RWorksheet_Perez#4a

2024-10-14

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

```
shoe_data <- read.csv("/cloud/project/ShoeData.csv")
shoe_data
```

```
##    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
```

a. Describe the data.

The data consists of the respondents' shoe sizes, as well as their height and gender.

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

```
males <- subset(shoe_data, Gender == "M", select = c(Shoe.size, Height))
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
```

```
females <- subset(shoe_data, Gender == "F", select = c(Shoe.size, Height))
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.

```
mean(shoe_data$Shoe.size)
```

```
## [1] 9.410714
```

```
mean(shoe_data$Height)
```

```
## [1] 68.57143
```

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

```
relationship <- cor(shoe_data$Height, shoe_data$Shoe.size)
relationship
```

```
## [1] 0.7766089
```

The results show that there is some kind of relationship between shoe size and height, but it's not the strongest. Some shoe sizes go up with height, but a few doesn't seem to be the case.

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

```
##  [1] March           April           January         November
##  [5] January         September       October         September
##  [9] November        August          January         November
## [13] November        February        May             August
## [17] July            December        August          August
## [21] September       November        February, April
## 12 Levels: April August December February February, April January ... September
```

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

```r
summary(months_vector)
```

```
##    Length     Class      Mode
##        23 character character
```

```r
summary(factor_months_vector)
```

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

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

Direction Frequency East 1 West 4 North 3

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

```r
factor_direction
```

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

```r
factor_frequency
```

```
## [1] 1 4 3
## Levels: 1 3 4
```

Note: Apply the factor function with required order of the level. new_order_data <- factor(factor_data,levels = c("East","West","North")) print(new_order_data)

```r
new_order_direction <- factor(factor_direction, levels = c("East", "West", "North"))
print(new_order_direction)
```

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

```r
new_order_frequency <- factor(factor_frequency, levels = c(1, 4, 3))
print(new_order_frequency)
```

```
## [1] 1 4 3
## Levels: 1 4 3
```

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

```r
read.csv("/cloud/project/import_march.csv")
```

```
##   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
```

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

```r
march <- read.table("/cloud/project/import_march.csv", header = TRUE, sep = ",")
```

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

```r
march
```

```
##   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
```

6. Full 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.

```r
search <- function(){
  num <- readline(prompt="Enter a number between 1 and 50: ")
  if (num < 1 || num > 50){
    print("The number selected is beyond the range of 1 to 50")
  } else if (num == 20){
    print("TRUE")
  } else {
    print(paste("The selected number is", num))
  }
}
search()
```

```
## Enter a number between 1 and 50:
## [1] "The number selected is beyond the range of 1 to 50"
```

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.

```
minBills <- function(price) {
    if(is.na(price%%50)) {price %% 50 ==0} else {if(price %% 50 != 0) {
        return("Price must be a multiple of 50")
    }}
    bills <- c(1000, 500, 200, 100, 50)
    count <- 0
    for (bill in bills) {
        num_bills <- price %/% bill
        count <- count + num_bills
        price <- price %% bill
    }

    return(count)
}

snack_price <- as.numeric(readline(prompt = "Enter price of snack: "))
```

```
## Enter price of snack:
```

```
minBills(snack_price)
```

```
## [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.

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

```
for (i in 1:nrow(grades)){
  ave <- sum(grades[i, 2:5]) / 4
  if (ave > 90){
    print(paste(grades$Name[i], "'s average grade this semester is", ave))
  }
```

```
}

## No student has gotten over 90 points.
```

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.

```
for(j in 2:5){
  avg <- sum(grades[ ,j]) / nrow(grades)
  if(avg < 80){
    print(paste0("The ", j - 1, "th test was difficult."))
  }
}
```

```
## [1] "The 2th 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.

```
for (i in 1:nrow(grades)) {
    sorted_grades <- sort(c(grades$Grade1[i], grades$Grade2[i], grades$Grade3[i], grades$Grade4[i]), de
    highest <- sorted_grades[1]
    if (highest > 90) {
        print(paste0(grades$Name[i], "'s highest grade this semester is ", highest))
    }
}
```

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