Assignment 2: Coding Basics

Ricky Prophete

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.
seq(1, 100, 4) #create sequence function
```

```
##  [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
```

```
one_hundred_by_four <- seq(1, 100, 4) #assign name to sequence function

#2.
mean(one_hundred_by_four) #compute mean of sequence function
```

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

```
median(one_hundred_by_four) #compute median of sequence function
```

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

```
#3.
x <- mean(one_hundred_by_four) #set x == to mean of sequence function
y <- median(one_hundred_by_four) #set y == to median of sequence function

x > y
```

```
## [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.

```
#a.
vector_student_names <- c("Sally", "Jack", "Chris", "Miranda") #character vector
#b.
vector_test_score <- c(58, 46, 83, 94) #numeric vector
#c.
vector_pass_result <- c(TRUE, FALSE, TRUE, TRUE) #logical vector
#data frame
dataframe_student_performance <- data.frame(vector_student_names,vector_test_score, vector_pass_result)
names(dataframe_student_performance) <- c("Name","Score","Passed"); View(dataframe_student_performance)

dataframe_student_performance #dataframe test
```

```
##      Name Score Passed
## 1   Sally    58   TRUE
## 2    Jack    46  FALSE
## 3   Chris    83   TRUE
## 4 Miranda    94   TRUE
```

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

   Answer: This data frame differs in that it combines elements of different types (numeric, logical, & character), whereas a matrix can only contain elements 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.

```
#10.
logical_test <- function(x){
  ifelse(x >= 50, print(TRUE), print(FALSE))
} #determine whether score is >= 50

#11.
logical_test(vector_test_score) #apply function to scores
```

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

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

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

    Answer: 'ifelse' worked as a shorthand alternative to the inclusion of both 'if' and 'else'. This function was a logical test of whether the x >=50 condition was met and included return values for true and false elements of the test condition.