Exercise2

Rotem and Gal

3/13/2023

In this exercise, you will submit questions presented on the green slides from our class *in addition* to questions not seen in class.

Remember to submit both the (.rmd) file and the generated pdf file after knitting the document.

The pdf should include both the command and the output of the code chunks.

Don't forget to update the "author" and date information accordingly.

Question 1 - The Iris data set project

1.1 Load the iris built-in dataset (requires the tidyverse library) and describe it - what are the dataset columns? How many observations ?

```
dataset<-data("iris")
colnames(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
cols_num<-ncol(iris)
cols_num
```

```
## [1] 5
```

1.2 Find the maximum sepal length for each species

```
 ind <- match(c("setosa","versicolor","virginica"),iris$Species)
  iris$Sepal.Length[ind]
```

```
## [1] 5.1 7.0 6.3
```

1.3 Find the row number of the iris with the longest petal, and use it to find its species

```
ind1<-max(iris$Petal.Length)
ind2<-match(ind1,iris$Petal.Length)
iris[ind2,5]
```

```
## [1] virginica
## Levels: setosa versicolor virginica
```

1.4 Do the same as in question 1.3, but now for the iris with the longest sepal.

```
ind1<-max(iris$Sepal.Length)
ind2<-match(ind1,iris$Sepal.Length)
iris[ind2,5]
```

```
## [1] virginica
## Levels: setosa versicolor virginica
```

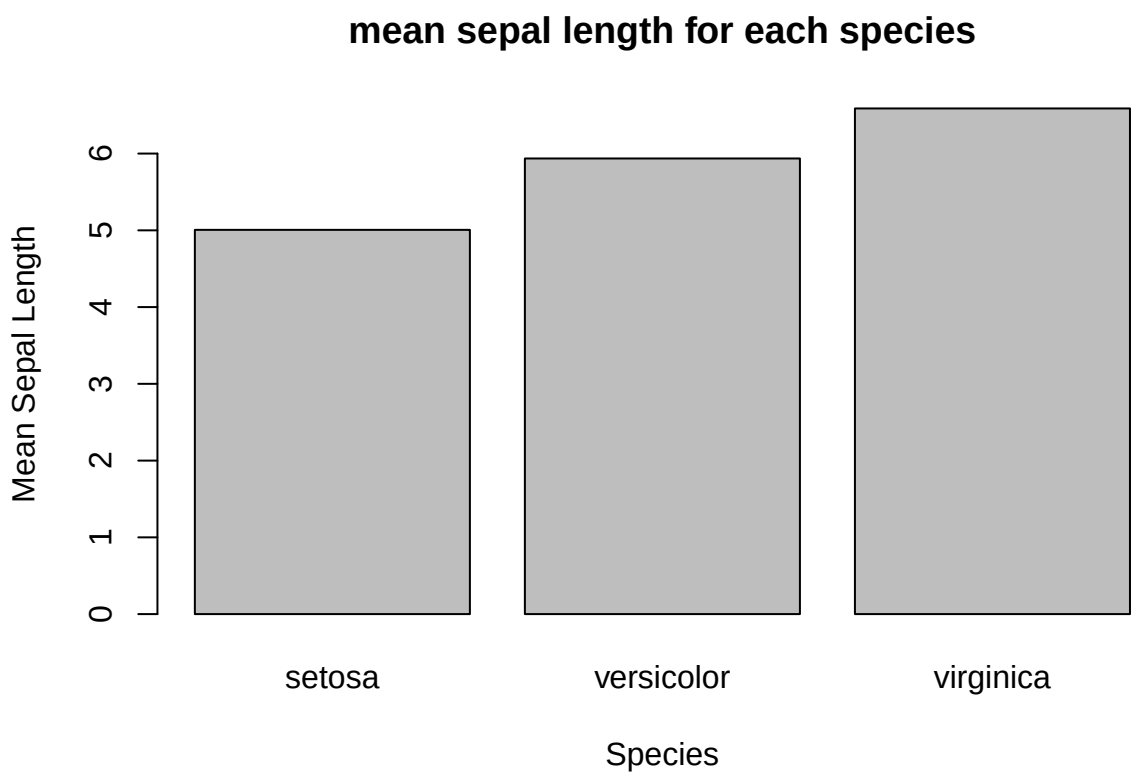
1.5 Create a bar plot (using "barplot()") of the mean sepal length for each species

```
setosa_subset<-subset(iris,Species=="setosa")
mean1<-mean(setosa_subset$Sepal.Length)

versicolor_subset<-subset(iris,Species=="versicolor")
mean2<-mean(versicolor_subset$Sepal.Length)

virginica_subset<-subset(iris,Species=="virginica")
mean3<-mean(virginica_subset$Sepal.Length)

meanVec=c(setosa=mean1,versicolor=mean2,virginica=mean3)
barplot(meanVec,main="mean sepal length for each species",xlab="Species",ylab="Mean Sepal Length")
```



Question 2 - Some more R

2.1 Print a message asking the user for her/his name and print a personalized (including his/her name) welcoming message

```
print("please enter your name")
```

```
## [1] "please enter your name"
```

```
x<-readline()
```

```
cat(paste0("Welcome ",x))
```

```
## Welcome
```

2.2 List all variables in memory (workspace). Save them to a file. Remove all variables from memory. Load the file where the variables have been saved.

```
ls()
```

```
##  [1] "cols_num"          "dataset"           "ind"
##  [4] "ind1"              "ind2"              "iris"
##  [7] "mean1"             "mean2"             "mean3"
## [10] "meanVec"           "setosa_subset"     "versicolor_subset"
## [13] "virginica_subset"  "x"
```

```
save(list = ls(),file = "myWork.RData")
rm(list =ls())
ls()
```

```
## character(0)
```

```
load("myWork.RData")
ls()
```

```
##  [1] "cols_num"          "dataset"           "ind"
##  [4] "ind1"              "ind2"              "iris"
##  [7] "mean1"             "mean2"             "mean3"
## [10] "meanVec"           "setosa_subset"     "versicolor_subset"
## [13] "virginica_subset"  "x"
```

2.3 Create 4 vectors:

v1 containing a sequence of numbers from 20 to 50

v2 contain the sum of numbers from 51 to 91

v3 containing 20 random numbers between 1 and 100

v4 containing the 5th to 15th letters of the english alphabet in lower case

```r
v1 <- 20:50
v2 <- sum(51:91)
v3 <-sample(1:100,20)
v4 <-letters[5:15]
v1
```

```
##  [1] 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
## [26] 45 46 47 48 49 50
```

```r
v2
```

```
## [1] 2911
```

```r
v3
```

```
##  [1] 100  31  53  85  55  23  35  66  96  15  57  94   5  44  69  50  36  70  63
## [20]  48
```

```r
v4
```

```
##  [1] "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o"
```

2.4 Write a function called "isPrime" that given a number it checks if the number is prime or not. If prime, the function will print "n is prime" (replacing n by the number received), otherwise the function will print all its divisors "n has the following divisors: d1,d2,..." (x is divisor of n if dividing n by x leaves no remainder. E.g. 3 is divisor of 9 bcs $9 \%\% 3 = 0$.).

```r
isPrime <- function(n) {
if (n < 2) {
cat(n, "is not prime\n")
return()
}
divisors <- c()
for (i in 2:(sqrt(n))) {
if (n %% i == 0) {
divisors <- c(divisors, i)
if (i != n/i) {
divisors <- c(divisors, n/i)
}
}
}
if (length(divisors) == 0) {
cat(n, "is prime\n")
} else {
cat(n, "has the following divisors:", paste(divisors, collapse = ","), "\n")
}
}
```

2.5 Write two functions, bindRows() and bindColumns(), that receive 3 vectors a,b,c, and concatenate them into a matrix. The rows of the matrix returned by bindRows() are the vectors a,b,c. The columns of the matrix returned by bindColumns() are the vectors a,b,c.

For example, given vectors v1= (1,2,3), v2=(4,5,6), v3=(7,8,9) bindRows() will return:

[1,2,3

4,5,6

7,8,9]

bindColumns() will return:

[1, 4, 7

2, 5, 8

3, 6, 9]

```
bindRows <- function(a, b, c) {
  matrix(c(a, b, c), nrow = 3, byrow = TRUE)
}

bindColumns <- function(a, b, c) {
  matrix(c(a, b, c), ncol = 3, byrow = FALSE)
}

a<-c(1,2,3)
b<-c(4,5,6)
c<-c(7,8,9)
bindRows(a, b, c)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

```
bindColumns(a, b, c)
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

2.6.1 Create three vectors: vector "a" containing numeric data, vector "b" containing characters and vector "c" with logical data (TRUE/FALSE). Display the content of the vectors and their type.

```
a <- c(2.3, 4.5, 1.2, 3.6, 7.8)
b <- c("apple", "banana", "cherry", "date", "elderberry")
c <- c(TRUE, FALSE, TRUE, FALSE, TRUE)
print(a)
```

```
## [1] 2.3 4.5 1.2 3.6 7.8
```

```
print(typeof(a))
```

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

```
print(b)
```

```
## [1] "apple"      "banana"     "cherry"     "date"       "elderberry"
```

```
print(typeof(b))
```

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

```
print(c)
```

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

```
print(typeof(c))
```

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

2.6.2 Create a data frame from the list (1,2,3). Display its type (with typeOf()) and its class (with class()), and the type and class of its first column.

```
my_list <- list(1, 2, 3)
my_df <- data.frame(my_list)
typeof(my_df)
```

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

```
class(my_df)
```

```
## [1] "data.frame"
```

```
typeof(my_df[,1])
```

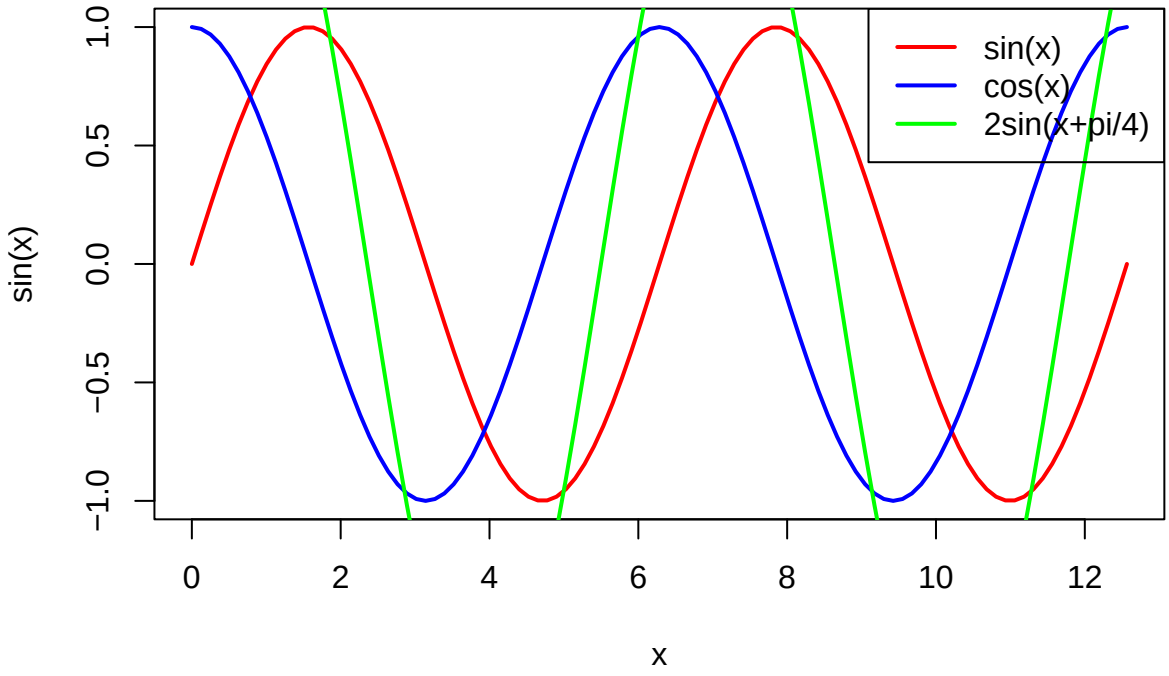
```
## [1] "double"
```

```
class(my_df[,1])
```

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

2.7 Plot as a smooth curve (a line and not dots) the following functions: sin(x), cos(x) and 2sin(x+pi/4) between [0,4pi] (all functions should be displayed on the same plot)

```
x <- seq(0, 4*pi, length.out = 1000)
curve(sin(x), from = 0, to = 4*pi, type = "l", col = "red", lwd = 2)
curve(cos(x), from = 0, to = 4*pi, type = "l", col = "blue", lwd = 2, add = TRUE)
curve(2*sin(x+pi/4), from = 0, to = 4*pi, type = "l", col = "green", lwd = 2, add = TRUE)
legend("topright", legend = c("sin(x)", "cos(x)", "2sin(x+pi/4)"),
       col = c("red", "blue", "green"), lty = 1, lwd = 2)
```



Question 3 - The HR dataset

Note: The questions below refer to the database "HR-Employee-employee" which you should download from the course's website.

3.1 Load the dataset into a dataframe named "df"

```
df<-read.csv("HR-Employee.csv")
```

3.2 How many employees were surveyed ?

```
nrow(df)
```

```
## [1] 1470
```

3.3 What are the names of the columns ?

```
colnames(df)
```

```
##  [1] "Age"                      "Attrition"
##  [3] "BusinessTravel"           "DailyRate"
##  [5] "Department"               "DistanceFromHome"
##  [7] "Education"                "EducationField"
##  [9] "EmployeeCount"            "EmployeeNumber"
## [11] "EnvironmentSatisfaction"  "Gender"
## [13] "HourlyRate"               "JobInvolvement"
## [15] "JobLevel"                 "JobRole"
## [17] "JobSatisfaction"          "MaritalStatus"
## [19] "MonthlyIncome"            "MonthlyRate"
## [21] "NumCompaniesWorked"       "Over18"
## [23] "OverTime"                 "PercentSalaryHike"
## [25] "PerformanceRating"        "RelationshipSatisfaction"
## [27] "StandardHours"            "StockOptionLevel"
## [29] "TotalWorkingYears"        "TrainingTimesLastYear"
## [31] "WorkLifeBalance"          "YearsAtCompany"
## [33] "YearsInCurrentRole"       "YearsSinceLastPromotion"
## [35] "YearsWithCurrManager"
```

3.4 What's the class of "distanceFromHome" column ?

```
class(df$DistanceFromHome)
```

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

3.5 List all the JobRoles in alphabetical order

```
sort(unique(df$JobRole))
```

```
## [1] "Healthcare Representative" "Human Resources"
## [3] "Laboratory Technician"     "Manager"
## [5] "Manufacturing Director"    "Research Director"
## [7] "Research Scientist"        "Sales Executive"
## [9] "Sales Representative"
```

3.6 What is the job role with highest monthly income ?

```
highest<-df[which.max(df$MonthlyIncome),"JobRole"]
highest
```

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

3.7 How many employees have salaries below the average monthly income ?

```r
avg_monthly_income <- mean(df$MonthlyIncome)
num_below_avg <- sum(df$MonthlyIncome < avg_monthly_income)
num_below_avg
```

```
## [1] 977
```

3.8.1 What's the education field of employees that have salaries below the average monthly income ?

```r
below_the_avr<-df[df$MonthlyIncome<avg_monthly_income, ]

table(below_the_avr$EducationField)
```

```
##
##  Human Resources    Life Sciences        Marketing          Medical
##               20              404               87              318
##            Other Technical Degree
##               59               89
```

3.8.2 Is there an education field that every employee with that education field have an average monthly income above average ? If yes, list which education fields

```r
avg <- aggregate(MonthlyIncome ~ EducationField, data = df, FUN = mean)
above_avg_fields <- avg[avg$MonthlyIncome >
mean(df$MonthlyIncome), "EducationField"]

if (length(above_avg_fields) > 0)
{
  print("all of the employess in the fields")
print(paste(" that earn above the average:",
            paste(above_avg_fields,collapse = ", ")))
} else
{
print("There are no fields where all of the employees has earnning
      above average.")
}
```

```
## [1] "all of the employess in the fields"
## [1] " that earn above the average: Human Resources, Marketing, Medical"
```

3.8.3 Is there job roles where all employees with this job role have incomes above average? If yes, list which job roles.

```r
avg <- aggregate(MonthlyIncome ~ JobRole, data = df, FUN = mean)
above_avg <- avg[avg$MonthlyIncome >
avg_monthly_income, "JobRole"]

if (length(above_avg) > 0)
{
  print("all of the employess in the jobs that earn above the average:")
  print(paste(above_avg,collapse = ", "))
```

```
} else
{
print("There are no jobs where all of the employees has earnning
      above average.")
}
```

```
## [1] "all of the employess in the jobs that earn above the average:"
## [1] "Healthcare Representative, Manager, Manufacturing Director, Research Director, Sales Executive"
```

3.9 What's the average age of employees that have salaries BELOW the average monthly income ? Is it lower than the average of those who have salaries ABOVE the average monthly income ?

```
below_age_avr <- mean(df$Age[df$MonthlyIncome < avg_monthly_income])
above_age_avr <- mean(df$Age[df$MonthlyIncome >= avg_monthly_income])
cat(paste("below:" ,below_age_avr,"\n"))
```

```
## below: 34.4114636642784
```

```
cat(paste("above:" ,above_age_avr, "\n"))
```

```
## above: 41.9026369168357
```

3.10 How many employees have salaries above average ? And below average?

```
above <- subset(df,df$MonthlyIncome > avg_monthly_income)
below <- subset(df,df$MonthlyIncome < avg_monthly_income)
nrow(above)
```

```
## [1] 493
```

```
nrow(below)
```

```
## [1] 977
```

3.11.1 What's the output and meaning of each one of the lines below ?

ind1 <- which(df$MonthlyIncome<meanMonthlyIncome)

ind2 <- df$MonthlyIncome<meanMonthlyIncome

Answer: _____ the line that start with ind1 creates avariable called ind1 and uses the which() function of ind the indices of all the rows in the df data frame where the monthlyincome(dfMonthlyIncome) is less than the mean monthly income of the data frame (meanMonthlyIncome). The result in ind1 variable is a vector of row indices.

the line that start with ind2 creates avariable called ind2 and uses alogical comparison (<) to compare the monthlyincome values in the df MonthlyIncome column to the mean monthly income value (meanMonthlyIncome). The result is a logical vector (TRUE or FALSE) that indicates whether each income value is less than the mean. The result in ind2 variable is a logical vector of the same length as the df$MonthlyIncome column, with TRUE values corresponding to income values that are less than the mean.

3.11.2 What's the result of the equation below (ind1 and ind2 as defined in the previous question), and explain the result.

df$EmployeeNumber[ind1] == df$EmployeeNumber[ind2]

This equation compares the values of the EmployeeNumber columns of ind1 & ind2

The result of this equation is a logical vector that tell us are they the same length (ind1 and ind2)

3.12 What can we say about the rate of the Job satisfaction ? (select all that is true and justify your answer w code)

(a) Most employees are satisfied (3) or very satisfied (4)

(b) Most employees are very satisfied (4)

(c) The amount of employees that are very satisfied (4) is bigger than the amount of employees that are either not satisfied (1) or medium satisfied (2)

```
satisfaction_freq <- table(df$JobSatisfaction)
satisfaction_pct <- prop.table(satisfaction_freq) * 100
print(satisfaction_pct)
```

```
##
##        1        2        3        4
## 19.65986 19.04762 30.06803 31.22449
```

3.13 Prove or contradict the following statement: "In average women have a higher percentage of very high JobSatisfaction (4) than men"

```
df_gender <- subset(df, Gender %in% c("Male", "Female"))
prop.table(table(df_gender$Gender, df_gender$JobSatisfaction), 1)
```

```
##
##                  1         2         3         4
##   Female 0.2023810 0.2006803 0.3078231 0.2891156
##   Male   0.1927438 0.1836735 0.2959184 0.3276644
```

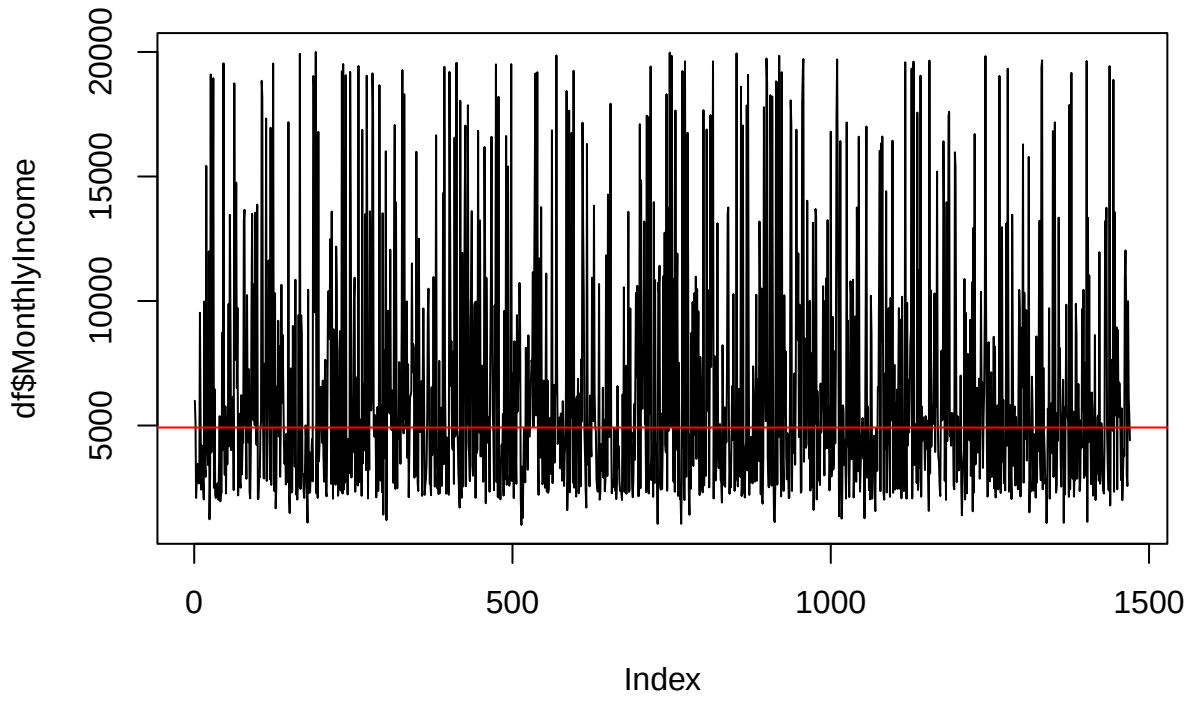
3.14 What is the marital status that is most satisfied with their jobs ? (the % of employees with JobSatisfaction= 4 is the highest across all other levels)

```
df_satisfied <- subset(df, JobSatisfaction == 4)
prop.table(table(df_satisfied$MaritalStatus))
```

```
##
##  Divorced   Married    Single
## 0.2222222 0.4357298 0.3420479
```

3.15 Plot the monthly income of all employees, adding a line to the median monthly income. What can we say based on the graph ?

```
plot(df$MonthlyIncome, type = "l")
abline(h = median(df$MonthlyIncome), col = "red")
```

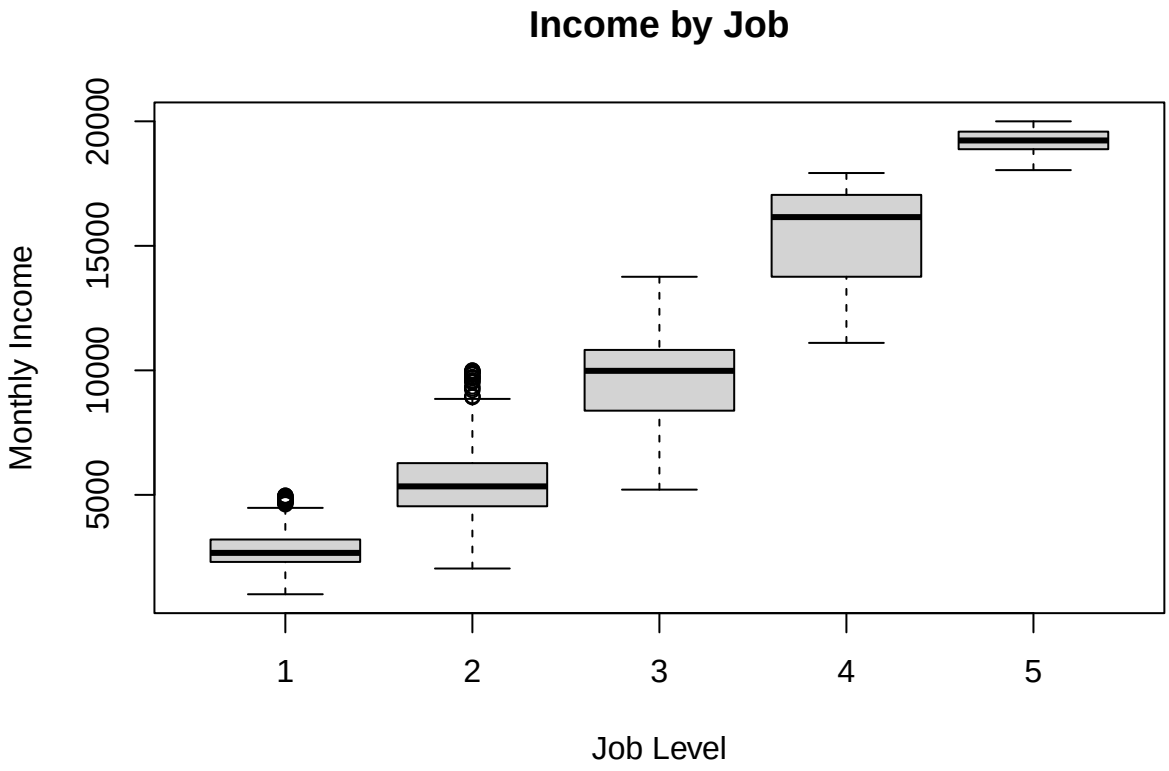


3.16 Boxplot the monthly income for each gender separately. What can we say about the difference in salaries between men and women?

```
df_men <- subset(df, Gender == "Male")
df_women <- subset(df, Gender == "Female")
par(mfrow=c(1,2))
boxplot(df_men$MonthlyIncome, main = "Monthly Income for Men",
        ylab = "Monthly Income")
boxplot(df_women$MonthlyIncome, main = "Monthly Income for Women",
        ylab = "Monthly Income")
```

3.17 Boxplot the monthly income for each job level separately. What can we conclude regarding the relationship between monthly income and the different job levels?

```
boxplot(df$MonthlyIncome ~ df$JobLevel, main = "Income by Job",
        xlab = "Job Level", ylab = "Monthly Income")
```


Income by Job
Monthly Income
5000
10000
15000
20000
1
2
3
4
5
Job Level

Good luck!