RWorksheet_Camarista#4a

John Lyxton Camarista

2024-10-16

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

```r
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, 72.0, 59.0, 62.0, 72.0, 66.0, 64.0, 67.0, 73.0, 69.0, 72.0, 70.0,
            69.0, 70.0)

shoeSize <- 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, 8.5, 5.0, 10.0, 6.5, 7.5, 8.5, 10.5, 8.5, 10.5, 11.0, 9.0, 13.0)

shoeData <- data.frame(Shoe_Size = shoeSize, Height = height)

#got help from chat gpt to achieve the same layout from the Figure 1: Household Data,
#having 2 halves of the data frame.
half <- ceiling(nrow(shoeData) / 2)
first_half <- shoeData[1:half, ]
second_half <- shoeData[(half+1):nrow(shoeData), ]

combined <- cbind(first_half, second_half)

combined
```

```
##    Shoe_Size Height Shoe_Size Height
## 1        6.5   66.0      13.0     77
## 2        9.0   68.0      11.5     72
## 3        8.5   64.5       8.5     59
## 4        8.5   65.0       5.0     62
## 5       10.5   70.0      10.0     72
## 6        7.0   64.0       6.5     66
## 7        9.5   70.0       7.5     64
## 8        9.0   71.0       8.5     67
## 9       13.0   72.0      10.5     73
## 10       7.5   64.0       8.5     69
## 11      10.5   74.5      10.5     72
## 12       8.5   67.0      11.0     70
## 13      12.0   71.0       9.0     69
## 14      10.5   71.0      13.0     70
```

```r
print(str(shoeData))
```

```
## 'data.frame':    28 obs. of  2 variables:
##  $ Shoe_Size: num  6.5 9 8.5 8.5 10.5 7 9.5 9 13 7.5 ...
##  $ Height   : num  66 68 64.5 65 70 64 70 71 72 64 ...
## NULL
```

- a. Describe the data.
  - The data is consist of 28 rows and columns(Shoe Size and Height) with numerical values.
- b. Create a subset by males and females with their corresponding shoe size and height. What its result? Show the R scripts.

```r
gender <- c("F", "F", "F", "F", "M", "F", "F", "F", "M", "F", "M", "F", "M", "M", "M",
            "M", "F", "F", "M", "F", "F", "M", "M", "F", "M", "M", "M", "M")

withGender <- cbind(shoeData, Gender = gender)

half <- ceiling(nrow(withGender) / 2)
first_half <- withGender[1:half, ]
second_half <- withGender[(half+1):nrow(withGender), ]

combinedWithGender <- cbind(first_half, second_half)

combinedWithGender
```

```
##    Shoe_Size Height Gender Shoe_Size Height Gender
## 1        6.5   66.0      F      13.0     77      M
## 2        9.0   68.0      F      11.5     72      M
## 3        8.5   64.5      F       8.5     59      F
## 4        8.5   65.0      F       5.0     62      F
## 5       10.5   70.0      M      10.0     72      M
## 6        7.0   64.0      F       6.5     66      F
## 7        9.5   70.0      F       7.5     64      F
## 8        9.0   71.0      F       8.5     67      M
## 9       13.0   72.0      M      10.5     73      M
## 10       7.5   64.0      F       8.5     69      F
## 11      10.5   74.5      M      10.5     72      M
## 12       8.5   67.0      F      11.0     70      M
## 13      12.0   71.0      M       9.0     69      M
## 14      10.5   71.0      M      13.0     70      M
```

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

```r
dataMeanShoeSize <- round(mean(withGender$Shoe_Size),2)
print(paste("The mean of the Shoe size is ", dataMeanShoeSize))
```

```
## [1] "The mean of the Shoe size is  9.41"
```

```r
dataMeanHeight <- round(mean(withGender$Height),2)
print(paste("The mean of the Height is ", dataMeanHeight))
```

```
## [1] "The mean of the Height is  68.57"
```

- d. Is there a relationship between shoe size and height? Why?
  - "Yes, there is. As height increases, shoe size also increases. This increase in shoe or foot size is necessary to support the person's relative height. Similarly, when constructing tall buildings, a deep foundation is required."

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.

Consider data consisting of the names of months: "March", "April", "January", "November", "January", "September", "October", "September", "November", "August", "January", "November", "November", "February", "May", "August", "July", "December", "August", "August", "September", "November", "February", "April"

```r
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)
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?

```r
summary_factor_months <- summary(factor_months_vector)
summary_factor_months
```

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

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

```
direction <- c("East", "West", "North")
frequency <- c(1, 4, 3)

factor_data <- factor(direction, levels = c("East", "West", "North"))

direction_data <- data.frame(Direction = factor_data, Frequency = frequency)

direction_data
```

```
##   Direction Frequency
## 1      East         1
## 2      West         4
## 3     North         3
```

5. Enter the data below in Excel with file name = import_march.csv

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

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

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

```
import_march_data
```

```
##   Students Strategy.1 Strategy.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
```

Using Conditional Statements (IF-ESE)

6. Fyll Search

Exhaustive search is a methodology for finding an answer by exploring all possible cases. When trying to find a desired number in a set of given numbers, the method of finding the corresponding number by checking all elements in the set one by one can be called an exhaustive search. Implement an exhaustive search function that meets the input/output conditions below. -a. Create an R Program that allows the User to randomly select numbers from 1 to 50. Then display the chosen number. If the number is beyond the range of the selected choice, it will have to display a string "The number selected is beyond the range of 1 to 50". If number 20 is inputted by the User, it will have to display "TRUE", otherwise display the input number.

```
num_selected <- readline(prompt = "Enter a number between 1 to 50: ")
```

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

```r
if(num_selected > 50){
  print("The number selected is beyond the range of 1 to 50")
}else if(num_selected == 20){
  print("TRUE")
}else{
  num_selected
}
```

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

7. Change

At ISATU University's traditional cafeteria, snacks can only be purchased with bills. A long-standing rule at the concession stand is that snacks must be purchased with as few coins as possible. There are three types of bills: 50 pesos, 100 pesos, 200 pesos, 500 pesos, 1000 pesos. - a. Write a function that prints the minimum number of bills that must be paid, given the price of the snack. Input: Price of snack (a random number divisible by 50) Output: Minimum number of bills needed to purchase a snack.

```r
minBills <- function(priceOfSnack){
  bills <- c(1000, 500, 200, 100, 50)

numOfBills <- 0
 for(bill in bills){
   if(!is.na(priceOfSnack) >= bill){
     count <- priceOfSnack %/% bill
     numOfBills <- numOfBills + count
     priceOfSnack <- priceOfSnack %% bill
   }
 }
  return(numOfBills)
}

priceOfSnack <- as.numeric(readline(prompt = "Input the price of the snack: "))
```

```
## Input the price of the snack:
```

```r
result <- minBills(priceOfSnack)
result
```

```
## [1] NA
```

##8. The following is each student's math score for one semester. Based on this, answer the following questions. - a. Create a dataframe from the above table. Write the R codes and its output.

```r
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)
```

```r
reportCard <- data.frame(Name = name, Grade1 = grade1, Grade2 = grade2, Grade3 = grade3, Grade4 = grade

reportCard
```

```
##    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.
  Example Output: Annie's average grade this semester is 88.75.

```r
for(i in 1:nrow(reportCard)){
  total <- reportCard$Grade1[i] + reportCard$Grade2[i] + reportCard$Grade3[i] + reportCard$Grade4[i]
  average <- total / 4

  if(average > 90){
    cat(reportCard$Name[i], "'s average grade this semester is", average)
  }
}
```

- 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.
  Example output: The nth test was difficult.

```r
gradeLvl <- c("Grade1", "Grade2", "Grade3", "Grade4")
numOfStudents <- nrow(reportCard)

for(i in 1:length(gradeLvl)){
  totalTest <- sum(reportCard[[gradeLvl[i]]])
  averageTest <- totalTest / numOfStudents

  if(averageTest < 80){
    cat("The", i, "th Test was difficult")
  }
}
```

```
## 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. Example Output: Annie's highest grade this semester is 95.

```r
for(i in 1:nrow(reportCard)){

    high <- reportCard$Grade1[i]

  if(reportCard$Grade2[i] > high){
    high <- reportCard$Grade2[i]
  }
  if(reportCard$Grade3[i] > high){
    high <- reportCard$Grade3[i]
  }
  if(reportCard$Grade4[i] > high){
    high <- reportCard$Grade4[i]
  }
  print(paste(reportCard$Name[i], "'s highest grade this semester is", high))
}
```

```
## [1] "Annie 's highest grade this semester is 100"
## [1] "Thea 's highest grade this semester is 90"
## [1] "Steve 's highest grade this semester is 85"
## [1] "Hanna 's highest grade this semester is 100"
```