PRACTICAL No. 1

BASIC SYNTAX

AIM: Using R execute the basic commands, array, list and frames.

SOURCE CODE & OUTPUT:

1. Hello world program:

```
> # My first program in R Programming
> helloString<-"Hello, World!!!"
> print(helloString)
[1] "Hello, World!!!"
> print(helloString,quote=FALSE)
[1] Hello, World!!!
```

2. R – Datatypes:

i. **Logical**
ii. **Numeric**
iii. **Integer**
iv. **Complex**
v. **Character**
vi. **Raw**

```
> # R - Datatypes
> # LOGICAL
> v<-TRUE
> print(class(v))
[1] "logical"
> # NUMERIC
> v<-2.5
> print(class(v))
[1] "numeric"
> v<-7
> print(class(v))
[1] "numeric"
> # INTEGER
> v<-7L
> print(class(v))
[1] "integer"
> # COMPLEX
> v<-2+5i
> print(class(v))
[1] "complex"
> # CHARACTER
> v<-"TRUE"
> print(class(v))
[1] "character"
> v<-'Welcome to R Programming!!!'
> print(class(v))
[1] "character"
> # RAW
> v<-charToRaw("TRUE")
> print(v)
[1] 54 52 55 45
> print(class(v))
[1] "raw"
```

3. R – Vectors:

- **Vector Creation**

i. Using Colon Operator:

```
> # Creating sequence from 2 to 9
> v<-2:9
> print(v)
[1] 2 3 4 5 6 7 8 9
> print(class(v))
[1] "integer"
> # Creating sequence from 2.5 to 12.5
> v<-2.5:12.5
> print(v)
 [1]  2.5  3.5  4.5  5.5  6.5  7.5  8.5  9.5 10.5 11.5 12.5
> print(class(v))
[1] "numeric"
> v<-2.6:10
> print(v)
[1] 2.6 3.6 4.6 5.6 6.6 7.6 8.6 9.6
```

ii. Using Sequence Operator:

```
> # Creating vector by using seq()
> v<-seq(3,7,by=0.5)
> print(v)
[1] 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0
> print(class(v))
[1] "numeric"
> v<-seq(1,14,by=2)
> print(v)
[1]  1  3  5  7  9 11 13
> print(class(v))
[1] "numeric"
> v<-seq(1,3,by=0.4)
> print(v)
[1] 1.0 1.4 1.8 2.2 2.6 3.0
```

iii. Using c() Function:

```
> # R - Vectors
> colors<-c('Red','Blue','Green','Yellow')
> print(colors)
[1] "Red"    "Blue"   "Green"  "Yellow"
> print(class(colors))
[1] "character"
> print(colors,quote=FALSE)
[1] Red    Blue   Green  Yellow
> age<-c(19L,23L,22L,27L)
> print(age)
[1] 19 23 22 27
> print(class(age))
[1] "integer"
```

```
> height<-c(4.11,5.2,5.5,5.7,5)
> print(height)
[1] 4.11 5.20 5.50 5.70 5.00
> print(class(height))
[1] "numeric"
> mix<-c(4L,3.14,1+3i)
> print(mix)
[1] 4.00+0i 3.14+0i 1.00+3i
> print(class(mix))
[1] "complex"
> mix2<-c('true',1.732,12L,2-i)
Error: object 'i' not found
> mix2<-c('true',1.732,12L,2-1i)
> print(mix2)
[1] "true"  "1.732" "12"    "2-1i"
> print(class(mix2))
[1] "character"
```

- **Accessing Vector Elements:**

  i. Using Position:

```
> # Accessing Vector Elements Using Position
> v<-c('SUN','MON','TUE','WED','THURS','FRI','SAT')
> print(v[4])
[1] "WED"
> print(v[c(1,3,5)])
[1] "SUN"   "TUE"   "THURS"
```

  ii. Using Logical Indexing:

```
> # Accessing Vector Elements Using Logical Indexing
> v<-c('SUN','MON','TUE','WED','THURS','FRI','SAT')
> print(v[c(FALSE,TRUE,FALSE,FALSE,TRUE,FALSE,TRUE)])
[1] "MON"   "THURS" "SAT"
> print(v[c(FALSE,TRUE)])
[1] "MON" "WED" "FRI"
> print(v[c(FALSE,TRUE,TRUE)])
[1] "MON"   "TUE"   "THURS" "FRI"
```

  iii. Using Negative Indexing:

```
> # Accessing Vector Elements Using Negative Indexing
> v<-c('SUN','MON','TUE','WED','THURS','FRI','SAT')
> print(v[-1])
[1] "MON"   "TUE"   "WED"   "THURS" "FRI"   "SAT"
> print(v[c(-2,-3,-5,-7)])
[1] "SUN" "WED" "FRI"
```

4. R – Lists:

- **Creating R – List:**

```
> # Creating a R-List
> firstList<-list('Monday',c(3,2,1),list('R','B','G'),sin)
> print(firstList)
[[1]]
[1] "Monday"

[[2]]
[1] 3 2 1

[[3]]
[[3]][[1]]
[1] "R"

[[3]][[2]]
[1] "B"

[[3]][[3]]
[1] "G"


[[4]]
function (x)  .Primitive("sin")
```

- **Naming List Elements:**

```
> # Naming List Elements
> secondList<-list('R-Programming',4L,list('SciLab Programming','C++ Programming','Mobile Programming',
+ 'R-Programming'))
> names(secondList)<-c('Current Programming Language:','Learning in Semester:','Learned in Semester')
> print(secondList)
$`Current Programming Language:`
[1] "R-Programming"

$`Learning in Semester:`
[1] 4

$`Learned in Semester`
$`Learned in Semester`[[1]]
[1] "SciLab Programming"

$`Learned in Semester`[[2]]
[1] "C++ Programming"

$`Learned in Semester`[[3]]
[1] "Mobile Programming"

$`Learned in Semester`[[4]]
[1] "R-Programming"
```

5. R – Matrices:

- **Creating R – Matrix:**

```
> # Creating R-Matrix by row
> A<-matrix(c(1,0,-1,2,3,6,1,2,0),nrow=3,ncol=3,byrow=TRUE)
> print(A)
     [,1] [,2] [,3]
[1,]    1    0   -1
[2,]    2    3    6
[3,]    1    2    0
> # Creating R-Matrix by column
> A<-matrix(c(1,0,-1,2,3,6,1,2,0),nrow=3,ncol=3,byrow=FALSE)
> print(A)
     [,1] [,2] [,3]
[1,]    1    2    1
[2,]    0    3    2
[3,]   -1    6    0
> A<-matrix(c(1,0,-1,2,3,6,1,2,0),nrow=3,ncol=3)
> print(A)
     [,1] [,2] [,3]
[1,]    1    2    1
[2,]    0    3    2
[3,]   -1    6    0
> B<-matrix(c(2,1,3,0,0,-1),nrow=2)
> print(B)
     [,1] [,2] [,3]
[1,]    2    3    0
[2,]    1    0   -1
```

- **Naming Rows and Columns of Matrix:**

```
> # Naming Rows and Columns of Matrix
> colNames<-c('No. of Girls','No. of Boys')
> rowNames<-c('O Grade','A+ Grade','A Grade','B+ Grade','B Grade','C Grade','D Grade','Fails/ATKT')
> ResultAnalysis<-matrix(c(3,1,2,3,5,2,2,1,4,11,5,3,3,0,3,10),ncol=2,byrow=TRUE,dimnames=list(rowNames,colNames))
> print(ResultAnalysis)
           No. of Girls No. of Boys
O Grade               3           1
A+ Grade              2           3
A Grade               5           2
B+ Grade              2           1
B Grade               4          11
C Grade               5           3
D Grade               3           0
Fails/ATKT            3          10
```

- **Accessing Elements of Matrix:**

```
> # Accessing Elements of Matrix
> A<-matrix(c(1,0,-1,2,3,6,1,2,0),nrow=3,byrow=TRUE)
> print(A)
     [,1] [,2] [,3]
[1,]    1    0   -1
[2,]    2    3    6
[3,]    1    2    0
> # Accessing Element at 2nd Row and 3rd Column
> print(A[2,3])
[1] 6
> # Accessing Element at 3rd Row and 2nd Column
> print(A[3,2])
[1] 2
> # Accessing Element in 2nd Row
> print(A[2,])
[1] 2 3 6
> # Accessing Element in 3rd Column
> print(A[,3])
[1] -1  6  0
```

6. R – Arrays:

- **Creating R – Array:**

```
> # Creating R-Array
> v1<-c(1,2,-1)
> v2<-c(3,2,6,-1,0,2)
> A<-array(c(v1,v2),dim=c(3,3,2))
> print(A)
, , 1

     [,1] [,2] [,3]
[1,]    1    3   -1
[2,]    2    2    0
[3,]   -1    6    2

, , 2

     [,1] [,2] [,3]
[1,]    1    3   -1
[2,]    2    2    0
[3,]   -1    6    2
```

- **Naming Dimensions of Array**

```
> # Naming Dimensions of Array
> v1<-c(2,4,6,3)
> v2<-c(1,0,2,3,-6,11,1,2)
> rowName<-c('R1','R2','R3','R4')
> colName<-c('C1','C2','C3')
> matName<-c('M1','M2')
> B<-array(c(v1,v2),dim=c(4,3,2),dimnames=list(rowName,colName,matName))
> print(B)
, , M1

   C1 C2 C3
R1  2  1 -6
R2  4  0 11
R3  6  2  1
R4  3  3  2

, , M2

   C1 C2 C3
R1  2  1 -6
R2  4  0 11
R3  6  2  1
R4  3  3  2
```

- **Accessing Elements of Array:**

```
> # Accessing Elements of Array
> A<-array(c(3,2,-1,1),dim=c(2,3,2))
> print(A)
, , 1

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

, , 2

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

> # Accessing Element at 1st Row and 2nd column of 2nd Matrix
> print(A[1,2,2])
[1] 3
> # Accessing Element at 2nd Row of 1st Matrix
> print(A[2,,1])
[1] 2 1 2
> # Accessing Element at 3rd Column of 2nd Matrix
> print(A[,3,2])
[1] -1  1
> # Accessing 2nd Matrix
> print(A[,,2])
     [,1] [,2] [,3]
[1,]   -1    3   -1
[2,]    1    2    1
```

7. R – Factors:

- **Creating Factors and Finding Number of Distinct Values:**

```
> age<-c(19,19,20,21,19,26,27,19,18,18,20,20,22,21,22,19,18,26,21)
> # Creating Factor Object
> factorAge<-factor(age)
> print(factorAge)
 [1] 19 19 20 21 19 26 27 19 18 18 20 20 22 21 22 19 18 26 21
Levels: 18 19 20 21 22 26 27
> print(nlevels(factorAge))
[1] 7
> # applying the nlevels function to factor object we can find the number of distinct values
```

8. R – Data Frames:

- **Creating R – Data Frames:**

```
> # Creating Data Frame
> Info<-data.frame(Name=c('Suresh','Ganesh','Mahesh','Dinesh'),
+      Age=c(27,23,25,20),
+      Salary=c(45000,17000,29000,15000))
> print(Info)
    Name Age Salary
1 Suresh  27  45000
2 Ganesh  23  17000
3 Mahesh  25  29000
4 Dinesh  20  15000
```

- **Getting Structure of Data Frame**

```
> # Creating Data Frame
> employee<-data.frame(emp_id=c(1001,1002,1003,1004,1005),
+    emp_name=c('Sadik','Pinky','Manoj','Krishna','Sonam'),
+    salary=c(56000,49000,45000,35000,28000),
+    DOJ=as.Date(c('2012-08-07','2013-01-15','2013-06-08',
+ '2014-11-10','2015-06-05')),
+    stringsAsFactors=FALSE)
> print(employee)
  emp_id emp_name salary        DOJ
1   1001    Sadik  56000 2012-08-07
2   1002    Pinky  49000 2013-01-15
3   1003    Manoj  45000 2013-06-08
4   1004  Krishna  35000 2014-11-10
5   1005    Sonam  28000 2015-06-05
> # Getting structure of data frame with the help of str()
> str(employee)
'data.frame':   5 obs. of  4 variables:
 $ emp_id  : num  1001 1002 1003 1004 1005
 $ emp_name: chr  "Sadik" "Pinky" "Manoj" "Krishna" ...
 $ salary  : num  56000 49000 45000 35000 28000
 $ DOJ     : Date, format: "2012-08-07" "2013-01-15" ...
```

- **Getting Statistical Summary**

```
> # Creating Data Frame
> employee<-data.frame(emp_id=c(1001,1002,1003,1004,1005),
+    emp_name=c('Sadik','Pinky','Manoj','Krishna','Sonam'),
+    salary=c(56000,49000,45000,35000,28000),
+    DOJ=as.Date(c('2012-08-07','2013-01-15','2013-06-08',
+ '2014-11-10','2015-06-05')),
+    stringsAsFactors=FALSE)
> print(employee)
  emp_id emp_name salary        DOJ
1   1001     Sadik  56000 2012-08-07
2   1002     Pinky  49000 2013-01-15
3   1003     Manoj  45000 2013-06-08
4   1004   Krishna  35000 2014-11-10
5   1005     Sonam  28000 2015-06-05
> # Getting statistical summary of data frame with the help of summary()
> summary(employee)
     emp_id       emp_name             salary           DOJ
 Min.   :1001   Length:5           Min.   :28000   Min.   :2012-08-07
 1st Qu.:1002   Class :character   1st Qu.:35000   1st Qu.:2013-01-15
 Median :1003   Mode  :character   Median :45000   Median :2013-06-08
 Mean   :1003                      Mean   :42600   Mean   :2013-11-14
 3rd Qu.:1004                      3rd Qu.:49000   3rd Qu.:2014-11-10
 Max.   :1005                      Max.   :56000   Max.   :2015-06-05
```

- **Extracting Data from Data Frame:**

```
> # Creating Data Frame
> employee<-data.frame(emp_id=c(1001,1002,1003,1004,1005),
+    emp_name=c('Sadik','Pinky','Manoj','Krishna','Sonam'),
+    salary=c(56000,49000,45000,35000,28000),
+    DOJ=as.Date(c('2012-08-07','2013-01-15','2013-06-08',
+ '2014-11-10','2015-06-05')),
+    stringsAsFactors=FALSE)
> # Extracting emp_name and DOJ from employee
> print(data.frame(employee$emp_name,employee$DOJ))
  employee.emp_name employee.DOJ
1             Sadik   2012-08-07
2             Pinky   2013-01-15
3             Manoj   2013-06-08
4           Krishna   2014-11-10
5             Sonam   2015-06-05
> # Extracting emp_id and salary from employee
> print(employee[,c(1,3)])
  emp_id salary
1   1001  56000
2   1002  49000
3   1003  45000
4   1004  35000
5   1005  28000
> # Extracting first three rows from employee
> print(employee[1:3,])
  emp_id emp_name salary        DOJ
1   1001     Sadik  56000 2012-08-07
2   1002     Pinky  49000 2013-01-15
3   1003     Manoj  45000 2013-06-08
> # Extracting 2nd and 5th row with 2nd and 4th column
> print(employee[c(2,5),c(2,4)])
  emp_name        DOJ
2    Pinky 2013-01-15
5    Sonam 2015-06-05
```

- **Expanding Data Frame**

i. Adding Column:

```
> # Expanding Data Frames
> # Creating Data Frame
> employee<-data.frame(emp_id=c(1001,1002,1003,1004,1005),
+    emp_name=c('Sadik','Pinky','Manoj','Krishna','Sonam'),
+    salary=c(56000,49000,45000,35000,28000),
+    DOJ=as.Date(c('2012-08-07','2013-01-15','2013-06-08',
+ '2014-11-10','2015-06-05')),
+    stringsAsFactors=FALSE)
> # Adding Depatment Column to employee
> employee$Department<-c('Finance','HR','Operations','IT','IT')
> print(employee)
  emp_id emp_name salary        DOJ Department
1   1001     Sadik  56000 2012-08-07    Finance
2   1002     Pinky  49000 2013-01-15         HR
3   1003     Manoj  45000 2013-06-08 Operations
4   1004  Krishna  35000 2014-11-10         IT
5   1005     Sonam  28000 2015-06-05         IT
```

ii. Adding Rows:

```
> # Expanding Data Frames
> # Creating Data Frame
> employee<-data.frame(emp_id=c(1001,1002,1003,1004,1005),
+    emp_name=c('Sadik','Pinky','Manoj','Krishna','Sonam'),
+    salary=c(56000,49000,45000,35000,28000),
+    DOJ=as.Date(c('2012-08-07','2013-01-15','2013-06-08',
+ '2014-11-10','2015-06-05')),
+    Department=c('Finance','HR','Operations','IT','IT'),
+    stringsAsFactors=FALSE)
> # Adding Rows to employee using rbind()
> # Creating Second Data Frame
> employeeNew<-data.frame(emp_id=c(1006,1007,1008),
+    emp_name=c('Shruti','Pawan','Raj'),
+    salary=c(46000,34000,32000),
+    DOJ=as.Date(c('2015-10-07','2015-10-15','2014-01-08')),
+    Department=c('Finance','Operations','IT'),
+    stringsAsFactors=FALSE)
> #Binding Data Frames
> employee2<-rbind(employee,employeeNew)
> print(employee2)
  emp_id emp_name salary        DOJ Department
1   1001     Sadik  56000 2012-08-07    Finance
2   1002     Pinky  49000 2013-01-15         HR
3   1003     Manoj  45000 2013-06-08 Operations
4   1004  Krishna  35000 2014-11-10         IT
5   1005     Sonam  28000 2015-06-05         IT
6   1006    Shruti  46000 2015-10-07    Finance
7   1007     Pawan  34000 2015-10-15 Operations
8   1008       Raj  32000 2014-01-08         IT
```

PRACTICAL No. 2

MATRIX COMPUTATIONS

AIM: Create a Matrix using R and Perform the operations addition, inverse, transpose and multiplication operations.

SOURCE CODE & OUTPUT:

```
> A<-matrix(c(3,2,-1,0,2,6,1,2,1),nrow=3)
> B<-matrix(c(1,0,-1,3,2,6,0,-2,-1),nrow=3)
> # Matrix Addition
> print(A+B)
     [,1] [,2] [,3]
[1,]    4    3    1
[2,]    2    4    0
[3,]   -2   12    0
> # Matrix Subtraction
> print(A-B)
     [,1] [,2] [,3]
[1,]    2   -3    1
[2,]    2    0    4
[3,]    0    0    2
> # Matrix Multiplication
> print(A%*%B)
     [,1] [,2] [,3]
[1,]    2   15   -1
[2,]    0   22   -6
[3,]   -2   15  -13
> # Matrix Transpose
> print(t(A))
     [,1] [,2] [,3]
[1,]    3    2   -1
[2,]    0    2    6
[3,]    1    2    1
> # Matrix Inverse
> print(solve(A))
       [,1]   [,2]   [,3]
[1,]  0.625 -0.375  0.125
[2,]  0.250 -0.250  0.250
[3,] -0.875  1.125 -0.375
```

PRACTICAL No. 3

STATISTICAL FUNCTIONS

Mean, Median, Mode, Quartiles, Range,

Inter-Quartile Range & Histogram

AIM: Using R Execute the statistical functions: mean, median, mode, quartiles, range, inter quartile range, histogram.

SOURCE CODE & OUTPUT:

1. Mean:

```
> # Creating Vector
> x<-c(84,91,72,68,87,78)
> # Finding Mean
> print(mean(x))
[1] 80
> # Creating Vector
> y<-c(2,3,4,11,14,17,23,25,27,28,80,84,88)
> # Finding Mean
> print(mean(y))
[1] 31.23077
> # Using trim Option
> print(mean(y,trim=0.3))
[1] 20.71429
> # Creating Vector
> z<-c(11,12,36,17,19,25,34,47,9,22,NA)
> # Finding Mean
> print(mean(z))
[1] NA
> # Using na.rm Option
> print(mean(z,na.rm=TRUE))
[1] 23.2
```

2. Median:

```
> # Creating Vector
> x<-c(84,91,72,68,87,78)
> # Finding Mean
> print(median(x))
[1] 81
> # Creating Vector
> y<-c(2,3,4,11,14,17,23,25,27,28,80,84,88)
> # Finding Mean
> print(median(y))
[1] 23
> # Creating Vector
> z<-c(11,12,36,17,19,25,34,47,9,22,NA)
> # Finding Mean
> print(median(z))
[1] NA
> # Using na.rm Option
> print(median(z,na.rm=TRUE))
[1] 20.5
```

3. Mode:

```
> # Creating getMode function
> getMode<-function(x)
+ {
+ u<-unique(x)
+ u[which.max(tabulate(match(x,u)))]
+ }
> # Creating Vector with Numeric Values
> x<-c(11,14,17,16,16,16,17,17,13,13,13,13)
> getMode(x)
[1] 13
> # Creating Vector with Character Values
> y<-c('IT','IT','CS','PM','CS','OS','IT','PM')
> getMode(y)
[1] "IT"
```

4. Quartiles:

```
> # Creating Vector
> v<-c(11,12,36,17,19,25,34,47,9,22)
> # Finding First Quartile
> Q1<-quantile(v,prob=0.25)
> cat('First quartile is:',Q1,'\n')
First quartile is: 13.25
> # Finding Second Quartile
> Q2<-quantile(v,prob=0.5)
> cat('Second quartile is:',Q2,'\n')
Second quartile is: 20.5
> # Finding Third Quartile
> Q3<-quantile(v,prob=0.75)
> cat('Third quartile is:',Q3,'\n')
Third quartile is: 31.75
> # Finding Quantiles
> quantile(v)
   0%   25%   50%   75%  100%
 9.00 13.25 20.50 31.75 47.00
```

5. Range:

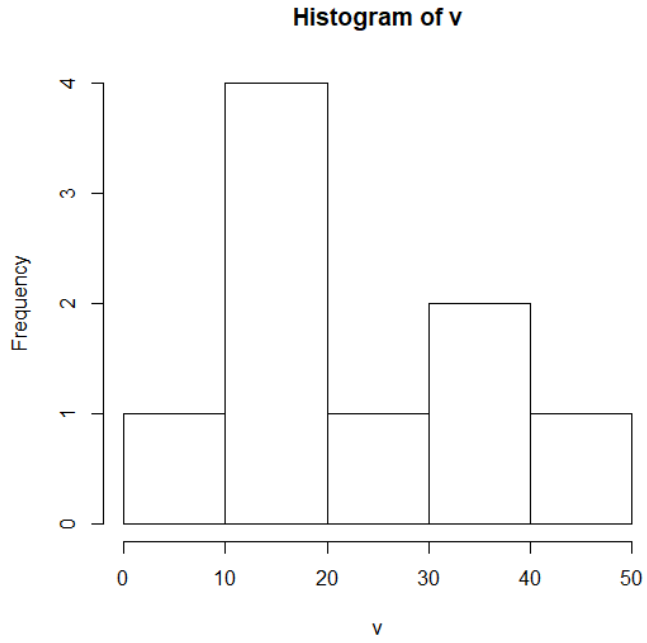
```
> v<-c(11,12,36,17,19,25,34,47,9,22)
> # Finding Range
> range<-max(v)-min(v)
> cat('Range is:',range,'\n')
Range is: 38
```

6. Inter-Quartile Range:

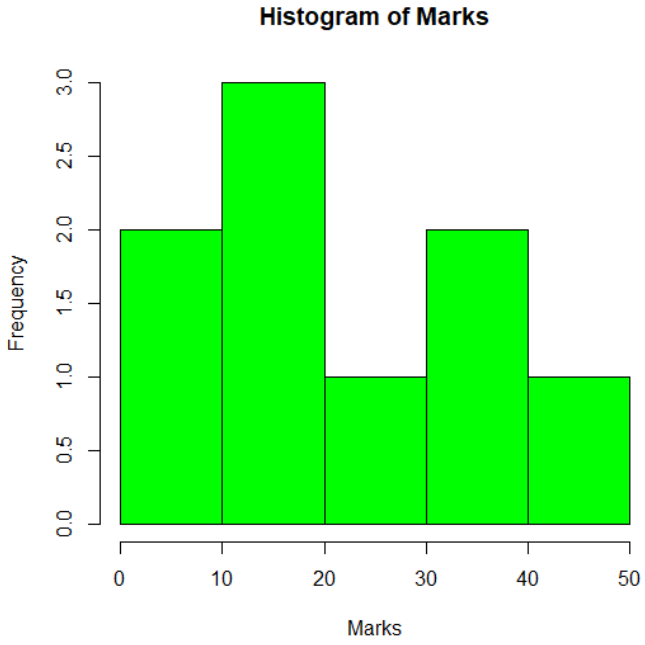
```
> # Creating Vector
> v<-c(11,12,36,17,19,25,34,47,9)
> # Finding Quartile Range
> quantile(v)
  0%  25%  50%  75% 100%
   9   12   19   34   47
> # Finding Inter-Quartile Range
> cat('Inter-Quartile Range is',IQR(v),'\n')
Inter-Quartile Range is 22
```

7. Histogram:

```
> # Creating Vector
> v<-c(11,12,36,17,19,25,34,47,9)
> # Creating Histogram
> hist(v)
```



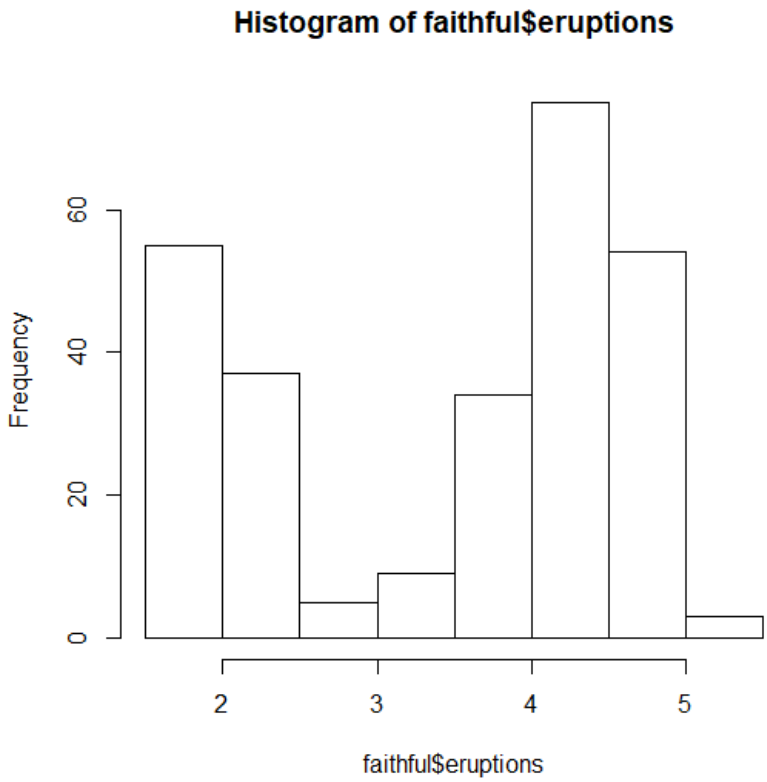

```
> # Creating Vector
> v<-c(8,12,36,17,19,25,34,47,9)
> # Creating Histogram with Various Available Options
> hist(v,main='Histogram of Marks',xlab='Marks',col='green')
```



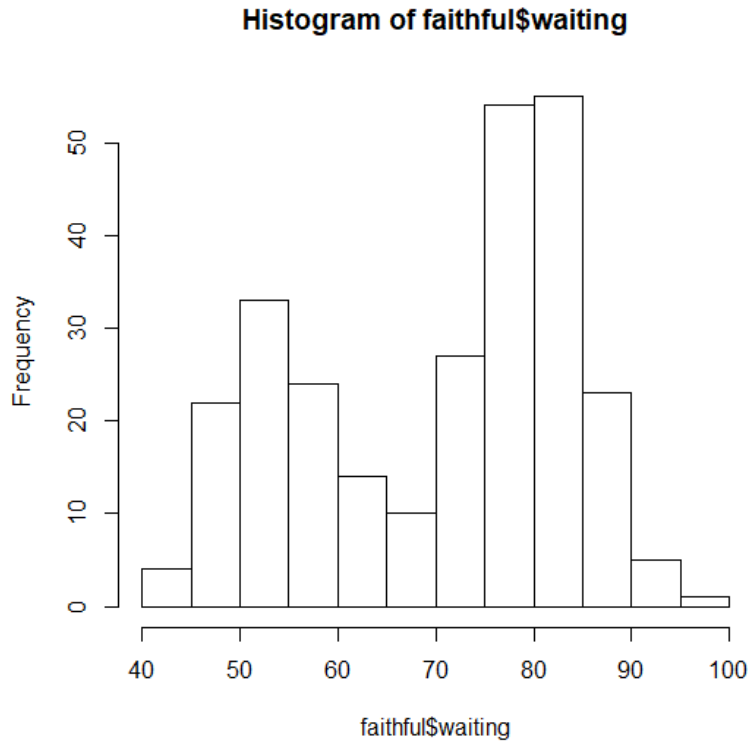
8. Performing Above Statistical Functions on 'faithful' Dataset:

```
> # Finding Mean of eruptions column of faithful dataset
> mean(faithful$eruptions)
[1] 3.487783
> # Finding Mean of waiting column of faithful dataset
> mean(faithful$waiting)
[1] 70.89706
> # Finding Median of eruptions column of faithful dataset
> median(faithful$eruptions)
[1] 4
> # Finding Median of waiting column of faithful dataset
> median(faithful$waiting)
[1] 76

> # Finding First and Third Quartile of eruptions column of faithful dataset
> quantile(faithful$eruptions,prob=0.25)
    25%
2.16275
> quantile(faithful$eruptions,prob=0.75)
    75%
4.45425
> # Finding First and Third Quartile of waiting column of faithful dataset
> quantile(faithful$waiting,prob=0.25)
25%
 58
> quantile(faithful$waiting,prob=0.75)
75%
 82
> # Finding Range of eruptions column of faithful dataset
> max(faithful$eruptions)-min(faithful$eruptions)
[1] 3.5
> # Finding Range of waiting column of faithful dataset
> max(faithful$waiting)-min(faithful$waiting)
[1] 53

> # Creating getMode function
> getMode<-function(x)
+ {
+ u<-unique(x)
+ u[which.max(tabulate(match(x,u)))]
+ }
> # Finding Mode of eruption column of faithful Dataset
> getMode(faithful$eruption)
[1] 1.867
> # Finding Mode of waiting column of faithful Dataset
> getMode(faithful$waiting)
[1] 78

> # Finding Inter-Quartile Range of eruptions column of faithful dataset
> IQR(faithful$eruptions)
[1] 2.2915
> # Finding Inter-Quartile Range of eruptions column of faithful dataset
> IQR(faithful$waiting)
[1] 24
```

```
> hist(faithful$eruptions)
```




```
> hist(faithful$waiting)
```



PRACTICAL No. 4

FINDING MEAN, MEDIAN, MODE, QUARTILES,

RANGE, INTER-QUARTILE RANGE,

& HISTOGRAM

OF EXCEL/.CSV DATA

AIM: Using R import the data from Excel/.CSV file and find mean, median, mode, quartiles, range, inter quartile range, histogram.

SOURCE CODE & OUTPUT:

- **Working with CSV File:**

1. **Copy and paste .csv file in working directory.**

2. **Importing Data from .CSV File:**

```
> emp <- read.csv("employee.csv")
> print(emp)
   Emp_Id Emp_Name        DOJ Salary Department
1    1001     Sadik 07-06-2012  47000    Finance
2    1002     Pinky 15-11-2012  45000         HR
3    1003     Manoj 03-03-2013  43000 Operations
4    1004      Aman 27-08-2013  38000         IT
5    1005     Sonam 15-12-2013  29000      Admin
6    1006    Rajesh 04-10-2014  23000      Admin
7    1007    Ramesh 04-10-2014  41000         HR
8    1008   Radhika 07-10-2014  40000 Operations
9    1009    Manish 17-10-2014  25000         IT
10   1010    Ritika 17-10-2014  33000         HR
11   1011     Aryan 28-10-2014  28000 Operations
12   1012      Ayan 28-10-2014  39000    Finance
13   1013    Suyash 07-11-2014  23000         IT
14   1014    Naresh 07-11-2014  27000      Admin
15   1015     Jyoti 09-11-2014  25000      Admin
```

3. Finding Mean, Median, Range, Quartiles, Inter-Quartile Range:

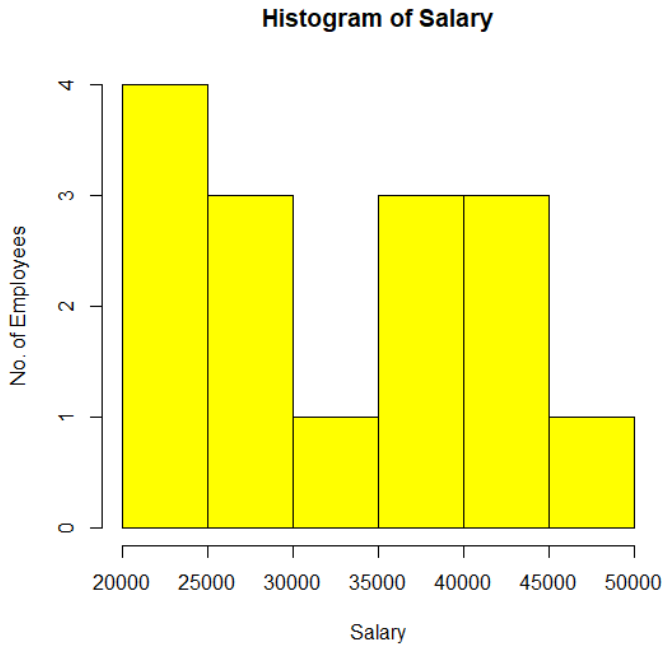
```
> # Finding Mean
> cat('Mean Salary =',mean(emp$Salary),'\n')
Mean Salary = 33733.33
> # Finding Meadian
> cat('Median Salary =',median(emp$Salary),'\n')
Median Salary = 33000
> # Finding Range
> cat('Range of Salary =',max(emp$Salary)-min(emp$Salary),'\n')
Range of Salary = 24000
> # Finding Quartile
> cat('First Quartile =',quantile(emp$Salary,prob=0.25),'\n')
First Quartile = 26000
> cat('Third Quartile =',quantile(emp$Salary,prob=0.75),'\n')
Third Quartile = 40500
> # Finding Inter-Quartile Range
> cat('Inter-Quartile Range =',IQR(emp$Salary),'\n')
Inter-Quartile Range = 14500
```

4. Finding Mode:

```
> # Creating getMode function
> getMode<-function(x)
+ {
+ u<-unique(x)
+ u[which.max(tabulate(match(x,u)))]
+ }
> # Finding Mode
> cat('Mode of Salary =',getMode(emp$Salary),'\n')
Mode of Salary = 23000
```

5. Histogram:

```
> hist(emp$Salary,main='Histogram of Salary',xlab='Salary',
+ ylab='No. of Employees',col='Yellow')
```



- **Working with Excel File:**

1. **Copy and paste .xlsx file in working directory.**

2. **Installing xlsx Package:**

```
> install.packages('xlsx')
```

3. **Importing Data from .xlsx File:**

```
> emp2<-read.xlsx('employee.xlsx',sheetIndex=1)
> print(emp2)
   Emp_Id Emp_Name        DOJ Salary Department
1    1001     Sadik 2012-06-07  47000    Finance
2    1002     Pinky 2012-11-15  45000         HR
3    1003     Manoj 2013-03-03  43000 Operations
4    1004      Aman 2013-08-27  38000         IT
5    1005     Sonam 2013-12-15  29000      Admin
6    1006    Rajesh 2014-10-04  23000      Admin
7    1007    Ramesh 2014-10-04  41000         HR
8    1008   Radhika 2014-10-07  40000 Operations
9    1009    Manish 2014-10-17  25000         IT
10   1010    Ritika 2014-10-17  33000         HR
11   1011     Aryan 2014-10-28  28000 Operations
12   1012      Ayan 2014-10-28  39000    Finance
13   1013    Suyash 2014-11-07  23000         IT
14   1014    Naresh 2014-11-07  27000      Admin
15   1015     Jyoti 2014-11-09  25000      Admin
```

4. **Finding Mean, Median, Range, Quartiles, Inter-Quartile Range:**

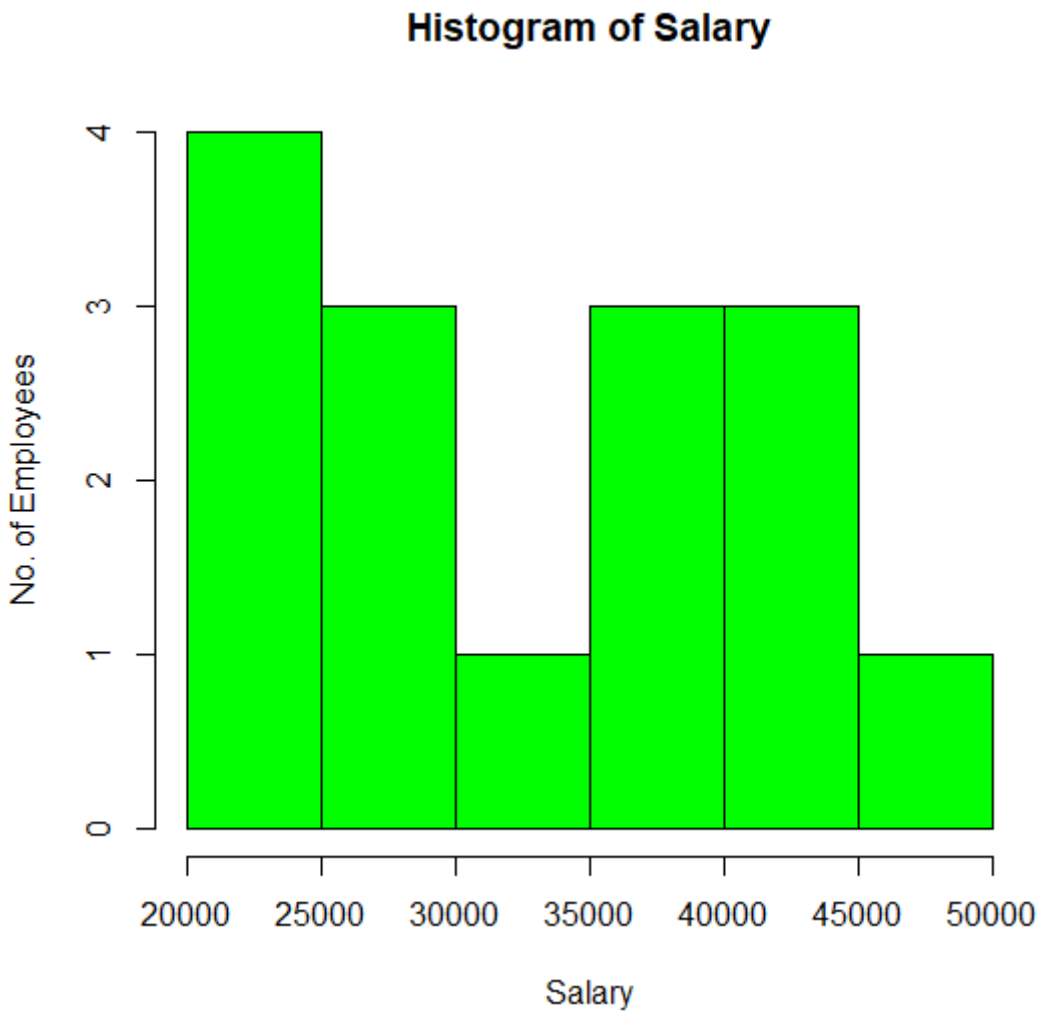
```
> # Finding Mean
> cat('Mean Salary =',mean(emp2$Salary),'\n')
Mean Salary = 33733.33
> # Finding Median
> cat('Median Salary =',median(emp2$Salary),'\n')
Median Salary = 33000
> # Finding Range
> cat('Range of Salary =',max(emp2$Salary)-min(emp2$Salary),'\n')
Range of Salary = 24000
> # Finding Quartile
> cat('First Quartile =',quantile(emp2$Salary,prob=0.25),'\n')
First Quartile = 26000
> cat('Third Quartile =',quantile(emp2$Salary,prob=0.75),'\n')
Third Quartile = 40500
> # Finding Inter-Quartile Range
> cat('Inter-Quartile Range=',IQR(emp2$Salary),'\n')
Inter-Quartile Range= 14500
```

5. Finding Mode:

```
> # Creating getMode function
> getMode<-function(x)
+ {
+ u<-unique(x)
+ u[which.max(tabulate(match(x,u)))]
+ }
> # Finding Mode
> cat('Mode of Salary =',getMode(emp2$Salary),'\n')
Mode of Salary = 23000
```

6. Histogram:

```
> hist(emp2$Salary,main='Histogram of Salary',xlab='Salary',ylab='No. of Employees',col='green')
```



PRACTICAL No. 5

FINDING STANDARD DEVIATION,

VARIANCE & CO-VARIANCE

OF EXCEL/.CSV DATA

AIM: Using R import the data from Excel/.CSV file and find standard deviation, variance and co-variance.

SOURCE CODE & OUTPUT:

- **Working with Excel File:**

1. **Copy and paste .xlsx file in working directory.**

2. **Importing Data from .xlsx File:**

```
> data<-read.xlsx('marks.xlsx',sheetIndex=1)
> print(data)
   Roll_No Maths Stats
1        1    56    73
2        2    43    55
3        3    35    50
4        4    47    52
5        5    55    75
6        6    41    49
7        7    65    69
8        8    39    45
9        9    72    77
10      10    44    51
```

3. **Finding Standard Deviation, Variance & Co-Variance:**

```
> # Finding Standard Deviation
> sd(data$Maths)
[1] 11.97265
> sd(data$Stats)
[1] 12.3756
> # Finding Variance
> var(data$Maths)
[1] 143.3444
> var(data$Stats)
[1] 153.1556
> # Finding Co-Variance
> cov(data$Maths,data$Stats)
[1] 131.9778
```

4. **Finding Standard Deviation, Variance & Co-Variance for given x & y:**

```
> x<-c(11,23,34,39,41,46,57,69)
> y<-c(73,65,61,56,53,44,34,18)
> sd(x)
[1] 18.26003
> sd(y)
[1] 17.86457
> var(x)
[1] 333.4286
> var(y)
[1] 319.1429
> cov(x,y)
[1] -318.1429
```

5. **Finding Standard Deviation, Variance & Co-Variance from faithful Dataset:**

```
> sd(faithful$eruptions)
[1] 1.141371
> sd(faithful$waiting)
[1] 13.59497
> var(faithful$eruptions)
[1] 1.302728
> var(faithful$waiting)
[1] 184.8233
> cov(faithful$eruptions,faithful$waiting)
[1] 13.97781
```

PRACTICAL No. 6

FINDING SKEWNESS & KURTOSIS

OF EXCEL/.CSV DATA

AIM: Using R import the data from Excel/.CSV file and find skewness and kutosis.

SOURCE CODE & OUTPUT:

- **Working with Excel File:**

1. **Copy and paste .xlsx file in working directory.**
2. **Installing moments package:**

```
> install.packages("moments")
```

3. **Importing Data from .xlsx File:**

```
> data <- read.xlsx("marks.xlsx", sheetIndex = 1)
> print(data)
   Roll_No Maths Stats
1        1    54    25
2        2    53    26
3        3    21    31
4        4    26    27
5        5    35    29
6        6    89    25
7        7    54    26
8        8    26    35
9        9    27    94
10      10    29    86
11      11    35    54
12      12    64    98
13      13    85    75
14      14    64    62
15      15    91    78
```

4. **Finding Skewness and Kurtosis:**

```
> library("moments")
> skewness(data$Maths)
[1] 0.4714073
> skewness(data$Stats)
[1] 0.484077
> kurtosis(data$Maths)
[1] 1.882526
> kurtosis(data$Stats)
[1] 1.592091
```

5. Finding Skewness and Kurtosis for given x:

```
> x<-c(1.25,3.15,2.27,3.16,1.56,1.85,2.99,3.36,2)
> skewness(x)
[1] -0.1042163
> kurtosis(x)
[1] 1.482559
```

6. Finding Skewness and Kurtosis from faithful Dataset:

```
> skewness(faithful$eruptions)
[1] -0.415841
> skewness(faithful$waiting)
[1] -0.4163188
> kurtosis(faithful$eruptions)
[1] 1.4994
> kurtosis(faithful$waiting)
[1] 1.857369
```

PRACTICAL No. 6

HYPOTHESIS TESTING

AIM: Perform hypothesis testing for the following:

Q.1 The mean breaking strength of cables produced by a manufacturer have a mean of 1800 lb and a standard deviation of 100 lb. A sample of 50 cables is tested and it is found that the mean breaking strength is 1780 lb. Test the hypothesis that the mean breaking strength of the cables has decreased at 0.05 significance level.

```
> # Left Tail Problem
> # H0: mu=1800 vs H1: mu<1800
> mu<-1800
> sigma<-100
> n<-50
> xbar<-1780
> zCal<-(xbar-mu)/(sigma/sqrt(n))
> print(zCal)
[1] -1.414214
> alpha<-0.05
> zTab<-qnorm(1-alpha)
> print(zTab)
[1] 1.644854
> if(abs(zCal)<zTab)
+ {
+ print('Accept H0.')
+ print('Breaking strength is not decreased.')
+ }else
+ {
+ print('Reject H0.')
+ print('Breaking strength is decreased.')
+ }
[1] "Accept H0."
[1] "Breaking strength is not decreased."
```

Q.2 The mean breaking strength of cables produced by a manufacturer have a mean of 1800 lb and a standard deviation of 100 lb. A sample of 50 cables is tested and it is found that the mean breaking strength is 1850 lb. Test the hypothesis that the mean breaking strength of the cables has increased at 0.05 significance level.

```
> # Right Tail Problem
> # H0: mu=1800 vs H1: mu>1800
> mu<-1800
> sigma<-100
> n<-50
> xbar<-1850
> zCal<-(xbar-mu)/(sigma/sqrt(n))
> print(zCal)
[1] 3.535534
> alpha<-0.05
> zTab<-qnorm(1-alpha)
> print(zTab)
[1] 1.644854
> if(zCal<zTab)
+ {
+ print('Accept H0.')
+ print('Breaking strength is not increased.')
+ }else
+ {
+ print('Reject H0.')
+ print('Breaking strength is increased.')
+ }
[1] "Reject H0."
[1] "Breaking strength is increased."
```

Q.3 The mean breaking strength of cables produced by a manufacturer have a mean of 1800 lb and a standard deviation of 100 lb. A sample of 50 cables is tested and it is found that the mean breaking strength is 1850 lb. Test the hypothesis that the mean breaking strength of the cables has changed at 0.05 significance level.

```
> # Two Tailed Problem
> # H0: mu=1800 vs H1: mu!=1800
> mu<-1800
> sigma<-100
> n<-50
> xbar<-1780
> zCal<-(xbar-mu)/(sigma/sqrt(n))
> print(zCal)
[1] -1.414214
> alpha<-0.05
> zTab<-qnorm(1-alpha/2)
> print(zTab)
[1] 1.959964
> if(abs(zCal)<zTab)
+ {
+ print('Accept H0.')
+ print("Breaking strength can't be increased.")
+ }else
+ {
+ print('Reject H0.')
+ print('Breaking strength can be increased.')
+ }
[1] "Accept H0."
[1] "Breaking strength can't be increased."
```

Q.4 The mean lifetime of electric light bulbs produced by a company has in the past been 1120h with a standard deviation of 125h. A sample of 8 electric bulbs recently chosen from supply of newly produced bulb showed a mean lifetime of 1030h. Test the hypothesis that the mean lifetime of the bulb has not changed at 0.05 significance level.

```
> # Student t-Test
> # Two Tailed Problem
> # H0: mu=1120 vs H1: mu!=1120
> mu<-1120
> sigma<-125
> n<-8
> xbar<-1030
> tCal<-(xbar-mu)/(sigma/sqrt(n-1))
> print(tCal)
[1] -1.904941
> alpha<-0.05
> df<-n-1
> tTab<-qt(1-alpha/2,df)
> print(tTab)
[1] 2.364624
> if(abs(tCal)<tTab)
+ {
+ print('Accept H0.')
+ print('Mean lifetime of the bulb has not changed.')
+ }else
+ {
+ print('Reject H0.')
+ print('Mean lifetime of the bulb has changed.')
+ }
[1] "Accept H0."
[1] "Mean lifetime of the bulb has not changed."
```

Q.5 The mean lifetime of electric light bulbs produced by a company has in the past been 1120h with a standard deviation of 125h. A sample of 8 electric bulbs recently chosen from supply of newly produced bulb showed a mean lifetime of 1030h. Test the hypothesis that the mean lifetime of the bulb has decreased at 0.05 significance level.

```
> # Student t-Test
> # Left Tailed Problem
> # H0: mu=1120 vs H1: mu<1120
> mu<-1120
> sigma<-125
> n<-8
> xbar<-1030
> tCal<-(xbar-mu)/(sigma/sqrt(n-1))
> print(tCal)
[1] -1.904941
> alpha<-0.05
> df<-n-1
> tTab<-qt(1-alpha,df)
> print(tTab)
[1] 1.894579
> if(abs(tCal)<tTab)
+ {
+ print('Accept H0.')
+ print('Mean lifetime of the bulb has not decreased.')
+ }else
+ {
+ print('Reject H0.')
+ print('Mean lifetime of the bulb has decreased.')
+ }
[1] "Reject H0."
[1] "Mean lifetime of the bulb has decreased."
```

Q.6 The mean lifetime of electric light bulbs produced by a company has in the past been 1120h with a standard deviation of 125h. A sample of 8 electric bulbs recently chosen from supply of newly produced bulb showed a mean lifetime of 1200h. Test the hypothesis that the mean lifetime of the bulb has decreased at 0.05 significance level.

```
> # Student t-Test
> # Right Tailed Problem
> # H0: mu=1120 vs H1: mu>1120
> mu<-1120
> sigma<-125
> n<-8
> xbar<-1200
> tCal<-(xbar-mu)/(sigma/sqrt(n-1))
> print(tCal)
[1] 1.693281
> alpha<-0.05
> df<-n-1
> tTab<-qt(1-alpha,df)
> print(tTab)
[1] 1.894579
> if(abs(tCal)<tTab)
+ {
+ print('Accept H0.')
+ print('Mean lifetime of the bulb has not increased.')
+ }else
+ {
+ print('Reject H0.')
+ print('Mean lifetime of the bulb has increased.')
+ }
[1] "Accept H0."
[1] "Mean lifetime of the bulb has not increased."
```