Assignment 2: Coding Basics

Halina Malinowski

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.
seq1_100 <- seq(1, 100, 4);seq1_100
```

```
##  [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
```

```
#Here, I created a sequence of numbers 1 to 100 by 4s


#2.
mean_seq1_100 <- mean(seq1_100); mean_seq1_100
```

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

```
median_seq1_100 <- median(seq1_100); median_seq1_100
```

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

```
#These 2 functions calculate the mean and median of the previously created sequence, respectively

#3.
(mean_seq1_100 > median_seq1_100)
```

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

```
#Here, I'm testing to see if the mean of the sequence is greater than the median of the sequence
```

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.
students <- c("Emma", "Alyssa", "Carli", "Lucy");students
```

```
## [1] "Emma"   "Alyssa" "Carli"  "Lucy"
```

```
#This is a vector of students, these are characters,and the data is qualitative and nominal
class(students)
```

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

```
testscores <- c(100, 48, 93, 87);testscores
```

```
## [1] 100  48  93  87
```

```
#This is a vector of student test scores, it is numerical, and the data is quantitative and discrete
class(testscores)
```

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

```
passfail <- c("TRUE", "FALSE", "TRUE", "TRUE");passfail
```

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

```
#This is a vector of of whether students passed or failed their test, it is a character, and the data i
class(passfail)
```

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

```r
#6. Labels for vectors in #5

#7. Create data frame
df <- cbind(students, testscores, passfail)
class_score_df <- as.data.frame(df)
class(class_score_df)
```

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

```r
#8. Label columns of data frame
names(class_score_df)
```

```
## [1] "students"   "testscores" "passfail"
```

```r
class_score_df <- data.frame("Student.Name"=students,"Test.Score"=testscores, "Pass.Fail" = passfail); c
```

```
##   Student.Name Test.Score Pass.Fail
## 1         Emma        100      TRUE
## 2       Alyssa         48     FALSE
## 3        Carli         93      TRUE
## 4         Lucy         87      TRUE
```

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

   Answer: A data frame is different from a matrix in that it can hold different types of data. The vectors that go into a data frame can have numbers, characters, or various types of data. Matrices can only hold data of the same type.

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.

```r
#10. Create if else function
test_scores_pass <- function(x) {
  if(x < 50) {
    print("FALSE")
  }
  else {
    print("TRUE")
  }
}

answer_passfail <- test_scores_pass(testscores)
```

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

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

```r
#OR
test_scores_pass2 <- function(x){
  ifelse(x < 50, print("FALSE"), print("TRUE"))

}

#11. Apply function to test scores vector
answers_passfail2 <- test_scores_pass2(testscores)
```

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

```r
answers_passfail2
```

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

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

    Answer: The 'ifelse' option worked because it was able to run through all of the scores in the testscores vector, while the 'if' and 'else' option did not work. The 'if' and 'else' option gave an error message explaining how the function length was for only one value and would only give the answer for the first test score. Therefore, this function could not run through the entire vector of testscores. However, you could use this function to check one score at a time. Otherwise, a for loop would be needed to be able to tell R to run through all of the options.