Assignment 2: Coding Basics

Jess Garcia

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.

```r
#1. Generating a sequence from 1 to 100, increasing by fours, and naming that sequence

#Generating sequence
seq(1,100,4)
```

```
##  [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
```

```r
#Naming sequence
new_sequence <- seq(1,100,4)


#2. Computing mean and median of newly created sequence
mean(new_sequence)
```

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

```r
median(new_sequence)
```

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

```r
#3. Asking R if mean is greater than median in sequence
mean(new_sequence)>median(new_sequence)
```

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

```r
#If true, then mean is greater than the median of this sequence
#If false, then mean is less than or equal to the median of this 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/#6. Creating Vectors

# Creating vector for student names
student_names <- c("Anna","Bill","Cece","Dan")
student_names
```

```
## [1] "Anna" "Bill" "Cece" "Dan"
```

```
  #Vector type =character
  class(student_names)
```

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

```
#Creating vector for student scores
test_score <- c(40,80,90,70)
test_score
```

```
## [1] 40 80 90 70
```

```
  #Vector type =numeric
  class(test_score)
```

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

```
#Creating vector for passing or failing test
test_passed <- c(FALSE,TRUE,TRUE,TRUE)
test_passed
```

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

```
  #Vector type =logical
  class(test_passed)
```

```
## [1] "logical"
```

```
#7/#8. Combining vectors into a data frame
#Creating data frame of student tests
Student_Tests_df <-data.frame ("Student"=student_names,"Score"=test_score,"Passed"=test_passed)
Student_Tests_df
```

```
##   Student Score Passed
## 1    Anna    40  FALSE
## 2    Bill    80   TRUE
## 3    Cece    90   TRUE
## 4     Dan    70   TRUE
```

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

   Answer: This data frame has vectors of equal length but has different modes (character, numeric, logical), unlike a matrix where all columns must have the same length and mode.

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/#11. Creating a function to determine passing grade & apply the function
#Create the function using "if" and "else"
test_status1 <-function(test_score){
  if(test_score>=50){print("Pass")}
  else{print("Fail")}
  }
test_status1(test_score)
```

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

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

```
  #Only works by inputting individual scores, e.g.:
  test_status1(50)
```

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

```
#Create the function using "ifelse"
test_status2 <-function(test_score){
  ifelse(test_score>=50,"Pass","Fail")}
  #Runs all the test scores from the test_score vector at once, apply to function to the vector
  test_status2(test_score)
```

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

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

    Answer: The combined "ifelse" statement worked while the separate "if" and "else" did not. Trying the "if" and "else" when inputing the test scores gave a warning that only the first element would be used because the vector has a length greater than one, so it will not apply to more than one score."Ifelse" applies it to all scores, and so running it will give you the test status for all of the scores in the created vector.