Assignment 2: Coding Basics

Gaby Antonova

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.

```r
seq1 <- seq(1,100,4) #1 assign the first function/sequence to "seq1"
seq1 #2 generate the sequence, 1 to 100 by 4s
```

```
##  [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
```

[1] 1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 [22] 85 89 93 97

2. Compute the mean and median of this sequence.

```r
summary(seq1) #3 compute summary statistic
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1      25      49      49      73      97
```

Min. 1st Qu. Median Mean 3rd Qu. Max. 1 25 49 49 73 97

3. Ask R to determine whether the mean is greater than the median.

```
mean <- mean(seq1) #4 assign "mean" to the mean value of seq1
median <- median(seq1) #5 assign "median" to the median value of seq1

#use if else statement to compare the median and mean

{
  if (mean>median)
{"mean"}
else if (median>mean)
{"median"}
else (mean=median)
  {"equal"}
}
```

```
## [1] "equal"
```

4. Insert comments in your code to describe what you are doing.

```
#1. see above comments
```

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.

```
#generate character vector for names of students
name <- c("Jenna", "Gal", "Lupita", "Nick")
name
```

```
## [1] "Jenna"  "Gal"    "Lupita" "Nick"
```

```
class(name)
```

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

```
#generate numeric vector for student test scores
score <- c(40, 52, 69, 61)
score
```

```
## [1] 40 52 69 61
```

```
class(score)
```

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

```
#generate logical vector for whether students passed
pass2 <- score >=50

pass2
```

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

```
class(pass2)
```

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

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.

```
df_name <- as.data.frame(name)
df_name
```

```
##     name
## 1  Jenna
## 2    Gal
## 3 Lupita
## 4   Nick
```

```
df_pass <- as.data.frame(pass2)
df_pass
```

```
##   pass2
## 1 FALSE
## 2  TRUE
## 3  TRUE
## 4  TRUE
```

```
df_score <- as.data.frame(score)
df_score
```

```
##   score
## 1    40
## 2    52
## 3    69
## 4    61
```

```
 df_StudentScores <- cbind(df_name, df_score, df_pass)
class(df_StudentScores)
```

```
## [1] "data.frame"
```

```
df_StudentScores
```

```
##     name score pass2
## 1  Jenna    40 FALSE
## 2    Gal    52  TRUE
## 3 Lupita    69  TRUE
## 4   Nick    61  TRUE
```

8. Label the columns of your data frame with informative titles.

```
names(df_StudentScores)[names(df_StudentScores) == "name"] <- "student_name"

names(df_StudentScores)[names(df_StudentScores) == "name"] <- "student_name"

names(df_StudentScores)[names(df_StudentScores) == "pass2"] <- "Passed"

df_StudentScores
```

```
##   student_name score Passed
## 1        Jenna    40  FALSE
## 2          Gal    52   TRUE
## 3       Lupita    69   TRUE
## 4         Nick    61   TRUE
```

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

   Answer: Matrices can only contain a single class of data while data frames can contain multiple classes of data. In this data frame, logical, character, and numeric data are all stored.

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.

```
df_score
```

```
##   score
## 1    40
## 2    52
## 3    69
## 4    61
```

```
Test_Scores_Pass <- function(df_score){
  ifelse(df_score>=50, "True", "False")
}

print(Test_Scores_Pass(df_score))
```

```
##      score
## [1,] "False"
## [2,] "True"
## [3,] "True"
## [4,] "True"
```

11. Apply your function to the vector with test scores that you created in number 5.

```
Test_Scores_Pass_vector <- function(score){
  ifelse(score>=50, "True", "False")
}

print(Test_Scores_Pass(score))
```

```
## [1] "False" "True"  "True"  "True"
```

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

    Answer: "Ifelse" worked for my code because conditional statement deal only with a single value, while my vector contains four values stored under the same name. If pass a vector in an if statement, it only checks the very first element and issues a warning.