingbyfours<median_increasingbyfours
```

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

```
mean_increasingbyfours==median_increasingbyfours
```

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

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.
studentnames<-c ("Aislinn", "Tay", "Eva", "John") #character

testscores<-c (96,60,45,5) #double

pass<-c (TRUE,FALSE,FALSE,FALSE) #logical

#6.
typeof(studentnames)
```

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

```
typeof(testscores)
```

```
## [1] "double"
```

```
typeof(pass)
```

```
## [1] "logical"
```

```
#7
studentscores<- data.frame(studentnames,testscores,pass)
studentscores
```

```
##   studentnames testscores  pass
## 1      Aislinn         96  TRUE
## 2          Tay         60 FALSE
## 3          Eva         45 FALSE
## 4         John          5 FALSE
```

```
#8
colnames(studentscores)<- c("First_Name", "Total_Score", "PASS(T/F)")
studentscores
```

```
##   First_Name Total_Score PASS(T/F)
## 1    Aislinn          96      TRUE
## 2        Tay          60     FALSE
## 3        Eva          45     FALSE
## 4       John           5     FALSE
```

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

   Answer: In a matrix all columns must have the same mode and length. A data frame can consist of many modes. This is demonstrated by the data frame above which consists of logical, character, and numerical modes.

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.

```
determinepass<- if(testscores>=50) {"Pass"} else{"Fail"}
```

```
## Warning in if (testscores >= 50) {: the condition has length > 1 and only the
## first element will be used
```

```
print(determinepass)
```

```
## [1] "Pass"
```

```
determinepass2<-ifelse(testscores>=50, "Pass","Fail")
print(determinepass2)
```

```
## [1] "Pass" "Pass" "Fail" "Fail"
```

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

    Answer: The 'ifelse' option worked better because in 'if' and 'else' only the first element was used, so it only indicated if the first test score was a pass or fail.