Assignment 2: Coding Basics

Becca Cox

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.

```
seq(1, 55, 5) # from, to, by
```

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

```
basicseq <- seq(1, 55, 5) #assign a name
```

2. Compute the mean and median of this sequence.

```
mean(basicseq) # calculate mean
```

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

```
median(basicseq) # calculate median
```

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

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

```
mean(basicseq) > median(basicseq) # mean > median is false
```

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

```r
mean(basicseq) < median(basicseq) # mean < median is also false
```

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

```r
mean(basicseq) == median(basicseq) #true -- they are the same
```

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

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

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.

```r
studentnames <- c("John","Beth","Sam","Becca") #student names # Character

testscores <- c(62,60,98,20) # test scores # Numeric

scholarship <- c(TRUE,FALSE,FALSE,TRUE) # scholarship # Logical
```

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
studentscoresandstatus <- data.frame(studentnames,testscores,scholarship) #Combine the vectors into a d

names(studentscoresandstatus) <- c("Name","Test Score","Scholarship Recipient") #Set the column names
```

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

   Answer: In a matrix, the data classes (i.e. the type of vector) need to be the same. Our dataframe allows us to use information from three different data classes: student names are character, test scores are numeric, and receipt of a scholarship is logical.

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
# Create a function using if...else

basics3 <- function(x) {
  if(x > 50) {
    print("Pass")
  }
  else {
    print("Fail")
  }
}

basics3(60) # example
```

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

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

```
# Create a function using ifelse()

basics3ifelse <- function(x){
  ifelse(x>50, "Pass", "Fail") #log_exp, if TRUE, if FALSE
}

basics3ifelse(60) # example
```

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

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

```
# Run the first function with the value 52.5

# Run the second function with the value 52.5

basics3(52.5)
```

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

```
basics3ifelse(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...)

```
# Run the first function with the vector of test scores

# Run the second function with the vector of test scores

# basics3(testscores) # Error

basics3ifelse(testscores) # Success
```

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

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

    Answer: "ifelse" worked and " 'if'...'else' " did not. My research on "R vectorization" revealed that "if" statements are not vectorized, and only work "if there is a single TRUE or FALSE value in the if statement" (source: Displayr), which cannot be the case with the multiple values of a vector. The "ifelse" function allows each component of the function to occur on each element of the vector at the same time, so there can be more than one value in the input/output.

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!)