Assignment 2: Coding Basics

Danielle Butler

OVERVIEW

This exercise accompanies the lessons/labs in Environmental Data Analytics on coding basics.

Directions

1. Rename this file `<FirstLast>_A02_CodingBasics.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure to **answer the questions** in this assignment document.
5. When you have completed the assignment, **Knit** the text and code into a single PDF file.
6. After Knitting, submit the completed exercise (PDF file) to Canvas.

Basics, Part 1

1. Generate a sequence of numbers from one to 55, increasing by fives. 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.

```r
#1. Creating a vector for a sequence of numbers 1-55 by 5
sequence1 <- c(seq(from=1,to=55,by=5))
sequence1
```

```
##  [1]  1  6 11 16 21 26 31 36 41 46 51
```

```r
#2. Computing the mean and median of the sequence
mean(sequence1)
```

```
## [1] 26
```

```r
median(sequence1)
```

```
## [1] 26
```

```r
#3. Determining if mean is greater than median
if (mean(sequence1) > median(sequence1)) {
  print("The mean is greater than the median.")
} else {
  print("The mean is not greater than the median.")
}
```

```
## [1] "The mean is not greater than the median."
```

Basics, Part 2

5. Create three vectors, each with four components, consisting of (a) student names, (b) test scores, and (c) whether they are on scholarship or not (TRUE or FALSE).
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.

```r
#Creating 3 vectors with data provided.
student_names <- c("Bob","Joe","Sally","Michelle") #Vector type: Character
test_scores <- c(100,90,80,70) #Vector type: Numeric
scholarships <- c(TRUE,FALSE,TRUE,FALSE) #Vector type: Logic

StudentInfo <- data.frame(student_names,test_scores,scholarships)
names(StudentInfo) <- c("Names","Scores","Scholarships")

#View(StudentInfo) #need to comment out view when knitting
```

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

   Answer: A matrix only has 2 dimensions that contains elements of the same type. This data frame has 3 different types of data.

10. Create a function with one input. In this function, use `if...else` to evaluate the value of the input: if it is greater than 50, print the word "Pass"; otherwise print the word "Fail".

```r
#10. Create a function using if...else
 function50 <- function(x){
  if(x > 50) {print("Pass")
  }
  else {
    print("Fail")
  }
 }
function50(55)
```

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

11. Create a second function that does the exact same thing as the previous one but uses `ifelse()` instead if `if...else`.

```
#11. Create a function using ifelse()

 newfunction50 <- function(x){
  ifelse(x>50,"Pass","Fail")
 }

newfunction50(10)
```

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

12. Run both functions using the value 52.5 as the input

```
#12a. Run the first function with the value 52.5

function50(52.5)
```

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

```
#12b. Run the second function with the value 52.5

newfunction50(52.5)
```

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

13. Run both functions using the **vector** of student test scores you created as the input. (Only one will work properly...)

```
#13a. Run the first function with the vector of test scores
#function50(test_scores)

#13b. Run the second function with the vector of test scores
newfunction50(test_scores)
```

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

14. QUESTION: Which option of `if...else` vs. `ifelse` worked? Why? (Hint: search the web for "R vectorization")

    Answer: ifelse worked! this function works with entire vectors (or columns) of dta at once while
    the if-else statement operates on individual values

NOTE Before knitting, you'll need to comment out the call to the function in Q13 that does not work. (A document can't knit if the code it contains causes an error!)