Assignment 2: Coding Basics

Queenie Wei

OVERVIEW

This exercise accompanies the lessons 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 Sakai.

Basics, Part 1

1. Generate a sequence of numbers from one to 30, increasing by threes. 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.

seqonetothirtybythree <- seq(1,30,3)
seqonetothirtybythree
```

```
##  [1]  1  4  7 10 13 16 19 22 25 28
```

```
# this is to generate a sequence that goes from 1 to 30, increasing by 3s.

#2.

mean(seqonetothirtybythree)
```

```
## [1] 14.5
```

```
median(seqonetothirtybythree)
```

```
## [1] 14.5
```

```
### This is for calculating the mean of the sequence in question 1

#3.

mean(seq(1,30,3)) > median(seq(1,30,3))
```

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

```
#### The code above tries to see if the mean of the sequence in question 1 is
####larger than the median of the sequence.
```

Basics, Part 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
## 5a
names <- as.vector(c("fudge", "muffin", "seal", "kangaroo")) ###vector type: char
names                            ##typeof(names)
```

```
## [1] "fudge"    "muffin"   "seal"     "kangaroo"
```

```
## 5b
test_score <- as.vector(c(100, 23, 76, 10))
test_score <- as.integer(test_score)
test_score                            ##typeof(testscore)
```

```
## [1] 100  23  76  10
```

```
####### vector type: originally: double, after conversion: integer
## 5c
passed <- as.vector((c(TRUE, FALSE, TRUE, FALSE))) ####vector type:logical
passed                            ##typeof(passed)
```

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

```
## alternatively
x <- c()
for (i in test_score){
  x <- append(x, i>50)
}
##x    ####vector type:logical
### 7
studentpassfail <-data.frame(names, test_score, passed)
colnames(studentpassfail) <- c("Names", "Test_scores", "Passed")
studentpassfail
```

```
##      Names Test_scores Passed
## 1    fudge         100   TRUE
## 2   muffin          23  FALSE
## 3     seal          76   TRUE
## 4 kangaroo          10  FALSE
```

```
#### this code creates a data frame called studentpassfail and combines the
####three vectors from above into a dataframe
```

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

   Answer: Dataframes can contain different classes of data, whereas matrices can only have one kind of data. since we have different classes of data, a matrix would not be suitable in this situation.

10. Create a function with an if/else statement. Your function should take a **vector** of test scores and print (not return) whether a given 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.

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

```
didtheypass <- function(scores){
  for (i in scores){
   if (i >=  50)
    print("TRUE")
   else
    print("FALSE")
  }
}
didtheypass(test_score)
```

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

```
didtheypass2 <- function(scores){
  result <- ifelse(scores >= 50, "TRUE", "FALSE")
  print(result)
}
didtheypass2(test_score)
```

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

```
#####
# works cited:
#   chatgpt
# prompt:
#   i fed chatgpt my code for if and else, and used the prompt "rewrite it
#   ifelse() so that when scores are equal or over fifty, "TRUE" is printed,
### otherwise, "FALSE" is printed"
# ####didtheypass2
####
```

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

    Answer: Both worked, with ifelse being more efficient. However, ifelse RETURNS results, and the way chatgpt did it is a very sneaky way and doesnt 100% do the job.