Assignment 2: Coding Basics

Claire Mullaney

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 last name into the file name (e.g., "Salk_A02_CodingBasics.Rmd") prior to submission.

The completed exercise is due on Tuesday, January 21 at 1:00 pm.

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.  Creating a sequence of numbers that goes from 1 to 100, counting by 4
seq100 <- seq(1, 100, 4)
seq100
```

```
##  [1]  1  5  9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89
## [24] 93 97
```

```
# 2.  Calculating the mean of this sequence
mean.100 <- mean(seq100)
mean.100
```

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

```
# Calculating the median of this sequence
median.100 <- median(seq100)
median.100
```

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

```
# 3. checking to see if the mean is greater than the median
mean.100 > median.100
```

```
## [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.

```
# 5 (a) character vector containing the names of students
students <- c("Maggie", "Sean", "Josh", "Brooklyn")
students
```

```
## [1] "Maggie"   "Sean"     "Josh"     "Brooklyn"
```

```
## (b) numeric vector containing test scores
test.scores <- c(97, 85, 43, 64)
test.scores
```

```
## [1] 97 85 43 64
```

```
## (c) logical vector indicating if students passed the test (T) or failed
## the test (F)
pass.fail <- c(T, T, F, T)
pass.fail
```

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

```
# 7 Combining vectors into a data frame
student.scores <- data.frame(students, test.scores, pass.fail)
student.scores
```

```
##   students test.scores pass.fail
## 1   Maggie          97      TRUE
## 2     Sean          85      TRUE
## 3     Josh          43     FALSE
## 4 Brooklyn          64      TRUE
```

```
# 8 renaming the columns of the data frame
names(student.scores) <- c("Student", "Test Score", "Score >= 50")
student.scores
```

```
##    Student Test Score Score >= 50
## 1   Maggie         97        TRUE
## 2     Sean         85        TRUE
## 3     Josh         43       FALSE
## 4 Brooklyn         64        TRUE
```

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

   Answer: A matrix is a 2D structure containing elements that are all of the same type, while a data frame is a 2D structure that can contain data of different types. This data frame contains numeric, character, and logical data; if it were a matrix, it could only contain elements of one of these modes.

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. function to find passing test scores using ifelse
pass.ifelse <- function(x) {
    ifelse(x >= 50, T, F)
}
```

```r
# function to find passing test scores using if and else
pass.if.else <- function(x) {
    if (x >= 50) {
        print(TRUE)
    } else {
        print(FALSE)
    }
}

# 11 applying functions to the test score vector above
pass.ifelse(test.scores)
```

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

```r
pass.if.else(test.scores)
```

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

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

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

    Answer: `ifelse` worked, while `if` and `else` did not. `ifelse` preserved the shape of the original test scores vector, evaluating each vector component to return the same number of T and F values as there were scores. Using `if` and `else` only allowed one value (the first) from the test scores vector to be evaluated; it is not set up to loop through all values in the vector.