Batch:-H2_1	**Roll No.: 16010122151**
Experiment No. 2	

Title : To apply the descriptive statistics techniques

Aim: To apply various descriptive statistics techniques, such as measures of central tendency, variability, and distribution, to analyze and summarize the key features of a dataset.

Expected Outcome of Experiment:
CO1 : Develop an understanding of data science and business analytics.

Books/ Journals/ Websites referred:

Select a built-in R dataset

You can see a list of all the built-in datasets using the data() function.

```
> data()
```

```
R data sets

Data sets in package 'datasets':

AirPassengers           Monthly Airline Passenger Numbers 1949-1960
BJsales                 Sales Data with Leading Indicator
BJsales.lead (BJsales)
                        Sales Data with Leading Indicator
BOD                     Biochemical Oxygen Demand
CO2                     Carbon Dioxide Uptake in Grass Plants
ChickWeight             Weight versus age of chicks on different diets
DNase                   Elisa assay of DNase
EuStockMarkets          Daily Closing Prices of Major European Stock
                        Indices, 1991-1998
Formaldehyde            Determination of Formaldehyde
HairEyeColor            Hair and Eye Color of Statistics Students
```

Here, we'll use the built-in R data set named *iris*. Every student in the batch has to choose a unique dataset.

```
> # Store the data in the variable my_data
> my_data <- iris
```

Check your data

You can inspect your data using the functions **head**() and **tails**(), which will display the first and the last part of the data, respectively.

```
> # Print the first 6 rows
> head(my_data, 6)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

To find number or row in data

>nrow(my_data)

What will be output of

>ncol(my_data)

R functions for computing descriptive statistics

Description	R function
Mean	**mean**()
Standard deviation	**sd**()
Variance	**var**()
Minimum	**min**()
Maximum	**maximum**()
Median	**median**()
Range of values (minimum and maximum)	**range**()
Sample quantiles	**quantile**()
Generic function	**summary**()
Interquartile range	**IQR**()

Descriptive statistics for a single group

Measure of central tendency: mean, median, mode

Mean is nothing but the average of the given set of values.

Median The median of a set of data is the **middlemost number or centre value in the set**. The median is also the number that is halfway into the set. To find the median, the data should be arranged first in order of least to greatest or greatest to the least value.

```
> # Compute the mean value
> mean(my_data$Sepal.Length)
[1] 5.843333
>
> # Compute the median value
> median(my_data$Sepal.Length)
[1] 5.8
```

The **mode** is the value that appears most often in a set of data. The mode of a discrete probability distribution is the value x at which its probability mass function takes its maximum value. In other words, it is the value that is most likely to be sampled.

The function mfv() [in the modeest R package] can be used to compute the mo de of a variable.

```
> # Compute the mode
> install.packages("modeest")
> require(modeest)
Loading required package: modeest
> mfv(my_data$Sepal.Length)
[1] 5
```

Measure of variability

Range: minimum & maximum

```
> # Compute the minimum value
> min(my_data$Sepal.Length)
[1] 4.3
> # Compute the maximum value
> max(my_data$Sepal.Length)
[1] 7.9
> # Range
> range(my_data$Sepal.Length)
[1] 4.3 7.9
```

Quantiles

A **quantile** is a **particular part of a data set** that determines how many values in a distribution are above or below a certain limit. Quantiles are cut points that divide the range of a probability distribution into continuous intervals with equal probabilities, or divide the observations in a sample in the same way. The word "quantile" comes from the word quantity, and it refers to dividing a sample or a probability distribution into equal-sized, adjacent subgroups.

So given data set is arranged in the ascending order. Assume there are 100 samples; then, 25% means THE value of 25th sample; in other words the value below which 25% of the samples lie.

```
> quantile(my_data$Sepal.Length)
  0%  25%  50%  75% 100%
 4.3  5.1  5.8  6.4  7.9
```

By default, the function returns the minimum, the maximum and three **quartiles** (the 0.25, 0.50 and 0.75 quantiles).

```
> quantile(my_data$Sepal.Length, seq(0, 1, 0.25))
  0%  25%  50%  75% 100%
 4.3  5.1  5.8  6.4  7.9
```

To compute deciles (0.1, 0.2, 0.3, …., 0.9), use this:

```
> quantile(my_data$Sepal.Length, seq(0, 1, 0.1))
  0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
4.30 4.80 5.00 5.27 5.60 5.80 6.10 6.30 6.52 6.90 7.90
```

Interquartile range

The difference between the **upper and lower quartile** is known as the interquartile range.

```
> IQR(my_data$Sepal.Length)
[1] 1.3
```

Variance and standard deviation

```
> # Compute the variance
> var(my_data$Sepal.Length)
[1] 0.6856935
> # Compute the standard deviation =
> # square root of th variance
> sd(my_data$Sepal.Length)
[1] 0.8280661
```

Median absolute deviation

```
> # Compute the median absolute deviation
> mad(my_data$Sepal.Length)
[1] 1.03782
```

Computing an overall summary of a variable and an entire data frame

summary() function

Summary of a single variable. Five values are returned: the mean, median, 25th and 75th quartiles, min and max in one single line call:

```
> summary(my_data$Sepal.Length)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  4.300   5.100   5.800   5.843   6.400   7.900 
```

Summary of a data frame. In this case, the function **summary**() is automatically applied to each column. The format of the result depends on the type of the data contained in the column. For example:

- If the column is a numeric variable, mean, median, min, max and quartiles are returned.
- If the column is a factor variable, the number of observations in each group is returned.

```
> summary(my_data, digits = 2)
  Sepal.Length  Sepal.Width   Petal.Length  Petal.Width       Species
 Min.   :4.3   Min.   :2.0   Min.   :1.0   Min.   :0.1   setosa    :50
 1st Qu.:5.1   1st Qu.:2.8   1st Qu.:1.6   1st Qu.:0.3   versicolor:50
 Median :5.8   Median :3.0   Median :4.3   Median :1.3   virginica :50
 Mean   :5.8   Mean   :3.1   Mean   :3.8   Mean   :1.2
 3rd Qu.:6.4   3rd Qu.:3.3   3rd Qu.:5.1   3rd Qu.:1.8
 Max.   :7.9   Max.   :4.4   Max.   :6.9   Max.   :2.5
```

sapply() function

It's also possible to use the function **sapply**() to apply a particular function over a list or vector. For instance, we can use it to compute for each column in a data frame, the mean, sd, var, min, quantile, …

```
> # Compute the mean of each column
> sapply(my_data[, -5], mean)
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
    5.843333     3.057333     3.758000     1.199333

> # Compute quartiles
> sapply(my_data[, -5], quantile)
     Sepal.Length Sepal.Width Petal.Length Petal.Width
0%            4.3         2.0         1.00         0.1
25%           5.1         2.8         1.60         0.3
50%           5.8         3.0         4.35         1.3
75%           6.4         3.3         5.10         1.8
100%          7.9         4.4         6.90         2.5
```

Graphical display of distributions

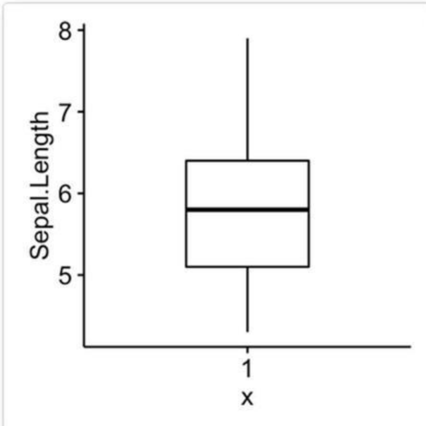
The R package **ggpubr** will be used to create graphs.

```
install.packages("ggpubr")
```

```
library(ggpubr)
```

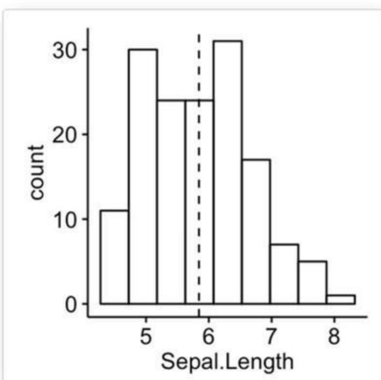
Box Plot

```
ggboxplot(my_data, y = "Sepal.Length", width = 0.5)
```



Histogram with mean line

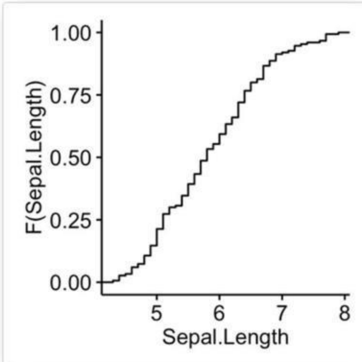
```
gghistogram(my_data, x = "Sepal.Length", bins = 9,
            add = "mean")
```



Empirical cumulative distribution function (ECDF)

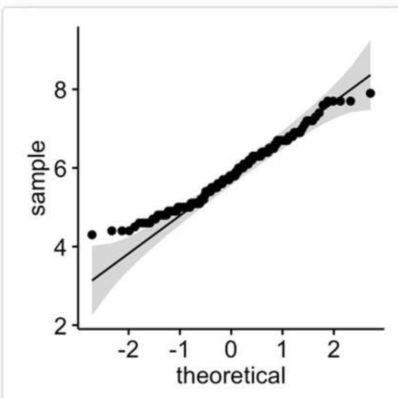
ECDF is the fraction of data smaller than or equal to x.

```
ggecdf(my_data, x = "Sepal.Length")
```



QQ plots are used to check whether the data is normally distributed.

```
ggqqplot(my_data, x = "Sepal.Length")
```



Descriptive statistics by groups

To compute summary statistics by groups, the functions **group_by**() and **summarise**() [in **dplyr** package] can be used.

- We want to group the data by *Species* and then:
  - compute the number of element in each group. R function: **n**()
  - compute the mean. R function **mean**()
  - and the standard deviation. R function **sd**()

Install **ddplyr** as follow:

```
install.packages("dplyr")
```

Descriptive statistics by groups:

To compute summary statistics by groups, the functions **group_by**() and **summarise**() [in **dplyr** package] can be used.

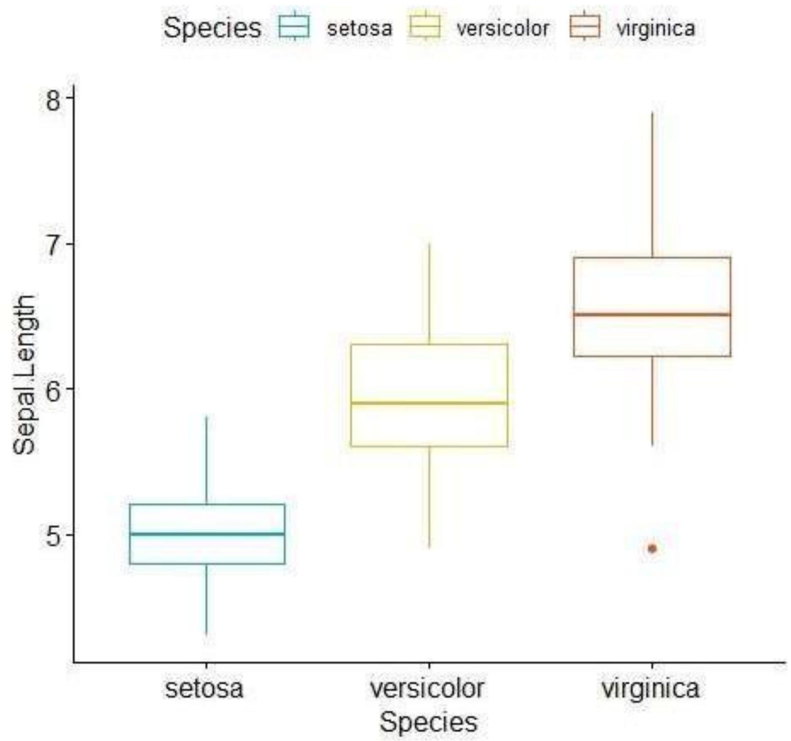
- We want to group the data by *Species* and then:
  - compute the number of element in each group. R function: **n**()
  - compute the mean. R function **mean**()
  - and the standard deviation. R function **sd**()

%>% is used to chain the operations.

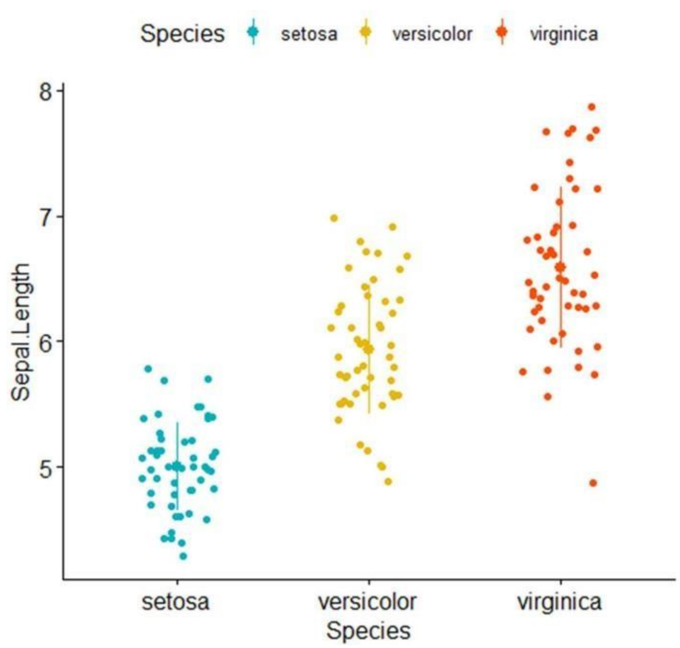
```
> library(dplyr)

Attaching package: ‘dplyr’

The following objects are masked from ‘package:stats’:

    filter, lag

The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union

> group_by(my_data, Species) %>%
+     summarise(
+         count = n(),
+         mean = mean(Sepal.Length, na.rm = TRUE),
+         sd = sd(Sepal.Length, na.rm = TRUE)
+     )
# A tibble: 3 × 4
  Species    count  mean    sd
  <fct>      <int> <dbl> <dbl>
1 setosa        50  5.01 0.352
2 versicolor    50  5.94 0.516
3 virginica     50  6.59 0.636
> 
```

Graphics for grouped data:

```
> # Box plot colored by groups: Species
> ggboxplot(my_data, x = "Species", y = "Sepal.Length",
+           color = "Species",
+           palette = c("#00AFBB", "#E7B800", "#FC4E07"))
```




```
> # Stripchart colored by groups: Species
> ggstripchart(my_data, x = "Species", y = "Sepal.Length",
+              color = "Species",
+              palette = c("#00AFBB", "#E7B800", "#FC4E07"),
+              add = "mean_sd")
```



Frequency tables

A frequency table (or contingency table) is used to describe categorical variables. It contains the counts at each combination of factor levels.

R function to generate tables: **table**()

For this section we will use the built-in R dataset that contains the distribution of hair and eye color by sex of 592 students:

```
> # Hair/eye color data
> df <- as.data.frame(HairEyeColor)
> hair_eye_col <- df[rep(row.names(df), df$Freq), 1:3]
> rownames(hair_eye_col) <- 1:nrow(hair_eye_col)
> head(hair_eye_col)
   Hair   Eye  Sex
1 Black Brown Male
2 Black Brown Male
3 Black Brown Male
4 Black Brown Male
5 Black Brown Male
6 Black Brown Male
```

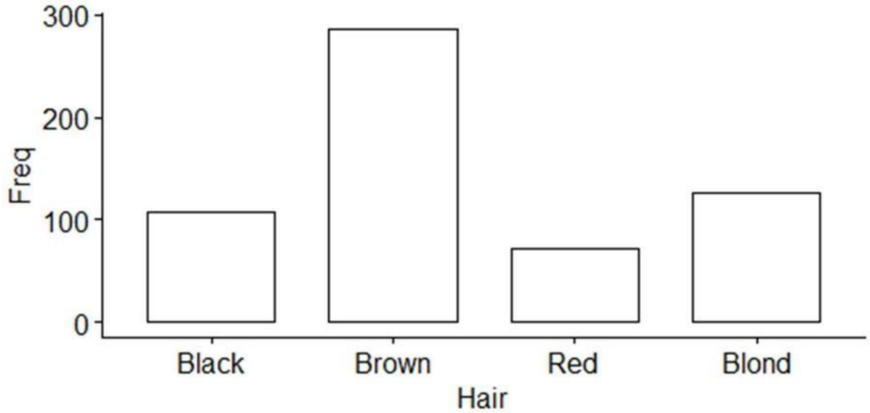
Simple frequency distribution: one categorical variable

Table of counts

```
> # hair/eye variables
> Hair <- hair_eye_col$Hair
> Eye <- hair_eye_col$Eye
> # Frequency distribution of hair color
> table(Hair)
Hair
Black Brown   Red Blond
  108   286    71   127
> # Frequency distribution of eye color
> table(Eye)
Eye
Brown  Blue Hazel Green
  220   215    93    64
```

Visualization:

```
> # Visualize using bar plot
> library(ggpubr)
> ggbarplot(df, x = "Hair", y = "Freq")
```



Two-way contingency table: Two categorical variables

```
> hair_eye <- table(Hair , Eye)
> hair_eye
       Eye
Hair    Brown Blue Hazel Green
  Black    68   20    15     5
  Brown   119   84    54    29
  Red      26   17    14    14
  Blond     7   94    10    16
```

Multiway tables: More than two categorical variables

- Hair and Eye color distributions by sex using **xtabs**():

```
> xtabs(~Hair + Eye + Sex, data = hair_eye_col)
, , Sex = Male

       Eye
Hair    Brown Blue Hazel Green
  Black    32   11    10     3
  Brown    53   50    25    15
  Red      10   10     7     7
  Blond     3   30     5     8

, , Sex = Female

       Eye
Hair    Brown Blue Hazel Green
  Black    36    9     5     2
  Brown    66   34    29    14
  Red      16    7     7     7
  Blond     4   64     5     8
```

You can also use the function **ftable**() [for flat contingency tables]. It returns a cleaner looking output compared to xtabs() when you have more than two variables:

```
> ftable(Sex + Hair ~ Eye, data = hair_eye_col)
      Sex   Male                    Female
      Hair Black Brown Red Blond  Black Brown Red Blond
Eye
Brown         32    53  10     3     36    66  16     4
Blue          11    50  10    30      9    34   7    64
Hazel         10    25   7     5      5    29   7     5
Green          3    15   7     8      2    14   7     8
```

Compute table margins and relative frequency

Table margins correspond to the sums of counts along rows or columns of the table.

```
> # Margin of rows
> margin.table(hair_eye, 1)
Hair
Black Brown   Red Blond
  108   286    71   127

> # Margin of columns
> margin.table(hair_eye, 2)
Eye
Brown  Blue Hazel Green
  220   215    93    64
```

Relative frequencies express table entries as proportions of table margins (i.e., row or column totals).

```
> # Frequencies relative to row total
> prop.table(hair_eye, 1)
       Eye
Hair         Brown       Blue      Hazel      Green
  Black 0.62962963 0.18518519 0.13888889 0.04629630
  Brown 0.41608392 0.29370629 0.18881119 0.10139860
  Red   0.36619718 0.23943662 0.19718310 0.19718310
  Blond 0.05511811 0.74015748 0.07874016 0.12598425
> # Table of percentages
> round(prop.table(hair_eye, 1), 2)*100
       Eye
Hair    Brown Blue Hazel Green
  Black    63   19    14     5
  Brown    42   29    19    10
  Red      37   24    20    20
  Blond     6   74     8    13
> # Table of percentages
> round(prop.table(hair_eye, 1), 4)*100
       Eye
Hair    Brown  Blue Hazel Green
  Black 62.96 18.52 13.89  4.63
  Brown 41.61 29.37 18.88 10.14
  Red   36.62 23.94 19.72 19.72
  Blond  5.51 74.02  7.87 12.60
```

EXECUTION:

CODE:

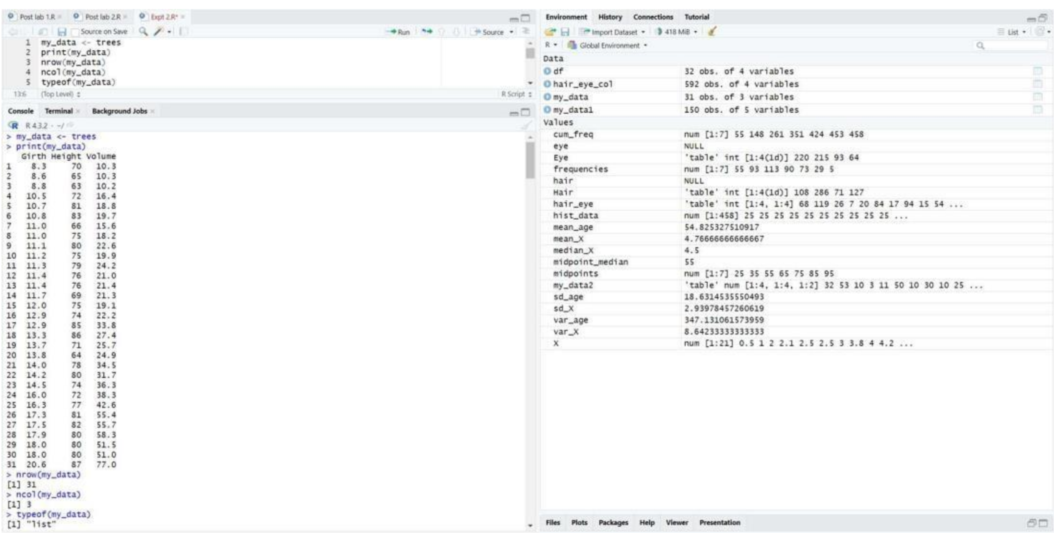
```
my_data <- trees
print(my_data)
nrow(my_data)
ncol(my_data)
typeof(my_data)

#Mean
mean(my_data$Girth)

#Median
median(my_data$Height)

#Mode
install.packages("modeest")
require(modeest)
mfv(my_data$Volume)

#Min
min(my_data$Height)

#Max
max(my_data$Height)

#Range
range(my_data$Girth)

#Quantiles
quantile(my_data$Girth)
quantile(my_data$Height)
quantile(my_data$Volume)

#Interquartile Range
IQR(my_data$Girth)

#Variance & Standard Deviation
var(my_data$Height)
sd(my_data$Volume)

#Median Absolute Deviation
mad(my_data$Girth)

#Summary
summary(my_data$Volume)
summary(my_data, digits = 2)
```

```
#Sapply
sapply(my_data[, -5], mean)
sapply(my_data[, -1], quantile)


#Box Plot
ggboxplot(my_data, y = "Girth", width = 0.5)

#Histogram
gghistogram(my_data, x = "Height", bins = 12, add = "mean")

#ecdf
ggecdf(my_data, x = "Volume")

#qqplot
ggqqplot(my_data, x = "Height")

#Descriptive Statistics
dplyr::group_by(my_data, Girth) %>%
dplyr::summarise(
   count = n(),
   mean = mean(Girth, na.rm = TRUE),
   sd = sd(Girth, na.rm = TRUE)
  )

#Box Plot colored by groups
my_data1 <- iris
print(my_data1)
ggstripchart(my_data1, x = "Species", y = "Sepal.Length", color = "Species",
        palette = c("#00AFBB", "#E7B800", "#FC4E07"))

#Frequency
my_data2 <- HairEyeColor
df <- as.data.frame(HairEyeColor)
hair_eye_col <- df[rep(row.names(df), df$Freq), 1:4]
rownames(hair_eye_col) <- 1:nrow(hair_eye_col)
head(hair_eye_col)

#Table of counts of categorical variables
Hair <- table(hair_eye_col$Hair)
Eye <- table(hair_eye_col$Eye)
print(Hair)
print(Eye)

#Visualization of the the counts
ggbarplot(df, x = "Hair", y = "Freq")

#Two-way contingency table
```

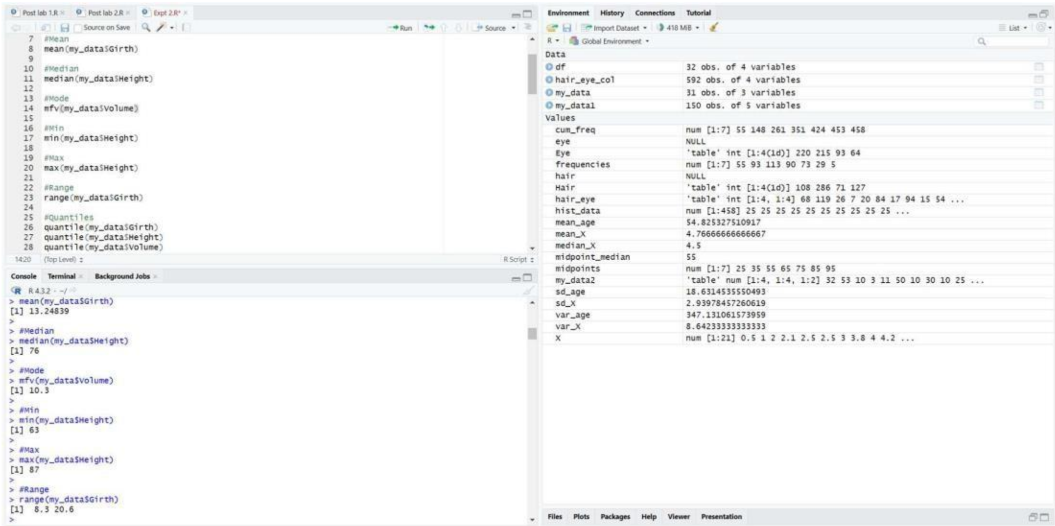
```
hair_eye <- table(hair_eye_col$Hair, hair_eye_col$Eye)
hair_eye

#Multiway Tables
xtabs(~Hair + Eye + Sex, data = hair_eye_col)

#Flat Contingency Table
ftable(Sex + Hair ~ Eye, data = hair_eye_col)

#Table Margins
margin.table(hair_eye, 1)
margin.table(hair_eye, 2)

#Relative Frequencies of row total
prop.table(hair_eye, 1)

#Table of Percentages
hair_eye <- table(hair_eye_col$Hair, hair_eye_col$Eye)
round(prop.table(hair_eye, 1), 2)*100
round(prop.table(hair_eye, 2), 4)*100
```

EXECUTION SCREENSHOTS:

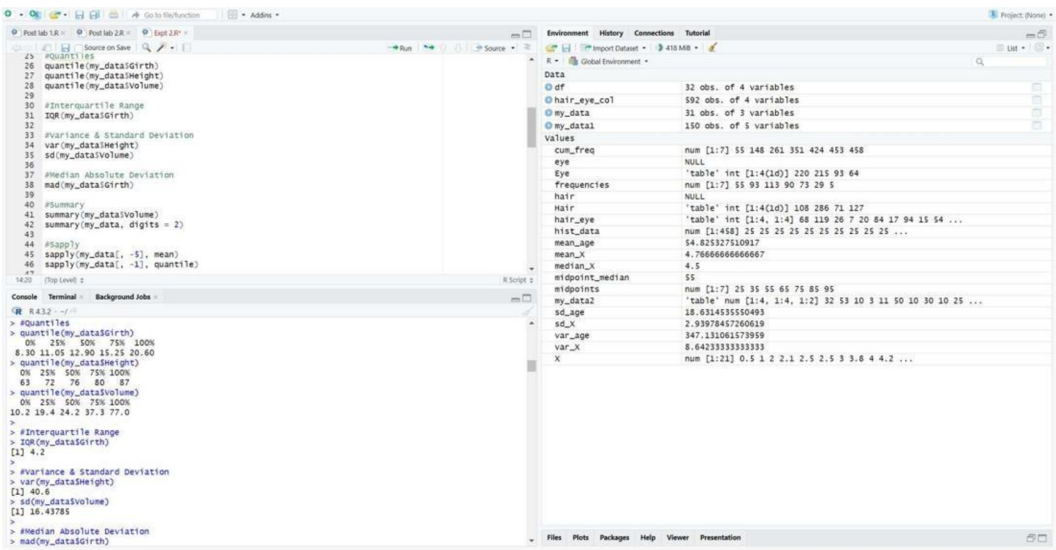
1]



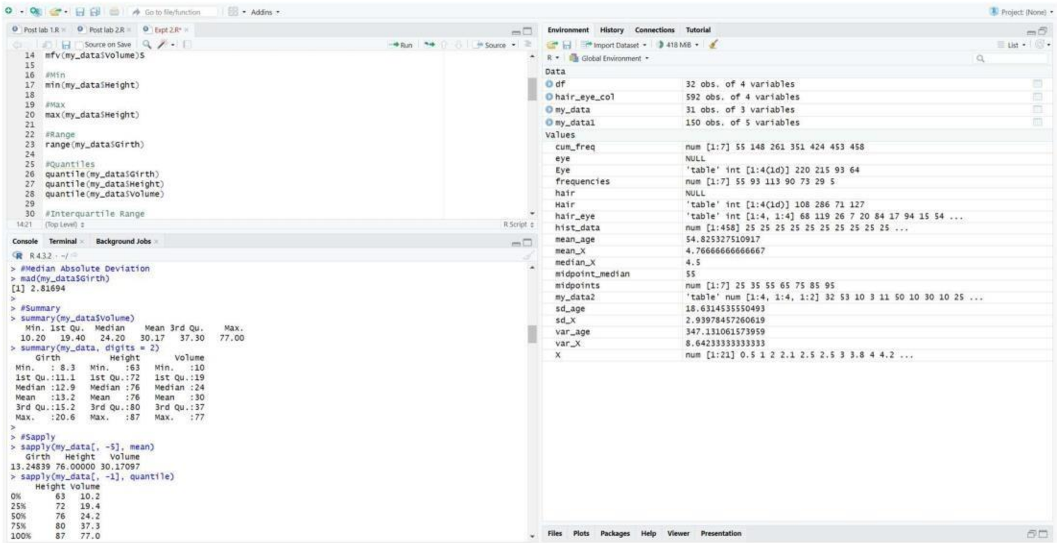
2]


#Mean
mean(my_data$Girth)
#Median
median(my_data$Height)
#Mode
mfv(my_data$Volume)
#Min
min(my_data$Height)
#Max
max(my_data$Height)
#Range
range(my_data$Girth)
#Quantiles
quantile(my_data$Girth)
quantile(my_data$Height)
quantile(my_data$Volume)
> mean(my_data$Girth)
[1] 13.24839
> #Median
> median(my_data$Height)
[1] 76
> #Mode
> mfv(my_data$Volume)
[1] 10.3
> #Min
> min(my_data$Height)
[1] 63
> #Max
> max(my_data$Height)
[1] 87
> #Range
> range(my_data$Girth)
[1] 8.3 20.6

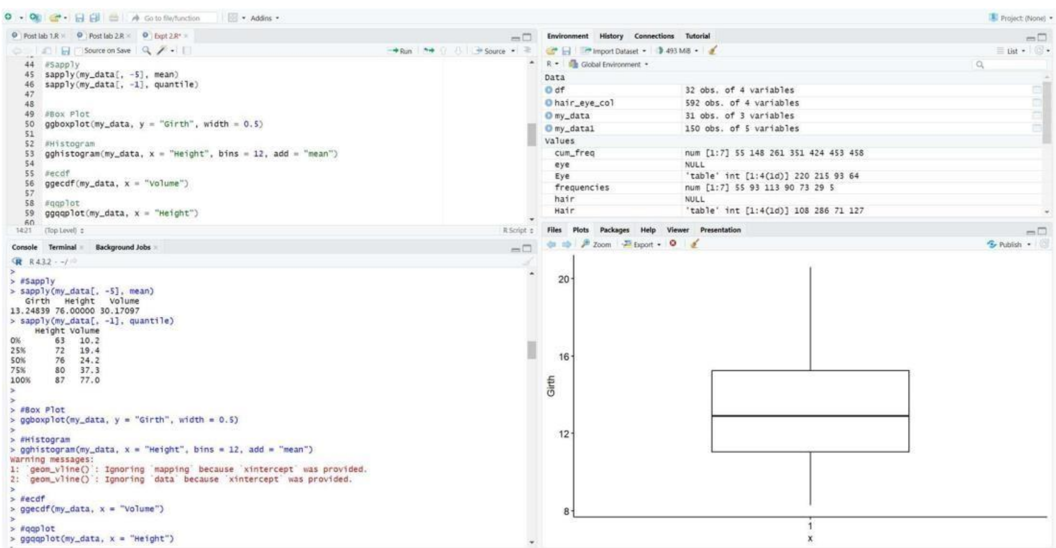
3]


#Quantiles
quantile(my_data$Girth)
quantile(my_data$Height)
quantile(my_data$Volume)
#Interquartile Range
IQR(my_data$Girth)
#Variance & Standard Deviation
var(my_data$Height)
sd(my_data$Volume)
#Median Absolute Deviation
mad(my_data$Girth)
#Summary
summary(my_data$Volume)
summary(my_data, digits = 2)
#Sapply
sapply(my_data[, -5], mean)
sapply(my_data[, -1], quantile)
> #Quantiles
> quantile(my_data$Girth)
0% 25% 50% 75% 100%
8.30 11.05 12.90 15.25 20.60
> quantile(my_data$Height)
0% 25% 50% 75% 100%
63 72 76 80 87
> quantile(my_data$Volume)
0% 25% 50% 75% 100%
10.2 19.4 24.2 37.3 77.0
> #Interquartile Range
> IQR(my_data$Girth)
[1] 4.2
> #Variance & Standard Deviation
> var(my_data$Height)
[1] 40.6
> sd(my_data$Volume)
[1] 16.43785
> #Median Absolute Deviation
> mad(my_data$Girth)

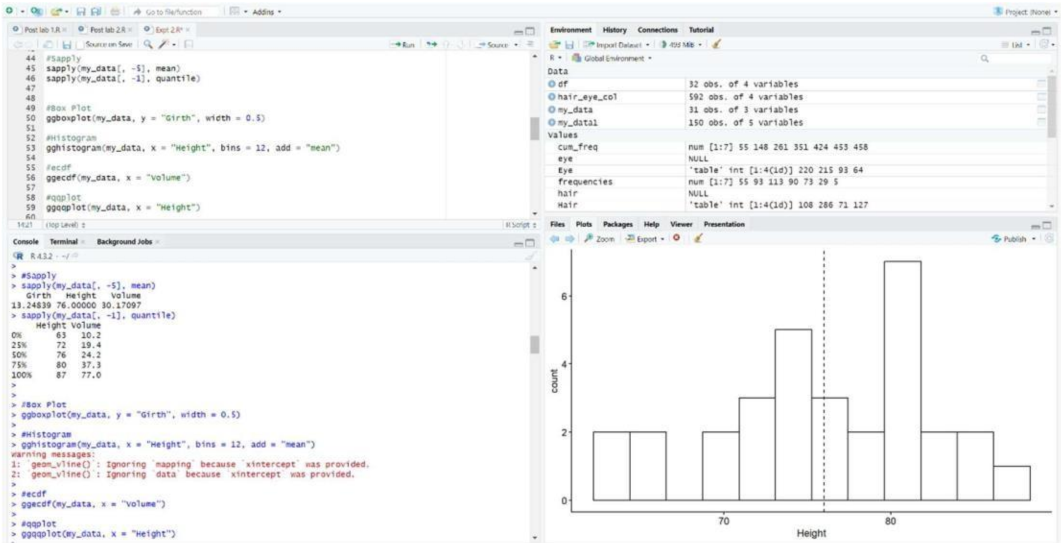
4]


mfv(my_data$Volume)$
#Min
min(my_data$Height)
#Max
max(my_data$Height)
#Range
range(my_data$Girth)
#Quantiles
quantile(my_data$Girth)
quantile(my_data$Height)
quantile(my_data$Volume)
#Interquartile Range
> #Median Absolute Deviation
> mad(my_data$Girth)
[1] 2.81694
> #Summary
> summary(my_data$Volume)
Min. 1st Qu. Median Mean 3rd Qu. Max.
10.20 19.40 24.20 30.17 37.30 77.00
> summary(my_data, digits = 2)
> #Sapply
> sapply(my_data[, -5], mean)
Girth Height Volume
13.24839 76.00000 30.17097
> sapply(my_data[, -1], quantile)
df 32 obs. of 4 variables
hair_eye_col 592 obs. of 4 variables
my_data 31 obs. of 3 variables
my_data1 150 obs. of 5 variables

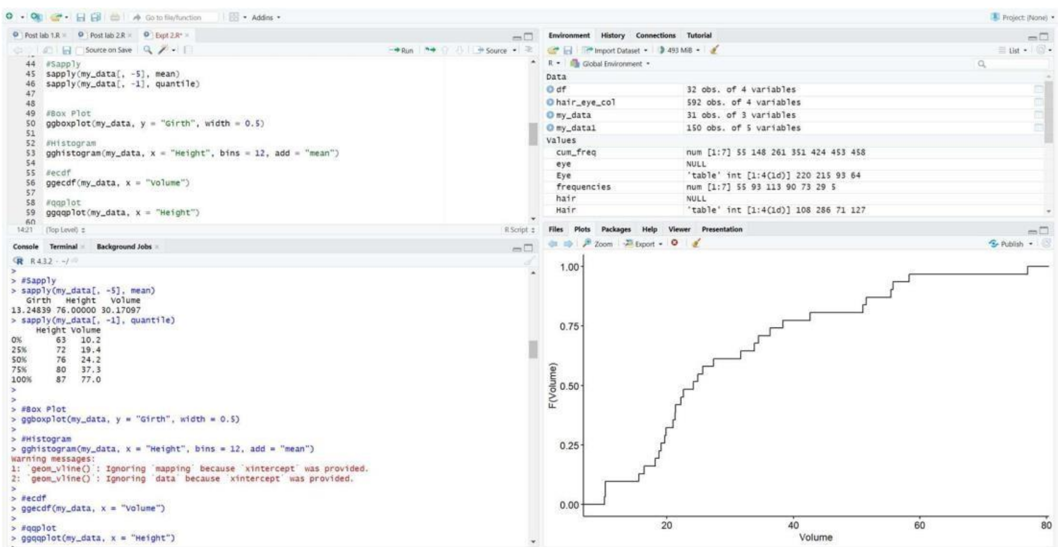
5]


#Sapply
sapply(my_data[, -5], mean)
sapply(my_data[, -1], quantile)
#Box Plot
ggboxplot(my_data, y = "Girth", width = 0.5)
#Histogram
gghistogram(my_data, x = "Height", bins = 12, add = "mean")
#ecdf
ggecdf(my_data, x = "Volume")
#qqplot
ggqqplot(my_data, x = "Height")
Warning messages:
1: `geom_vline()`: Ignoring `mapping` because `xintercept` was provided.
2: `geom_vline()`: Ignoring `data` because `xintercept` was provided.
Girth
x

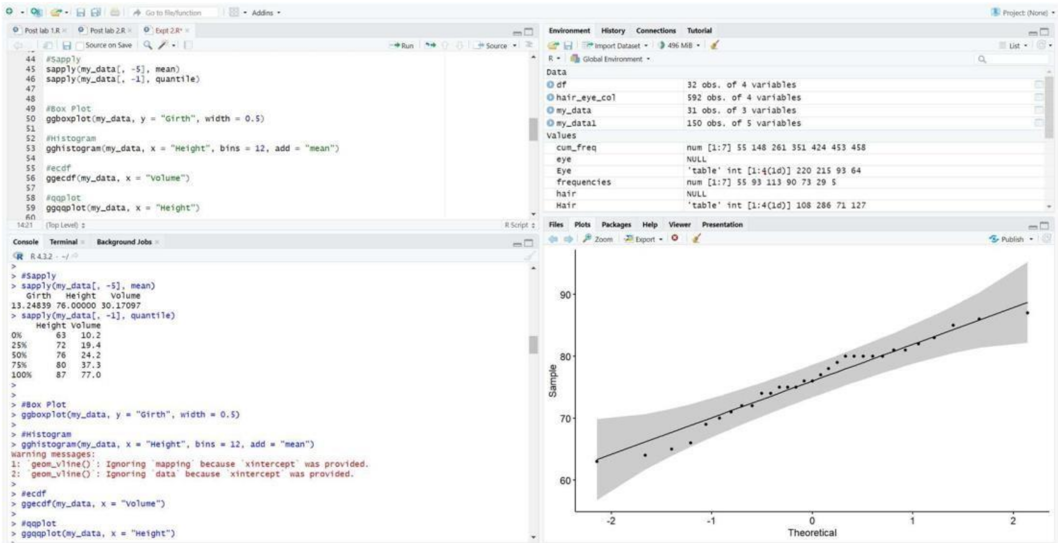
6]



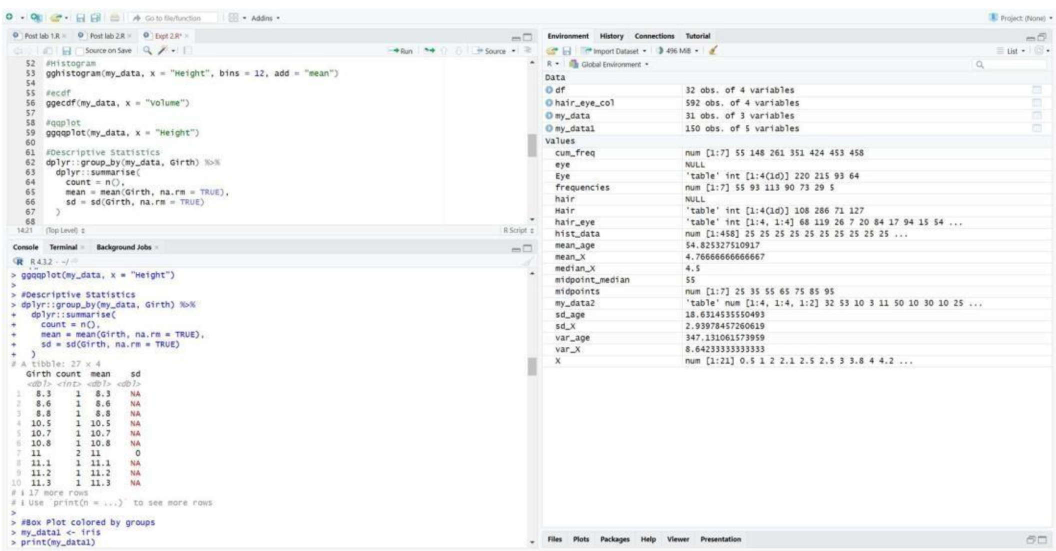
7]



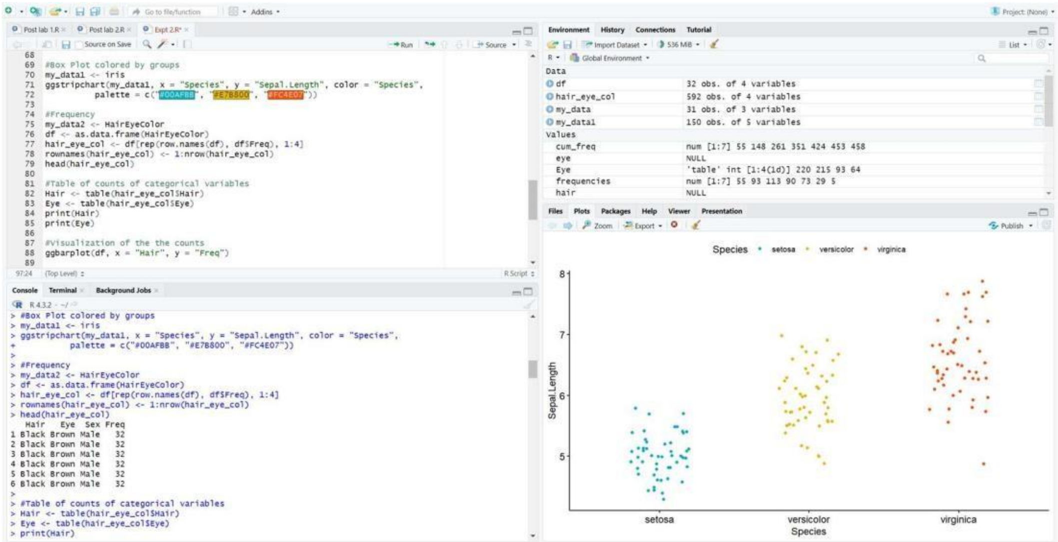
8]



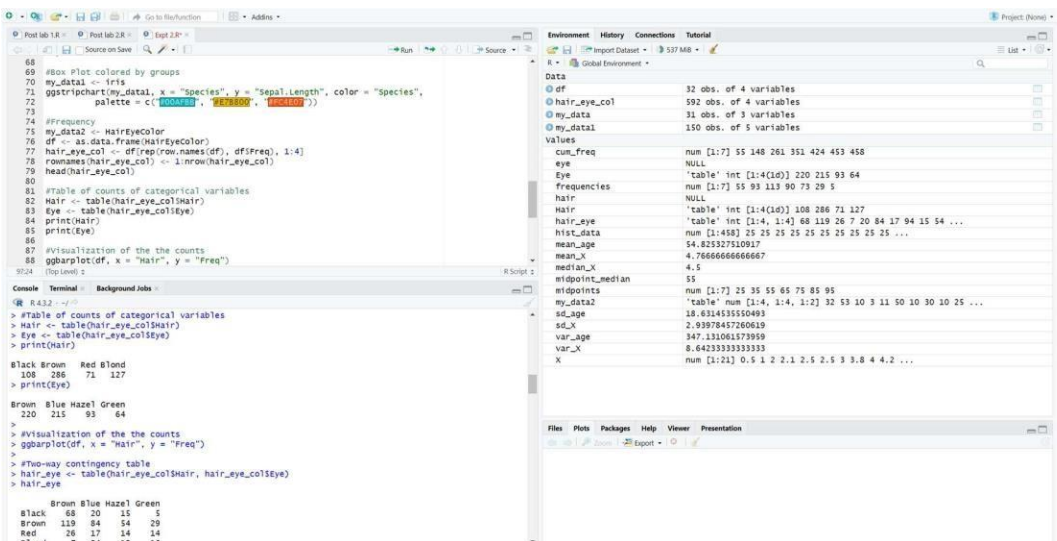
9]



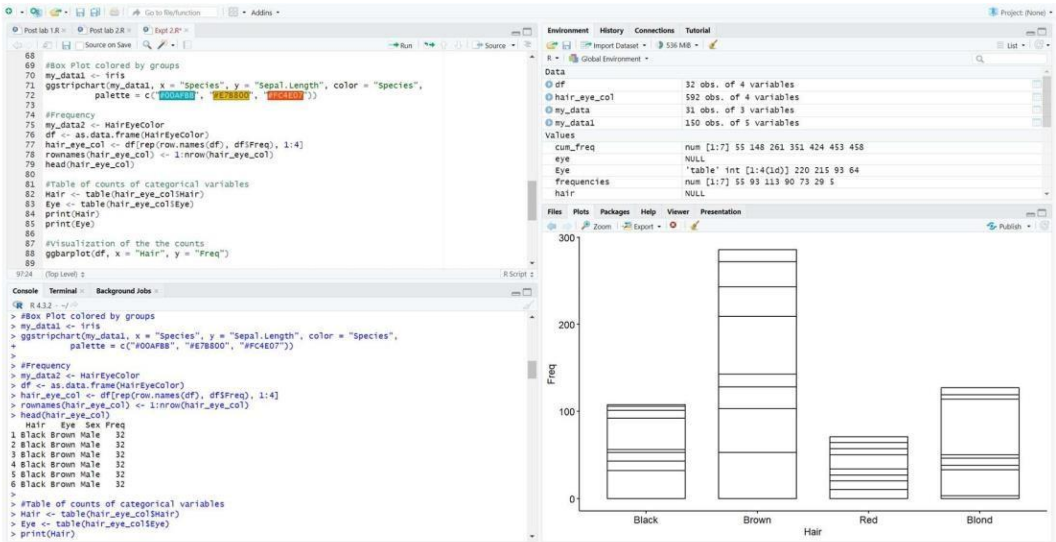
10]


Species setosa versicolor virginica
Sepal.Length
setosa
versicolor
virginica
Species

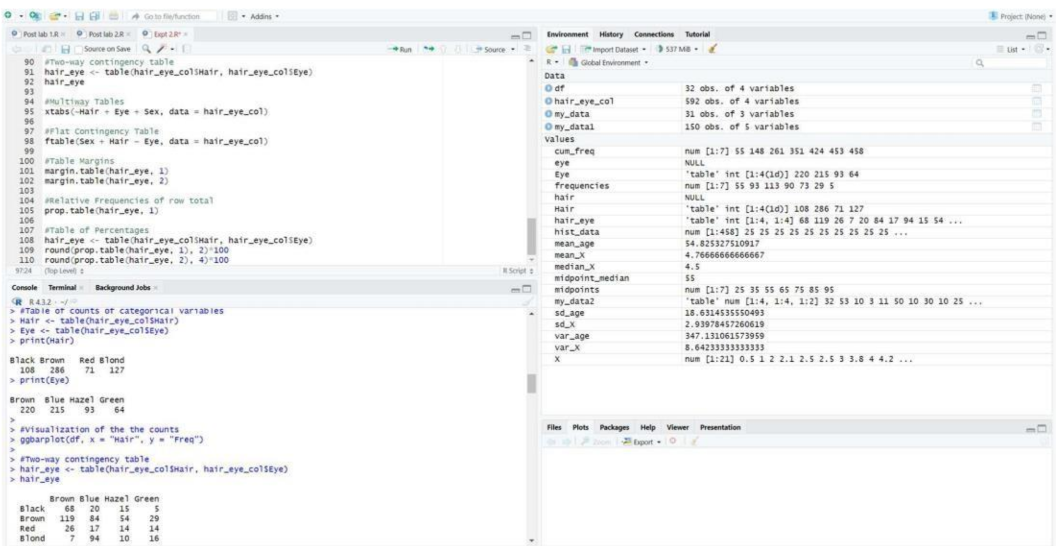
11]



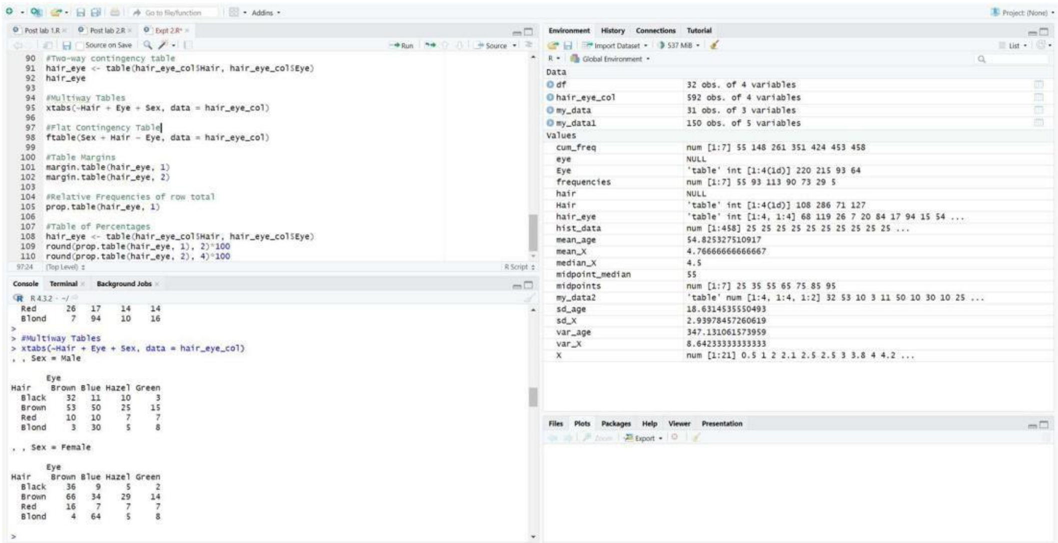
12]



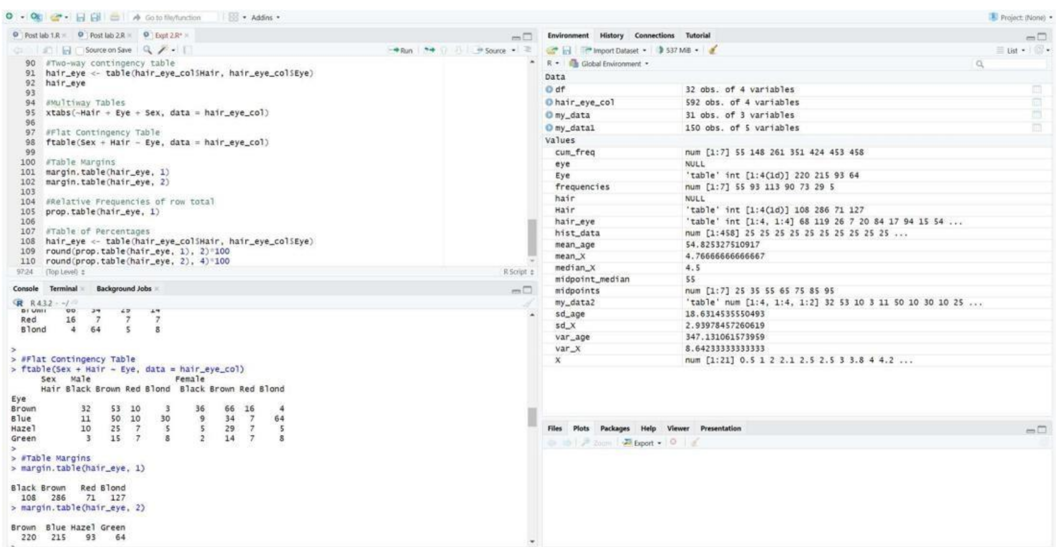
13]



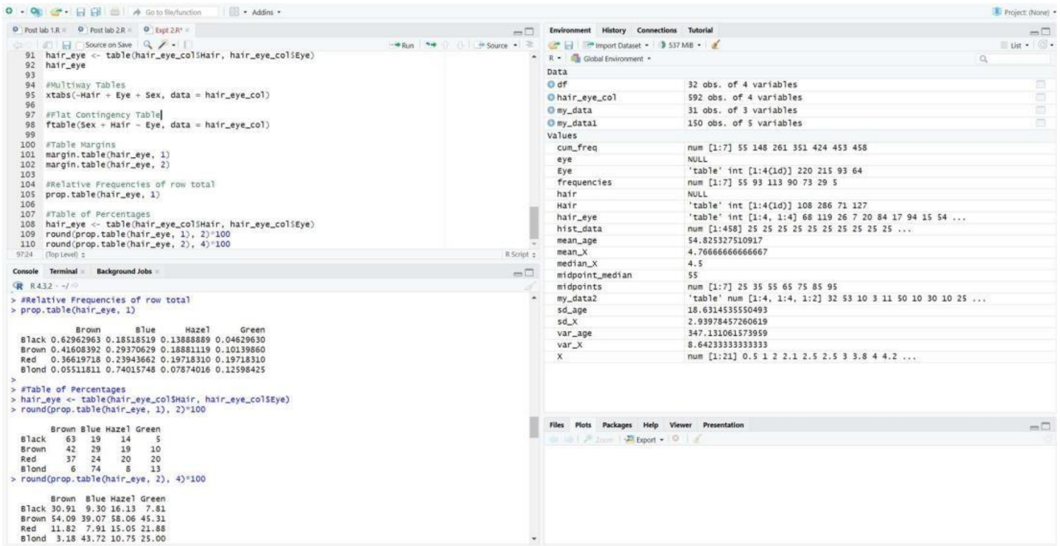
14]


#Two-way contingency table
hair_eye <- table(hair_eye_col$Hair, hair_eye_col$Eye)
hair_eye
#Multiway Tables
xtabs(~Hair + Eye + Sex, data = hair_eye_col)
#Flat Contingency Table
ftable(Sex + Hair ~ Eye, data = hair_eye_col)
#Table Margins
margin.table(hair_eye, 1)
margin.table(hair_eye, 2)
#Relative Frequencies of row total
prop.table(hair_eye, 1)
#Table of Percentages
hair_eye <- table(hair_eye_col$Hair, hair_eye_col$Eye)
round(prop.table(hair_eye, 1), 2)*100
round(prop.table(hair_eye, 2), 4)*100
> #Multiway Tables
> xtabs(~Hair + Eye + Sex, data = hair_eye_col)
, , Sex = Male
, , Sex = Female

15]


> #Flat Contingency Table
> ftable(Sex + Hair ~ Eye, data = hair_eye_col)
> #Table Margins
> margin.table(hair_eye, 1)
Black Brown Red Blond
108 286 71 127
> margin.table(hair_eye, 2)
Brown Blue Hazel Green
220 215 93 64

16]



Conclusion: Applying descriptive statistics in Applied Data Science reveals insights into central tendencies, variabilities, and distributions. It forms a foundational step for informed decision-making and strategic data-driven approaches.

Post Lab questions

Write R commands for the following

1. In an article in American Journal of Pathology, Pitts et al (2001) have taken the measurements on diameters in centimetres of the neoplasm removed from the breasts of 20 subjects with pure sarcoma. Following is the dataset: 0.5, 1, 2, 2.1, 2.5, 2.5, 3.0, 3.8, 4.0, 4.2, 4.5, 5.0, 5.0, 5.0, 5.0, 6.0, 6.5, 7.0, 8.0, 9.5, 13.0

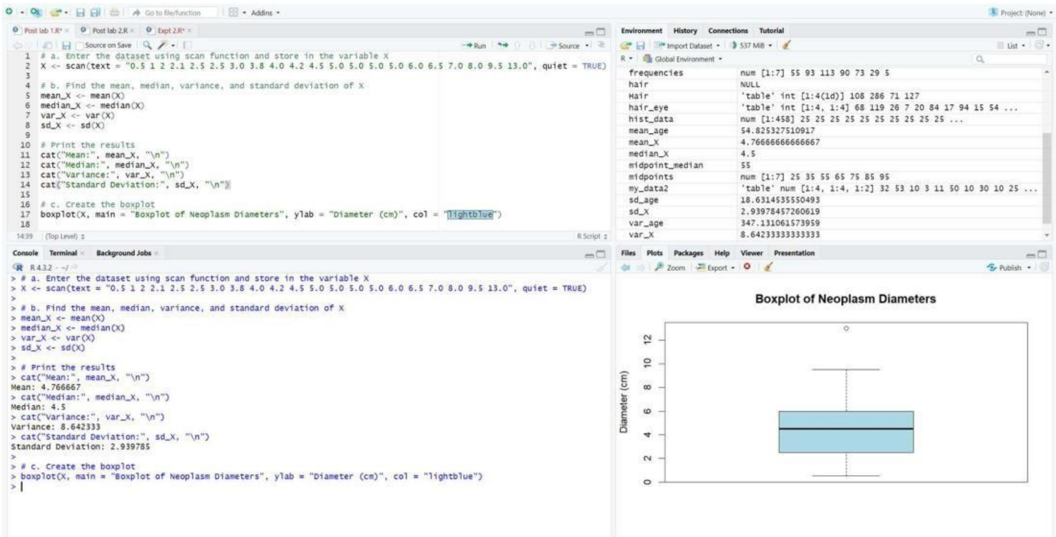
a. Enter the dataset using scan function and store in the variable X
b. Find the mean, median, variance and standard deviation of x
c. Create the boxplot

SOLUTION:

CODE:

```
# a. Enter the dataset using scan function and store in the variable X
X <- scan(text = "0.5 1 2 2.1 2.5 2.5 3.0 3.8 4.0 4.2 4.5 5.0 5.0 5.0 5.0 6.0 6.5 7.0 8.0
9.5 13.0")

# b. Find the mean, median, variance, and standard deviation of X
mean_X <- mean(X)
median_X <- median(X)
var_X <- var(X)
sd_X <- sd(X)

# Print the results
cat("Mean:", mean_X, "\n")
cat("Median:", median_X, "\n")
cat("Variance:", var_X, "\n")
cat("Standard Deviation:", sd_X, "\n")

# c. Create the boxplot
boxplot(X, main = "Boxplot of Neoplasm Diameters", ylab = "Diameter (cm)", col =
"lightblue")
```

EXECUTION SCREENSHOT:



2. American Journal of psychiatry conducted a study of the presence of significant psychiatric illness in heterozygous carriers of the gene for the Wolfram syndrome. Among the subject studied were 543 blood relatives of patients of Wolfram syndrome. Following is the frequency distribution of ages of these blood relatives:

Age(Mid-point)	25	35	55	65	75	85	95
Number(Frequency)	55	93	113	90	73	29	5

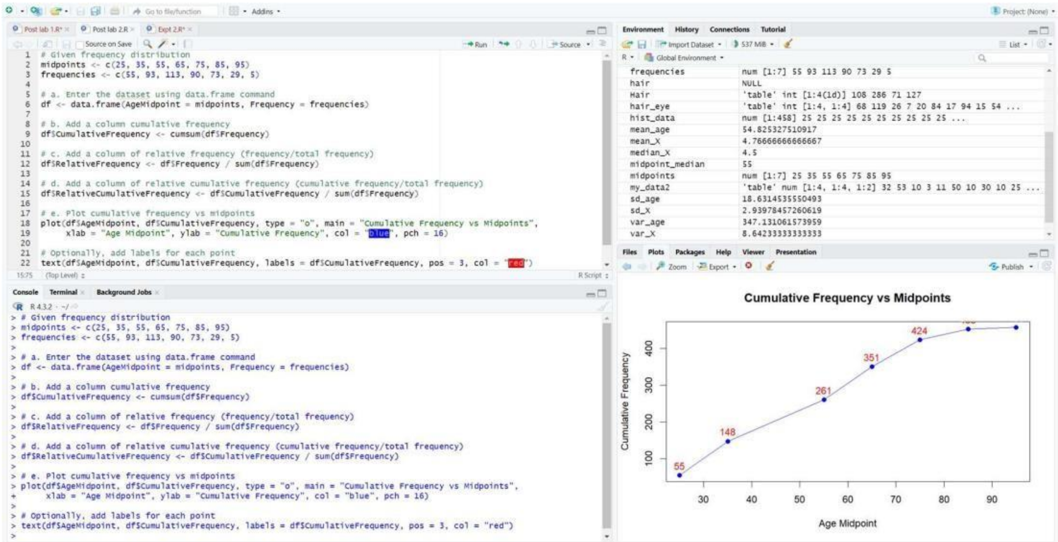
a. Enter the dataset using data.frame command
b. Add a column cumulative frequency
c. Add a column of relative frequency (frequency/total frequency)
d. Add a column of relative cumulative frequency (cumulative frequency/total frequency)
e. Plot cumulative frequency vs mid points

SOLUTION:

CODE:

```
# Given frequency distribution
midpoints <- c(25, 35, 55, 65, 75, 85, 95)
frequencies <- c(55, 93, 113, 90, 73, 29, 5)

# a. Enter the dataset using data.frame command
df <- data.frame(AgeMidpoint = midpoints, Frequency = frequencies)

# b. Add a column cumulative frequency
df$CumulativeFrequency <- cumsum(df$Frequency)

# c. Add a column of relative frequency (frequency/total frequency)
df$RelativeFrequency <- df$Frequency / sum(df$Frequency)

# d. Add a column of relative cumulative frequency (cumulative frequency/total frequency)
df$RelativeCumulativeFrequency <- df$CumulativeFrequency / sum(df$Frequency)

# e. Plot cumulative frequency vs midpoints
plot(df$AgeMidpoint, df$CumulativeFrequency, type = "o", main = "Cumulative Frequency vs
Midpoints",
    xlab = "Age Midpoint", ylab = "Cumulative Frequency", col = "blue", pch = 16)

# Optionally, add labels for each point
text(df$AgeMidpoint, df$CumulativeFrequency, labels = df$CumulativeFrequency, pos = 3, col
= "red")
```

EXECUTION SCREENSHOT:



3. Critically assess the limitations of using only measures of central tendency in data analysis.

Limitations of Using Only Measures of Central Tendency:

Central tendency measures (mean, median, mode) have limitations:

Ignoring Distribution Shape: They don't provide information about the shape ofthe distribution. Two datasets with the same mean might have very different distributions.

Sensitivity to Outliers: The mean is sensitive to extreme values (outliers), and a few outliers can significantly distort the mean. Median is less affected, but still may not be entirely robust.

Not Descriptive of Spread: Central tendency measures don't give insights into the spread or dispersion of data. Two datasets with the same mean can have different levels of variability.

Applicability to Different Distributions: Different central tendency measures might be more suitable for different types of data (e.g., median for skewed data, mean for symmetric).

4. Compare and contrast the different measures of variability, with the focus on when one measure might be more informative than the other.

Comparison of Measures of Variability:

Measures of variability (range, variance, standard deviation, interquartile range) have distinct characteristics:

Range: Simple but sensitive to outliers; it doesn't capture the overall spread effectively.

Variance and Standard Deviation: Provide a more nuanced understanding of the spread around the mean; sensitive to outliers.

Interquartile Range (IQR): Captures the spread of the middle 50% of the data, less sensitive to extreme values.

When to Use One Measure Over Another:

Use Range for Simplicity: When simplicity is crucial and extreme values are not a significant concern.

Use Variance/SD for Precision: When a more precise measure of spread is needed and outliers need to be considered.

Use IQR for Robustness: When you want a measure that is less sensitive to extreme values.

5. Imagine you are presented with a dataset from a research study. Discuss how applying descriptive statistics techniques could aid in understanding the key features and trends in the data. Take any real life examples to aid your analysis.

Descriptive Statistics Techniques in Understanding Data:
Example Scenario: Examining Exam Scores

Mean: Provides an average score, indicating the overall performance.

Median: Shows the middle point of the scores, helpful if there are extreme scores.

Mode: Identifies the most common score.

Variance/SD: Indicates the spread of scores around the mean.

Histogram/Boxplot: Visual representations to grasp the distribution shape and detect outliers.

Understanding these descriptive statistics aids in identifying trends, variations, and potential outliers, helping researchers make informed decisions and interpretations.

Real-Life Example: Examining Income Distribution

Mean Income: Gives an average income level.

Median Income: Provides the income level at the middle point.

Mode Income: Shows the most frequently occurring income range.

Standard Deviation: Indicates the variation in income levels.

Boxplot: Visualizes the income distribution and identifies potential outliers.

Analysing these descriptive statistics helps policymakers understand income disparities, target interventions, and make evidence-based decisions.