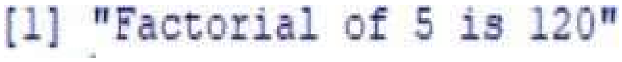
Practical 1:- Write an R-Program to demonstrate working with Arithmetic operators.

```
> a=1
> b=2
> c=a+b
> c
[1] 3
> a=10
> b=5
> c=a-b
> c
[1] 5
> a=5
> b=3
> c=a*b
> c
[1] 15
> a=6
> b=3
> c=a/b
> c
[1] 2
> vc1<-c(1,2)
> vc2<-c(3,4) > cat("add=",vc1+vc2)

add= 4 6>

> vc1<-c(2,1)
> vc2<-c(1,0)
> cat("sub=",vc1-vc2)
sub= 1 1>
> vc1<-c(3,4)
> vc2<-c(2,2) > cat("mul=",vc1*vc2)

mul= 6 8>

> vc1<-c(4,8)
> vc2<-c(2,2) > cat("div=",vc1/vc2)

div= 2 4>
```

Practical 2:- Write an R-Program to find the Factorial of a Number.

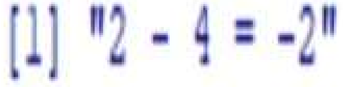
```
> findfact<-function (n) {

+ factorial<-1

+ if(n<0)

+ print("Factorial of negative numbers is not possible")

+ else{

+ for (i in 1:n)

+ factorial<-factorial*i

+ print(paste("Factorial of",n,"is",factorial))

+ }

+ }

> findfact(5)
```



```
[1] "Factorial of 5 is 120"
```

Practical 3 :-Write an R-Program to make a Simple Calculator.

```
> add<-function(x,y) {
+ return(x+y)
+ }
> subtract<-function(x,y) {
+ return(x-y)
+ }
> multiply<-function(x,y) {
+ return(x*y)
+ }
> divide<-function(x,y) {
+ return(x/y)
+ }
> print("Select operation.")
[1] "Select operation."
> print("1.Add")
[1] "1.Add"
> print("2.Subtract") [1] "2.Subtract"

> print("3.Multiply")
[1] "3.Multiply"
> print("4.Divide")
[1] "4.Divide"
> choice=as.integer(readline(prompt="Enter choice[1/2/3/4]:"))
Enter choice[1/2/3/4]:2
> num1=as.integer(readline(prompt="Enter first number:"))
Enter first number:2
> num2=as.integer(readline(prompt="Enter second number:"))
Enter second number:4
> operator<-switch(choice,"+","-","*","/")
> result<-
switch(choice,add(num1,num2),subtract(num1,num2),multiply(num1,num2),divide(num1,n um2))
> print(paste(num1,operator,num2,"=",result))
```



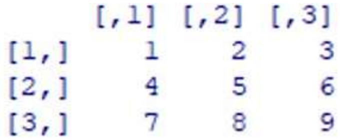
```
[1] "2 - 4 = -2"
```

Practical 4:- Write an R-Program to create a Matrix and access rows and columns using functions colnames() and rownames.

```
A=matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=TRUE)
> rownames(A)<-letters[1:3]
> print(A)
  [,1] [,2] [,3] a    1    2    3 b

4    5    6 c    7    8    9


> A=matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=TRUE)
> colnames(A)<-letters[1:3]
> print(A)
     a b c
[1,] 1 2 3
[2,] 4 5 6
[3,] 7 8 9
```

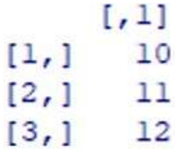
Practical 5:- Write an R-Program to create a Matrix using cbind() and rbind() functions.

```
> A=matrix(
+ c(1,2,3,4,5,6,7,8,9),
+ nrow=3,
+ ncol=3,
+ byrow=TRUE
+ )
> cat("The 3*3 matrix:\n")
The 3*3 matrix:
> print(A)
```



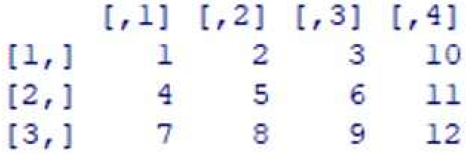
```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```
> B=matrix(
+ c(10,11,12),
+ nrow=3,
+ ncol=1,
+ byrow=TRUE
+ )
> cat("The 3*1 matrix:\n")
The 3*1 matrix:
> print(B)
```



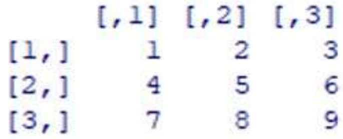
```
     [,1]
[1,]   10
[2,]   11
[3,]   12
```

```
> C=cbind(A,B)
> cat("After concatenation of column:\n")
After concatenation of column:
> print(C)
```



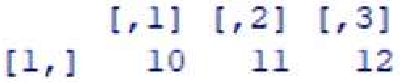
```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3   10
[2,]    4    5    6   11
[3,]    7    8    9   12
```

```
> A=matrix(
+ c(1,2,3,4,5,6,7,8,9),
+ nrow=3,
+ ncol=3,
+ byrow=TRUE
+ )
> cat("The 3*3 matrix:\n")
The 3*3 matrix:
> print(A)
```



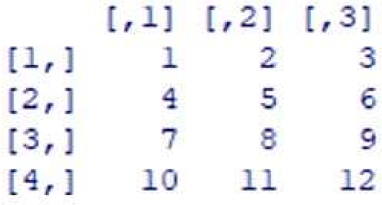
```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```
> B=matrix(
+ c(10,11,12),
+ nrow=1,
+ ncol=3,
+ )
> cat("The 1*3 matrix:\n")
The 1*3 matrix:
> print(B)
```



```
     [,1] [,2] [,3]
[1,]   10   11   12
```

```
> C=rbind(A,B)
> cat("After concatenation of a column:\n")
After concatenation of a column:
> print(C)
```



```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
```

Practical 6:- Write an R-Program to create a Matrix from a vector using dim() function.

```
> #Getting the number from 1 to 9

> x<-rep(1:9)

> x

[1] 1 2 3 4 5 6 7 8 9

> #Calling the dim()function to

> #Set dimension of 3*3

> dim(x)<-c(3,3)

> x

    [,1] [,2] [,3]

[1,]    1    4    7

[2,]    2    5    8 [3,]    3    6    9
```

Practical 7:- Write an R-Program to create list and modify its components.

> lt<-list(a=1,

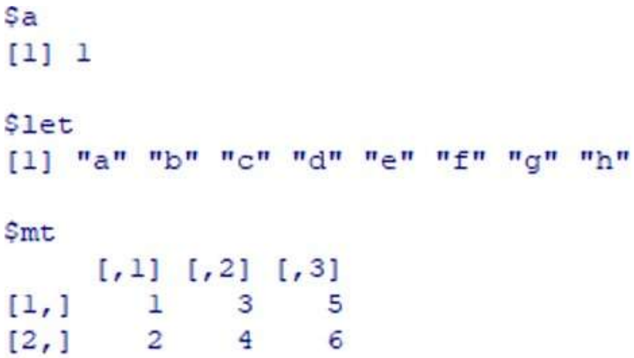
+ let=letters[1:8],

+ mt=matrix(1:6,nrow=2)) > cat("List before

modifying:\n")

List before modifying:

> print(lt)



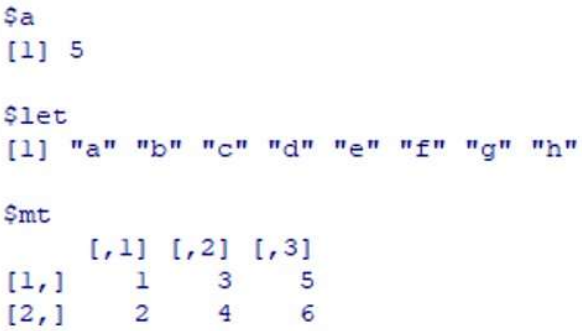
```
$a
[1] 1

$let
[1] "a" "b" "c" "d" "e" "f" "g" "h"

$mt
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

> lt$a<-5

> cat("List after modefying:\n")

List after modefying:

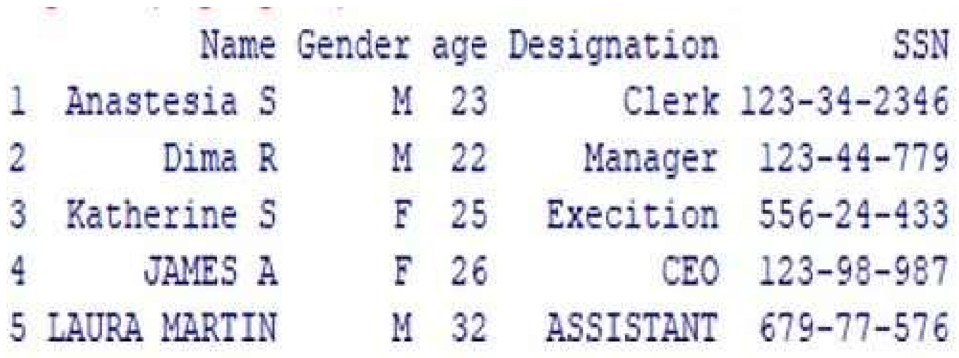
> print(lt)



```
$a
[1] 5

$let
[1] "a" "b" "c" "d" "e" "f" "g" "h"

$mt
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Practical 8:- Write a R-Program to create a Data Frame.

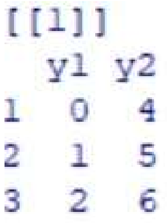
```
> Employees=data.frame(Name=c("Anastesia S","Dima R","Katherine S","James R","Laura M"),

+ Gender=c("M","M","F","F","M"),

+ Age=c(23,22,25,26,32),

+ Designation=c("Clerk","Manager","Execition","CEO","Assistant"),

+ SSN=c("123-34-2346","123-44-779","556-24-443","123-98-987","679-77-576")

+ )

> print("Details of the Employees:")

[1] "Details of the Employees:"

> print(Employees)
```



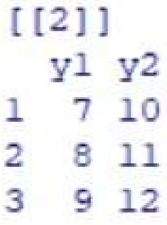
```
          Name Gender age Designation         SSN
1  Anastesia S      M  23       Clerk 123-34-2346
2       Dima R      M  22     Manager  123-44-779
3  Katherine S      F  25   Execition  556-24-433
4      JAMES A      F  26         CEO  123-98-987
5 LAURA MARTIN      M  32   ASSISTANT  679-77-576
```

Practical 9:- Write an R-Program to access a Data Frame like a list.

```
> df1=data.frame(y1=c(0,1,2), y2=c(4,5,6))

> df2=data.frame(y1=c(7,8,9), y2=c(10,11,12))

> new_list=list(df1,df2)

> print("New_list:")

[1] "New_list:"

> print(new_list)
```

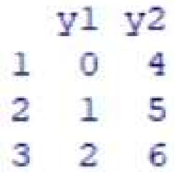


```
[[1]]
  y1 y2
1  0  4
2  1  5
3  2  6
```



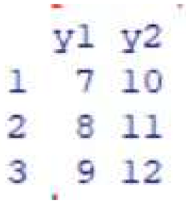
```
[[2]]
  y1 y2
1  7 10
2  8 11
3  9 12
```

```
> print("Data frame-1")

[1] "Data frame-1"

> print(new_list[[1]])
```



```
  y1 y2
1  0  4
2  1  5
3  2  6
```

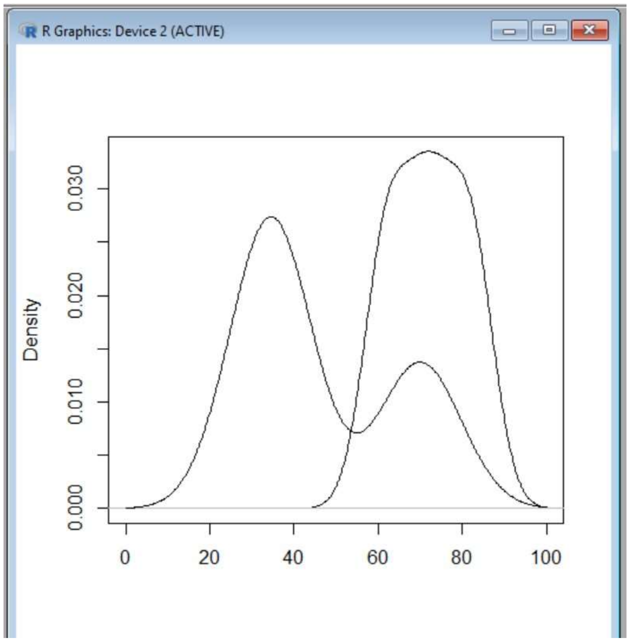
```
> print("Data frame-2")

[1] "Data frame-2"

> print(new_list[[2]])
```



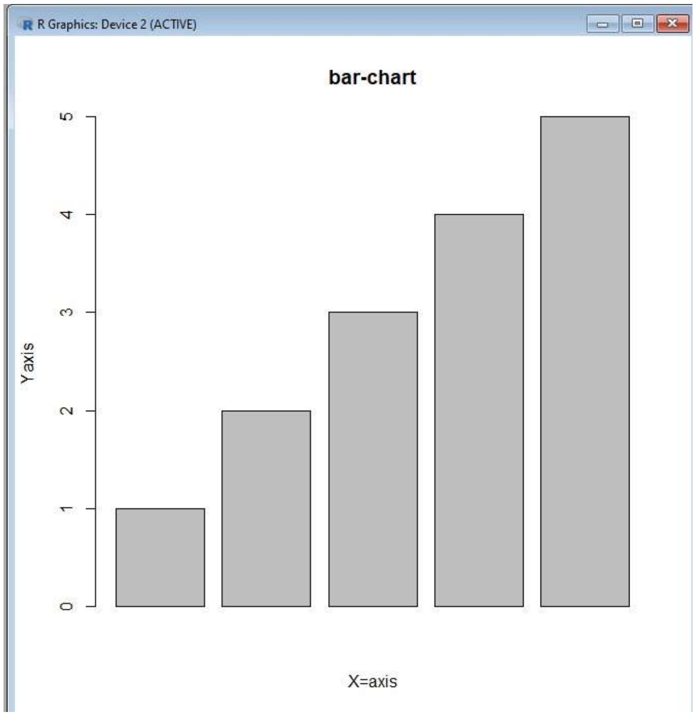
```
  y1 y2
1  7 10
2  8 11
3  9 12
```

Practical 10:- Write R command to create a bar plot of the data by using argument of the used function for filling up the bars and to give the labels to the axix 5,10,8,7,8,5,8,7,5,8,9,6,8,8,8

```
> testscore=data.frame(Exam1=c(62,72,82),Exam2=c(70,34,35))
> d1=
+ density(testscore$Exam1,from=0,to=100)
> d2=
+ density(testscore$Exam2,from=0,to=100)
> plot(d1,main="",xlab="")
> lines(d2)
```

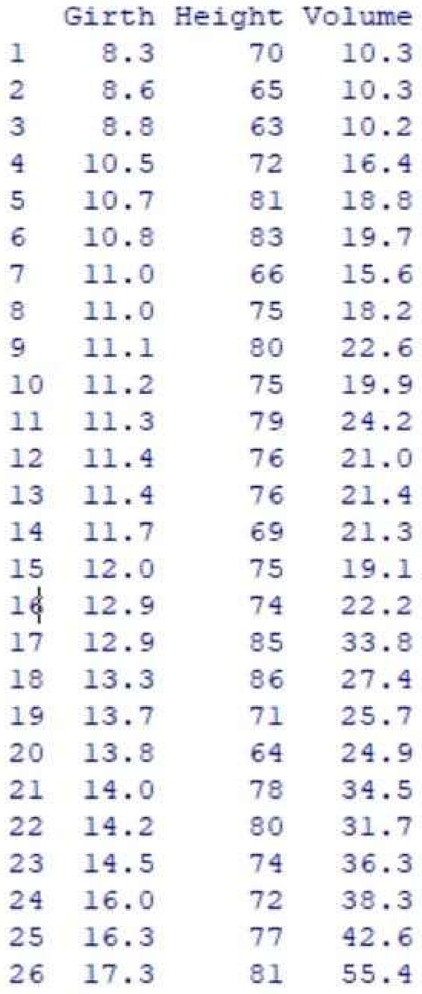


```
>A<-c(1,2,3,4,5)
>Barplot(A,xlab=”X=axis”,ylab=”Yaxis”,main=”bar-chart”)
```

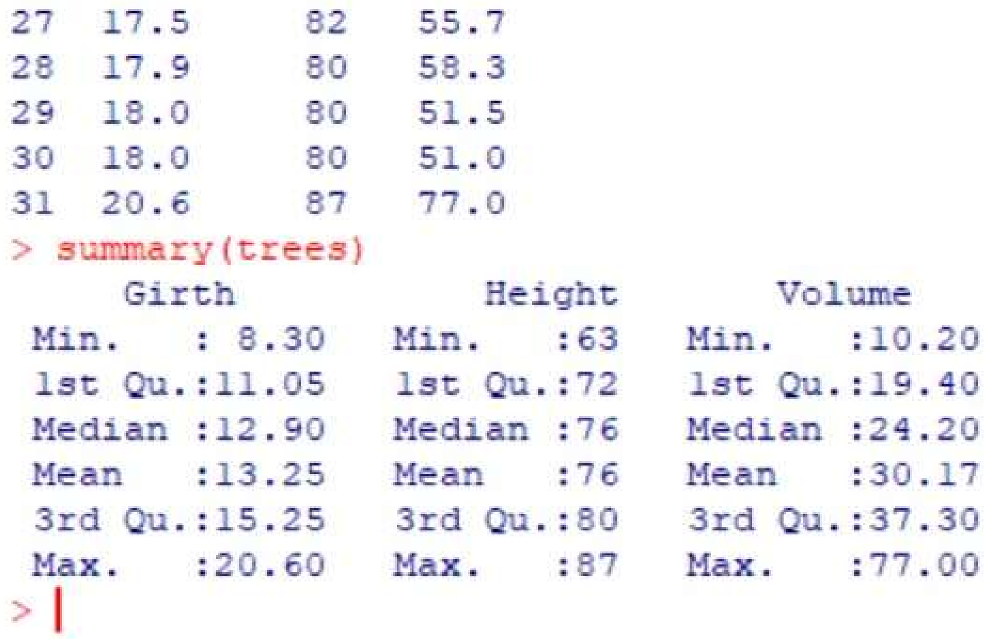


Practical 11: In R we have a built-in data set “trees”. Calculate summary, first 4 rows of data and last 4 rows of data from trees dataset.

```
> df<-datasets::trees
> df
```



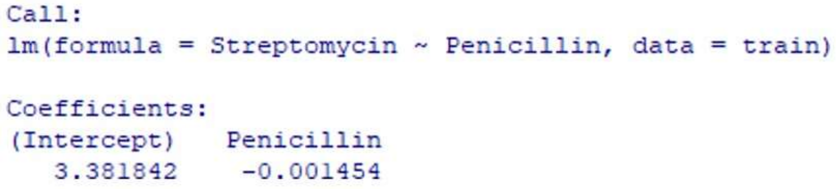
```
   Girth Height Volume
1    8.3     70   10.3
2    8.6     65   10.3
3    8.8     63   10.2
4   10.5     72   16.4
5   10.7     81   18.8
6   10.8     83   19.7
7   11.0     66   15.6
8   11.0     75   18.2
9   11.1     80   22.6
10  11.2     75   19.9
11  11.3     79   24.2
12  11.4     76   21.0
13  11.4     76   21.4
14  11.7     69   21.3
15  12.0     75   19.1
16  12.9     74   22.2
17  12.9     85   33.8
18  13.3     86   27.4
19  13.7     71   25.7
20  13.8     64   24.9
21  14.0     78   34.5
22  14.2     80   31.7
23  14.5     74   36.3
24  16.0     72   38.3
25  16.3     77   42.6
26  17.3     81   55.4
```



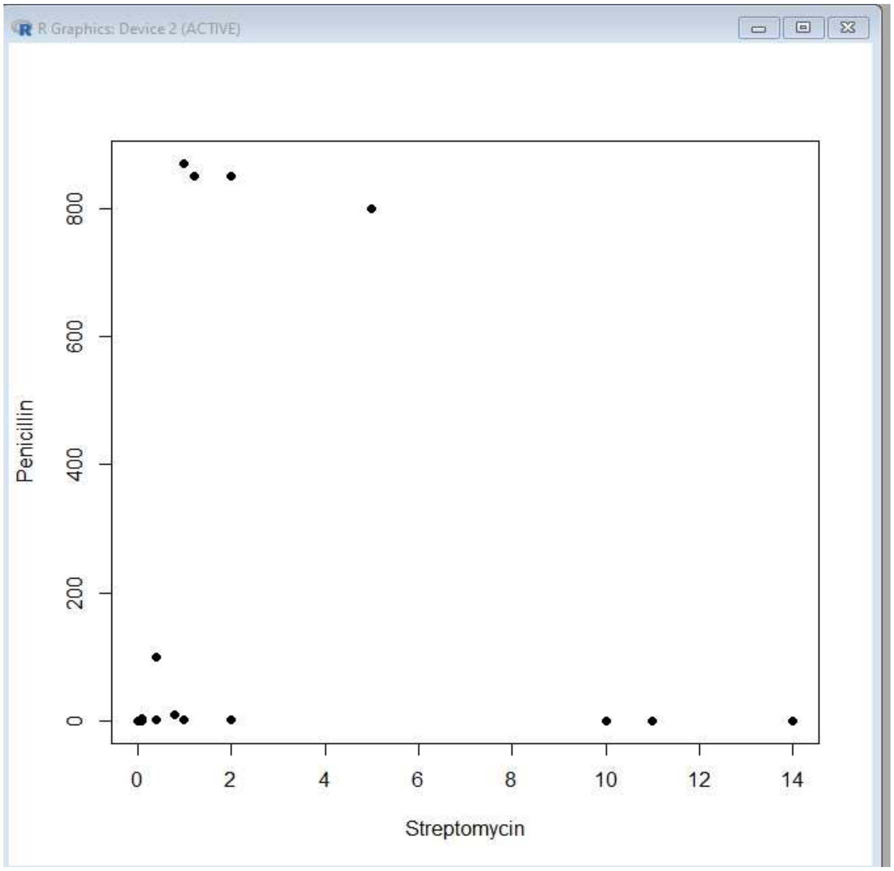
```
27  17.5     82   55.7
28  17.9     80   58.3
29  18.0     80   51.5
30  18.0     80   51.0
31  20.6     87   77.0
> summary(trees)
     Girth           Height       Volume
 Min.   : 8.30   Min.   :63   Min.   :10.20
 1st Qu.:11.05   1st Qu.:72   1st Qu.:19.40
 Median :12.90   Median :76   Median :24.20
 Mean   :13.25   Mean   :76   Mean   :30.17
 3rd Qu.:15.25   3rd Qu.:80   3rd Qu.:37.30
 Max.   :20.60   Max.   :87   Max.   :77.00
> |
```

Practical 12:- Write R-Program to work which is examined the antibiotics dataset. Create a training data and model using Antibiotics database.

```
>data<-read.csv(“F:/Thesis Topic/badneradb/Antibiotics.csv”)

>attach(data)

>train=data.frame(Streptomycin,Pencillin)

>plot(train,pch=16)

>model<-Im(Streptomycin Pencillin,train)

>model
```

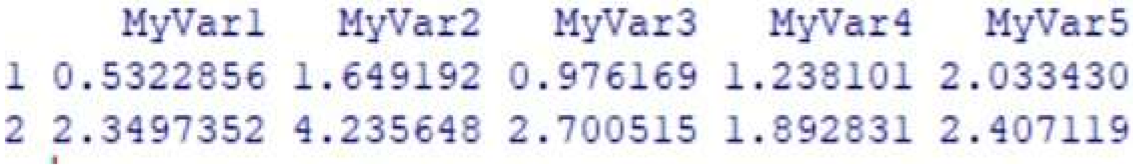


```
Call:
lm(formula = Streptomycin ~ Penicillin, data = train)

Coefficients:
(Intercept)   Penicillin
   3.381842    -0.001454
```



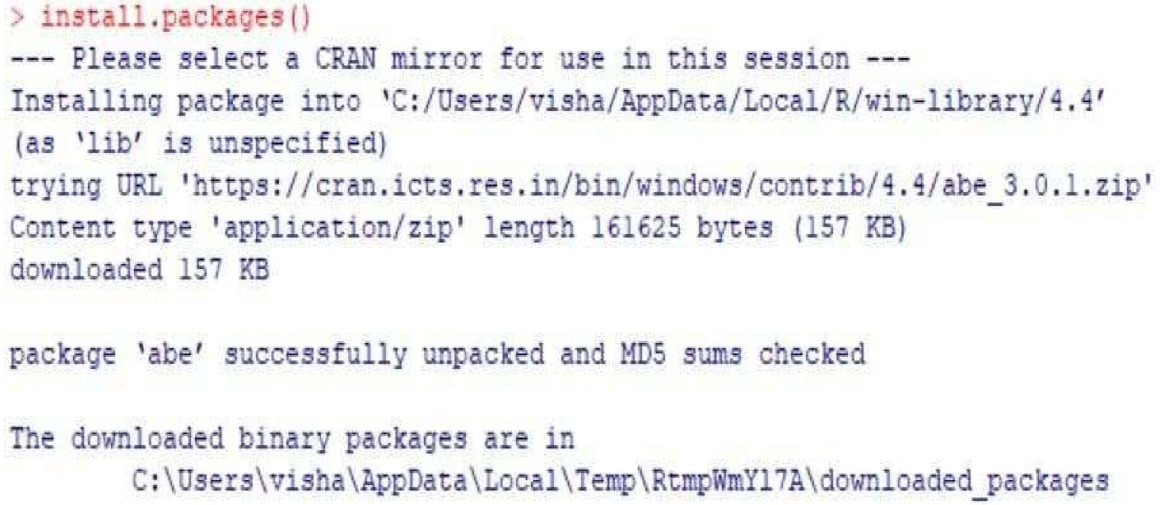
Practical 13:- Write a data matrix or data frame using the write table() unction write table() has similar arguments to read table.

```
> myResults<-matrix(rnorm(100,mean=2),nrow=20)

> write.table(myResults,file="results.txt")

> df1<-data.frame(myResults)

> colnames(df1)<-paste("MyVar",1:5,sep="")

> write.table(df1,file="results2.txt",row.names=FALSE,

+ col.names=TRUE)

> read.table(file="results2.txt",head=TRUE)[1:2,]
```



```
     MyVar1   MyVar2   MyVar3   MyVar4   MyVar5
1 0.5322856 1.649192 0.976169 1.238101 2.033430
2 2.3497352 4.235648 2.700515 1.892831 2.407119
```

Practical 14:- How to Install and use the Package in R



```
> install.packages()
--- Please select a CRAN mirror for use in this session ---
Installing package into ‘C:/Users/visha/AppData/Local/R/win-library/4.4’
(as ‘lib’ is unspecified)
trying URL 'https://cran.icts.res.in/bin/windows/contrib/4.4/abe_3.0.1.zip'
Content type 'application/zip' length 161625 bytes (157 KB)
downloaded 157 KB

package ‘abe’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\visha\AppData\Local\Temp\RtmpWmYl7A\downloaded_packages
```