Assignment 2: Coding Basics

Camille DeSisto

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. create a sequence from 1 to 100 by 4 and name the sequence "seq1"
seq1 <- seq(1, 100, 4)

#2. calculate the mean and median of the sequence

mean(seq1) #mean is 49
```

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

```
median(seq1) #median is 49
```

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

```
#the mean and median of the sequence are both 49

#3. use conditional statement to determine if the mean is greater than the median

mean(seq1)>median(seq1)
```

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

```
#this returns a "FALSE" because they are equal so the mean cannot be greater than the median
```

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

# A) vector of student names. This is a character vector
names <- c("Martin", "Isabelle", "Peter", "Cooper")

# B) vector of test scores out of 100. This is a numeric vector. Specifically, this double vector.
scores <- c(98, 43, 83, 94)

#C) function for whether or not the students pass the test and then make vector from output (although I

#function
pass_or_fail <- function(x){
  x>50
}

#vector from function output
pass <- pass_or_fail(scores)

#6 (comments included in the code above)

#7 create data from from vectors

exam_results <- data.frame("Names"=names, "Scores"=scores, "Pass"=pass)

# 8 create informative column names in data frame
#I did this already above in the "data.frame" function.

str(exam_results)
```

```
## 'data.frame':    4 obs. of  3 variables:
##  $ Names : chr  "Martin" "Isabelle" "Peter" "Cooper"
##  $ Scores: num  98 43 83 94
##  $ Pass  : logi  TRUE FALSE TRUE TRUE
```

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

   Answer: A matrix is data in a two-dimensional space that is composed of multiple vectors and has a fixed number of rows and columns. Unlike matrices, data frames always have column names, and having columns is one reason why one would transform a vector into a data frame. A data frame is a type of matrix that stores multiple data types and is made up of vectors that have equal lengths. Data frames can have different numbers of rows and columns. In a data frame, but not in a matrix, data can be multiple types (character, numeric, or factor). Converting data to a data frame allows one more freedom to work with the dataset in future analyses. When you add a vector to a data frame, the restulting object will always be a data frame. There are also functions in R to transform vectors into data frames.

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.

```
# function to determine if the student passes or fails the test. This is a slightly modified version of

pass_or_fail2 <- function(x){
  ifelse(x>50, TRUE, FALSE)
}

#also try with an if/else statement
pass_or_fail3 <- function(x){
  if(x>50){
    "TRUE"
  } else {
    "FALSE"
  }
}
```

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

```
#vector from function output
pass <- pass_or_fail2(scores)

#also try with the if/else statement function
pass2 <- pass_or_fail3(scores)
```

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

```
# this function does not work
```

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

    Answer: the "ifelse" statement worked best because the condition will work for vectors that have a length of greater than 1, whereas the "if/else" statement returns an error then vectors are used and only provides an output for the first number in the vector. Thus, the "if/else" function works well for single values but does not work well for multiples values that are in a vector.