Worksheet-4a in R

Erl Syron R. Espadon

#1. The table below shows the data about shoe size and height. Create a data frame. #a. Describe the data.

```
# Create a data frame from the table
df <- data.frame(
  Shoe_size = c(6.5, 9.0, 8.5, 8.5, 10.5, 7.0, 9.5, 9.0, 13.0, 7.5, 10.5, 8.5, 12.0, 10.5, 13.0, 11.5, 8
  Height = c(66.0, 68.0, 64.5, 65.0, 70.0, 64.0, 70.0, 71.0, 72.0, 64.0, 74.5, 67.0, 71.0, 71.0, 77.0, 7
  Gender = c("F", "F", "F", "F", "M", "F", "F", "F", "M", "F", "M", "F", "M", "M", "M", "M", "F", "F", "
)

# Print the data frame
print(df)
```

```
##    Shoe_size Height Gender
## 1        6.5   66.0      F
## 2        9.0   68.0      F
## 3        8.5   64.5      F
## 4        8.5   65.0      F
## 5       10.5   70.0      M
## 6        7.0   64.0      F
## 7        9.5   70.0      F
## 8        9.0   71.0      F
## 9       13.0   72.0      M
## 10       7.5   64.0      F
## 11      10.5   74.5      M
## 12       8.5   67.0      F
## 13      12.0   71.0      M
## 14      10.5   71.0      M
## 15      13.0   77.0      M
## 16      11.5   72.0      M
## 17       8.5   59.0      F
## 18       5.0   62.0      F
## 19      10.0   72.0      M
## 20       6.5   66.0      F
## 21       7.5   64.0      F
## 22       8.5   67.0      M
## 23      10.5   73.0      M
## 24       8.5   69.0      F
## 25      10.5   72.0      M
## 26      11.0   70.0      M
## 27       9.0   69.0      M
## 28      13.0   70.0      M
```

#b. Create a subset by males and females with their corresponding shoe size and height.What its result? Show the R scripts.

```r
# Create subsets for males and females
males <- df[df$Gender == "M", c("Shoe_size", "Height")]
females <- df[df$Gender == "F", c("Shoe_size", "Height")]

# Print the subsets
print(males)
```

```
##    Shoe_size Height
## 5       10.5   70.0
## 9       13.0   72.0
## 11      10.5   74.5
## 13      12.0   71.0
## 14      10.5   71.0
## 15      13.0   77.0
## 16      11.5   72.0
## 19      10.0   72.0
## 22       8.5   67.0
## 23      10.5   73.0
## 25      10.5   72.0
## 26      11.0   70.0
## 27       9.0   69.0
## 28      13.0   70.0
```

```r
print(females)
```

```
##    Shoe_size Height
## 1        6.5   66.0
## 2        9.0   68.0
## 3        8.5   64.5
## 4        8.5   65.0
## 6        7.0   64.0
## 7        9.5   70.0
## 8        9.0   71.0
## 10       7.5   64.0
## 12       8.5   67.0
## 17       8.5   59.0
## 18       5.0   62.0
## 20       6.5   66.0
## 21       7.5   64.0
## 24       8.5   69.0
```

#c. Find the mean of shoe size and height of the respondents. Write the R scripts and its result.

```r
# Calculate the mean of shoe size and height
mean_shoe_size <- mean(df$Shoe_size)
mean_height <- mean(df$Height)

# Print the means
print(paste("Mean shoe size:", mean_shoe_size))
```

```
## [1] "Mean shoe size: 9.41071428571429"
```

```r
print(paste("Mean height:", mean_height))
```

```
## [1] "Mean height: 68.5714285714286"
```

#d. Is there a relationship between shoe size and height? Why?

```
#
```

#2. Construct character vector months to a factor with factor() and assign the result to factor_months_vector. Print out factor_months_vector and assert that R prints out the factor levels below the actual values.

```
months <- c("March", "April", "January", "November", "January",
            "September", "October", "September", "November", "August",
            "January", "November", "November", "February", "May", "August",
            "July", "December", "August", "August", "September", "November",
            "February", "April")
factor_months_vector <- factor(months)
print(factor_months_vector)
```

```
##  [1] March     April     January   November  January   September October
##  [8] September November  August    January   November  November  February
## [15] May       August    July      December  August    August    September
## [22] November  February  April
## 11 Levels: April August December February January July March May ... September
```

#3. Then check the summary() of the months_vector and factor_months_vector. | Interpret the results of both vectors. Are they both equally useful in this case?

```
summary(months)
```

```
##    Length     Class      Mode
##        24 character character
```

```
summary(factor_months_vector)
```

```
##     April    August  December  February   January      July     March       May
##         2         4         1         2         3         1         1         1
##  November   October September
##         5         1         3
```

```
#In this case, factor_months_vector is more useful than months_vector. The summary of the factor vector
```

#4. Create a vector and factor for the table below.

```
direction_vector <- c("East", "West", "West", "West", "West", "North", "North", "North")
factor_data <- factor(direction_vector)
new_order_data <- factor(factor_data, levels = c("East", "West", "North"))
print(new_order_data)
```

```
## [1] East  West  West  West  West  North North North
## Levels: East West North
```

#5 #a. a. Import the excel file into the Environment Pane using read.table() function.Write the code

```
march_data <- read.table("import_march.csv", header = TRUE, sep = ",")
```

#b. View the dataset. Write the R scripts and its result.

```
print(march_data)
```

```
##   Students Strategy.1 Stategy.2 Strategy.3
## 1     Male          8        10          8
## 2                   4         8          6
## 3                   0         6          4
## 4   Female         14         4         15
## 5                  10         2         12
```

```
## 6                   6         0          9
## 7                  NA        NA         NA
## 8                  NA        NA         NA
```

#6. Full Search (Exhaustive Search Function) # Get user input

```
exhaustive_search <- function(number) {
  if (number < 1 || number > 50) {
    return("The number selected is beyond the range of 1 to 50")
  } else if (number == 20) {
    return(TRUE)
  } else {
    return(number)
  }
}

# Get user input
user_number <- readline("Enter a number between 1 and 50: ")
```

```
## Enter a number between 1 and 50:
```

```
# Check if the input is a valid number
if (is.numeric(user_number)) {
  user_number <- as.numeric(user_number)
  # Call the function and display the result
  result <- exhaustive_search(user_number)
  print(result)
} else {
  print("Invalid input. Please enter a number.")
}
```

```
## [1] "Invalid input. Please enter a number."
```

#7. Change (Minimum Bills Function)

```
min_bills <- function(price) {
  bill_values <- c(1000, 500, 200, 100, 50)
  bills_needed <- rep(0, length(bill_values))

  for (i in 1:length(bill_values)) {
    while (price >= bill_values[i]) {
      bills_needed[i] <- bills_needed[i] + 1
      price <- price - bill_values[i]
    }
  }

  return(sum(bills_needed))
}

# Get a random price divisible by 50
price <- sample(seq(50, 1000, by = 50), 1)

# Call the function and display the result
num_bills <- min_bills(price)
print(paste("Minimum bills needed:", num_bills))
```

```
## [1] "Minimum bills needed: 1"
```

#8. #a. Create a dataframe from the above table. Write the R codes and its output.

```
# Create the dataframe
grades <- data.frame(
  Name = c("Annie", "Thea", "Steve", "Hanna"),
  Grade1 = c(85, 65, 75, 95),
  Grade2 = c(65, 75, 55, 75),
  Grade3 = c(85, 90, 80, 100),
  Grade4 = c(100, 90, 85, 90)
)

# Print the dataframe
print(grades)
```

```
##    Name Grade1 Grade2 Grade3 Grade4
## 1 Annie     85     65     85    100
## 2  Thea     65     75     90     90
## 3 Steve     75     55     80     85
## 4 Hanna     95     75    100     90
```

#b. Without using the rowMean function, output the average score of students whose average math score over 90 points during the semester. write R code and its output.

```
for (i in 1:nrow(grades)) {
  student_name <- grades$Name[i]
  average_score <- (grades$Grade1[i] + grades$Grade2[i] + grades$Grade3[i] + grades$Grade4[i]) / 4
  if (average_score > 90) {
    print(paste(student_name, "'s average grade this semester is", round(average_score, 2)))
  }
}
```

#c. Without using the mean function, output as follows for the tests in which the average score was less than 80 out of 4 tests.

```
for (j in 2:ncol(grades)) {  # Start from column 2 (Grade1) to avoid the Name column
  test_average <- mean(grades[, j])
  if (test_average < 80) {
    print(paste("The", j - 1, "th test was difficult."))  # Adjust the test number
  }
}
```

```
## [1] "The 2 th test was difficult."
```

#d. Without using the max function, output as follows for students whose highest score for a semester exceeds 90 points.

```
for (i in 1:nrow(grades)) {
  student_name <- grades$Name[i]
  highest_score <- max(grades[i, 2:5])
  if (highest_score > 90) {
    print(paste(student_name, "'s highest grade this semester is", highest_score))
  }
}
```

```
## [1] "Annie 's highest grade this semester is 100"
## [1] "Hanna 's highest grade this semester is 100"
```