JiahuanLi_A02_CodingBasics

Jiahuan Li

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 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.
## c1: name of the sequence.
c1 <- seq(100, from = 1, by = 4)
c1
```

```
##  [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
#2.
## the mean of ci is 49
mean <- mean(c1)
## the median of c1 is also 49
median <- median(c1)

#3.
## a simple if command. The reply is false, indicating the mean is not greater than the median.
if (mean > median){
  print ("True")
} else {
  print ("False")
}
```

```
## [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.
## name is a character vector
name <- c("student 1", "student 2", "student 3", "student 4")
typeof (name)
```

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

```
## score is a numeric vector
score <- c(85, 78, 45, 47)
typeof (score)
```

```
## [1] "double"
```

```
## pass is a logical vector
pass <- c(TRUE, TRUE, FALSE, FALSE)
typeof (pass)
```

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

```
#7.
## the dataframe is named "student records"
combine <- cbind(name, score, pass)
student_records <- as.data.frame(combine)
student_records
```

```
##        name score  pass
## 1 student 1    85  TRUE
## 2 student 2    78  TRUE
## 3 student 3    45 FALSE
## 4 student 4    47 FALSE
```

```
#8.
## the dataframe already has the column names because it is made from vectors
## rename the columns
names(student_records) <- c("Name", "Score","Pass")
```

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

Answer: in a matrix, only similar data types can be stored whereas in a data frame like this, it can contain different data types like characters, integers, or other data frames. Basically Data Frame is advanced version of a Matrix where various types of data input types can be stored.

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.

```r
#10.
Judgement <- function (v) {
  ifelse (v >= 50, "TRUE", "FALSE")
}

#11.
Judgement(student_records$Score)
```

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

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

Answer: the `ifelse` command worked. That is because this function is a type of vector operation and checks the condition for every element of a vector. If using the `if` and `else` command, only the first element of a vector will be checked and a warning will be reported.