

BIG DATA ANALYTICS

ASSIGNMENT-3

Name-Anish Khandelwal

Sec-c(19)

Roll No.-201500093

Q1) Write a R program to perform below operations using R User-Defined Functions –

- (a) Creating and Calling User-Define Function
- (b) Number of Arguments
- (c) Default Arguments
- (d) Return Values
- (e) Nested Functions
- (f) Recursion

a)

```
R 4.2.1 · ~/ ~
> x <- function(fname="neeraj"){
+   paste(fname,"Kumar")
+ }
> x()
[1] "neeraj Kumar"
```

b)

```
> x <-function(fname, lname)
+ {
+   paste(fname, lname)
+ }
> x("Anish", "Khandelwal")
[1] "Anish Khandelwal"
>
```

c)

```
> x <-function(fname, cname)
+ {
+   paste("My name is " ,fname , " and i am from " ,cname)
+ }
> x("Anish Khandelwl" , "India")
[1] "My name is Anish Khandelwl and i am from India"
>
```

d)

```
> y <-function(a){
+   return(a*a)
+ }
> y(10)
[1] 100
>
```

e)

```
> x <- function(a){  
+   return(a)  
+ }  
> y <- function(b){  
+   print(x(a)+b)  
+ }  
> x(5)  
[1] 5  
> y(12)  
[1] 17  
> | .
```

f)

```
> factorial <- function(a)  
+ {  
+   if(a==1){  
+     return(a)  
+   }else{  
+     return(a*factorial(a-1))  
+   }  
+ }  
> factorial(7) .  
[1] 5040  
> |
```

2) Write a R program to perform below operations using Data Frames –

- (a) Create Data Frame
- (b) Summarize the Data
- (c) Access Items
- (d) Add Rows & Columns
- (e) Remove Rows and Columns
- (f) Amount of Rows and Columns
- (g) Data Frame Length
- (h) Combining Data Frames

a)

```
R 4.2.1 · ~/ ↗
> a <- data.frame(
+   Training = c("Strength", "Stamina", "Other"),
+   Pulse = c(100, 150, 120),
+   Duration = c(60, 30, 45)
+ )
> print(a)
  Training Pulse Duration
1 Strength    100       60
2 Stamina     150       30
3 Other        120       45
> |
```

b)

```
> summary(a)
  Training          Pulse          Duration
Length:3           Min.   :100.0   Min.   :30.0
Class :character  1st Qu.:110.0  1st Qu.:37.5
Mode  :character  Median  :120.0  Median  :45.0
                           Mean   :123.3  Mean   :45.0
                           3rd Qu.:135.0  3rd Qu.:52.5
                           Max.   :150.0  Max.   :60.0
> |
```

c)

```
> a[1]
Training
1 Strength
2 Stamina
3 Other
> a[["Training"]]
[1] "Strength" "Stamina" "other"
> a#Training
Training Pulse Duration
1 Strength    100      60
2 Stamina     150      30
3 Other        120      45
> |
```

d)

```
> new_row_DF <- rbind(a, c("Strength", 110, 110))
> b <- rbind(a, c("Strength", 110, 110))
> print(b)
Training Pulse Duration
1 Strength    100      60
2 Stamina     150      30
3 Other        120      45
4 Strength    110      110
|
```

e)

```
> d <- a[-c(1), -c(1)]
> print(d)
Pulse Duration
2    150      30
3    120      45
> |
```

f)

```
> dim(a)
[1] 3 3
> |
```

g)

```
+ )
> length(a)
[1] 3
> |
```

h)

```
> a <- data.frame(  
+   Training = c("Strength", "Stamina", "Other"),  
+   Pulse = c(100, 150, 120),  
+   Duration = c(60, 30, 45)  
+ )  
> b <- data.frame (  
+   Training = c("Stamina", "Stamina", "Strength"),  
+   Pulse = c(140, 150, 160),  
+   Duration = c(30, 30, 20)  
+ )  
>  
> b <- rbind(a, b)  
> print(b)  
  Training Pulse Duration  
1 Strength    100       60  
2 Stamina     150       30  
3 Other        120       45  
4 Stamina     140       30  
5 Stamina     150       30  
6 Strength    160       20  
> |
```

Q3) Write a R program to perform below operations using R Factors – (a) Create a factor (b) Factor Length (c) Access Factors (d) Change Item Value (e) Print the levels using levels() function:

a)

```
> a <- factor(c("Jazz", "Rock", "Classic", "Classic", "Pop", "Jazz", "Rock", "Jazz"))
> print(a)
[1] Jazz      Rock      Classic  Classic Pop      Jazz      Rock      Jazz
Levels: Classic Jazz Pop Rock
> |
```

b)

```
> a <- factor(c("Jazz", "Rock", "Classic", "Classic", "Pop", "Jazz", "Rock", "Jazz"))
> length(a)
[1] 8
> |
```

c)

```
> a <- factor(c("Jazz", "Rock", "Classic", "Classic", "Pop", "Jazz", "Rock", "Jazz"))
> a[2]
[1] Rock
Levels: Classic Jazz Pop Rock
```

d)

```
> print(a)
[1] Jazz      <NA>    Classic  Classic Pop      Jazz      Rock      Jazz
Levels: Classic Jazz Pop Rock
> |
```

Q4) Create R Factors in Data Frame that prints data with a column of text into categorical form using R Factor.

```
> data <- c("East","West","East","North","North","East","West","West","West","East","North")
>
> print(data)
[1] "East"  "West"  "East"  "North" "North" "East"  "West"  "West"  "West"  "East"  "North"
> print(is.factor(data))
[1] FALSE
>
>
> factor_data <- factor(data)
>
> print(factor_data)
[1] East  West  East  North North East  West  West  West  East  North
Levels: East North West
> print(is.factor(factor_data))
[1] TRUE
> |
```

GLA UNIVERSITY, MATHURA

Mathura- 281406, UP - INDIA



Session:- 2022-2023

Institute Of engineering & technology

Department Of Computer Engineering & Applications

Course Name: Big Data and Analytics Lab (Code: BCSE0183)

Semester: 5th

Section: 3-C

Academic Session: Aug 2022 – Dec 2022

Lab Assignment – 06

Basic of R Programming Language using R Studio

Submitted to: Dr. Robin Singh Bhaduria

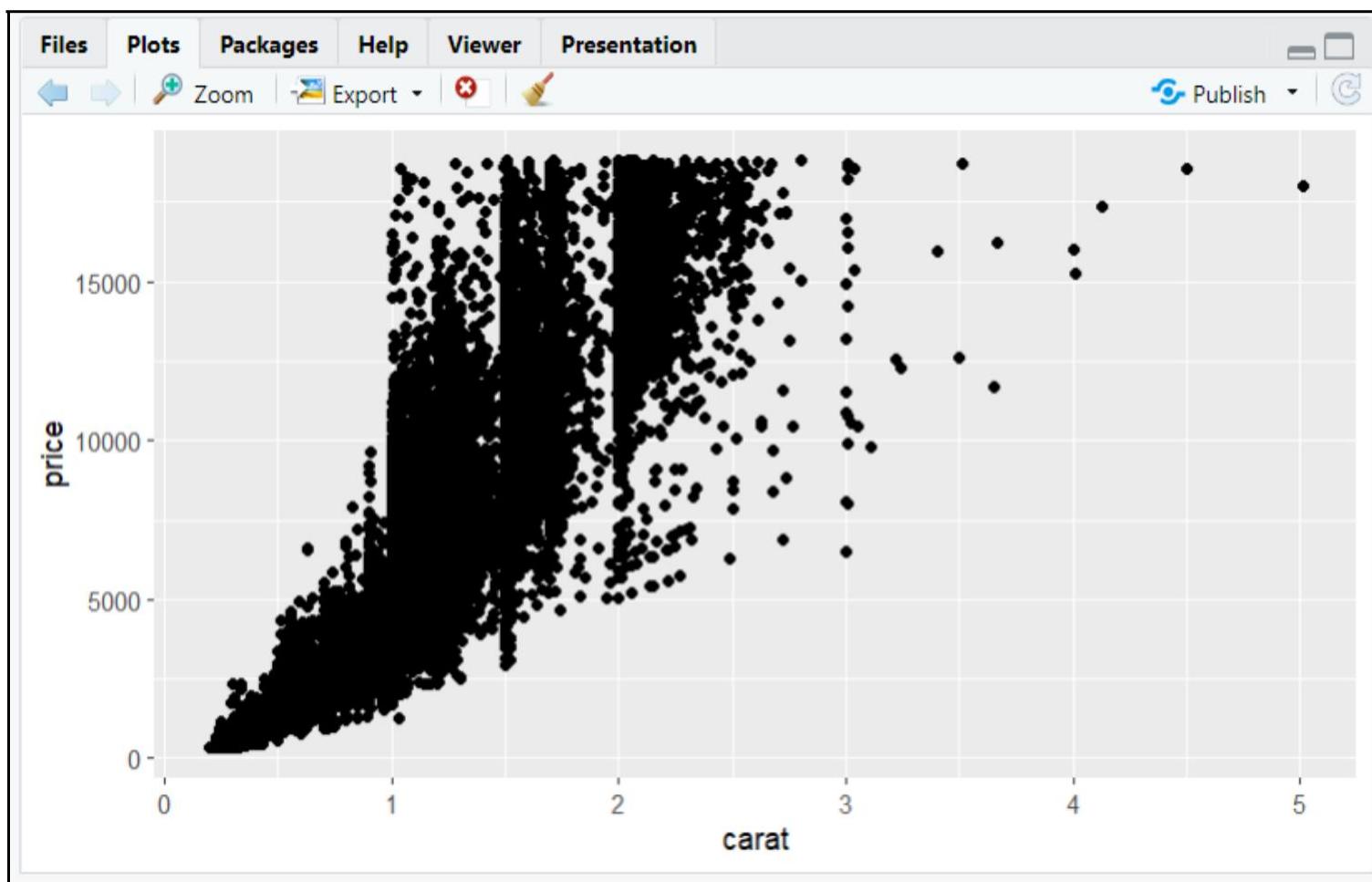
Submitted by: Aniruddha bajpai

Date: 22-11-2022

1) Use 'diamonds' dataset and qplot () of ggplot2 Package for following operations:

- (a) Plot a relationship between parameters like price and carats (weight)
- (b) Customize colour, size, & shape
- (c) Adding a smoother to a plot using geoms
- (d) Boxplots and jittered points using geoms
- (e) Histogram and density plots using geoms
- (f) Bar charts using geoms
- (g) faceting method in qplot()
- (h) Building a scatterplot using factor () function

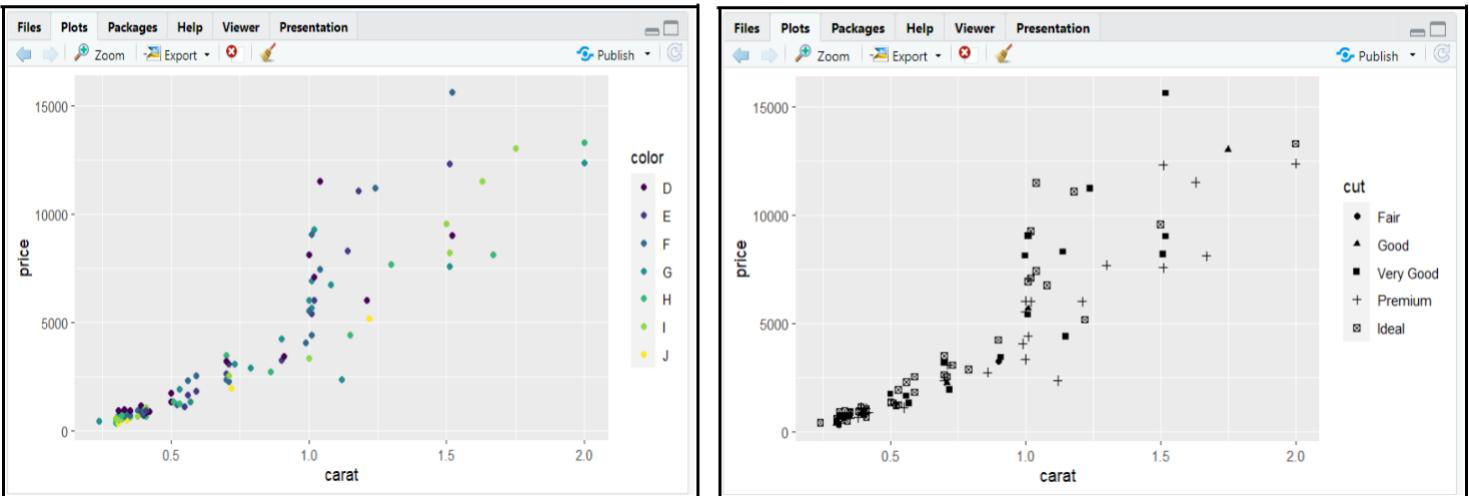
```
>
> qplot(carat, price, data = diamonds)
>
>
>
>
>
```



```

>
> qplot(carat, price, data = dsmall, colour = color)
> qplot(carat, price, data = dsmall, shape = cut)

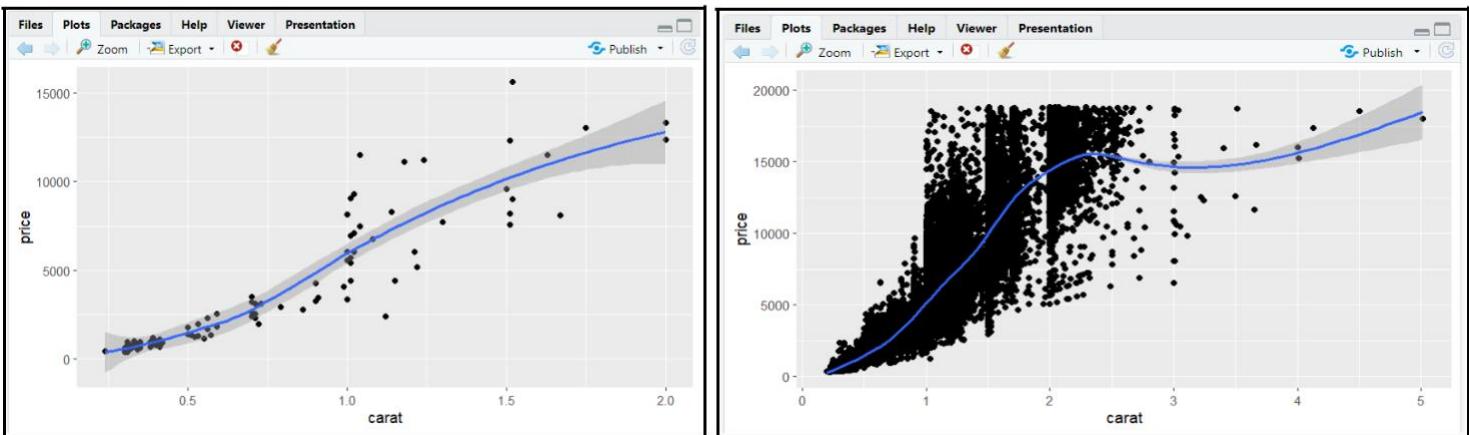
```



```

> qplot(carat, price, data = dsmall, geom = c("point", "smooth"))
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
>
> qplot(carat, price, data = diamonds, geom = c("point", "smooth"))
`geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
>

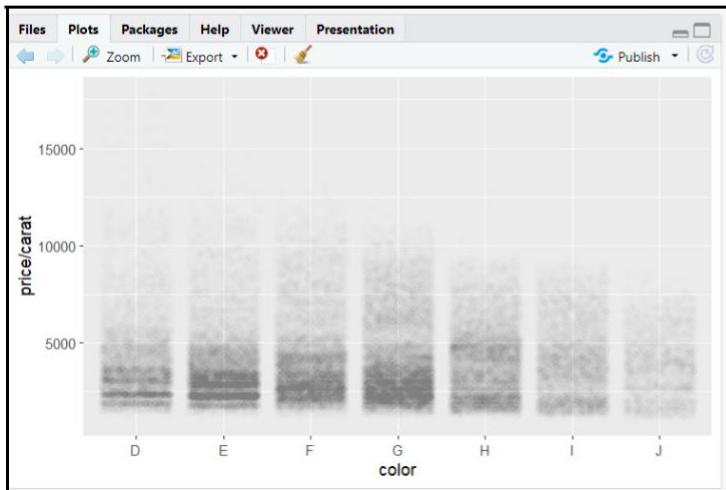
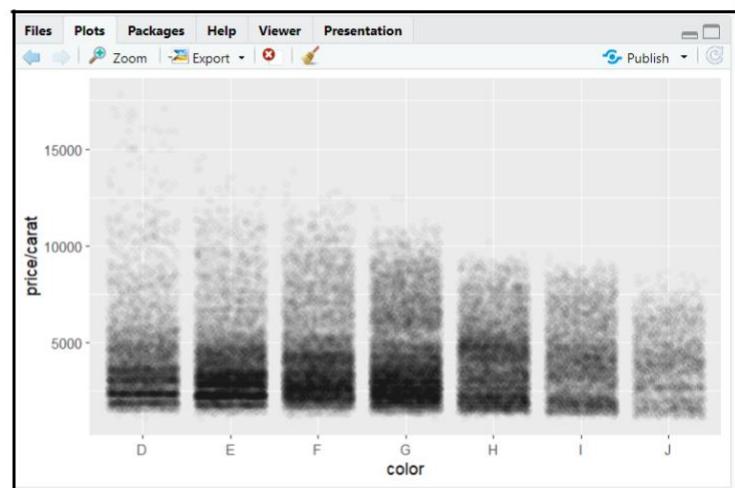
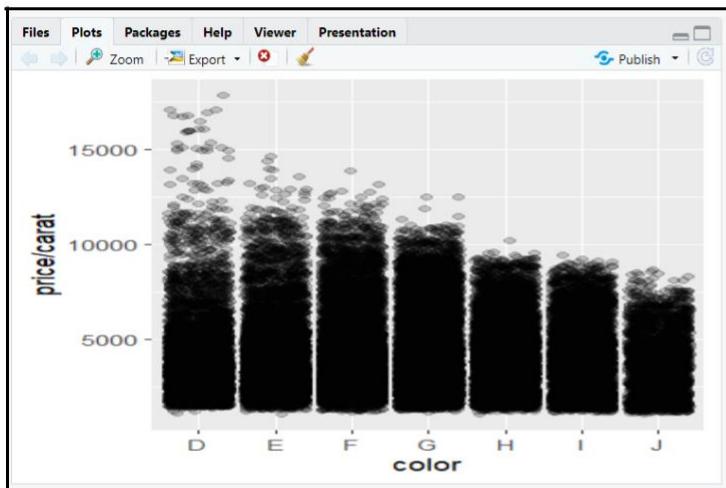
```



```

package 'ggplot2' was built under R version 4.2.2
> qplot(color, price / carat, data = diamonds, geom = "jitter",alpha = I(1 / 5))
Warning message:
`qplot()` was deprecated in ggplot2 3.4.0.
This warning is displayed once every 8 hours.
Call `lifecycle::last_lifecycle_warnings()` to
see where this warning was generated.
>
> qplot(color, price / carat, data = diamonds, geom = "jitter",alpha = I(1 / 50))
> qplot(color, price / carat, data = diamonds, geom = "jitter",alpha = I(1 / 200))
> |

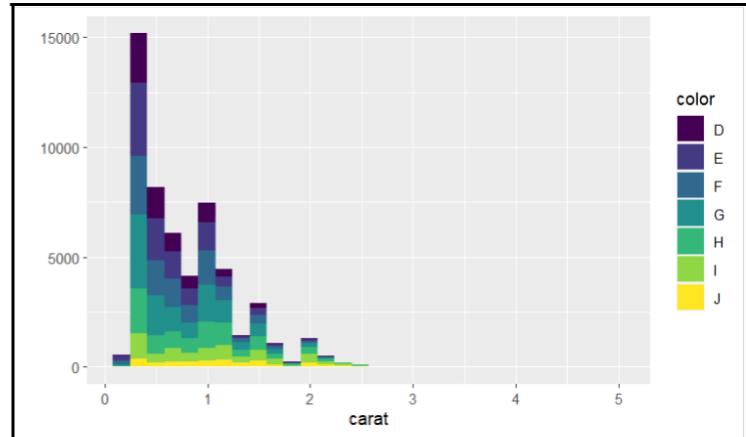
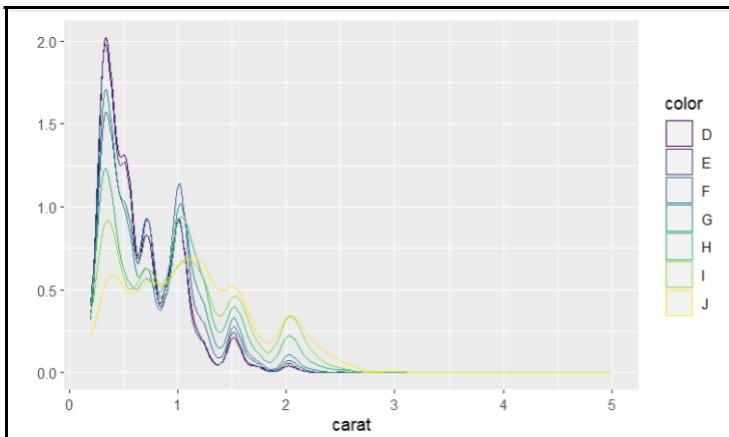
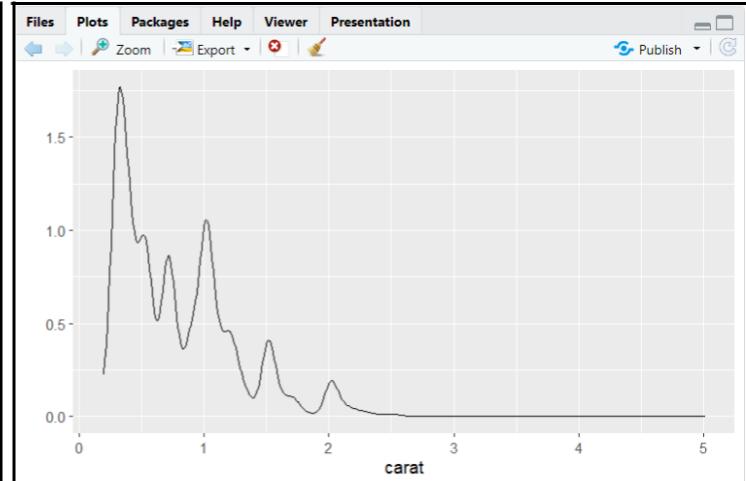
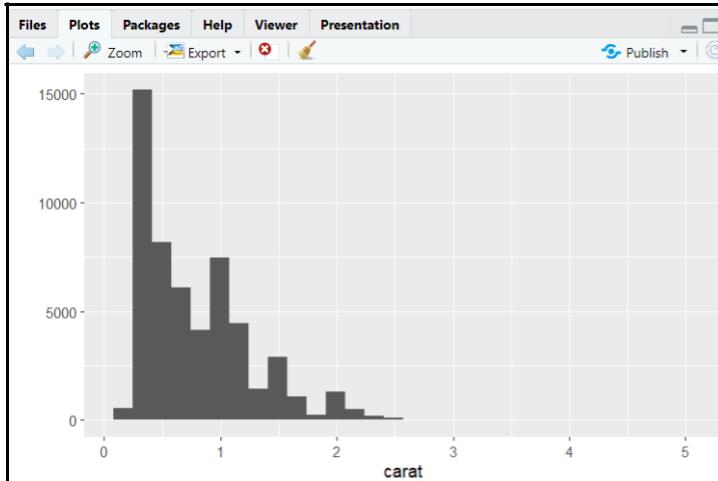
```



```

> qplot(carat, data = diamonds, geom = "histogram")
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> qplot(carat, data = diamonds, geom = "density")
> qplot(carat, data = diamonds, geom = "density", colour = color)
> qplot(carat, data = diamonds, geom = "histogram", fill = color)
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
>

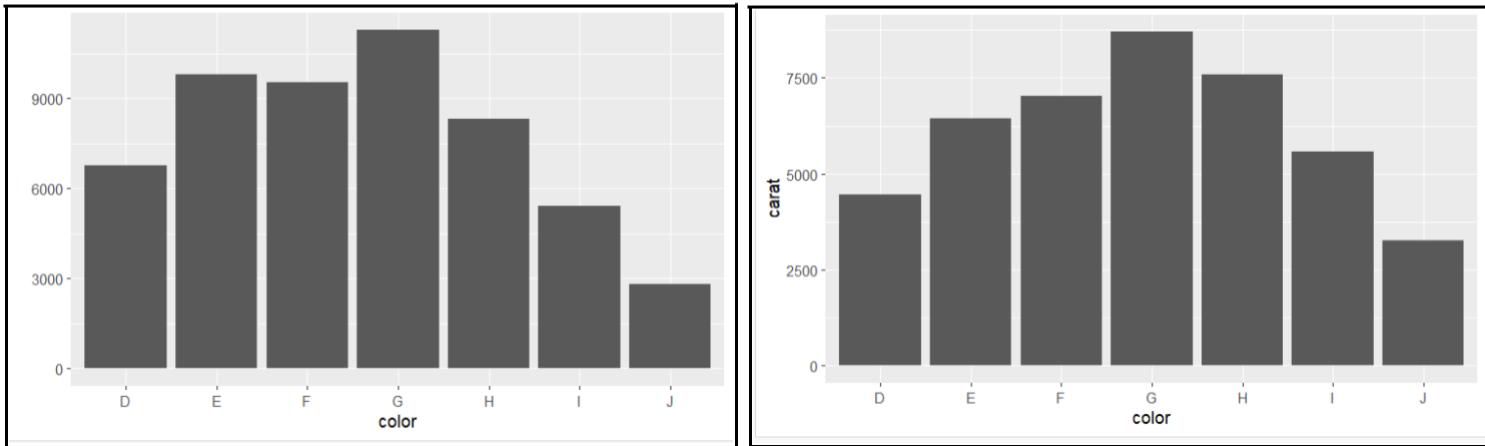
```



```

>
> qplot(color, data = diamonds, geom = "bar")
> qplot(color, data = diamonds, geom = "bar", weight = carat) + scale_y_continuous("carat")
>

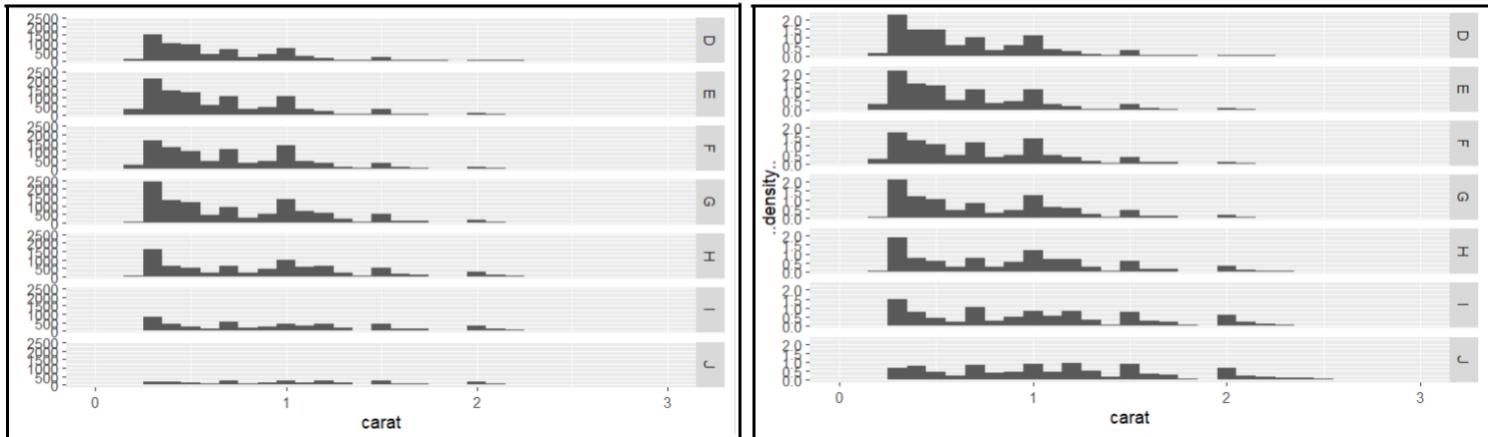
```



```

> qplot(carat, data = diamonds, facets = color ~ .,geom = "histogram", binwidth = 0.1, xlim = c(0, 3))
Warning messages:
1: Removed 32 rows containing non-finite values (`stat_bin()`).
2: Removed 14 rows containing missing values (`geom_bar()`).
> qplot(carat, ..density.., data = diamonds, facets = color ~ .,geom = "histogram", binwidth = 0.1, xlim = c(0, 3))
Warning messages:
1: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
  Please use `after_stat(density)` instead.
This warning is displayed once every 8 hours.
Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
2: Removed 32 rows containing non-finite values (`stat_bin()`).
3: Removed 14 rows containing missing values (`geom_bar()`).

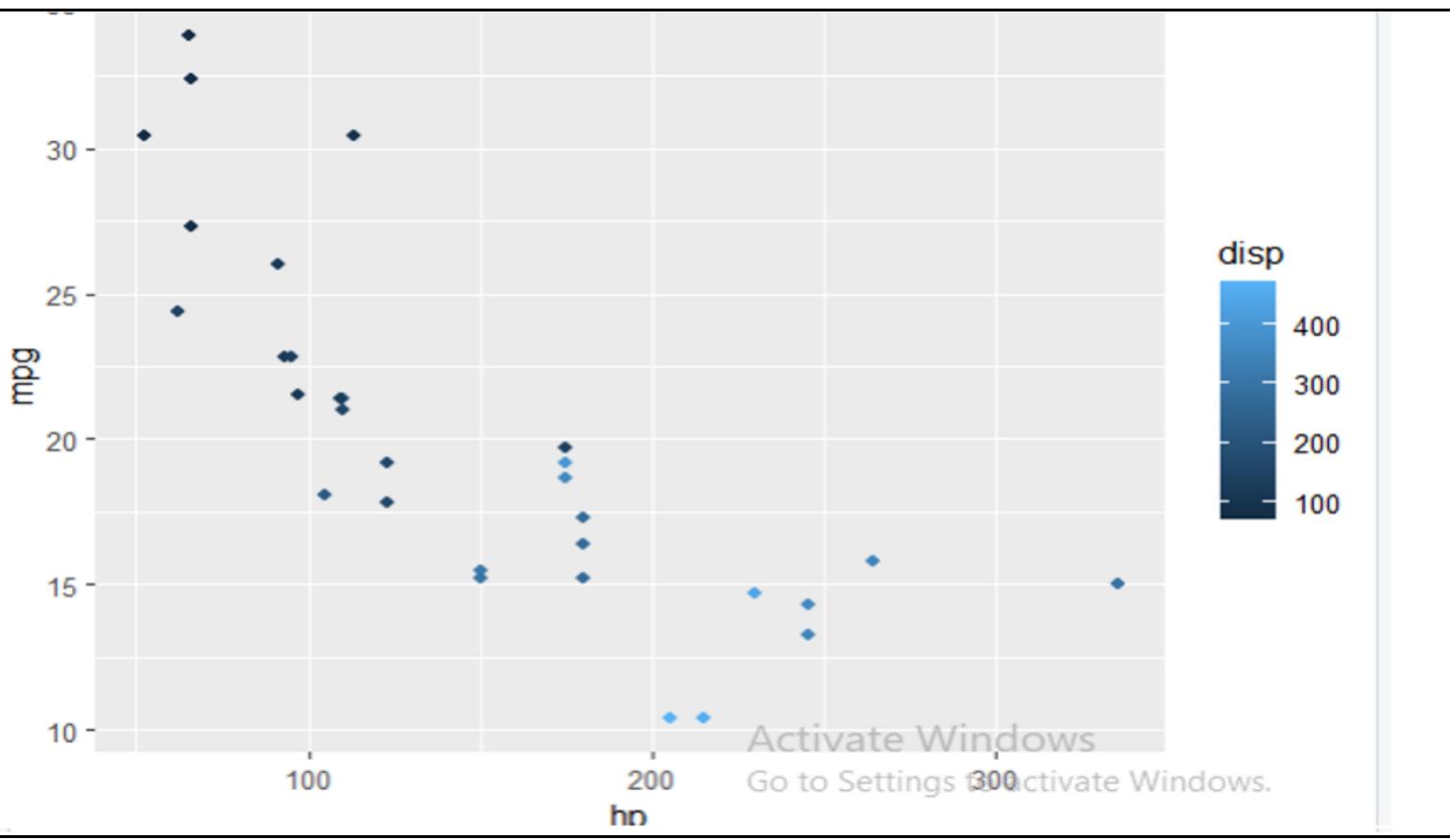
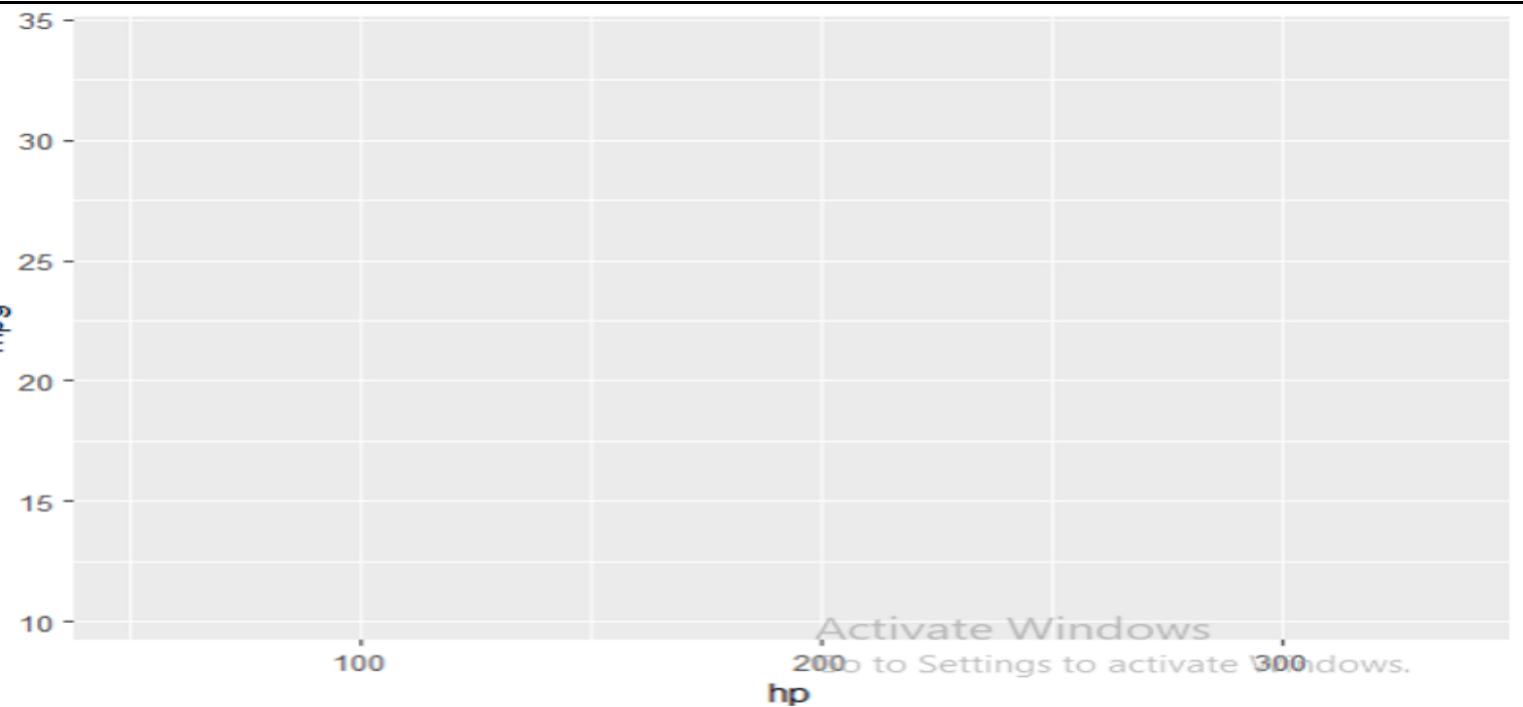
```

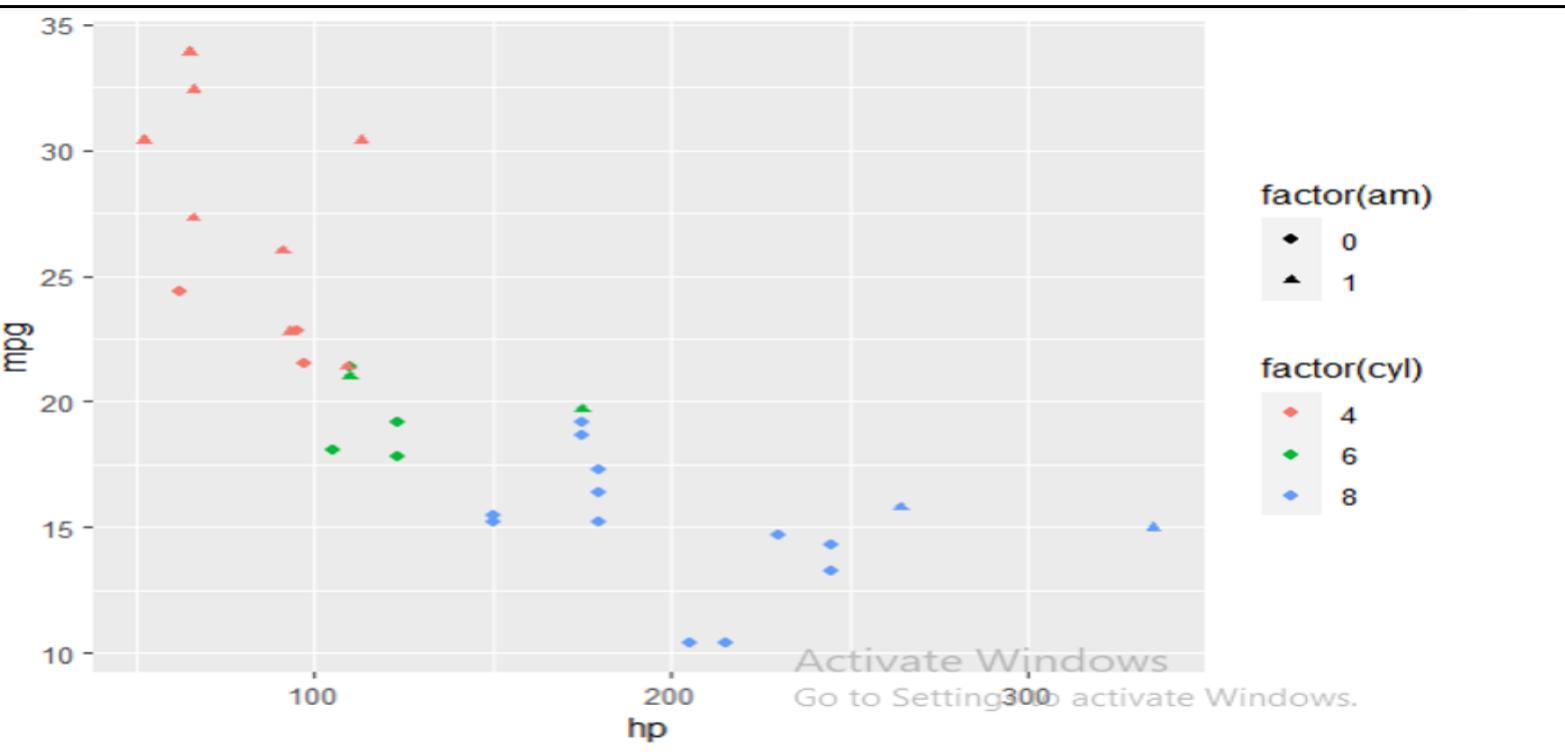
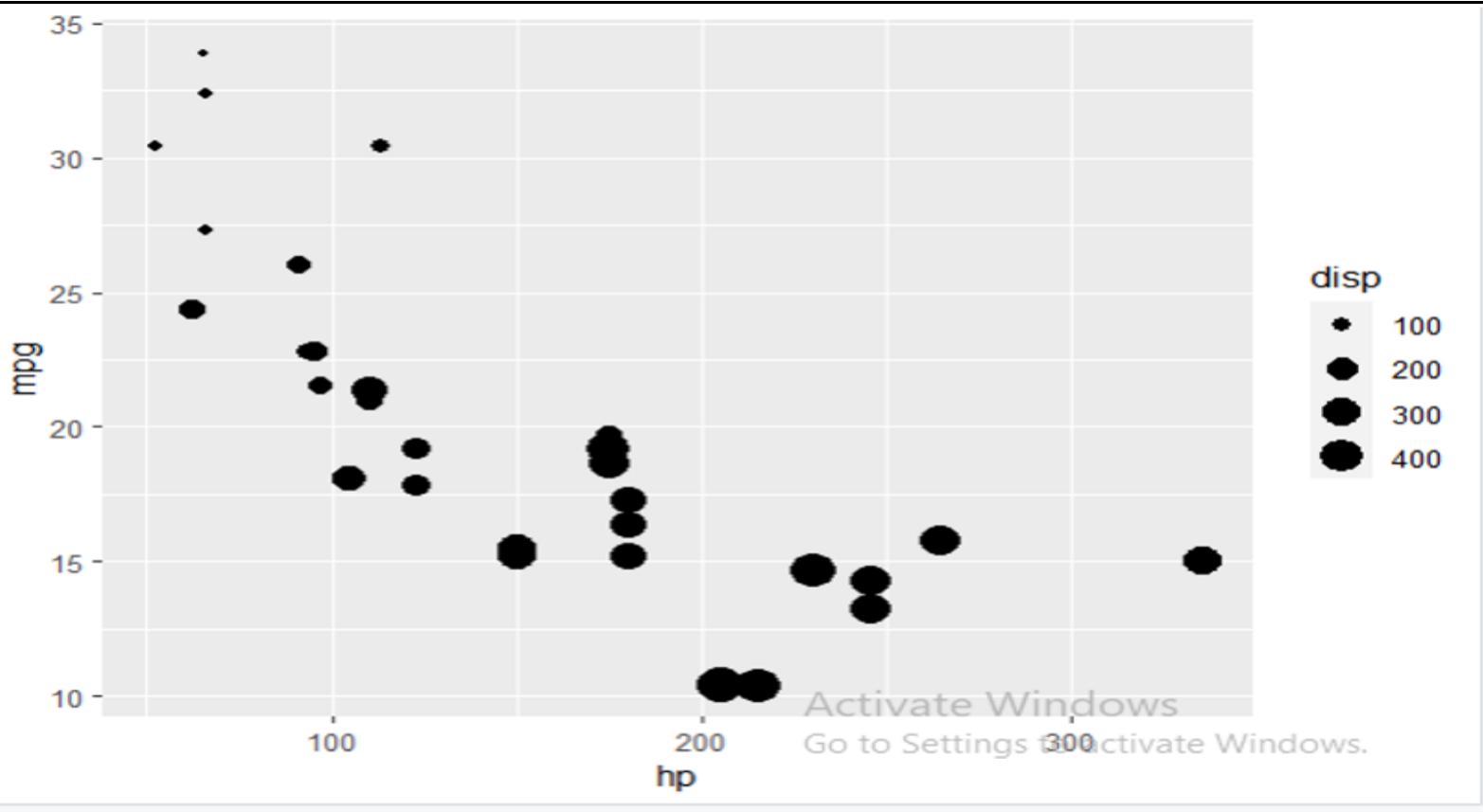


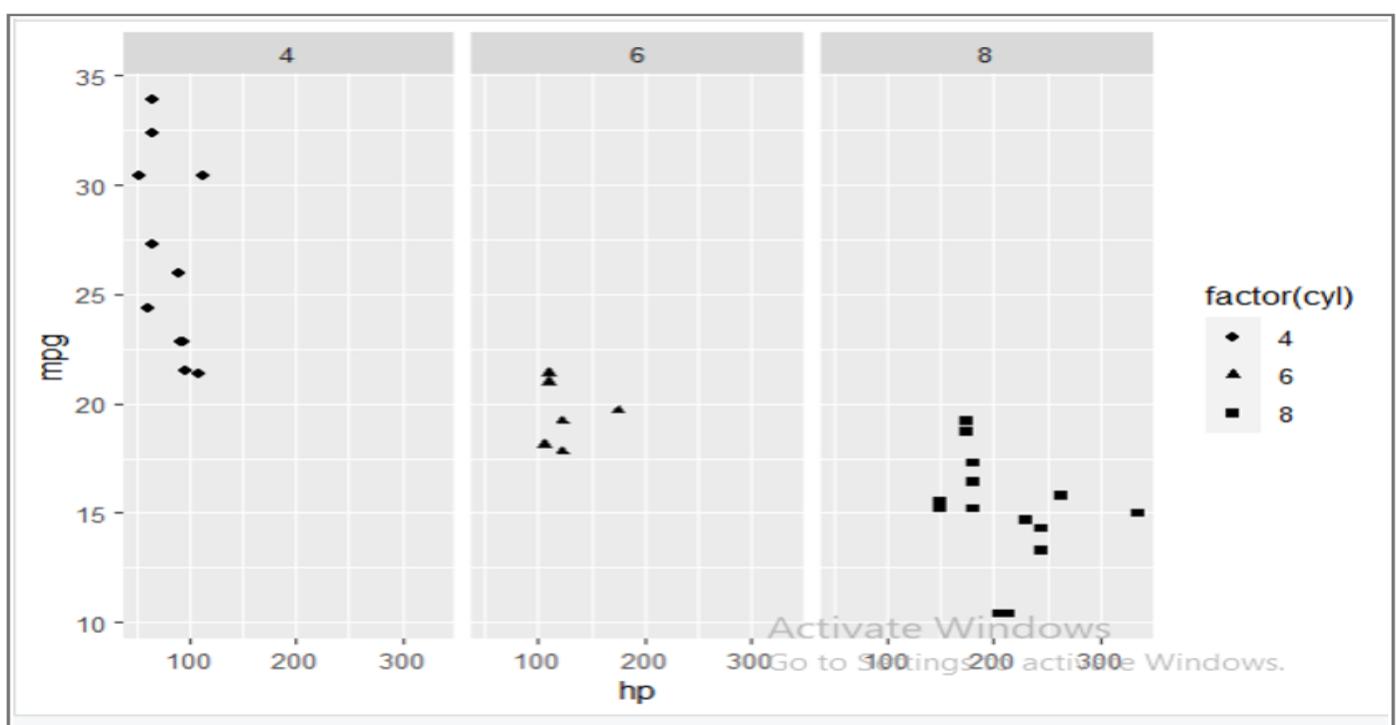
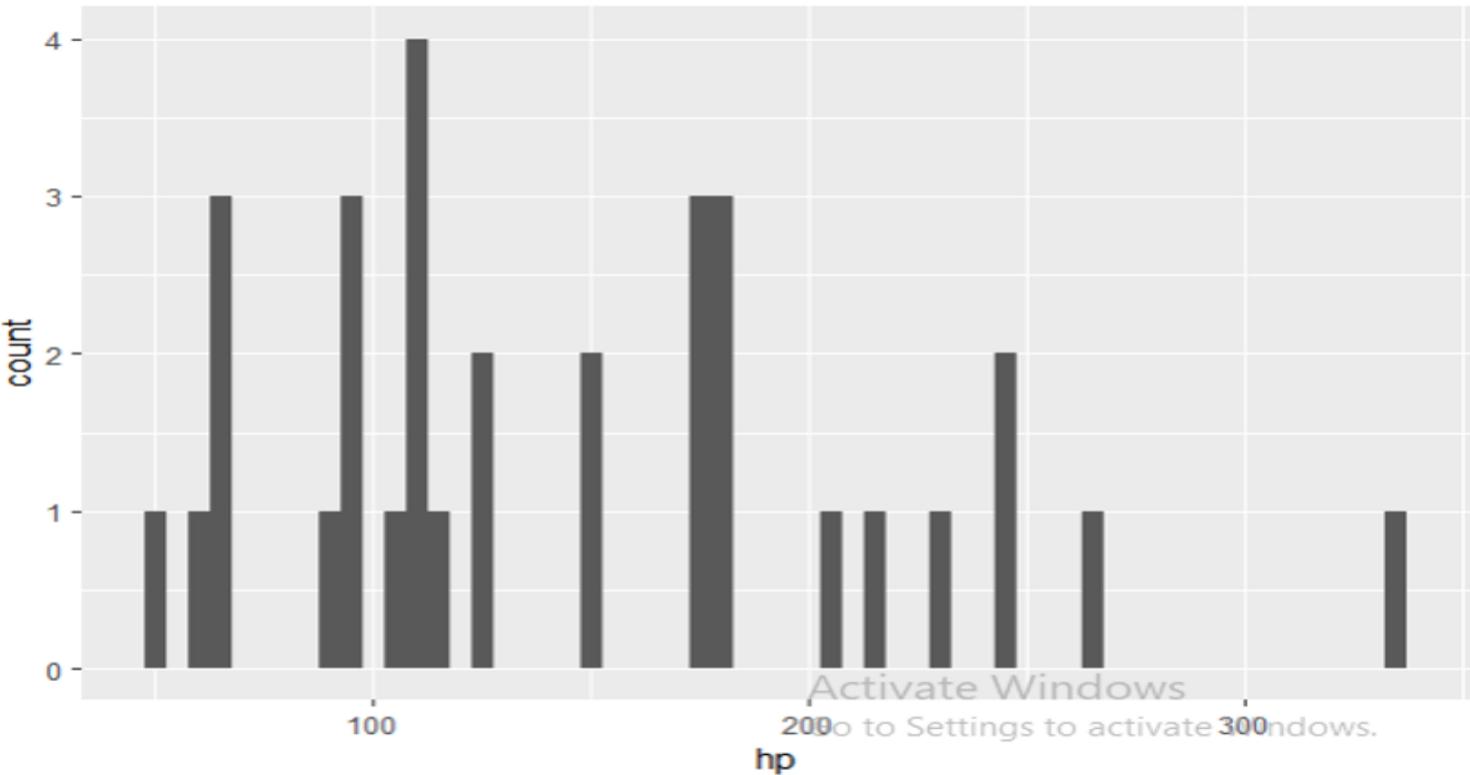
2) Use 'mtcars' dataset and ggplot () of ggplot2 Package for following operations

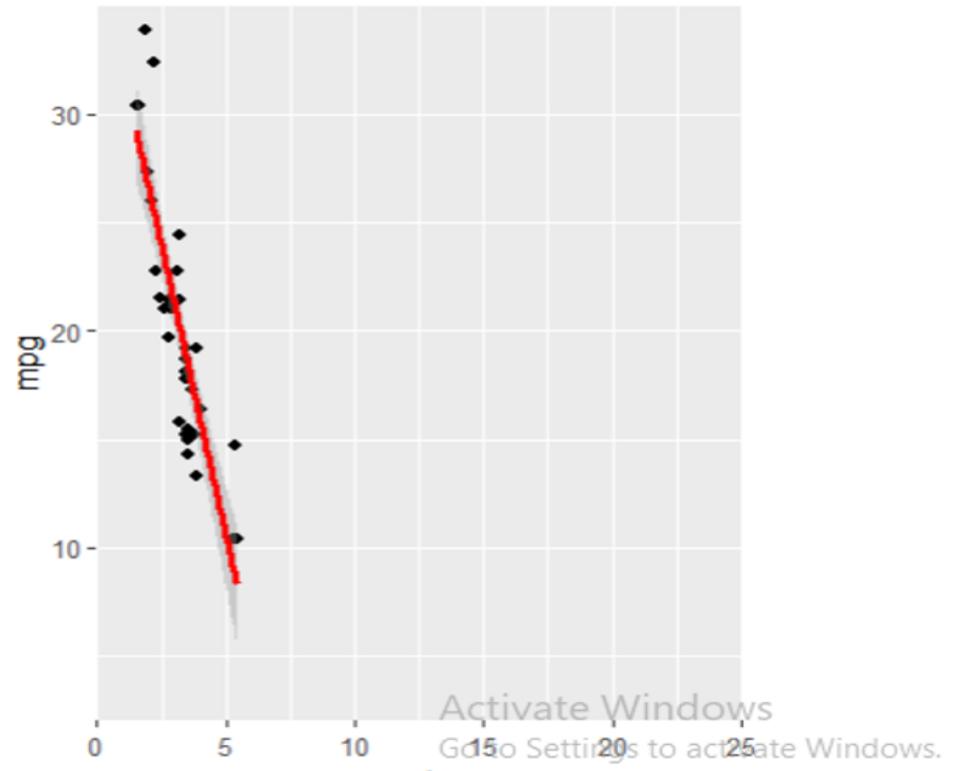
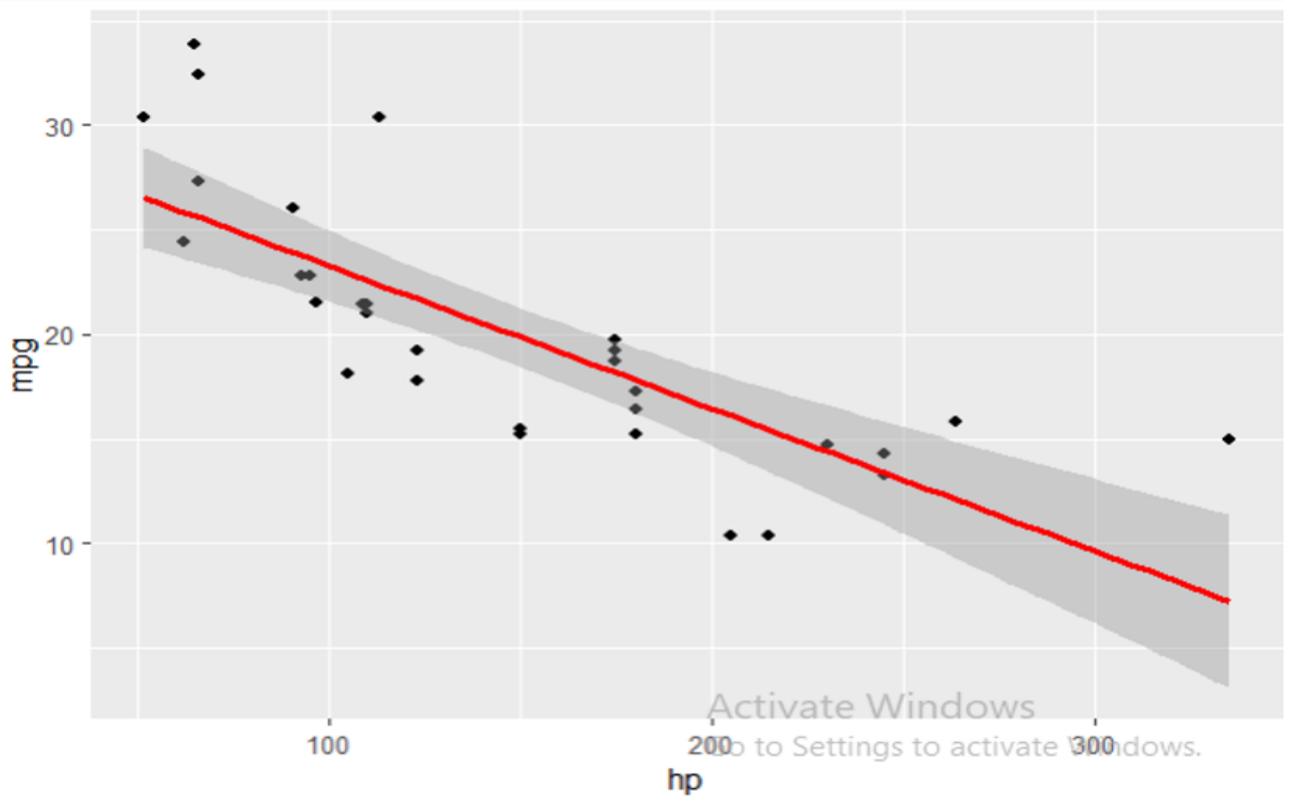
- (a) Data Layer
- (b) Aesthetic Layer
- (c) Geometric layer
- (d) Adding size
- (e) Adding color and shape
- (f) Histogram plot
- (g) Facet Layer
- (h) Statistics layer
- (i) Coordinates layer: Control plot dimensions

```
1 install.packages("dplyr")
2 library(ggplot2)
3 library(dplyr)
4 View(mtcars)
5
6
7 ggplot(data = mtcars)
8 ggplot(data = mtcars, aes(x = hp, y = mpg, col = disp))
9 ggplot(data = mtcars,
0     aes(x = hp, y = mpg, col = disp)) + geom_point()
1 ggplot(data = mtcars,
2     aes(x = hp, y = mpg, size = disp)) + geom_point()
3
4 ggplot(data = mtcars,
5     aes(x = hp, y = mpg, col = factor(cyl),
6         shape = factor(am))) +
7     geom_point()
8
9 ggplot(data = mtcars, aes(x = hp)) +
0     geom_histogram(binwidth = 5)
1
2 p <- ggplot(data = mtcars,
3     aes(x = hp, y = mpg,
4         shape = factor(cyl))) + geom_point()
5
6 p + facet_grid(am ~ .)
7
8 p + facet_grid(. ~ cyl)
9 ggplot(data = mtcars, aes(x = hp, y = mpg)) +
0     geom_point() +
stat_smooth(method = lm, col = "red")
```









3)) Display Import and load dataset from external sources like:

- (a) Excel File
- (b) Statistical Analysis System (SAS) File
- (c) Text File (base)

Import Statistical Data

File/URL:

Data Preview:

Import Options:

Name: Model: Format: Open Data Viewer

Code Preview:

```
library(haven)
dataset <- read_sav(NULL)
View(dataset)
```

[? Reading data using haven](#)

Import Excel Data

File/URL:

Data Preview:

Import Options:

Name: Max Rows: First Row as Names
Sheet: Skip: Open Data Viewer
Range: NA:

Code Preview:

```
library(readxl)
dataset <- read_excel(NULL)
View(dataset)
```

[? Reading Excel files using readxl](#)

4) Read file using below function commands:

- (a) **read.delim()** and **read.delim2()**
- (b) **read.table()**
- (c) **read.csv()**



A screenshot of the RStudio interface showing R code. The code reads three files: 'cancer_csv.csv' using both `read.delim()` and `read.delim2()`, 'adi.txt' using `read.table()`, and 'cancer_csv.csv' again using `read.csv()`. The code is as follows:

```
my_file <- read.delim(file = "C:/Users/Glau/Desktop/cancer_csv.csv")
my_file2 <- read.delim2(file = "C:/Users/Glau/Desktop/cancer_csv.csv")

df <- read.table("adi.txt", header = TRUE)

dfcsv<-read.csv("C:/Users/Glau/Desktop/cancer_csv.csv",header=TRUE)
dfcsv
```

5) Write a program in Java for counting the number of words.



A screenshot of a Java IDE showing a code editor and a terminal window. The code in the editor is a Java program that counts the number of words in a string. It initializes a counter for words, iterates through the string, and increments the counter whenever it finds a space character. The terminal window shows the output of the program, which is 'Number of words in a string : 4'. The code is as follows:

```
1 public class Main
2 {
3     public static void main(String[] args) {
4         String str = "This is Aditya Prakash";
5         int count = 1;
6         for (int i = 0; i < str.length() - 1; i++) {
7             if ((str.charAt(i) == ' ') && (str.charAt(i + 1) != ' '))
8             {
9                 count++;
10            }
11        }
12        System.out.println("Number of words in a string : " + count);
13    }
14 }
```

Number of words in a string : 4

ASSIGNMENT-07

Big data analytics

Name - Aniruddha Bajpai

Roll no - 201500092

Section - C(18)

Q1 Download VMware 17 Pro for Window.



The image shows a promotional page for VMware Workstation Player 17. At the top left is the product logo with the text "VMWARE WORKSTATION PLAYER™ 17". To the right is a laptop screen displaying the software's main menu. Below the logo is a large, light gray rectangular area containing the text "Try Workstation 17 Player for Windows" and a "DOWNLOAD NOW >" button.

VMWARE
WORKSTATION
PLAYER™ 17

Try Workstation 17 Player for Windows

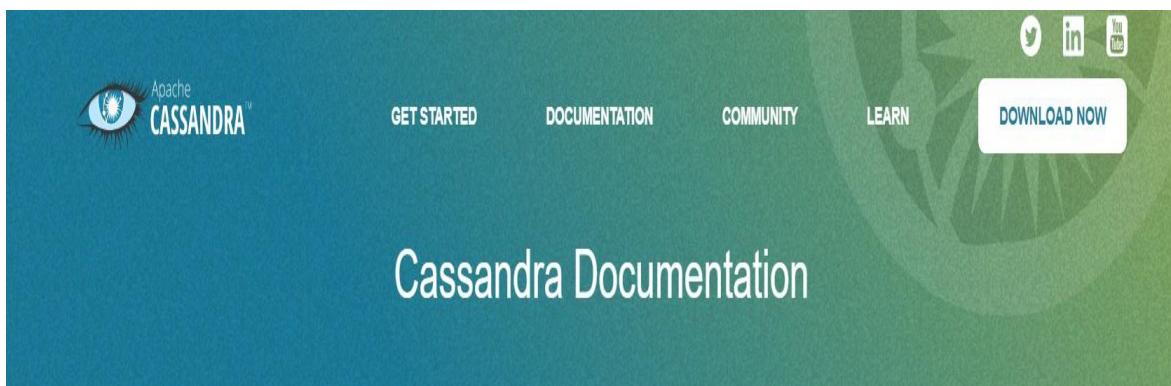
DOWNLOAD NOW >

Q 2 Download CloudEra Quick Start

The screenshot shows a section titled "CDH 5.13.x Download Information". Below the title, a paragraph states: "The section describes download information for CDH 5.13.x releases. For a different release, see the corresponding section." A bulleted list follows:

- CDH 5.13.3
- CDH 5.13.2
- CDH 5.13.1
- CDH 5.13.0

Q3 Download Cassandra cqlsh: Command Line Interface



Q4 Download MongoDB and CRUD Operations on Mongosh:

```
C:\Users\ASUS>mongosh
Current Mongosh Log ID: 638b8035a446f276f846741e
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1
Using MongoDB:      6.0.3
Using Mongosh:      1.6.0

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

-----
The server generated these startup warnings when booting
2022-11-29T14:42:36.412+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
-----
test> |
```

Lab Assignment – 04

Additional Problems on Basics of R Programming Language

Aniruddha bajpai

Section C , Roll-number 201500092 (18)

1) Write R Program to Find Sum, Mean and Product of Vector.

```
R 4.2.1 · ~/Assignment_3/ ↵
> # Question 1
> demo <- c(19,29,12,32,1,23,21)
> print(mean(demo))
[1] 19.57143
> print(sum(demo))
[1] 137
> print(prod(demo))
[1] 102195072
>
```

2) Write R Program to Take Input From User

```
R 4.2.1 · ~/Assignment_3/ ↵
> # Question 2
> name <- readline(prompt="Enter you name ")
Enter you name Aniruddha bajpai
> print(name)
[1] "Aniruddha bajpai"
>
```

3) Write R Program to sample() function on given example

```
R 4.2.1 · ~/Assignment_3/ ↵
> # Question 3
> sampledemo <- c(102,12,21,42,54,65,76,12,34,21,43654,01)
> print(sample(sampledemo,5))
[1] 21 43654 102 65 42
>
```

4) Write R Program to minimum, maximum and range of a given Vector

```
R 4.2.1 · ~/Assignment_3/ ↗
> # Question 4
> v <- c(10,2,1,21,3,54,21,09)
> print(min(v))
[1] 1
> print(max(v))
[1] 54
> print(range(v))
[1] 1 54
> |
```

5) Write R Program to Sort a given Vector

```
Console Terminal Background Jobs
R 4.2.1 · ~/Assignment_3/ ↗
> # Question 5
> v <- c(19,21,42,1,421,53,12,122)
> v <- sort(v)
> print(v)
[1] 1 12 19 21 42 53 122 421
> |
```

6) Write R Program to Find the Factorial of a Number

```
R 4.2.1 · ~/Assignment_3/ ↗
> # Question 6
> print(factorial(5))
[1] 120
> |
```

7) Write R Program to Print Table of given Number

```
> x=5
> a=1
> while(a<11){
+
+   print(x*a)
+   a=a+1
+ }
[1] 5
[1] 10
[1] 15
[1] 20
[1] 25
[1] 30
[1] 35
[1] 40
[1] 45
[1] 50
> |
```

8) Write R Program to Check Prime Number

```
> num=5
> flag = 0
> if(num > 1) {
+   flag = 1
+   for(i in 2:(num-1)) {
+     if ((num %% i) == 0) {
+       flag = 0
+       break
+     }
+   }
> if(num == 2)    flag = 1
> if(flag == 1) {
+   print(paste(num,"is a prime number"))
+ } else {
+   print(paste(num,"is not a prime number"))
+ }
[1] "5 is a prime number"
>
```

9) Write R Program to check Armstrong Number

```
> num =12
> sum = 0
> temp = num
> while(temp > 0) {
+   digit = temp %% 10
+   sum = sum + (digit ^ 3)
+   temp = floor(temp / 10)
+ }
> if(num == sum) {
+   print(paste(num, "is an Armstrong number"))
+ } else {
+   print(paste(num, "is not an Armstrong number"))
+ }
[1] "12 is not an Armstrong number"
>
```

10) Write R Program to Print the Fibonacci Sequence

```
> nterms=5
> n1 = 0
> n2 = 1
> count = 2
> if(nterms == 1) {
+   print(n1)
+ } else {
+   print(n1)
+   print(n2)
+   while(count < nterms) {
+     nth = n1 + n2
+     print(nth)
+     n1 = n2
+     n2 = nth
+     count = count + 1
+   }
+ }
[1] 0
[1] 1
[1] 1
[1] 2
[1] 3
>
```

11) Write R Program to find nth highest value in a given vector.

```
> x = c(10, 20, 30, 20, 20, 25, 9, 26)
> print("n = 1")
[1] "n = 1"
> n = 1
> print(sort(x, TRUE)[n])
[1] 30
> print("n = 2")
[1] "n = 2"
> n = 2
> print(sort(x, TRUE)[n])
[1] 26
.
```

- 12) Write R Program to create a vector using : operator and seq() function.

```
> x = 1:15
> y = seq(1, 3, by=0.3)
> print(y)
[1] 1.0 1.3 1.6 1.9 2.2 2.5 2.8
> z = seq(1, 5, length.out = 6)
> print(z)
[1] 1.0 1.8 2.6 3.4 4.2 5.0
> |
```

- 13) Write R Program to convert a given list to vector.

```
> n1 = list(1,2,3)
> c1 = list(4,5,6)
> v1 = unlist(n1)
> v2 = unlist(c1)
> print(v1)
[1] 1 2 3
> print(v2)
[1] 4 5 6
> v = v1 + v2
> print(v)
[1] 5 7 9
> |
```

- 14) Write R Program for Convert Decimal into Binary using Recursion

```
> binary <- function(num) {
+   if(num > 1) {
+     binary(as.integer(num/2))
+   }
+   cat(num %% 2)
+ }
> binary(5)
101
> | .
```

- 15) Write R Program to Find H.C.F. or G.C.D.

```

> hcf <- function(x, y) {
+   if(x > y) {
+     smaller = y
+   } else {
+     smaller = x
+   }
+   for(i in 1:smaller) {
+     if((x %% i == 0) && (y %% i == 0)) {
+       hcf = i
+     }
+   }
+   return(hcf)
+ }
> hcf(15,25)
[1] 5
>

```

- 16) Write R Program to create an array of two 3×3 matrices each with 3 rows and 3 columns from two given two vectors. Print the second row of the second matrix of the array and the element in the 3rd row and 3rd column of the 1st matrix.

```

> v1 = c(1,3,4,5)
> v2 = c(10,11,12,13,14,15)
> result = array(c(v1,v2),dim = c(3,3,2))
> print(result)
, , 1

[,1] [,2] [,3]
[1,]    1    5   12
[2,]    3   10   13
[3,]    4   11   14

, , 2

[,1] [,2] [,3]
[1,]   15    4   11
[2,]    1    5   12
[3,]    3   10   13

> print(result[2,,2])
[1] 1 5 12
> print(result[3,3,1])
[1] 14

```

- 17) Write R Program to create two 2×3 matrix and add, subtract, multiply and divide the matrixes.

```

> m1 = matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
> m2 = matrix(c(0, 1, 2, 3, 0, 2), nrow = 2)
> result = m1 + m2
> print(result)
[,1] [,2] [,3]
[1,]    1    5    5
[2,]    3    7    8
>
> result = m1 - m2
> print(result)
[,1] [,2] [,3]
[1,]    1    1    5
[2,]    1    1    4
>

```

- 18) Write R Program to access the element at 3rd column and 2nd row, only the 3rd row and only the 4th column

of a given matrix.

```
> row_names = c("row1", "row2", "row3", "row4")
> col_names = c("col1", "col2", "col3", "col4")
> M = matrix(c(1:16), nrow = 4, byrow = TRUE, dimnames = list(row_names, col_names))
>
> print(M[2,3])
[1] 7
> print("Access only the 3rd row:")
[1] "Access only the 3rd row:"
> print(M[3,])
col1 col2 col3 col4
  9   10   11   12
> print("Access only the 4th column:")
[1] "Access only the 4th column:"
> print(M[,4])
row1 row2 row3 row4
  4    8   12   16
```

- 19) Write R Program to find row and column index of maximum and minimum value in a given matrix.

```
> m = matrix(c(1:16), nrow = 4, byrow = TRUE)
> result = which(m == max(m), arr.ind=TRUE)
> print("max value")
[1] "max value"
> print(result)
  row col
[1,]  4   4
> result = which(m == min(m), arr.ind=TRUE)
> print("min value")
[1] "min value"
> print(result)
  row col
[1,]  1   1
>
```

- 20) Write R Program to concatenate two given matrices of same column but different rows.

```
> x = matrix(1:12, ncol=3)
> y = matrix(13:24, ncol=3)
> result = dim(rbind(x,y))
> print(result)
[1] 8 3
>
```

- 21) Write R Program to extract specific column from a data frame using column name.

```
exam_data = data.frame(
  name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
  score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
  attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),
  qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
)
print("Original dataframe:")
print(exam_data)
print("Extract Specific columns:")
result <- data.frame(exam_data$name, exam_data$score)
print(result)|
```

ASSIGNMENT-05

BIG DATA ANALYTICS

**NAME - ANIRUDDHA BAJPAI
SECTION-C(18)
ROLL NO.-201500092**

Q1) Install below packages using `install.package()` function :

- Coin
- XML
- Hmisc
- Graphics
- Ggplot

and list all available set of Packages under system library & user library.

Hint: For specific help in particular package use `library (help= "graphics")`.

Also, use

`edit()`, `search()` and `detach()` functions for different package utilities.

The screenshot shows the RStudio IDE. At the top, there's a menu bar with tabs like 'File', 'Edit', 'View', etc., and a toolbar with icons for file operations. Below that is a status bar showing 'Untitled1*' and 'R Script'. The main area has two panes: a 'Script' pane on the left containing R code, and a 'Console' pane on the right showing the output of the code execution.

In the Script pane, the code is:

```
1 # installing packages
2 install.packages('coin')
3 install.packages('XML')
4 install.packages('Hmisc')
5 install.packages('Graphics')
6 install.packages('ggplot2')
```

In the Console pane, the output is:

```
R 4.2.1 · ~/Assignment5/
see the 'ideas at'
https://cran.r-project.org/doc/manuals/r-patched/R-admin.html#Installing-packages
> install.packages('ggplot2')
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/bajpa/AppData/Local/R/win-library/4.2'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.2/ggplot2\_3.3.6.zip'
Content type 'application/zip' length 4122980 bytes (3.9 MB)
downloaded 3.9 MB

package 'ggplot2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\bajpa\AppData\Local\Temp\RtmpmzmZ111\downloaded_packages
> |
```

```

6  install.packages('ggplot2')
7
8  # available packages
9  installed.packages()
10

7:1 (Top Level) ▾

Console Terminal x Background Jobs x
R 4.2.1 · ~/Assignment5/ ↗

```

package ‘ggplot2’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
 C:\Users\bajpa\AppData\Local\Temp\Rtmpmmz111\downloaded_packages

```

> # available packages
> installed.packages()
  Package      LibPath
askpass      "askpass"    "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
backports     "backports"  "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
base64enc     "base64enc"  "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
checkmate     "checkmate"  "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
cli          "cli"        "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
coin          "coin"       "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
colorspace    "colorspace" "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
crandep       "crandep"   "C:/Users/bajpa/AppData/Local/R/win-library/4.2"
curl          "curl"      "C:/Users/bajpa/AppData/Local/R/win-library/4.2"

```

Q2) Display below datasets available under specific packages:

- Graphics
- MASS
- Datasets
- Cluster

Ans)

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

Data sets in package ‘cluster’:

agriculture	European Union Agricultural Workforces
animals	Attributes of Animals
chorSub	Subset of C-horizon of Kola Data
flower	Flower Characteristics
plantTraits	Plant Species Traits Data
pluton	Isotopic Composition Plutonium Batches
ruspini	Ruspini Data
votes.repub	Votes for Republican Candidate in Presidential Elections
xclara	Bivariate Data Set with 3 Clusters

```

> data()
> data(package = .packages(all.available = TRUE))
> data(package = .packages(all.available = TRUE))
>

```

Data sets in package 'MASS':

Aids2	Australian AIDS Survival Data
Animals	Brain and Body Weights for 28 Species
Boston	Housing Values in Suburbs of Boston
Cars93	Data from 93 Cars on Sale in the USA in 1993
Cushings	Diagnostic Tests on Patients with Cushing's Syndrome
DDT	DDT in Kale
GAGurine	Level of GAG in Urine of Children
Insurance	Numbers of Car Insurance claims
Melanoma	Survival from Malignant Melanoma
OME	Tests of Auditory Perception in Children with OME
Pima.te	Diabetes in Pima Indian Women
Pima.tr	Diabetes in Pima Indian Women
Pima.tr2	Diabetes in Pima Indian Women
Rabbit	Blood Pressure in Rabbits
Rubber	Accelerated Testing of Tyre Rubber
SP500	Returns of the Standard and Poors 500
Sitka	Growth Curves for Sitka Spruce Trees in 1988
Sitka89	Growth Curves for Sitka Spruce Trees in 1989
Skye	AFM Compositions of Aphyric Skye Lavas
Traffic	Effect of Swedish Speed Limits on Accidents
UScereal	Nutritional and Marketing Information on US Cereals
UScrime	The Effect of Punishment Regimes on Crime Rates
VA	Veteran's Administration Lung Cancer Trial
abbey	Determinations of Nickel Content
accdeaths	Accidental Deaths in the US 1973-1978
anorexia	Anorexia Data on Weight Change
bacteria	Presence of Bacteria after Drug Treatments

Q3) For given ‘mtcars’ dataset, use below functions– (a) dim () (b) sort () [for individual parameter with ‘\$’ Dollar Sign] (c) names () (d) rownames () (e) Print Variable Values using ‘\$’ Dollar Sign (f) summary () (g) max () and which.max () (h) min () and which.min () (i) mean () (j) median () (k) quantile ()

```
R 4.2.1 : ~/Assignment5/ 
> dim(mtcars)
[1] 32 11
> names(mtcars)
[1] "mpg" "cyl" "disp" "hp" "drat" "wt" "qsec" "vs" "am" "gear" "carb"
> rownames(mtcars)
[1] "Mazda RX4"          "Mazda RX4 Wag"      "Datsun 710"        "Hornet 4 Drive"    "Hornet Sportabout"
[6] "Valiant"            "Duster 360"       "Merc 240D"        "Merc 230"         "Merc 280"
[11] "Merc 280C"          "Merc 450SE"       "Merc 450SL"       "Merc 450SLC"     "Cadillac Fleetwood"
[16] "Lincoln Continental" "Chrysler Imperial" "Fiat 128"         "Honda Civic"     "Toyota Corolla"
[21] "Toyota Corona"     "Dodge Challenger" "AMC Javelin"     "Camaro Z28"      "Pontiac Firebird"
[26] "Fiat x1-9"          "Porsche 914-2"    "Lotus Europa"    "Ford Pantera L"   "Ferrari Dino"
[31] "Maserati Bora"      "Volvo 142E"
> summary(mtcars)
   mpg           cyl          disp          hp          drat         wt          qsec
Min. :10.40  Min. :4.000  Min. :71.1  Min. :52.0  Min. :2.760  Min. :1.513  Min. :14.50
1st Qu.:15.43 1st Qu.:4.000  1st Qu.:120.8 1st Qu.:96.5  1st Qu.:3.080  1st Qu.:2.581  1st Qu.:16.89
Median :19.20  Median :6.000  Median :196.3  Median :123.0  Median :3.695  Median :3.325  Median :17.71
Mean   :20.09  Mean   :6.188  Mean   :230.7  Mean   :146.7  Mean   :3.597  Mean   :3.217  Mean   :17.85
3rd Qu.:22.80 3rd Qu.:8.000  3rd Qu.:326.0  3rd Qu.:180.0  3rd Qu.:3.920  3rd Qu.:3.610  3rd Qu.:18.90
Max.   :33.90  Max.   :8.000  Max.   :472.0   Max.   :335.0  Max.   :4.930  Max.   :5.424  Max.   :22.90
   vs           am          gear          carb
Min. :0.00000  Min. :0.0000  Min. :3.000  Min. :1.000
1st Qu.:0.00000 1st Qu.:0.0000  1st Qu.:3.000  1st Qu.:2.000
Median :0.00000  Median :0.0000  Median :4.000  Median :2.000
Mean   :0.43750  Mean   :0.4062  Mean   :3.688  Mean   :2.812
3rd Qu.:1.00000 3rd Qu.:1.0000  3rd Qu.:4.000  3rd Qu.:4.000
Max.   :1.00000  Max.   :1.0000  Max.   :5.000  Max.   :8.000
> 
> max(mtcars)
[1] 472
> which.max(mtcars$mpg)
[1] 20
> min(mtcars)
[1] 0
> which.min(mtcars$mpg)
[1] 15
> mean(mtcars$mpg)
[1] 20.09062
> median(mtcars$mpg)
[1] 19.2

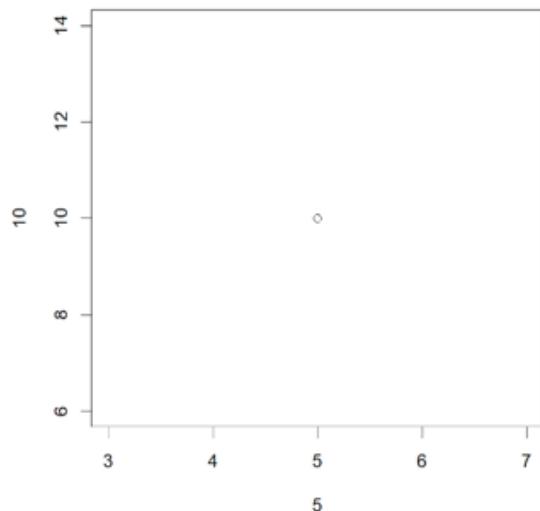
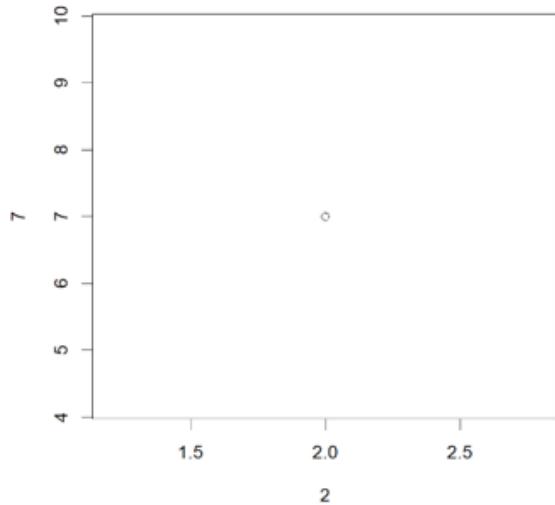
> quantile(mtcars$mpg)
 0%    25%    50%    75%   100%
10.400 15.425 19.200 22.800 33.900
>
```

```
> mpg <- mtcars$mpg
> sort(mpg)
[1] 10.4 10.4 13.3 14.3 14.7 15.0 15.2 15.5 15.8 16.4 17.3 17.8 18.1 18.7 19.2 19.2 19.7 21.0 21.0 21.4 21.4 21.5 22.8
[25] 22.8 24.4 26.0 27.3 30.4 30.4 32.4 33.9
> mtcars$vs
[1] 0 0 1 0 1 0 1 1 1 0 0 0 0 0 0 1 1 1 0 0 0 0 1 0 1 0 0 0 1
> mtcars$cyl
[1] 6 6 4 6 8 6 8 4 4 6 6 8 8 8 8 4 4 4 8 8 8 8 4 4 4 8 6 8 4
> mtcars$disp
[1] 160.0 160.0 108.0 258.0 360.0 225.0 360.0 146.7 140.8 167.6 167.6 275.8 275.8 275.8 472.0 460.0 440.0 78.7 75.7 71.1
[21] 120.1 318.0 304.0 350.0 400.0 79.0 120.3 95.1 351.0 145.0 301.0 121.0
> mtcars$hp
[1] 110 110 93 110 175 105 245 62 95 123 123 180 180 180 205 215 230 66 52 65 97 150 150 245 175 66 91 113 264 175
[31] 335 109
> mtcars$drat
[1] 3.90 3.90 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 3.92 3.07 3.07 3.07 2.93 3.00 3.23 4.08 4.93 4.22 3.70 2.76 3.15 3.73
[25] 3.08 4.08 4.43 3.77 4.22 3.62 3.54 4.11
> mtcars$wt
[1] 2.620 2.875 2.320 3.215 3.440 3.460 3.570 3.190 3.150 3.440 3.440 4.070 3.730 3.780 5.250 5.424 5.345 2.200 1.615 1.835
[21] 2.465 3.520 3.435 3.840 3.845 1.935 2.140 1.513 3.170 2.770 3.570 2.780
> mtcars$qsec
[1] 16.46 17.02 18.61 19.44 17.02 20.22 15.84 20.00 22.90 18.30 18.90 17.40 17.60 18.00 17.98 17.82 17.42 19.47 18.52 19.90
[21] 20.01 16.87 17.30 15.41 17.05 18.90 16.70 16.90 14.50 15.50 14.60 18.60
> mtcars$vs
[1] 0 0 1 1 0 1 0 1 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 0 1 0 0 0 1
> mtcars$am
[1] 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 1 1 1 1 1
> mtcars$gear
[1] 4 4 4 3 3 3 3 4 4 4 3 3 3 3 3 4 4 4 3 3 3 3 3 4 5 5 5 5 5 4
```

Q4) For given function plot (), use below operations: (a) Draw two points with position (2, 7) and (5, 10)

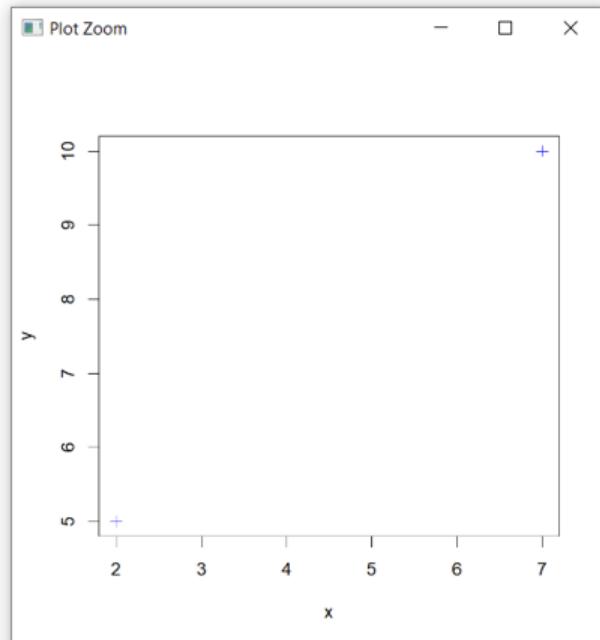
```
R 4.2.1 · ~/Assignment5/ ↗
```

```
> plot(2,7)
> plot(5,10)
```



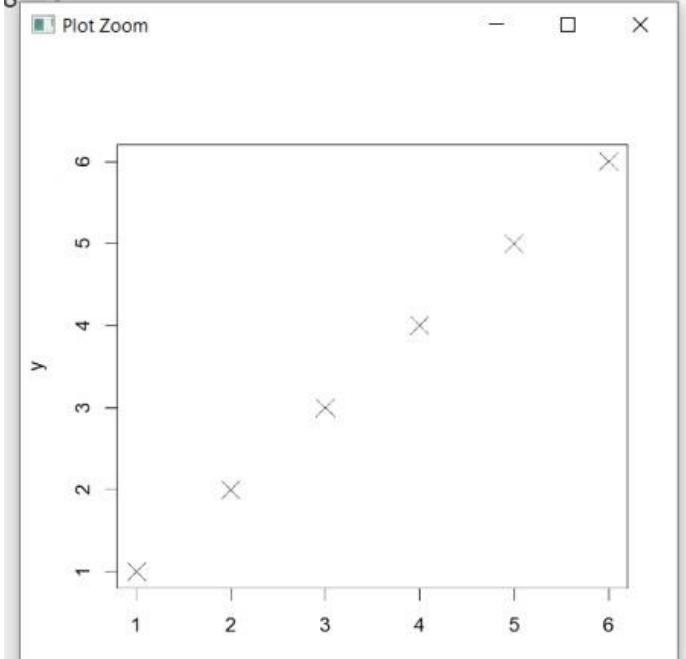
(c) Draw multiple points with variable positions

```
x <- c(2,7)
y <- c(5,10)
plot(x,y,cex=1,pch=3,xlab="x",ylab="y",col="blue")
```



(d) Draw multiple points with vector

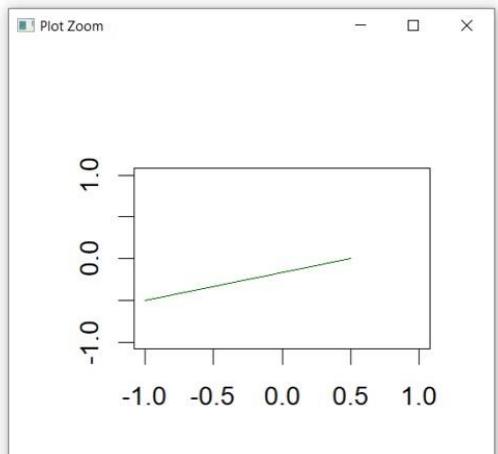
```
5 x <- c(1,2,3,4,5,6)
6 y <- c(1,2,3,4,5,6)
7 plot(x,y,cex=2,pch=4,xlab="x",ylab="y",col="BLACK")
8
```



(e) Draw sequences of points

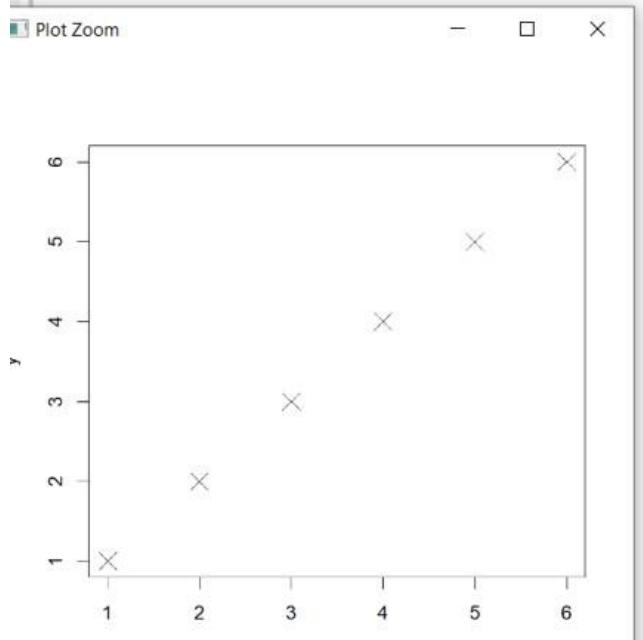
```
# Create empty example plot
plot(0, 0, col = "white", xlab = "", ylab = "")

# Draw one line
segments(x0 = - 1, y0 = - 0.5, x1 = 0.5, y1 = 0, col = "darkgreen")
```



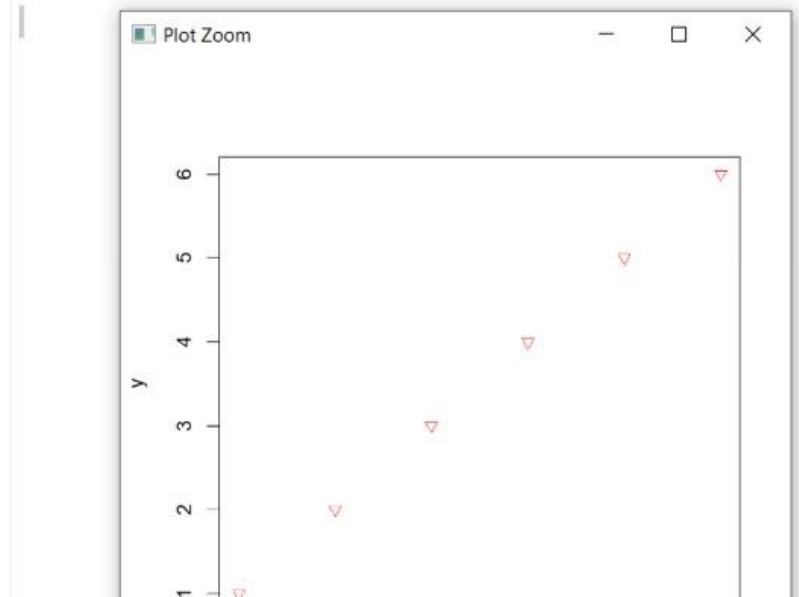
(f) Draw multiple points with parameters like main, xlab and ylab

```
x <- c(1,2,3,4,5,6)
y <- c(1,2,3,4,5,6)
plot(x,y,cex=2,pch=4,xlab="x",ylab="y",col="BLACK")
```



(g) Draw multiple points with point shape, colors and size

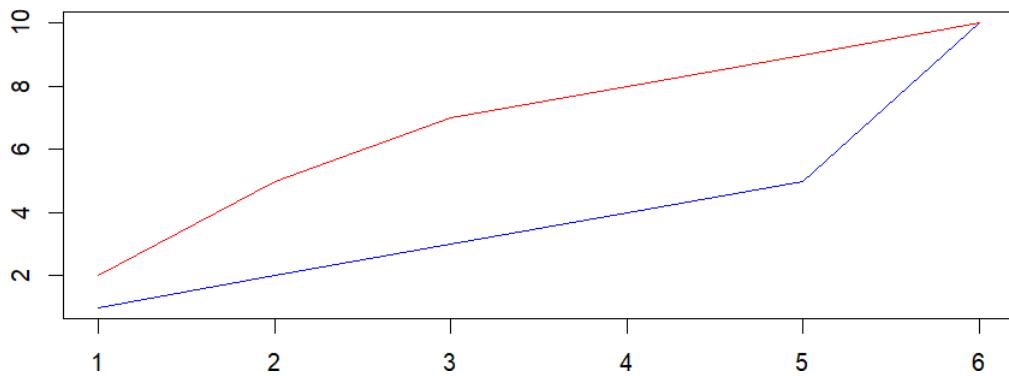
```
x <- c(1,3,1,7,5,1)
y <- c(1,2,3,4,7,6)
plot(x,y,cex=1,pch=6,xlab="x",ylab="y",col="red")
```



Q5) Compare two plots on following criteria: (a) Draw Multiple Lines with style, width and color
(f) Draw observations for bar texture using density, width & color parameters

```
line1 <- c(1,2,3,4,5,10)
line2 <- c(2,5,7,8,9,10)

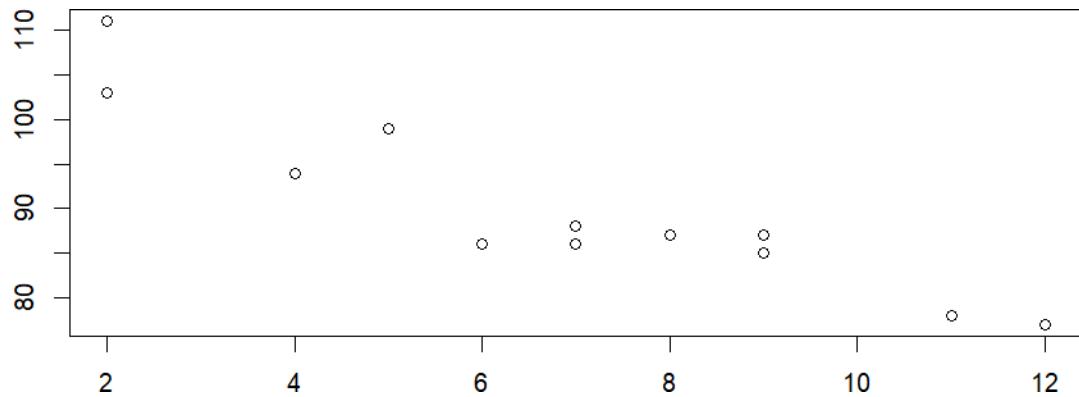
plot(line1, type = "l", col = "blue")
lines(line2, type="l", col = "red")|
```



(b) Draw the observations for two different dataset using points() function

```
x <- c(5,7,8,7,2,2,9,4,11,12,9,6)
y <- c(99,86,87,88,111,103,87,94,78,77,85,86)

plot(x, y)|
```

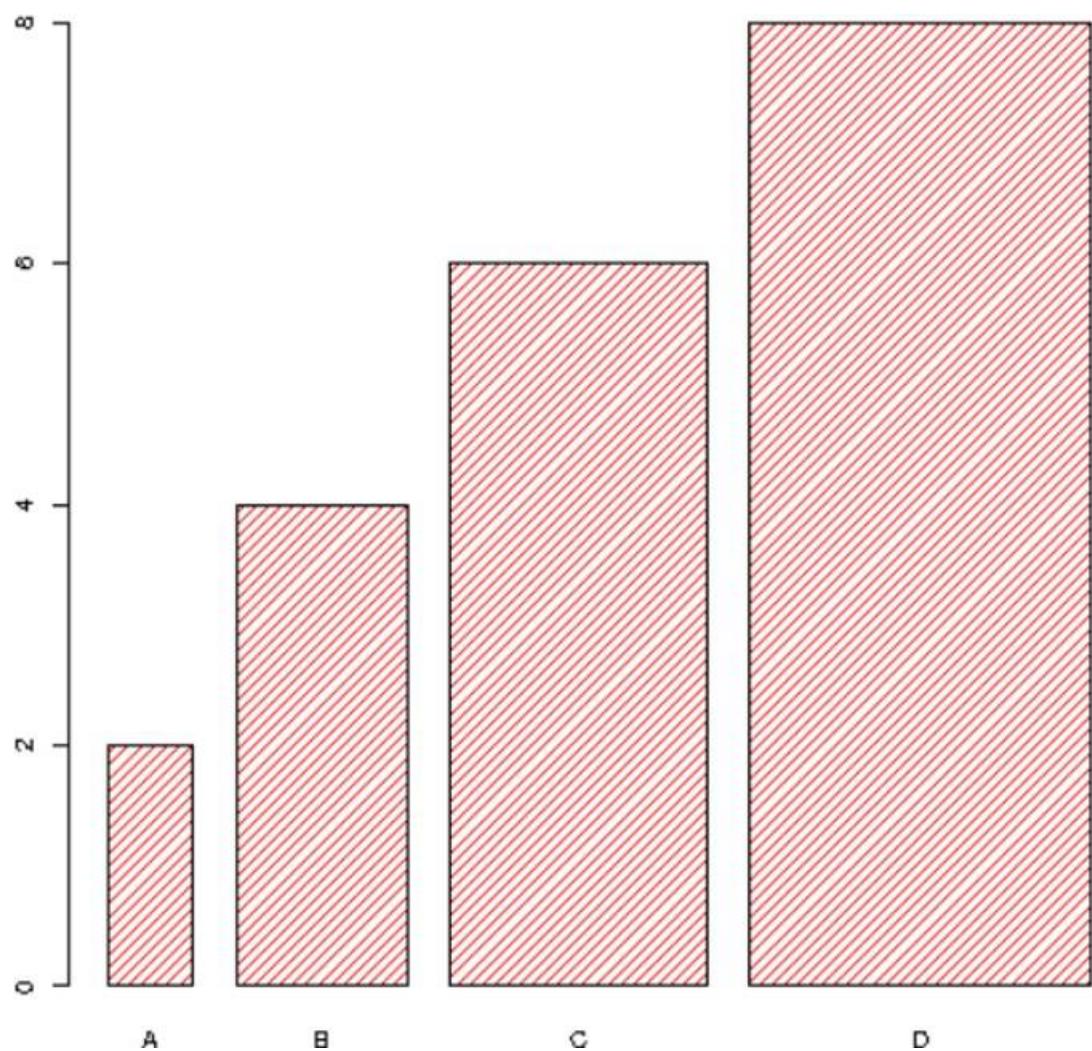


(c) Draw circular data visualization (pie chart) using pie() function

```
x<-c(20,20,15,45)
labels<-c("one", "two", "three", "four")
colors <- c("blue", "yellow", "green", "black")
pie(x, label=labels, init.angle = 45, main = "number", col=colors)
legend("bottomright", labels, fill = colors)
```

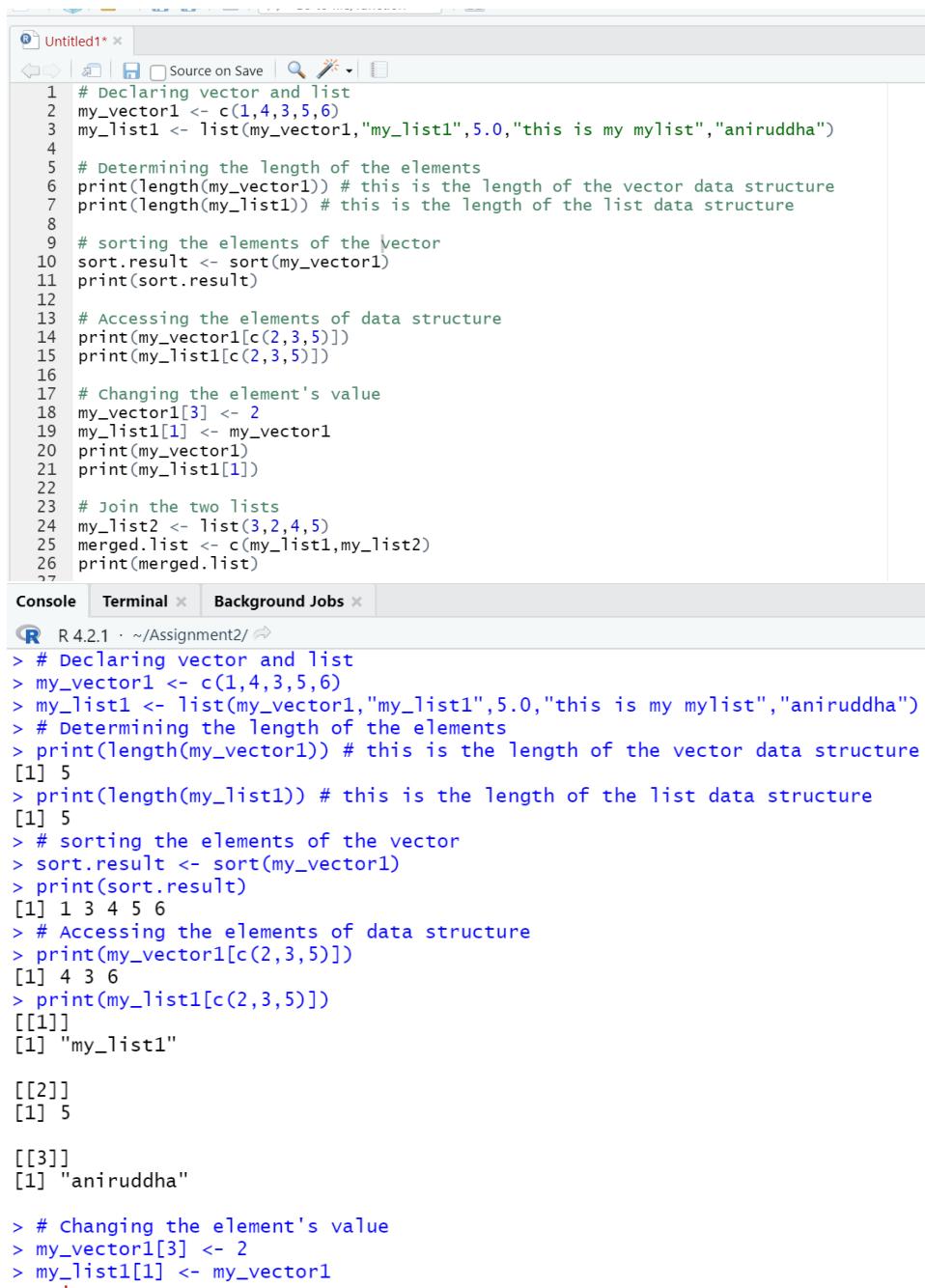


(e) Draw rectangular data visualization (bar chart) using barplot () function.



Lab Assignment – 02**Decision Making, Selection & Looping Structures using R Programming Language****Projected submitted by Aniruddha Bajpai(18-C1) (University roll-number : 201500092)****1) Write a R program to perform below operations on Data Structures like **vector** and **list** –**

- (a) Determining the length of elements
- (b) Sorting the elements
- (c) Accessing the elements
- (d) Changing the element's value
- (e) Join the two lists
- (f) Repeat elements of vector



The screenshot shows the RStudio interface with two panes. The top pane is the 'Code' editor containing R code for performing various operations on vectors and lists. The bottom pane is the 'Console' showing the execution of this code in R 4.2.1.

```

# Declaring vector and list
my_vector1 <- c(1,4,3,5,6)
my_list1 <- list(my_vector1,"my_list1",5.0,"this is my mylist","aniruddha")
# Determining the length of the elements
print(length(my_vector1)) # this is the length of the vector data structure
print(length(my_list1)) # this is the length of the list data structure
# sorting the elements of the vector
sort.result <- sort(my_vector1)
print(sort.result)
# Accessing the elements of data structure
print(my_vector1[c(2,3,5)])
print(my_list1[c(2,3,5)])
# Changing the element's value
my_vector1[3] <- 2
my_list1[1] <- my_vector1
print(my_vector1)
print(my_list1[1])
# Join the two lists
my_list2 <- list(3,2,4,5)
merged.list <- c(my_list1,my_list2)
print(merged.list)

```

R 4.2.1 · ~/Assignment2/ ↗

```

> # Declaring vector and list
> my_vector1 <- c(1,4,3,5,6)
> my_list1 <- list(my_vector1,"my_list1",5.0,"this is my mylist","aniruddha")
> # Determining the length of the elements
> print(length(my_vector1)) # this is the length of the vector data structure
[1] 5
> print(length(my_list1)) # this is the length of the list data structure
[1] 5
> # sorting the elements of the vector
> sort.result <- sort(my_vector1)
> print(sort.result)
[1] 1 3 4 5 6
> # Accessing the elements of data structure
> print(my_vector1[c(2,3,5)])
[1] 4 3 6
> print(my_list1[c(2,3,5)])
[[1]]
[1] "my_list1"

[[2]]
[1] 5

[[3]]
[1] "aniruddha"

> # changing the element's value
> my_vector1[3] <- 2
> my_list1[1] <- my_vector1
Warning message:

```

Console Terminal Background Jobs

R 4.2.1 · ~/Assignment2/ ↗

```
> # Changing the element's value
> my_vector1[3] <- 2
> my_list1[1] <- my_vector1
Warning message:
In my_list1[1] <- my_vector1 :
  number of items to replace is not a multiple of replacement length
> print(my_vector1)
[1] 1 4 2 5 6
> print(my_list1[1])
[[1]]
[1] 1

> # Join the two lists
> my_list2 <- list(3,2,4,5)
> merged.list <- c(my_list1,my_list2)
> print(merged.list)
[[1]]
[1] 1

[[2]]
[1] "my_list1"

[[3]]
[1] 5

[[4]]
[1] "this is my mylist"
```

[[5]]

Console Terminal Background Jobs

R 4.2.1 · ~/Assignment2/ ↗

```
> my_list2 <- list(3,2,4,5)
> merged.list <- c(my_list1,my_list2)
> print(merged.list)
[[1]]
[1] 1

[[2]]
[1] "my_list1"

[[3]]
[1] 5

[[4]]
[1] "this is my mylist"

[[5]]
[1] "aniruddha"

[[6]]
[1] 3

[[7]]
[1] 2

[[8]]
[1] 4

[[9]]
[1] 5
```

[[10]]

28 # Repeat the elements of two vectors
29 v2 <- rep(c(4,11),each=10,times=2)
30 print(v2)

[[11]]

[[12]]

[[13]]

[[14]]

[[15]]

[[16]]

[[17]]

[[18]]

[[19]]

[[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]]

[[45]]

[[46]]

[[47]]

[[48]]

[[49]]

[[50]]

[[51]]

[[52]]

[[53]]

[[54]]

[[55]]

[[56]]

[[57]]

[[58]]

[[59]]

[[60]]

[[61]]

[[62]]

[[63]]

[[64]]

[[65]]

[[66]]

[[67]]

[[68]]

[[69]]

[[70]]

[[71]]

[[72]]

[[73]]

[[74]]

[[75]]

[[76]]

[[77]]

[[78]]

[[79]]

[[80]]

[[81]]

[[82]]

[[83]]

[[84]]

[[85]]

[[86]]

[[87]]

[[88]]

[[89]]

[[90]]

[[91]]

[[92]]

[[93]]

[[94]]

[[95]]

[[96]]

[[97]]

[[98]]

[[99]]

[[100]]

[[101]]

[[102]]

[[103]]

[[104]]

[[105]]

[[106]]

[[107]]

[[108]]

[[109]]

[[110]]

[[111]]

[[112]]

[[113]]

[[114]]

[[115]]

[[116]]

[[117]]

[[118]]

[[119]]

[[120]]

[[121]]

[[122]]

[[123]]

[[124]]

[[125]]

[[126]]

[[127]]

[[128]]

[[129]]

[[130]]

[[131]]

[[132]]

[[133]]

[[134]]

[[135]]

[[136]]

[[137]]

[[138]]

[[139]]

[[140]]

[[141]]

[[142]]

[[143]]

[[144]]

[[145]]

[[146]]

[[147]]

[[148]]

[[149]]

[[150]]

[[151]]

[[152]]

[[153]]

[[154]]

[[155]]

[[156]]

[[157]]

[[158]]

[[159]]

[[160]]

[[161]]

[[162]]

[[163]]

[[164]]

[[165]]

[[166]]

[[167]]

[[168]]

[[169]]

[[170]]

[[171]]

[[172]]

[[173]]

[[174]]

[[175]]

[[176]]

[[177]]

[[178]]

[[179]]

[[180]]

[[181]]

[[182]]

[[183]]

[[184]]

[[185]]

[[186]]

[[187]]

[[188]]

[[189]]

[[190]]

[[191]]

[[192]]

[[193]]

[[194]]

[[195]]

[[196]]

[[197]]

[[198]]

[[199]]

[[200]]

[[201]]

[[202]]

[[203]]

[[204]]

[[205]]

[[206]]

[[207]]

[[208]]

[[209]]

[[210]]

[[211]]

[[212]]

[[213]]

[[214]]

[[215]]

[[216]]

[[217]]

[[218]]

[[219]]

[[220]]

[[221]]

[[222]]

[[223]]

[[224]]

[[225]]

[[226]]

[[227]]

[[228]]

[[229]]

[[230]]

[[231]]

[[232]]

[[233]]

[[234]]

[[235]]

[[236]]

[[237]]

[[238]]

[[239]]

[[240]]

[[241]]

[[242]]

[[243]]

[[244]]

[[245]]

[[246]]

[[247]]

[[248]]

[[249]]

[[250]]

[[251]]

[[252]]

[[253]]

[[254]]

[[255]]

[[256]]

[[257]]

[[258]]

[[259]]

[[260]]

[[261]]

[[262]]

[[263]]

[[264]]

[[265]]

[[266]]

[[267]]

[[268]]

[[269]]

[[270]]

[[271]]

[[272]]

[[273]]

[[274]]

[[275]]

[[276]]

[[277]]

[[278]]

[[279]]

[[280]]

[[281]]

[[282]]

[[283]]

[[284]]

[[285]]

[[286]]

[[287]]

[[288]]

[[289]]

[[290]]

[[291]]

[[292]]

[[293]]

[[294]]

[[295]]

[[296]]

[[297]]

[[298]]

[[299]]

[[300]]

[[301]]

[[302]]

[[303]]

[[304]]

[[305]]

[[306]]

[[307]]

[[308]]

[[309]]

[[310]]

[[311]]

[[312]]

[[313]]

[[314]]

[[315]]

[[316]]

[[317]]

[[318]]

[[319]]

[[320]]

[[321]]

[[322]]

[[323]]

[[324]]

[[325]]

[[326]]

[[327]]

[[328]]

[[329]]

[[330]]

[[331]]

[[332]]

[[333]]

[[334]]

[[335]]

[[336]]

[[337]]

[[338]]

[[339]]

[[340]]

[[341]]

[[342]]

[[343]]

[[344]]

[[345]]

[[346]]

[[347]]

[[348]]

[[349]]

[[350]]

[[351]]

[[352]]

[[353]]

[[354]]

[[355]]

[[356]]

[[357]]

[[358]]

[[359]]

[[360]]

[[361]]

[[362]]

[[363]]

[[364]]

[[365]]

[[366]]

[[367]]

[[368]]

[[369]]

[[370]]

[[371]]

[[372]]

[[373]]

[[374]]

[[375]]

[[376]]

[[377]]

[[378]]

[[379]]

[[380]]

[[381]]

[[382]]

[[383]]

[[384]]

[[385]]

[[386]]

[[387]]

[[388]]

[[389]]

[[390]]

[[391]]

[[392]]

[[393]]

[[394]]

[[395]]

[[396]]

[[397]]

[[398]]

[[399]]

[[400]]

[[401]]

[[402]]

[[403]]

[[404]]

[[405]]

[[406]]

[[407]]

[[408]]

[[409]]

[[410]]

[[411]]

[[412]]

[[413]]

[[414]]

[[415]]

[[416]]

[[417]]

[[418]]

[[419]]

[[420]]

[[421]]

[[422]]

[[423]]

[[424]]

[[425]]

[[426]]

[[427]]

[[428]]

[[429]]

[[430]]

[[431]]

[[432]]

[[433]]

[[434]]

[[435]]

[[436]]

[[437]]

[[438]]

[[439]]

[[440]]

[[441]]

[[442]]

[[443]]

[[444]]

[[445]]

[[446]]

[[447]]

[[448]]

[[449]]

[[450]]

[[451]]

[[452]]

[[453]]

[[454]]

[[455]]

[[456]]

[[457]]

[[458]]

[[459]]

[[460]]

[[461]]

[[462]]

[[463]]

[[464]]

[[465]]

[[466]]

[[467]]

[[468]]

[[469]]

[[470]]

[[471]]

[[472]]

[[473]]

[[474]]

[[475]]

[[476]]

[[477]]

[[478]]

[[479]]

[[480]]

[[481]]

[[482]]

[[483]]

[[484]]

[[485]]

[[486]]

[[487]]

[[488]]

[[489]]

[[490]]

[[491]]

[[492]]

[[493]]

[[494]]

[[495]]

[[496]]

[[497]]

[[498]]

[[499]]

[[500]]

[[501]]

[[502]]

[[503]]

[[504]]

[[505]]

[[506]]

[[507]]

[[508]]

[[509]]

[[510]]

[[511]]

[[512]]

[[513]]

[[514]]

[[515]]

[[516]]

[[517]]

[[518]]

[[519]]

[[520]]

[[521]]

[[522]]

[[523]]

[[524]]

[[525]]

[[526]]

[[527]]

[[528]]

[[529]]

[[530]]

[[531]]

[[532]]

[[533]]

[[534]]

[[535]]

[[536]]

[[537]]

[[538]]

[[539]]

[[540]]

[[541]]

[[542]]

[[543]]

[[544]]

[[545]]

[[546]]

[[547]]

[[548]]

[[549]]

[[550]]

[[551]]

[[552]]

[[553]]

[[554]]

[[555]]

[[556]]

[[557]]

[[558]]

[[559]]

[[560]]

[[561]]

[[562]]

[[563]]

[[564]]

[[565]]

[[566]]

[[567]]

[[568]]

[[569]]

[[570]]

[[571]]

[[572]]

[[573]]

[[574]]

[[575]]

[[576]]

[[577]]

[[578]]

[[579]]

[[580]]

[[581]]

[[582]]

[[583]]

[[584]]

[[585]]

[[586]]

[[587]]

[[588]]

[[589]]

[[590]]

[[591]]

[[592]]

[[593]]

[[594]]

[[595]]

[[596]]

[[597]]

[[598]]

[[599]]

[[600]]

[[601]]

[[602]]

[[603]]

[[604]]

[[605]]

[[606]]

[[607]]

[[608]]

[[609]]

[[610]]

[[611]]

[[612]]

[[613]]

[[614]]

[[615]]

[[616]]

[[617]]

[[618]]

[[619]]

[[620]]

[[621]]

[[622]]

[[623]]

[[624]]

[[625]]

[[626]]

[[627]]

[[628]]

[[629]]

[[630]]

[[631]]

[[632]]

[[633]]

[[634]]

[[635]]

[[636]]

[[637]]

[[638]]

[[639]]

[[640]]

[[641]]

[[642]]

[[643]]

[[644]]

[[645]]

[[646]]

[[647]]

[[648]]

[[649]]

[[650]]

[[651]]

[[652]]

[[653]]

[[654]]

[[655]]

[[656]]

[[657]]

[[658]]

[[659]]

[[660]]

[[661]]

[[662]]

[[663]]

[[664]]

[[665]]

[[666]]

[[667]]

[[668]]

[[669]]

[[670]]

[[671]]

[[672]]

[[673]]

[[674]]

[[675]]

[[676]]

[[677]]

[[678]]

[[679]]

[[680]]

[[681]]

[[682]]

[[683]]

[[684]]

[[685]]

[[686]]

[[687]]

[[688]]

[[689]]

[[690]]

[[691]]

[[692]]

[[693]]

[[694]]

[[695]]

[[696]]

[[697]]

[[698]]

[[699]]

[[700]]

[[701]]

[[702]]

[[703]]

[[704]]

[[705]]

[[706]]

[[707]]

[[708]]

[[709]]

[[710]]

[[711]]

[[712]]

[[713]]

[[714]]

[[715]]

[[716]]

[[717]]

[[718]]

[[719]]

[[720]]

[[721]]

[[722]]

[[723]]

[[724]]

[[725]]

[[726]]

[[727]]

[[728]]

[[729]]

[[730]]

[[731]]

[[732]]

[[733]]

[[734]]

[[735]]

[[736]]

[[737]]

[[738]]

[[739]]

[[740]]

[[741]]

[[742]]

[[743]]

[[744]]

[[745]]

[[746]]

[[747]]

[[748]]

[[749]]

[[750]]

[[751]]

[[752]]

[[753]]

[[754]]

[[755]]

[[756]]

[[757]]

[[758]]

[[759]]

[[760]]

[[761]]

[[762]]

[[763]]

[[764]]

[[765]]

[[766]]

[[767]]

[[768]]

[[769]]

[[770]]

[[771]]

[[772]]

[[773]]

[[774]]

[[775]]

[[776]]

[[777]]

[[778]]

[[779]]

[[780]]

[[781]]

[[782]]

[[783]]

[[784]]

[[785]]

[[786]]

[[787]]

[[788]]

[[789]]

[[790]]

[[791]]

[[792]]

[[793]]

[[794]]

[[795]]

[[796]]

[[797]]

[[798]]

[[799]]

[[800]]

[[801]]

[[802]]

[[803]]

[[804]]

[[805]]

[[806]]

[[807]]

[[808]]

[[809]]

[[810]]

[[811]]

[[812]]

[[813]]

[[814]]

[[815]]

[[816]]

[[817]]

[[818]]

[[819]]

[[820]]

[[821]]

[[822]]

[[823]]

[[824]]

[[825]]

[[826]]

[[827]]

[[828]]

[[829]]

[[830]]

[[831]]

[[832]]

[[833]]

[[834]]

[[835]]

[[836]]

[[837]]

[[838]]

[[839]]

[[840]]

[[841]]

[[842]]

[[843]]

[[844]]

[[845]]

[[846]]

[[847]]

[[848]]

[[849]]

[[850]]

[[851]]

[[852]]

[[853]]

[[854]]

[[855]]

[[856]]

[[857]]

[[858]]

[[859]]

[[860]]

[[861]]

[[862]]

[[863]]

[[864]]

[[865]]

[[866]]

[[867]]

[[868]]

[[869]]

[[870]]

[[871]]

[[872]]

[[873]]

[[874]]

[[875]]

[[876]]

[[877]]

[[878]]

[[879]]

[[880]]

[[881]]

[[882]]

[[883]]

[[884]]

[[885]]

[[886]]

[[887]]

[[888]]

[[889]]

[[890]]

[[891]]

[[892]]

[[893]]

[[894]]

[[895]]

[[896]]

[[897]]

[[898]]

[[899]]

[[900]]

[[901]]

[[902]]

[[903]]

[[904]]

[[905]]

[[906]]

[[907]]

[[908]]

[[909]]

[[910]]

[[911]]

[[912]]

[[913]]

[[914]]

[[915]]

[[916]]

[[917]]

[[918]]

[[919]]

[[920]]

[[921]]

[[922]]

[[923]]

[[924]]

[[925]]

[[926]]

[[927]]

[[928]]

[[929]]

[[930]]

[[931]]

[[932]]

[[933]]

[[934]]

[[935]]

[[936]]

[[937]]

[[938]]

[[939]]

[[940]]

[[941]]

[[942]]

[[943]]

[[944]]

[[945]]

[[946]]

[[947]]

[[948]]

[[949]]

[[950]]

[[951]]

[[952]]

[[953]]

[[954]]

[[955]]

[[956]]

[[957]]

[[958]]

[[959]]

[[960]]

[[961]]

[[962]]

[[963]]

[[964]]

[[965]]

[[966]]

[[967]]

[[968]]

[[969]]

[[970]]

[[971]]

[[972]]

[[973]]

[[974]]

[[975]]

[[976]]

[[977]]

[[978]]

[[979]]

[[980]]

[[981]]

[[982]]

[[983]]

[[984]]

[[985]]

[[986]]

[[987]]

[[988]]

[[989]]

[[990]]

[[991]]

[[992]]

[[993]]

[[994]]

[[995]]

[[996]]

[[997]]

[[998]]

[[999]]

[[1000]]

[[1001]]

[[1002]]

[[1003]]

[[1004]]

[[1005]]

[[1006]]

[[1007]]

[[1008]]

[[1009]]

[[1010]]

[[1011]]

[[1012]]

[[1013]]

[[1014]]

[[1015]]

[[1016]]

[[1017]]

[[1018]]

[[1019]]

[[1020]]

[[1021]]

[[1022]]

[[1023]]

[[1024]]

[[1025]]

[[1026]]

[[1027]]

[[1028]]

[[1029]]

[[1030]]

[[1031]]

[[1032]]

[[1033]]

[[1034]]

[[1035]]

[[1036]]

[[1037]]

[[1038]]

[[1039]]

[[1040]]

[[1041]]

[[1042]]

[[1043]]

[[1044]]

[[1045]]

[[1046]]

[[1047]]

[[1048]]

[[1049]]

[[1050]]

[[1051]]

[[1052]]

[[1053]]

[[1054]]

[[1055]]

[[1056]]

[[1057]]

[[1058]]

[[1059]]

[[1060]]

[[1061]]

[[1062]]

[[1063]]

[[1064]]

[[1065]]

[[1066]]

[[1067]]

[[1068]]

[[1069]]

[[1070]]

[[1071]]

[[1072]]

[[1073]]

[[1074]]

[[1075]]

[[1076]]

[[1077]]

[[1078]]

[[1079]]

[[1080]]

[[1081

2) Write a R program to perform below operations on Data Structures like **matrix** and **array** –

- (a) Determining the length
- (b) Accessing the elements
- (c) Check element exists or not
- (d) Add row and column
- (e) Remove row and column
- (f) Combining the two matrices

The screenshot shows the RStudio interface. The top panel is the code editor with the tab 'Untitled1*' selected. The code defines a matrix and an array, then performs various operations on them. The bottom panel is the console, showing the R session where the code is run and its output.

```
# Declaring matrix and array
mtrix <- matrix(c(0:9), nrow=5, byrow=TRUE)
arr <- array(mtrix, dim=c(3, 3, 2))
arr

# Determining the length
dim(mtrix) # matrix dimension
length(mtrix) # matrix length
length(arr)

# Accessing the elements in a matrix
print(mtrix[1,]) # accessing all elements in row 1
print(mtrix[,1]) # accessing all elements in column 1
print(arr[1,2,1]) # accessing elements in array

# check if element exist or not
print(5 %in% mtrix) # 5 exist in matrix hence true
print(15 %in% mtrix) # 15 does not exist in matrix hence false
print(8 %in% arr) # 8 exist in array
```



```
Console Terminal Background Jobs
R 4.2.1 · ~/Assignment2/ ↗
> # Declaring matrix and array
> mtrix <- matrix(c(0:9), nrow=5, byrow=TRUE)
> arr <- array(mtrix, dim=c(3, 3, 2))
> # Determining the length
> dim(mtrix) # matrix dimension
[1] 5 2
> length(mtrix) # matrix length
[1] 10
> length(arr)
[1] 18
> # Accessing the elements in a matrix
> print(mtrix[1,]) # accessing all elements in row 1
[1] 0 1
> print(mtrix[,1]) # accessing all elements in column 1
[1] 0 2 4 6 8
> print(arr[1,2,1]) # accessing elements in array
[1] 6
> # check if element exist or not
> print(5 %in% mtrix) # 5 exist in matrix hence true
[1] TRUE
> print(15 %in% mtrix) # 15 does not exist in matrix hence false
[1] FALSE
> print(8 %in% arr) # 8 exist in array
[1] TRUE
> |
```

```

# adding row or column
cmtrix <- cbind(c(0,1),mtrix)      # adding column in matrix
cmtrix
rmtrix <- rbind(c(10,11),mtrix)    # adding row in matrix
rmtrix

# removing row and column in matrix
cmtrix <- cmtrix[,-c(1)]
cmtrix
rmtrix <- rmtrix[-c(1),]
rmtrix

# combining the two matrix
combinedmtrix <- cbind(cmtrix,rmtrix) # combined the matrix
combinedmtrix
arr2 <- array(c(1:9))
merged.array <- cbind(arr,arr2) # combined the row
merged.array

```

Console Terminal × Background Jobs ×

R 4.2.1 · ~/Assignment2/ ↗

```

> # adding row or column
> cmtrix <- cbind(c(1,2,3,4,5),mtrix)      # adding column in matrix
> cmtrix
 [,1] [,2] [,3]
[1,]    1    0    1
[2,]    2    2    3
[3,]    3    4    5
[4,]    4    6    7
[5,]    5    8    9
> rmtrix <- rbind(c(10,11),mtrix)    # adding row in matrix
> rmtrix
 [,1] [,2]
[1,]   10   11
[2,]    0    1
[3,]    2    3
[4,]    4    5
[5,]    6    7
[6,]    8    9
> # removing row and column in matrix
> cmtrix <- cmtrix[,-c(1)]
> cmtrix
 [,1] [,2]
[1,]    0    1
[2,]    2    3
[3,]    4    5
[4,]    6    7
[5,]    8    9

```

Source

Console Terminal × Background Jobs ×

R 4.2.1 · ~/Assignment2/ ↗

```

[5,]    8    9
> rmtrix <- rmtrix[-c(1),]
> rmtrix
 [,1] [,2]
[1,]    0    1
[2,]    2    3
[3,]    4    5
[4,]    6    7
[5,]    8    9
> # combining the two matrix
> combinedmtrix <- cbind(cmtrix,rmtrix) # combined the matrix
> combinedmtrix
 [,1] [,2] [,3] [,4]
[1,]    0    1    0    1
[2,]    2    3    2    3
[3,]    4    5    4    5
[4,]    6    7    6    7
[5,]    8    9    8    9

```

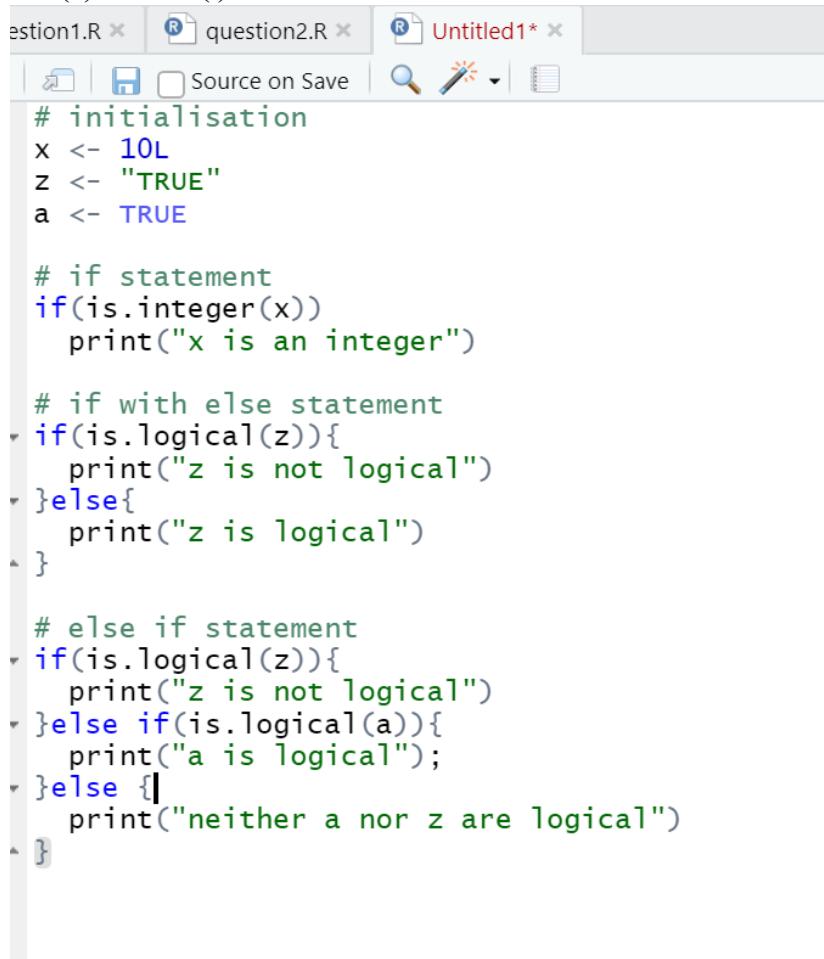
```

> arr2 <- array(c(1:9))
> merged.array <- cbind(arr,arr2) # combined the row
> merged.array
      arr arr2
[1,]    0    1
[2,]    2    2
[3,]    4    3
[4,]    6    4
[5,]    8    5
[6,]    1    6
[7,]    3    7
[8,]    5    8
[9,]    7    9
[10,]   9    1
[11,]   0    2
[12,]   2    3
[13,]   4    4
[14,]   6    5
[15,]   8    6
[16,]   1    7
[17,]   3    8
[18,]   5    9

```

3) Create R Code for Condition Statements -

- (a) if()
- (b) if() ... else {...}
- (c) else if()



The screenshot shows the RStudio interface with the following details:

- File Bar:** Contains tabs for "question1.R", "question2.R", and "Untitled1*".
- Toolbar:** Includes icons for file operations (New, Open, Save), source control (Source on Save), and search.
- Code Area:** Displays the following R code:

```

# initialisation
x <- 10L
z <- "TRUE"
a <- TRUE

# if statement
if(is.integer(x))
  print("x is an integer")

# if with else statement
if(is.logical(z)){
  print("z is not logical")
} else{
  print("z is logical")
}

# else if statement
if(is.logical(z)){
  print("z is not logical")
} else if(is.logical(a)){
  print("a is logical");
} else {
  print("neither a nor z are logical")
}

```

R 4.2.1 · ~/Assignment2/ ↗

```

> # initialisation
> x <- 10L
> z <- "TRUE"
> a <- TRUE
> # if statement
> if(is.integer(x))
+   print("x is an integer")
[1] "x is an integer"
> # if with else statement
> if(is.logical(z)){
+   print("z is not logical")
+ }else{
+   print("z is logical")
+ }
[1] "z is logical"
> # else if statement
> if(is.logical(z)){
+   print("z is not logical")
+ }else if(is.logical(a)){
+   print("a is logical");
+ }else {
+   print("neither a nor z are logical")
+ }
[1] "a is logical"
>

```

4) Create R Code for Selection Control Structures –

- (a) switch statement
- (b) next statement
- (c) break statement

The screenshot shows the RStudio interface with four tabs at the top: question1.R, question2.R, question3.R, and Untitled1*. Below the tabs is a toolbar with icons for file operations, source saving, and search. The main area displays the following R code:

```

question1.R x question2.R x question3.R x Untitled1* x
1 # switch statement
2 result <- switch(paste("myname",sep = " "),
3                     "friend1" = "harsh",
4                     "friend2" = "aryan",
5                     "friend3" = "shivang",
6                     "myname" = "aniruddha")
7 print(result)
8
9 # next & break
L0 i <- 1
L1 repeat
L2 {
L3   if(i==5){
L4     break}
L5   if(i==5) {
L6     next
L7   }
L8   print(i)
L9   i <- i+1
L10 }

```

Console Terminal Background Jobs

R 4.2.1 · ~/Assignment2/ ↗

```
> # switch statement
> result <- switch(paste("myname",sep = " "), 
+                     "friend1" = "harsh",
+                     "friend2" = "aryan",
+                     "friend3" = "shivang",
+                     "myname"   = "aniruddha")
> print(result)
[1] "aniruddha"
> # next & break
> i <- 1
> repeat
+ {
+   if(i==5){
+     break}
+   if(i==5) {
+     next
+   }
+   print(i)
+   i <- i+1
+ }
[1] 1
[1] 2
[1] 3
[1] 4
> |
```

5) Create R Code for Looping Statements –

- (a) repeat Loop
- (b) while Loop
- (c) for Loop

```
# repeat program in r

v <- c(5,2,12,12,32)
mx <- 0
i <- 1
repeat {
  if(length(v)<i){
    break
  }
  if(v[i]>mx){
    mx <- v[i]
  }
  i<- i+1
}
print(paste("Maximum number= ", mx))
```

Console Terminal × Background Jobs ×

R 4.2.1 · ~/Assignment2/ ↗

```
> v <- c(5,2,12,12,32)
> mx <- 0
> i <- 1
> repeat {
+   if(length(v)<i){
+     break
+   }
+   if(v[i]>mx){
+     mx <- v[i]
+   }
+   i<- i+1
+ }
> print(paste("Maximum number= ", mx))
[1] "Maximum number= 32"
>
```

Source on Save | 🔎 | ✎ | 📁

```
1 v <- c(5,2,12,12,32)
2
3 k <- 32
4 i <- 1
5 while(length(v)>i){
6   if(k==v[i])
7     break
8   i<-i+1
9 }
10 print(paste("Element found at index ",k))
11
12
```

Console Terminal × Background Jobs ×

R 4.2.1 · ~/Assignment2/ ↗

```
v <- c(5,2,12,12,32)
k <- 32
i <- 1
while(length(v)>i){
  if(k==v[i])
    break
  i<-i+1
}
print(paste("Element found at index ",i))
[1] "Element found at index 5"
```

Source on Save | 🔎 | ✎ | 📁

```
1 fruit <- c('Apple', 'Orange', "Guava", 'Pinapple', 'Banana', 'Grapes')
2 # creating for statement for displaying the fruits
3 for ( i in fruit){
4   print(i)
5 }
6
```

Console Terminal Background Jobs

```
R 4.2.1 · ~/Assignment2/ ↵
> fruit <- c('Apple', 'orange','Guava", 'Pinapple', 'Banana','Grapes')
> # creating for statement for displaying the fruits
> for ( i in fruit){
+   print(i)
+
[1] "Apple"
[1] "Orange"
[1] "Guava"
[1] "Pinapple"
[1] "Banana"
[1] "Grapes"
> |
```

6) Write a R program to perform below operations using String functions –

- (a) String Length
- (b) Check a String
- (c) Multiline Strings
- (d) Combine Two Strings
- (e) Escape Characters

The screenshot shows two RStudio panes. The left pane contains R code for string manipulation, including string length, checking for substrings, creating multiline strings, combining two strings, and using escape characters. The right pane shows the output of this code, which is a story about a father and son playing with a globe.

```
# string length function
str <- "aniruddha"
nchar(str)

# check a string
grepl("ruddh",str) #--> TRUE : ruddh is present in string "aniruddha"

# creating multiline string
ms <- 'One night a father was relaxing with his newspaper,
after a long day at office. His son, who wanted to play,kept,
on pestering him. Finally, fed up ,the father ripped out a picture of the globe that was in the paper and tore it into a
tuby pieces ." Here,son go ahead and try to put this back together" ,he said,hoping that this would keep the little boy b
long enough to finish reading the newspaper .To his amazement, his son returned after only one minute with the globe per
together , When the startled father asked how he achieved this feat, the smiled gently and replied, "Dad, on the other s
there was a picture of a person and once I got the person together ,the worl was okay"
cat(ms)

# combine two string
str1 <- " My name is "
str2 <- "Aniruddha"
res <- paste(str1,str2)
print(res)

# escape character
s <- "Ram said,\n\"I am on a voyage to find my true aim in life\""
cat(s)
```



```
R 4.2.1 · ~/Assignment2/ ↵
> # string length function
> str <- "aniruddha"
> nchar(str)
[1] 9
> # check a string
> grepl("ruddh",str) #--> TRUE : ruddh is present in string "aniruddha"
[1] TRUE
> # creating multiline string
> ms <- 'One night a father was relaxing with his newspaper,
+ after a long day at office. His son, who wanted to play,kept,
+ on pestering him. Finally, fed up ,the father ripped out a picture of the globe that was in the paper and tore it into a hundred
+ tuby pieces ." Here,son go ahead and try to put this back together" ,he said,hoping that this would keep the little boy busy,
+ long enough to finish reading the newspaper .To his amazement, his son returned after only one minute with the globe perfectly b
ack,
+ together , when the startled father asked how he achieved this feat, the smiled gently and replied, "Dad, on the other side of t
he",
+ there was a picture of a person and once I got the person together ,the worl was okay"
> cat(ms)
[1] "One night a father was relaxing with his newspaper,
+ after a long day at office. His son, who wanted to play,kept,
+ on pestering him. Finally, fed up ,the father ripped out a picture of the globe that was in the paper and tore it into a hundred ,
+ tuby pieces ." Here,son go ahead and try to put this back together" ,he said,hoping that this would keep the little boy busy,
+ long enough to finish reading the newspaper .To his amazement, his son returned after only one minute with the globe perfectly bac
k,
+ together , when the startled father asked how he achieved this feat, the smiled gently and replied, "Dad, on the other side of th
e",
+ there was a picture of a person and once I got the person together ,the worl was okay"
> # combine two string
> str1 <- " My name is "
> str2 <- "Aniruddha"
> res <- paste(str1,str2)
> print(res)
[1] " My name is Aniruddha"
> # escape character
> s <- "Ram said,\n\"I am on a voyage to find my true aim in life\""
> cat(s)
[1] "Ram said,
+I am on a voyage to find my true aim in life"
```