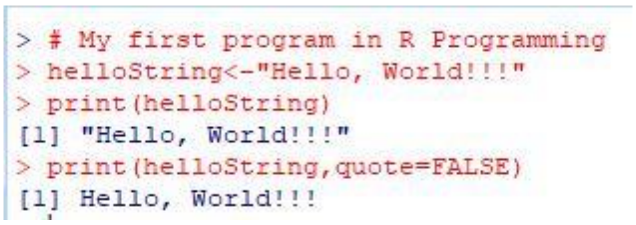
PRACTICAL NO. 1

BASICS OF R

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

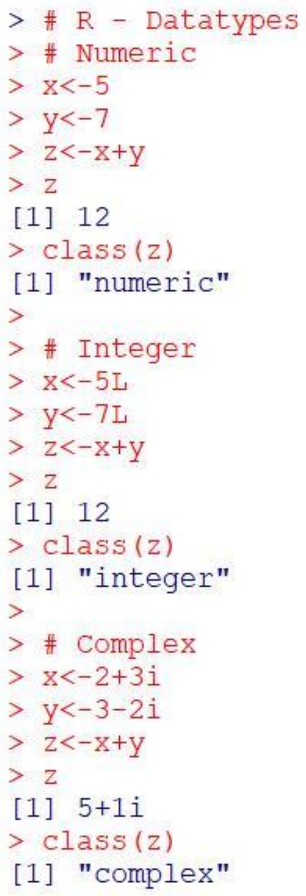
SOURCE CODE & OUTPUT:

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. **Numeric**
   ii. **Integer**
   iii. **Complex**
   iv. **Logical**
   v. **Character**

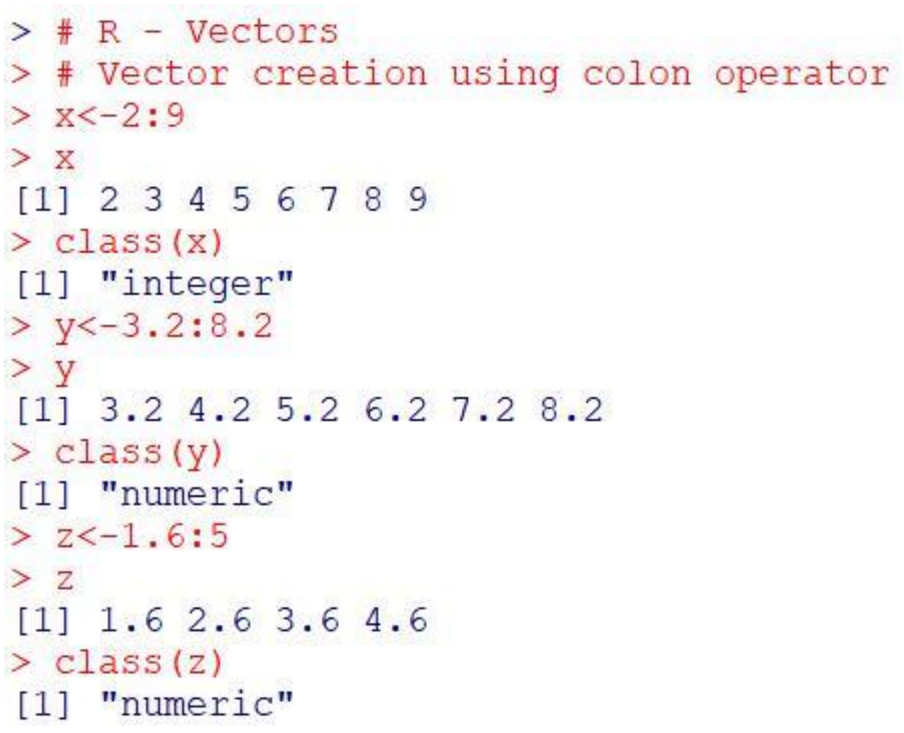


```
> # R - Datatypes
> # Numeric
> x<-5
> y<-7
> z<-x+y
> z
[1] 12
> class(z)
[1] "numeric"
>
> # Integer
> x<-5L
> y<-7L
> z<-x+y
> z
[1] 12
> class(z)
[1] "integer"
>
> # Complex
> x<-2+3i
> y<-3-2i
> z<-x+y
> z
[1] 5+1i
> class(z)
[1] "complex"
```

```
> # Logical
> x<-TRUE
> y<-FALSE
> z<-x&y
> z
[1] FALSE
> class(z)
[1] "logical"
> x<-T
> y<-F
> z<-x|y
> z
[1] TRUE
> class(z)
[1] "logical"
>
> # Character
> course<-"Bioinformatics"
> course
[1] "Bioinformatics"
> class(course)
[1] "character"
> x<-"TRUE"
> x
[1] "TRUE"
> class(x)
[1] "character"
```

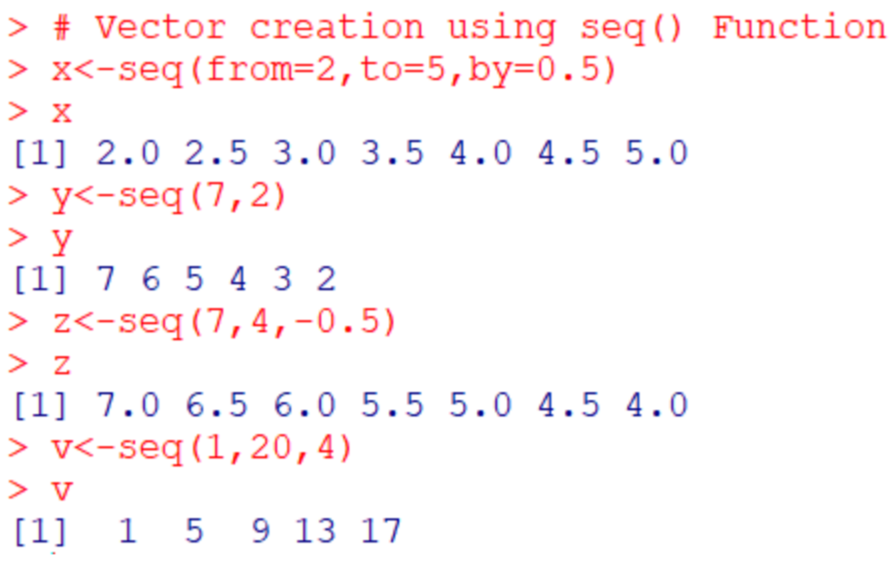
3. R – Vectors:

i. Vector Creation Using Colon Operator:

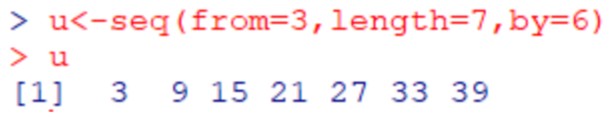


```
> # R - Vectors
> # Vector creation using colon operator
> x<-2:9
> x
[1] 2 3 4 5 6 7 8 9
> class(x)
[1] "integer"
> y<-3.2:8.2
> y
[1] 3.2 4.2 5.2 6.2 7.2 8.2
> class(y)
[1] "numeric"
> z<-1.6:5
> z
[1] 1.6 2.6 3.6 4.6
> class(z)
[1] "numeric"
```

ii. Vector Creation Using seq() Function:

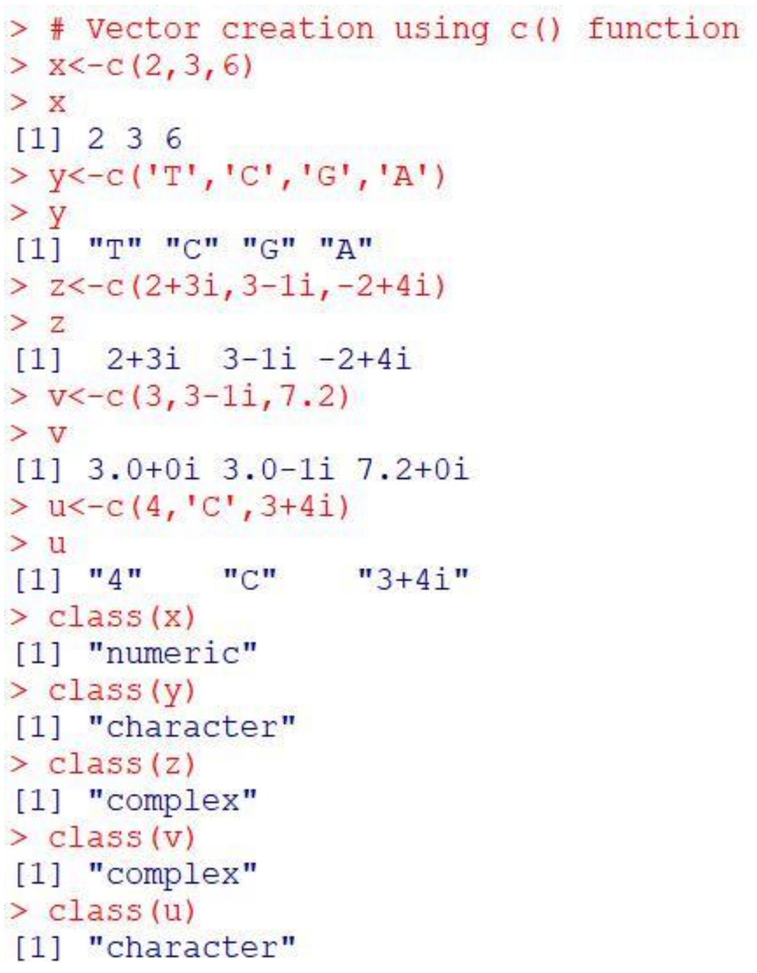


```
> # Vector creation using seq() Function
> x<-seq(from=2,to=5,by=0.5)
> x
[1] 2.0 2.5 3.0 3.5 4.0 4.5 5.0
> y<-seq(7,2)
> y
[1] 7 6 5 4 3 2
> z<-seq(7,4,-0.5)
> z
[1] 7.0 6.5 6.0 5.5 5.0 4.5 4.0
> v<-seq(1,20,4)
> v
[1]  1  5  9 13 17
```



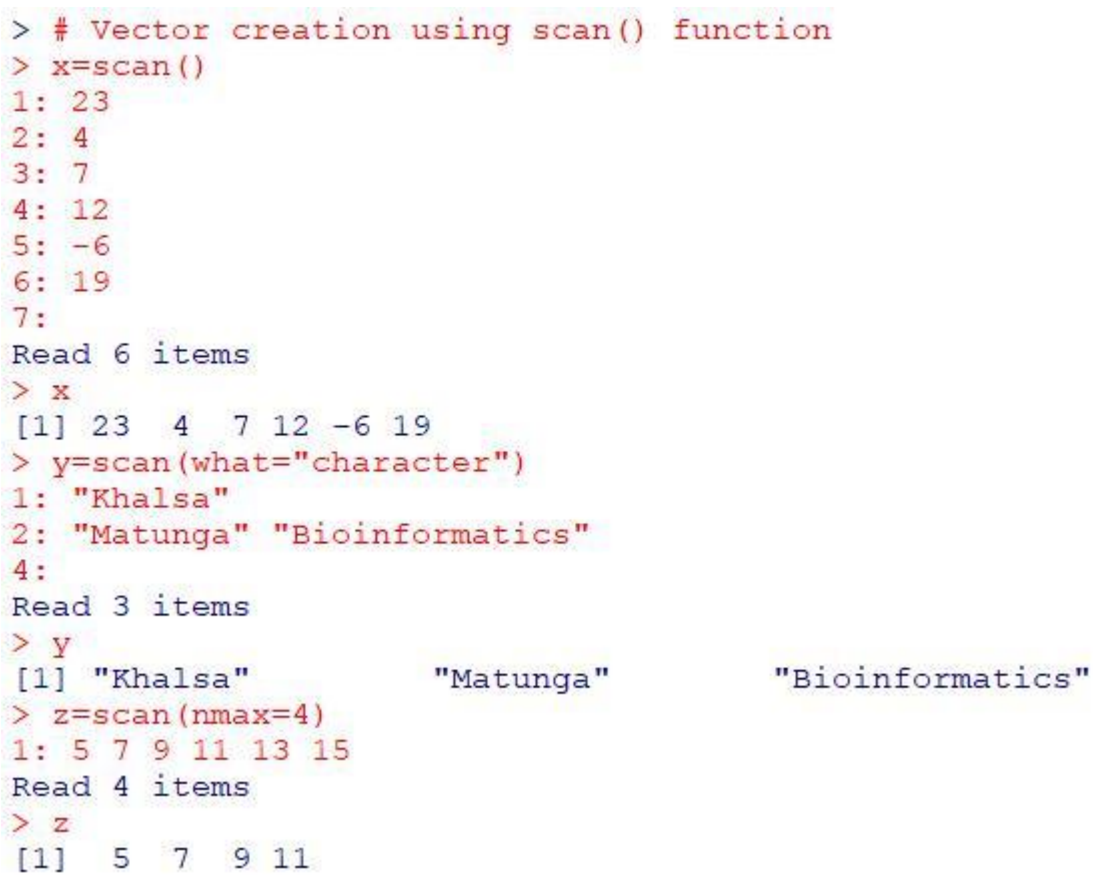
```
> u<-seq(from=3,length=7,by=6)
> u
[1]  3  9 15 21 27 33 39
```

iii. Vector Creation Using c() Function:



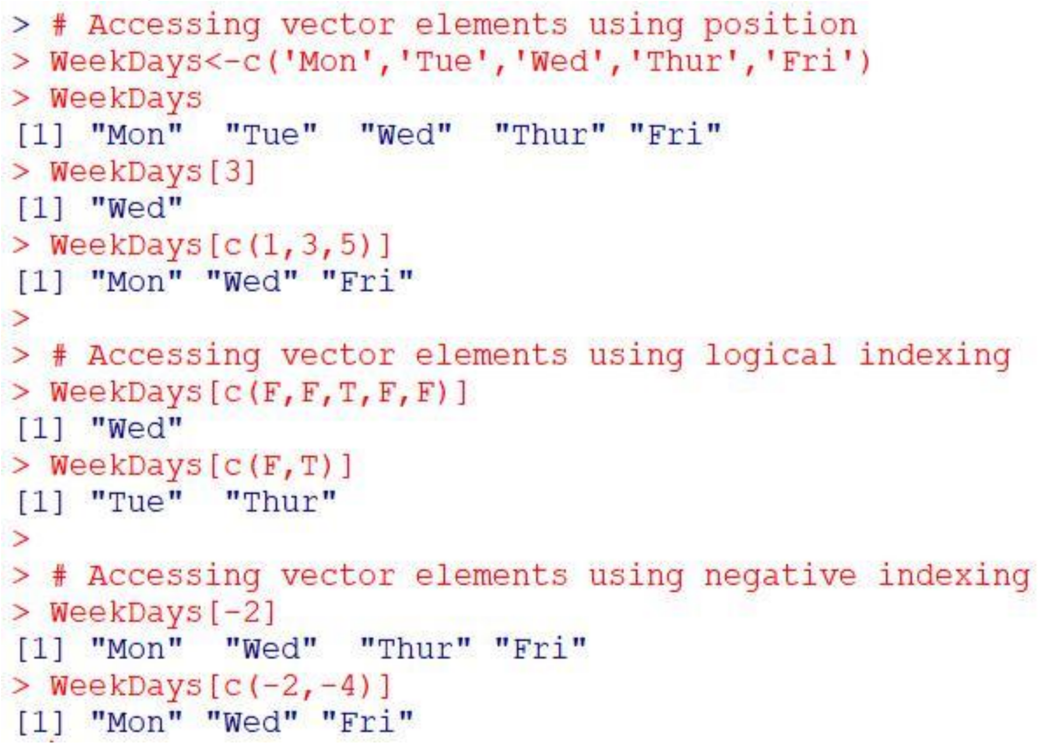
```
> # Vector creation using c() function
> x<-c(2,3,6)
> x
[1] 2 3 6
> y<-c('T','C','G','A')
> y
[1] "T" "C" "G" "A"
> z<-c(2+3i,3-1i,-2+4i)
> z
[1]  2+3i  3-1i -2+4i
> v<-c(3,3-1i,7.2)
> v
[1] 3.0+0i 3.0-1i 7.2+0i
> u<-c(4,'C',3+4i)
> u
[1] "4"    "C"    "3+4i"
> class(x)
[1] "numeric"
> class(y)
[1] "character"
> class(z)
[1] "complex"
> class(v)
[1] "complex"
> class(u)
[1] "character"
```

iv. Vector Creation Using scan() Function:



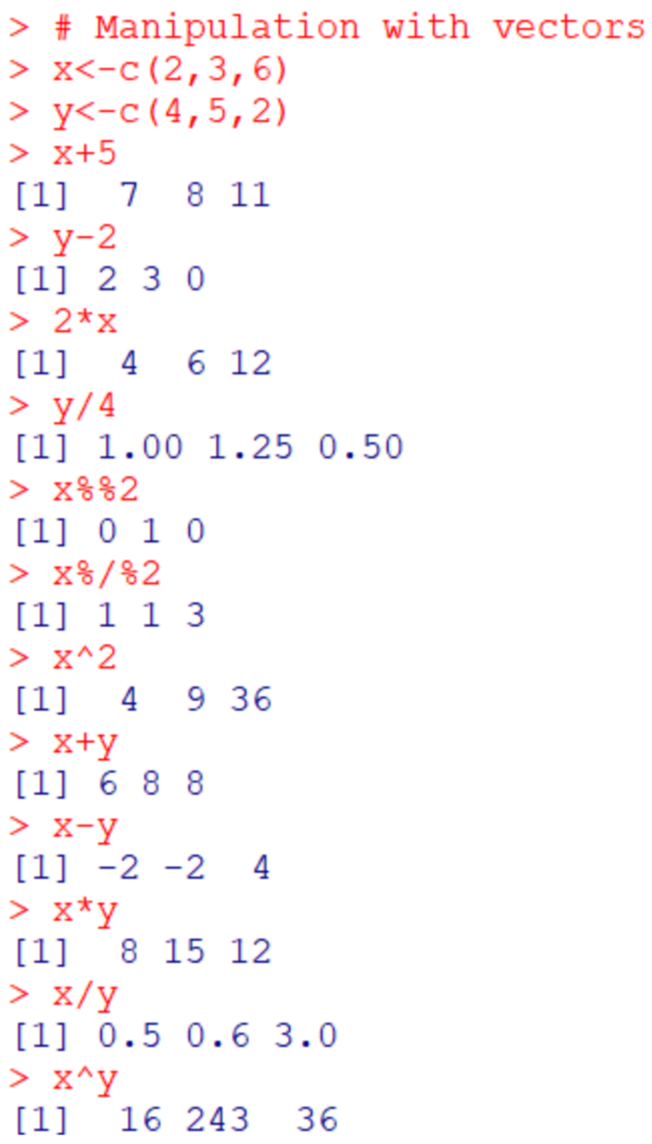
```
> # Vector creation using scan() function
> x=scan()
1: 23
2: 4
3: 7
4: 12
5: -6
6: 19
7:
Read 6 items
> x
[1] 23  4  7 12 -6 19
> y=scan(what="character")
1: "Khalsa"
2: "Matunga" "Bioinformatics"
4:
Read 3 items
> y
[1] "Khalsa"          "Matunga"          "Bioinformatics"
> z=scan(nmax=4)
1: 5 7 9 11 13 15
Read 4 items
> z
[1]  5  7  9 11
```

- **Accessing Vector Elements:**

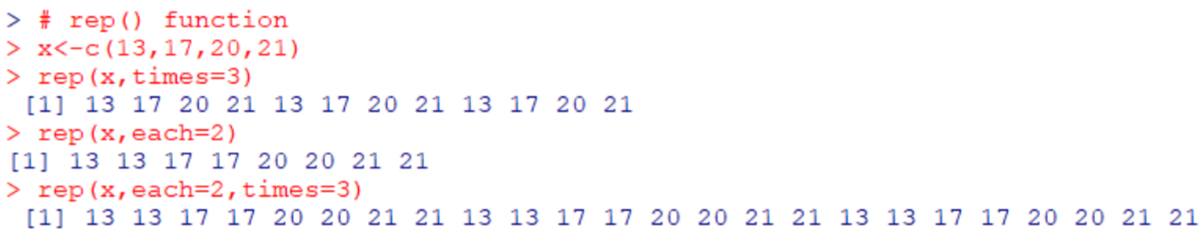


```
> # Accessing vector elements using position
> WeekDays<-c('Mon','Tue','Wed','Thur','Fri')
> WeekDays
[1] "Mon"  "Tue"  "Wed"  "Thur" "Fri"
> WeekDays[3]
[1] "Wed"
> WeekDays[c(1,3,5)]
[1] "Mon" "Wed" "Fri"
>
> # Accessing vector elements using logical indexing
> WeekDays[c(F,F,T,F,F)]
[1] "Wed"
> WeekDays[c(F,T)]
[1] "Tue"  "Thur"
>
> # Accessing vector elements using negative indexing
> WeekDays[-2]
[1] "Mon"  "Wed"  "Thur" "Fri"
> WeekDays[c(-2,-4)]
[1] "Mon" "Wed" "Fri"
```

- **Manipulation with Vectors:**

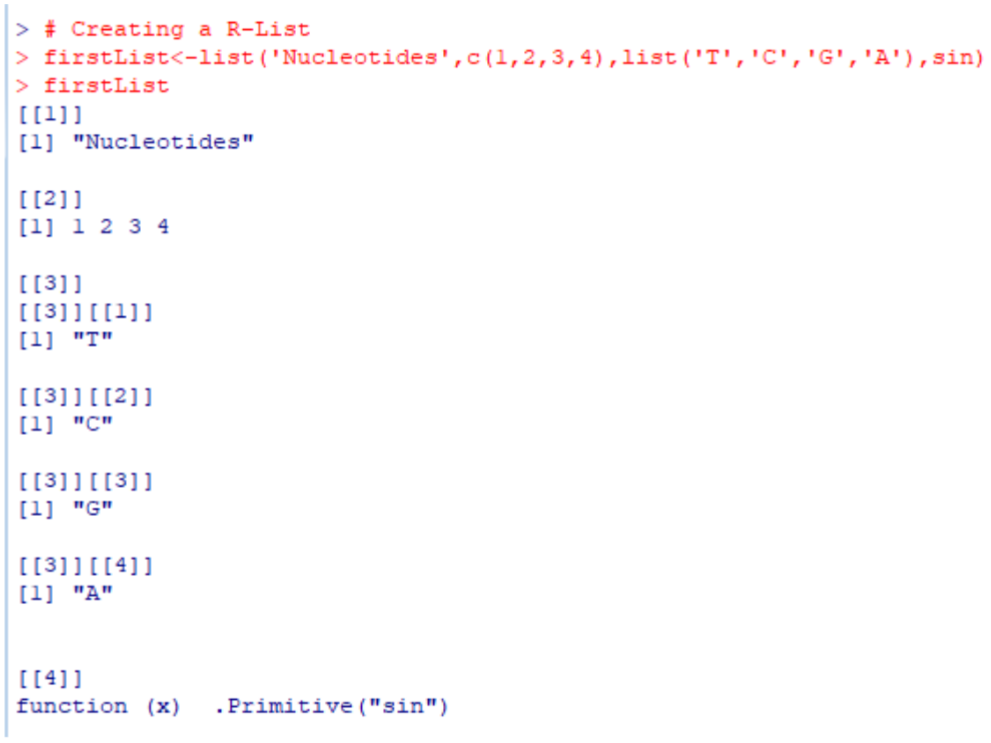


```
> # Manipulation with vectors
> x<-c(2,3,6)
> y<-c(4,5,2)
> x+5
[1]  7  8 11
> y-2
[1] 2 3 0
> 2*x
[1]  4  6 12
> y/4
[1] 1.00 1.25 0.50
> x%%2
[1] 0 1 0
> x%/%2
[1] 1 1 3
> x^2
[1]  4  9 36
> x+y
[1] 6 8 8
> x-y
[1] -2 -2  4
> x*y
[1]  8 15 12
> x/y
[1] 0.5 0.6 3.0
> x^y
[1]  16 243  36
```

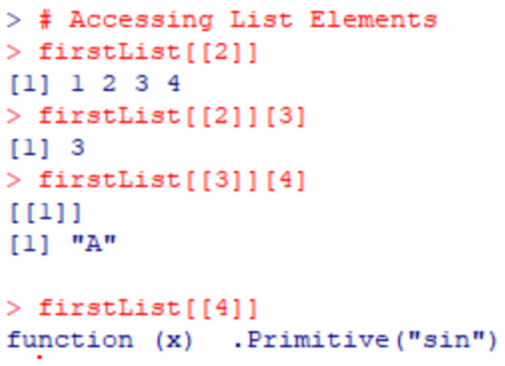


```
> # rep() function
> x<-c(13,17,20,21)
> rep(x,times=3)
 [1] 13 17 20 21 13 17 20 21 13 17 20 21
> rep(x,each=2)
[1] 13 13 17 17 20 20 21 21
> rep(x,each=2,times=3)
 [1] 13 13 17 17 20 20 21 21 13 13 17 17 20 20 21 21 13 13 17 17 20 20 21 21
```

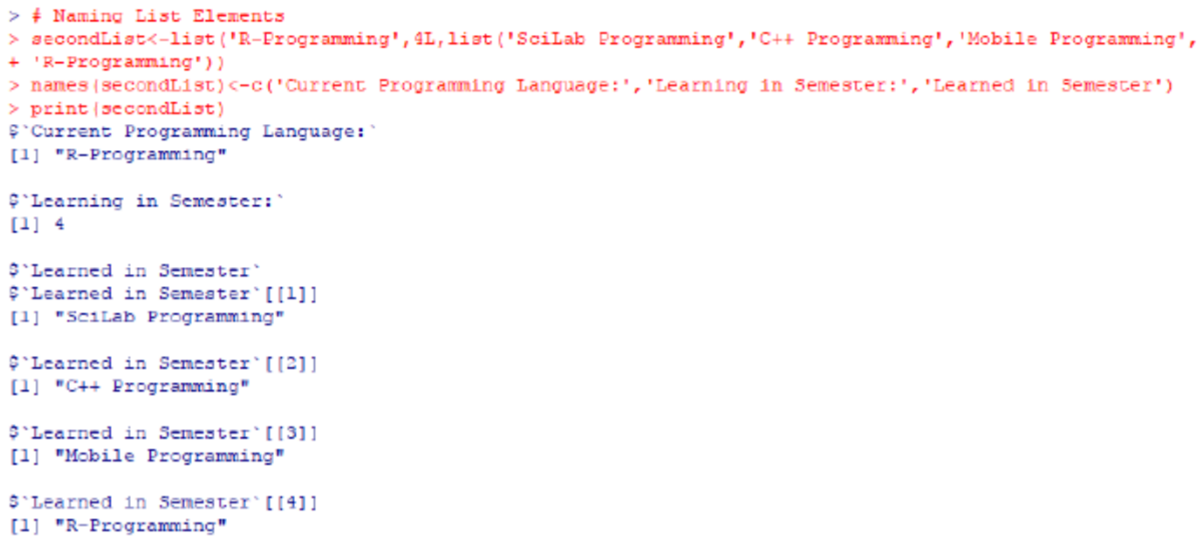
4. R – Lists:



```
> # Creating a R-List
> firstList<-list('Nucleotides',c(1,2,3,4),list('T','C','G','A'),sin)
> firstList
[[1]]
[1] "Nucleotides"

[[2]]
[1] 1 2 3 4

[[3]]
[[3]][[1]]
[1] "T"

[[3]][[2]]
[1] "C"

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

[[3]][[4]]
[1] "A"


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



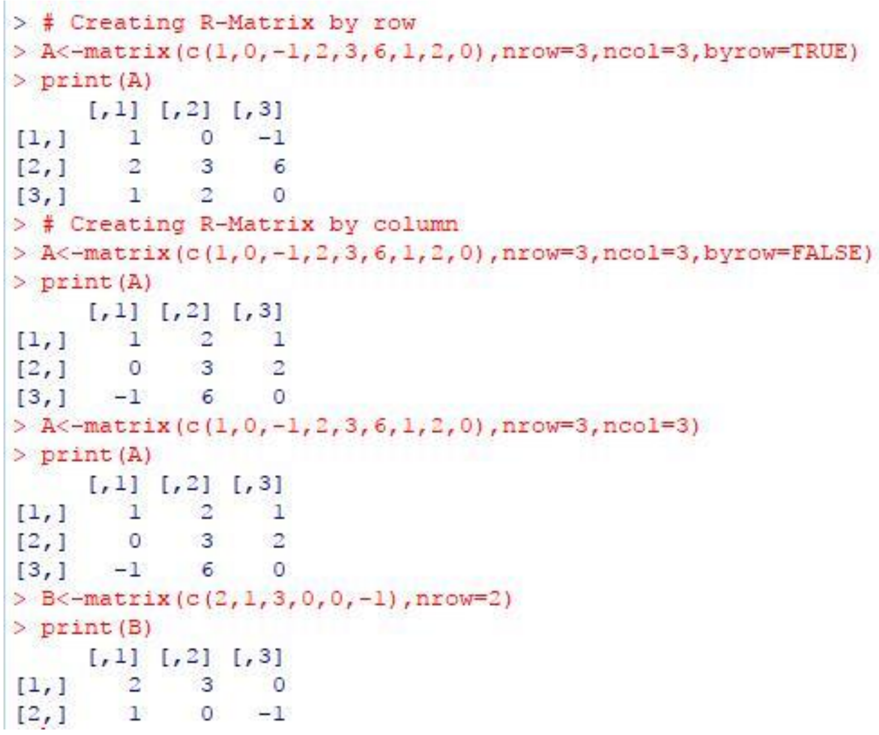
```
> # Accessing List Elements
> firstList[[2]]
[1] 1 2 3 4
> firstList[[2]][3]
[1] 3
> firstList[[3]][4]
[[1]]
[1] "A"

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



```
> # 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"
```

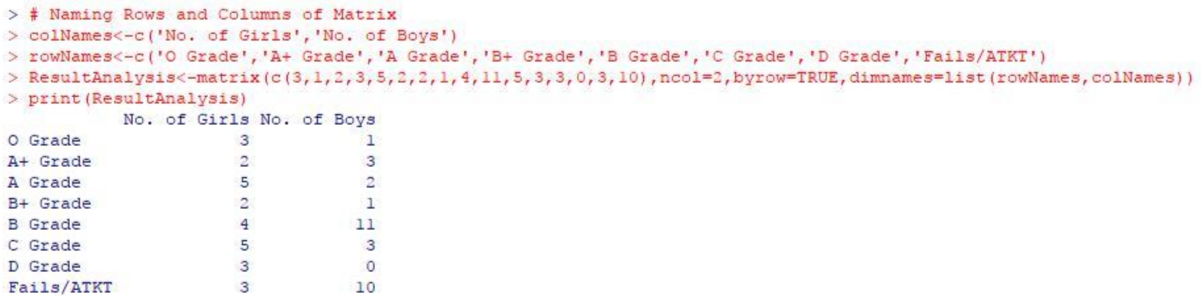
5. R – Matrices:

- **Creating R – Matrix:**



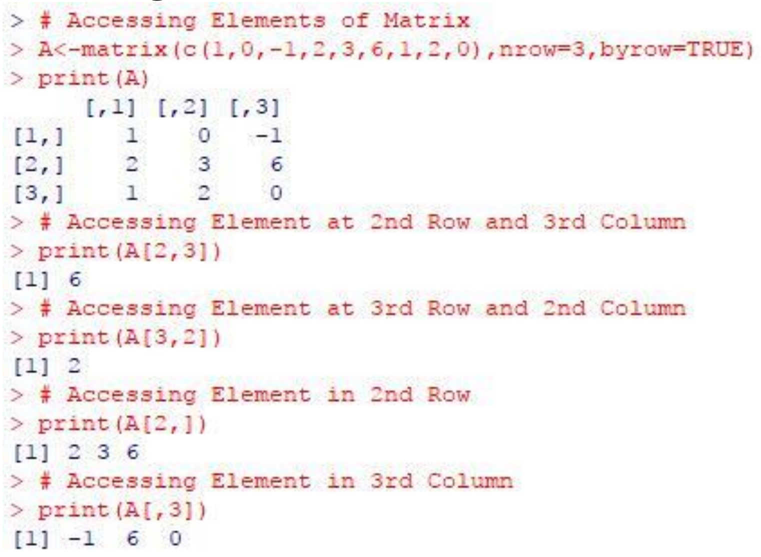
```
> # Creating R-Matrix by row
> A<-matrix(c(1,0,-1,2,3,6,1,2,0),nrow=3,ncol=3,byrow=TRUE)
> print(A)
     [,1] [,2] [,3]
[1,]    1    0   -1
[2,]    2    3    6
[3,]    1    2    0
> # Creating R-Matrix by column
> A<-matrix(c(1,0,-1,2,3,6,1,2,0),nrow=3,ncol=3,byrow=FALSE)
> print(A)
     [,1] [,2]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

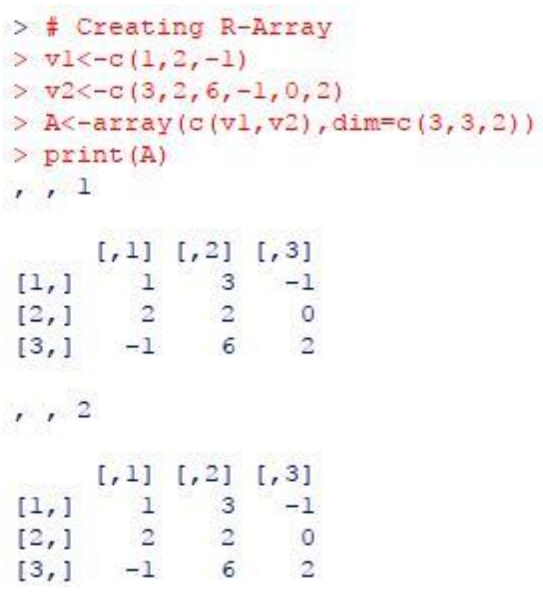
- **Accessing Elements of Matrix:**



```
> # 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
```

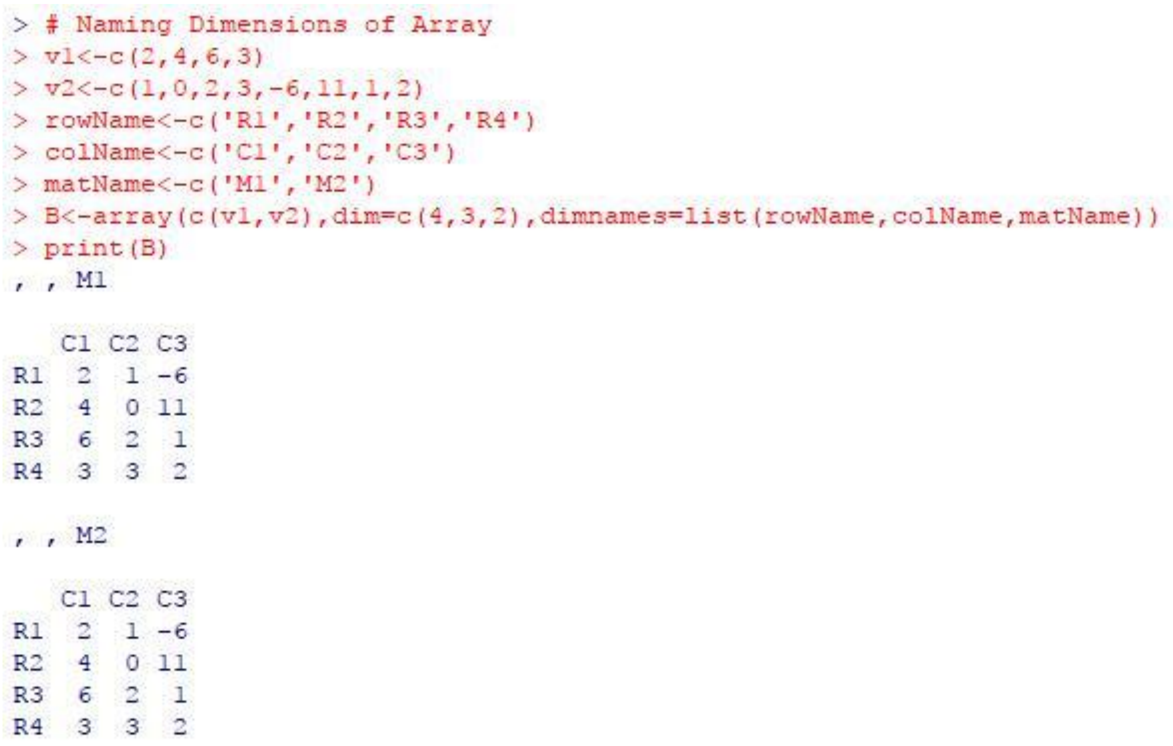
6. R – Arrays:

- **Creating R – Array:**



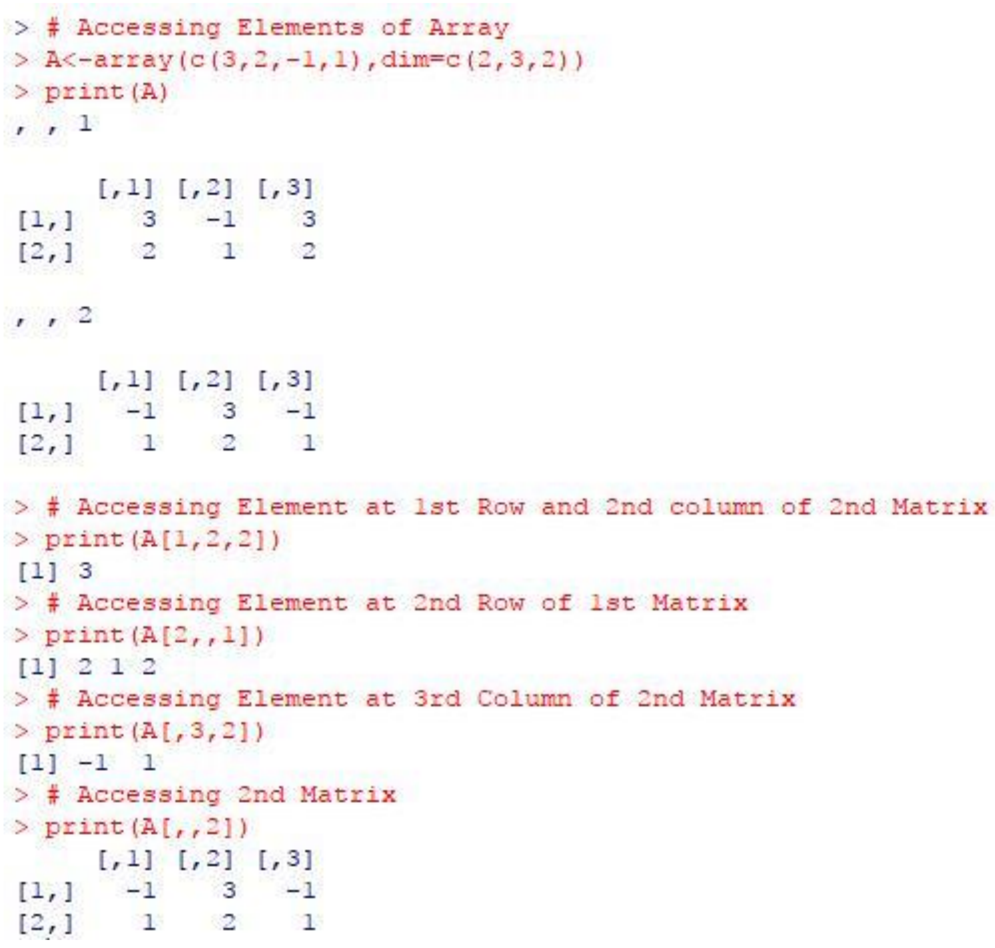
```
> # Creating R-Array
> v1<-c(1,2,-1)
> v2<-c(3,2,6,-1,0,2)
> A<-array(c(v1,v2),dim=c(3,3,2))
> print(A)
, , 1

     [,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**



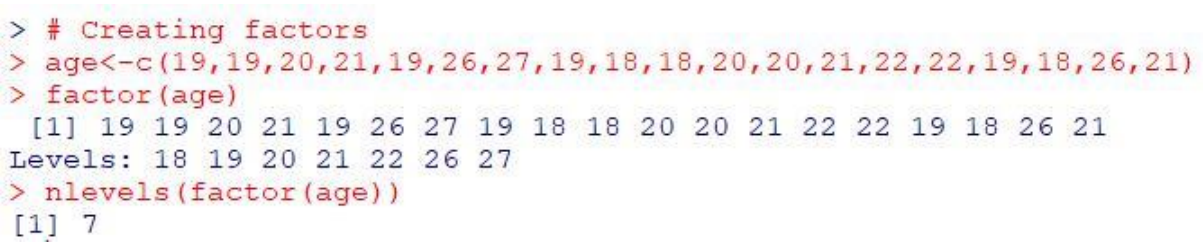
```
> # Naming Dimensions of Array
> v1<-c(2,4,6,3)
> v2<-c(1,0,2,3,-6,11,1,2)
> rowName<-c('R1','R2','R3','R4')
> colName<-c('C1','C2','C3')
> matName<-c('M1','M2')
> B<-array(c(v1,v2),dim=c(4,3,2),dimnames=list(rowName,colName,matName))
> print(B)
, , M1

   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
```

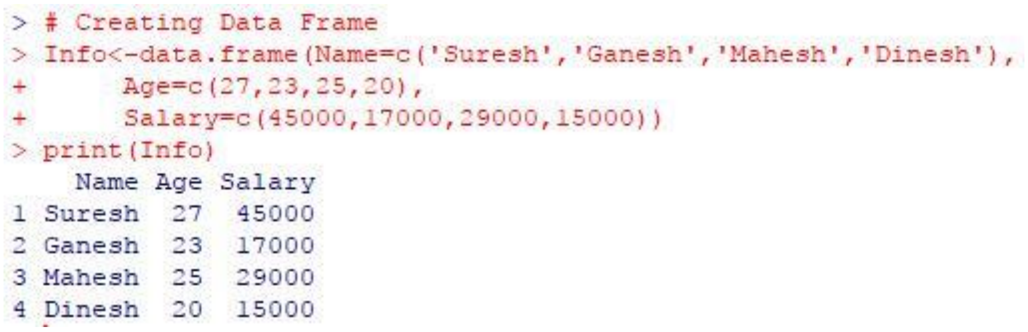
7. R – Factors



```
> # Creating factors
> age<-c(19,19,20,21,19,26,27,19,18,18,20,20,21,22,22,19,18,26,21)
> factor(age)
 [1] 19 19 20 21 19 26 27 19 18 18 20 20 21 22 22 19 18 26 21
Levels: 18 19 20 21 22 26 27
> nlevels(factor(age))
[1] 7
```

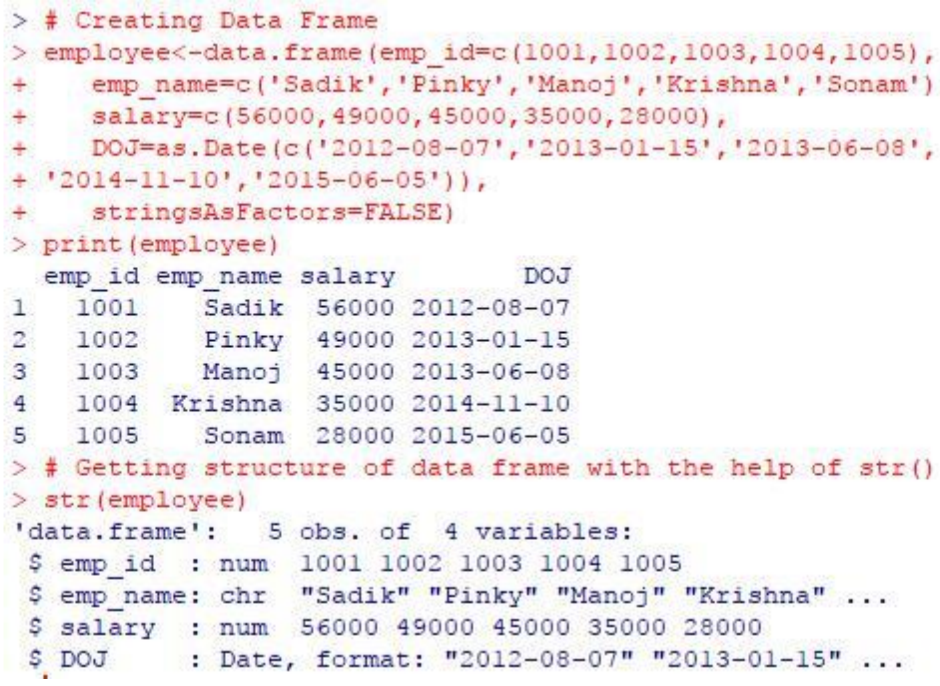
8. R – Data Frames:

- **Creating R – Data Frames:**



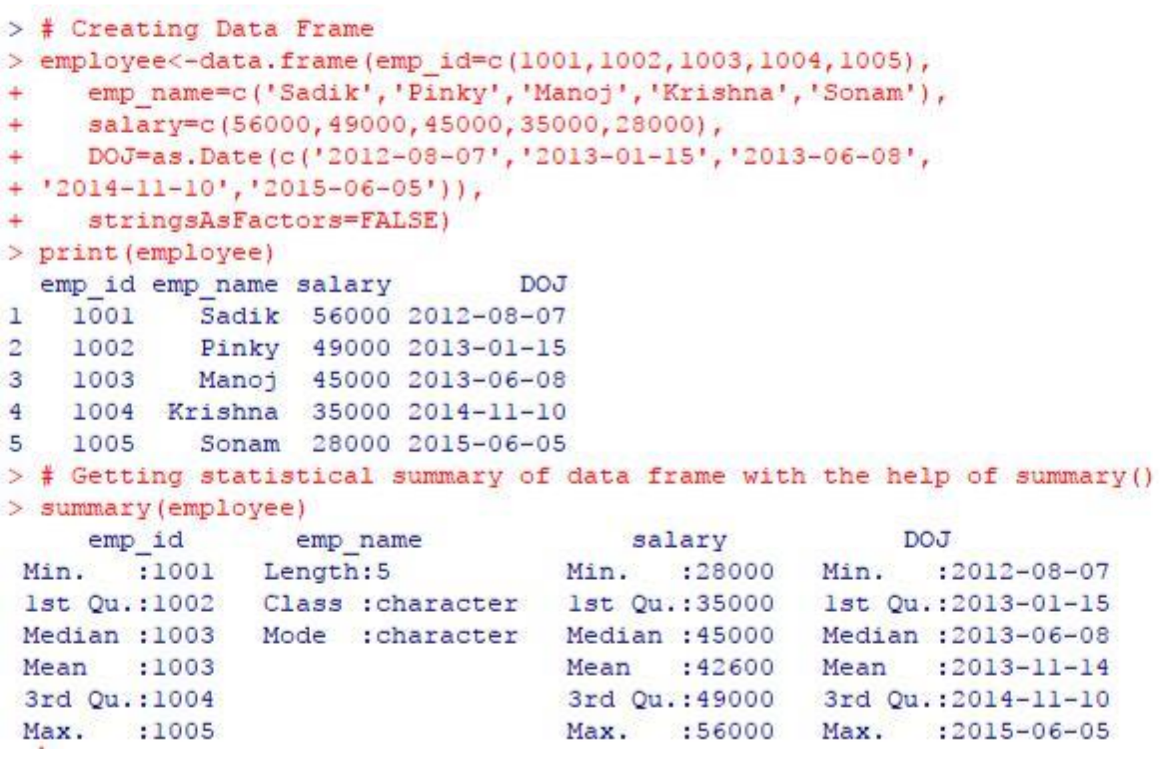
```
> # Creating Data Frame
> Info<-data.frame(Name=c('Suresh','Ganesh','Mahesh','Dinesh'),
+      Age=c(27,23,25,20),
+      Salary=c(45000,17000,29000,15000))
> print(Info)
    Name Age Salary
1 Suresh  27  45000
2 Ganesh  23  17000
3 Mahesh  25  29000
4 Dinesh  20  15000
```

- **Getting Structure of Data Frame**



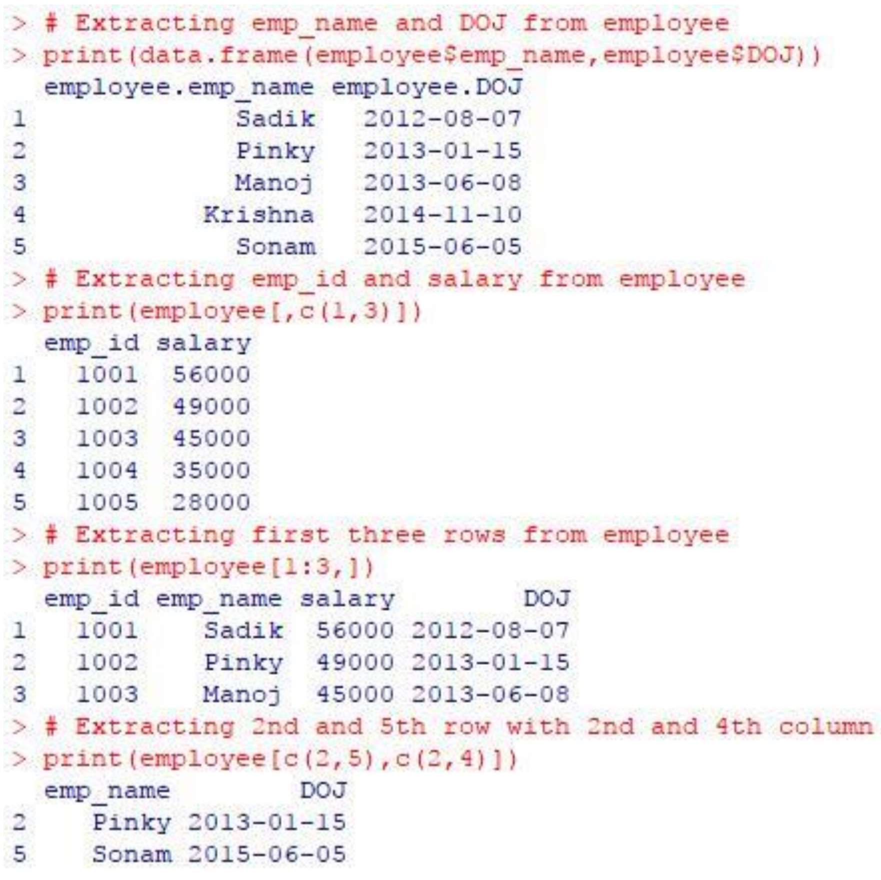
```
> # Creating Data Frame
> employee<-data.frame(emp_id=c(1001,1002,1003,1004,1005),
+    emp_name=c('Sadik','Pinky','Manoj','Krishna','Sonam'),
+    salary=c(56000,49000,45000,35000,28000),
+    DOJ=as.Date(c('2012-08-07','2013-01-15','2013-06-08',
+ '2014-11-10','2015-06-05')),
+    stringsAsFactors=FALSE)
> 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
```

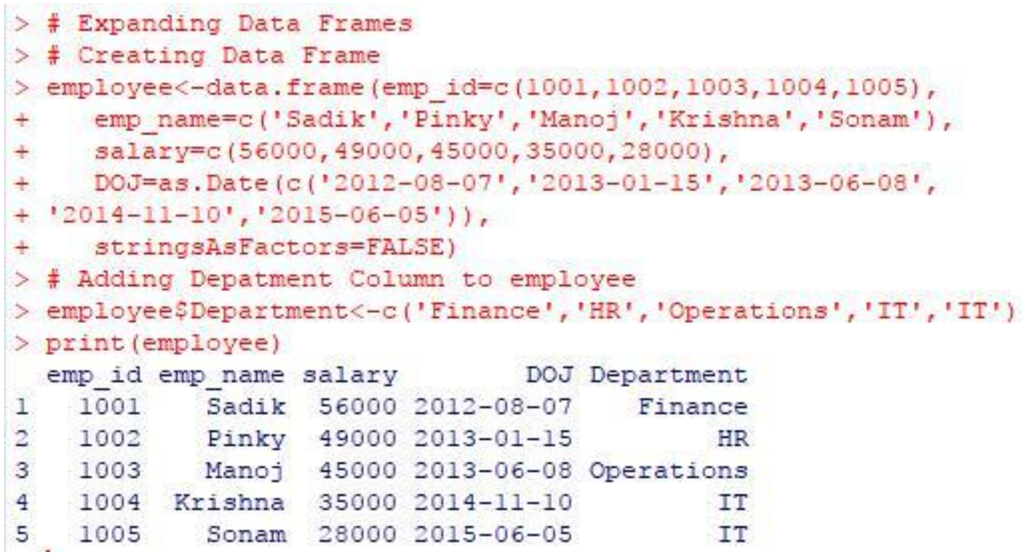
- **Extracting Data from Data Frame:**



```
> # 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
```

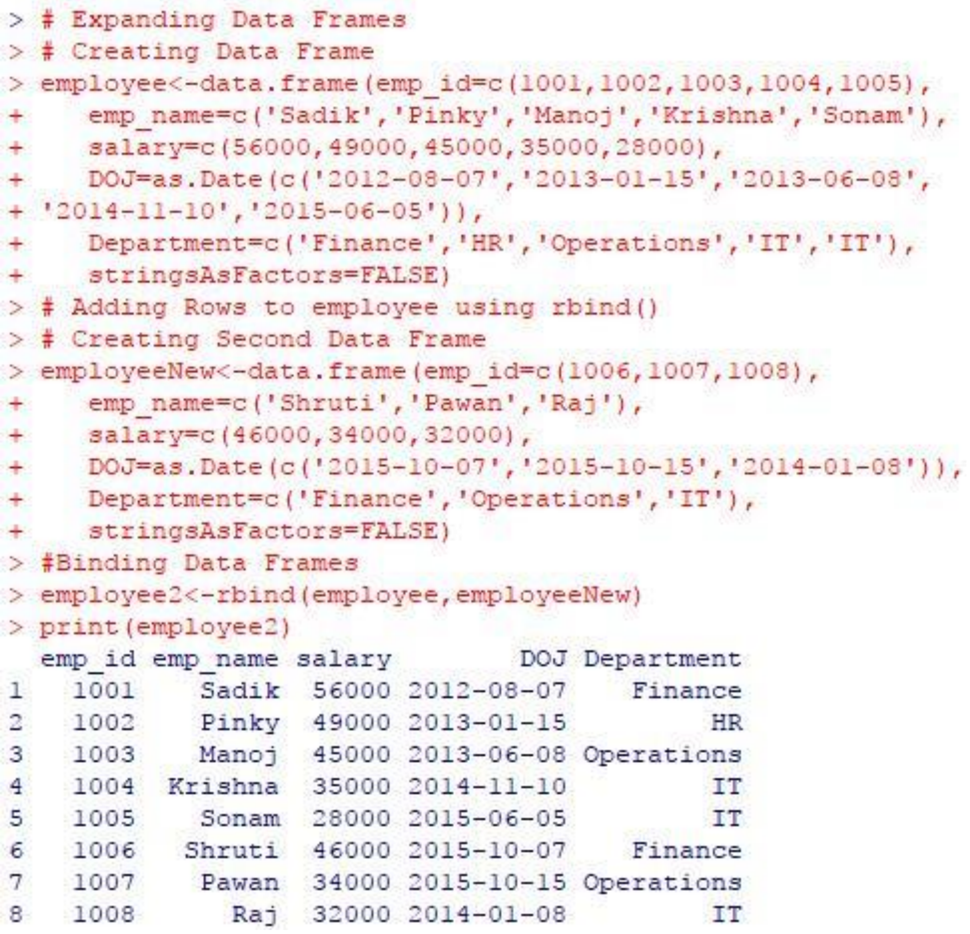
- **Expanding Data Frame**

  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
```

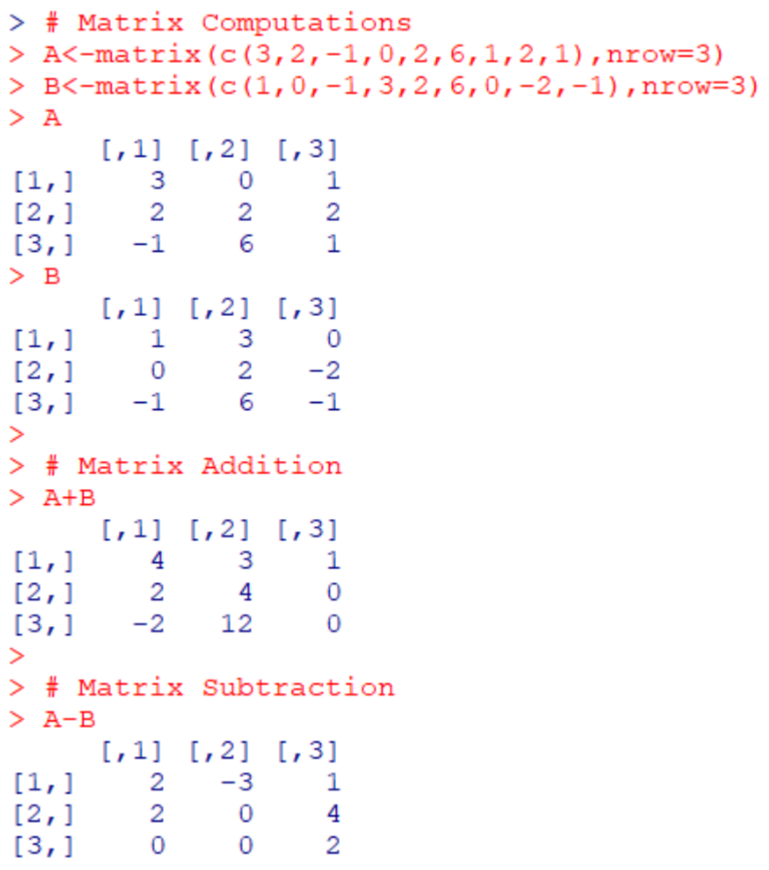
```
> # Creating Data Frame
> Name<-c('Manish','Danish','David','Sifa')
> Age<-c(32,21,28,25)
> Salary<-c(41000,20000,32000,28000)
> Info<-data.frame(Name,Age,Salary)
> Info
    Name Age Salary
1 Manish  32  41000
2 Danish  21  20000
3  David  28  32000
4   Sifa  25  28000
> Info$Name
[1] Manish Danish David  Sifa
Levels: Danish David Manish Sifa
> class(Info$Name)
[1] "factor"
> Info$Name<-as.character(Name)
> class(Info$Name)
[1] "character"
> Info$Salary
[1] 41000 20000 32000 28000
> Info$Name[3]
[1] "David"
> Info$Salary[3]
[1] 32000
> # Adding a column to existing data frame
> Emp_ID<-c(1001,1002,1003,1004)
> InfoNew<-cbind(Emp_ID,Info)
> InfoNew
  Emp_ID   Name Age Salary
1   1001 Manish  32  41000
2   1002 Danish  21  20000
3   1003  David  28  32000
4   1004   Sifa  25  28000
> # Adding a row to existing data frame
> NewRow<-c(1005,'Ashok',20,19000)
> InfoNew2<-rbind(InfoNew,NewRow)
> InfoNew2
  Emp_ID   Name Age Salary
1   1001 Manish  32  41000
2   1002 Danish  21  20000
3   1003  David  28  32000
4   1004   Sifa  25  28000
5   1005  Ashok  20  19000
```

PRACTICAL No. 2

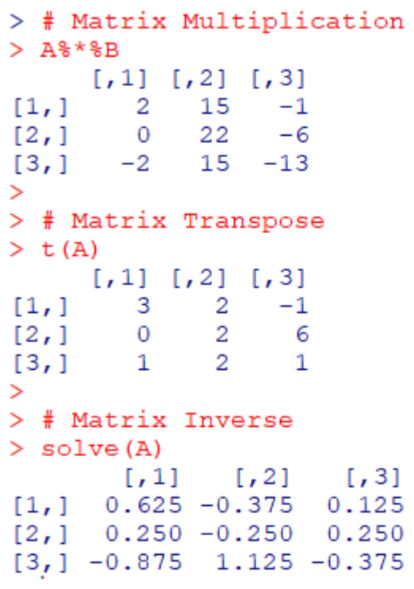
MATRIX COMPUTATIONS

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

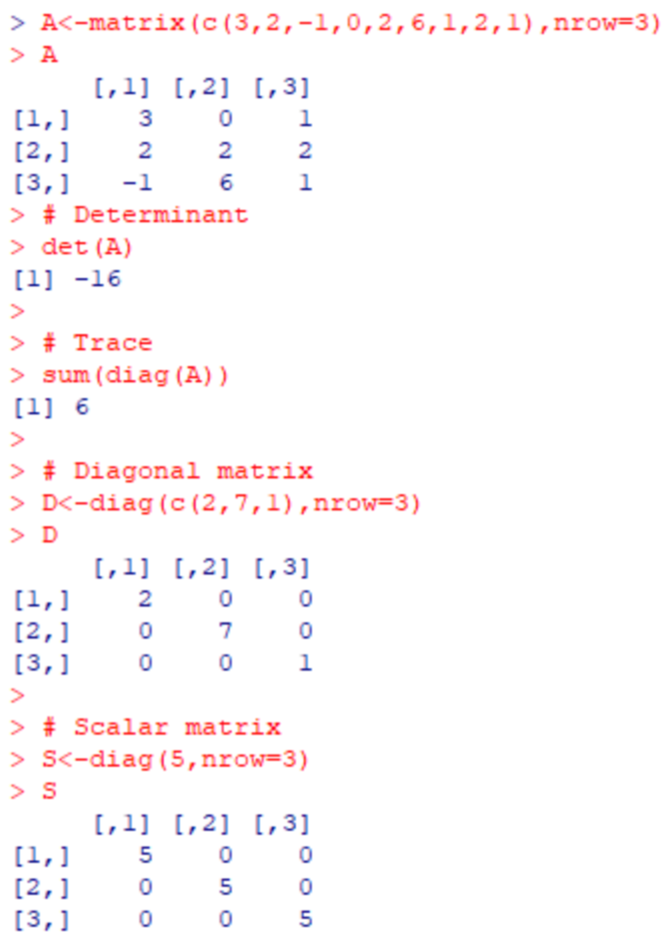
SOURCE CODE & OUTPUT:



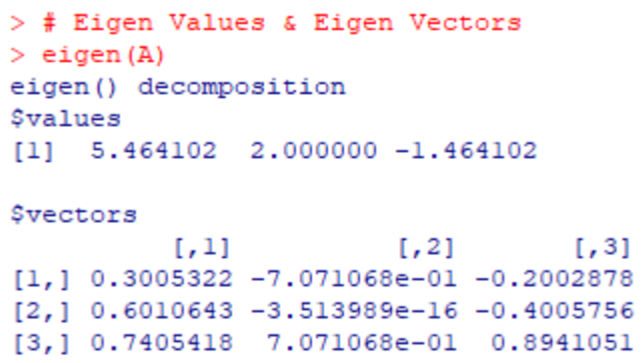
```
> # Matrix Computations
> A<-matrix(c(3,2,-1,0,2,6,1,2,1),nrow=3)
> B<-matrix(c(1,0,-1,3,2,6,0,-2,-1),nrow=3)
> A
     [,1] [,2] [,3]
[1,]    3    0    1
[2,]    2    2    2
[3,]   -1    6    1
> B
     [,1] [,2] [,3]
[1,]    1    3    0
[2,]    0    2   -2
[3,]   -1    6   -1
>
> # Matrix Addition
> A+B
     [,1] [,2] [,3]
[1,]    4    3    1
[2,]    2    4    0
[3,]   -2   12    0
>
> # Matrix Subtraction
> A-B
     [,1] [,2] [,3]
[1,]    2   -3    1
[2,]    2    0    4
[3,]    0    0    2
```



```
> # Matrix Multiplication
> A%*%B
     [,1] [,2] [,3]
[1,]    2   15   -1
[2,]    0   22   -6
[3,]   -2   15  -13
>
> # Matrix Transpose
> t(A)
     [,1] [,2] [,3]
[1,]    3    2   -1
[2,]    0    2    6
[3,]    1    2    1
>
> # Matrix Inverse
> 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
```



```
> A<-matrix(c(3,2,-1,0,2,6,1,2,1),nrow=3)
> A
     [,1] [,2] [,3]
[1,]    3    0    1
[2,]    2    2    2
[3,]   -1    6    1
> # Determinant
> det(A)
[1] -16
>
> # Trace
> sum(diag(A))
[1] 6
>
> # Diagonal matrix
> D<-diag(c(2,7,1),nrow=3)
> D
     [,1] [,2] [,3]
[1,]    2    0    0
[2,]    0    7    0
[3,]    0    0    1
>
> # Scalar matrix
> S<-diag(5,nrow=3)
> S
     [,1] [,2] [,3]
[1,]    5    0    0
[2,]    0    5    0
[3,]    0    0    5
```



```
> # Eigen Values & Eigen Vectors
> eigen(A)
eigen() decomposition
$values
[1]  5.464102  2.000000 -1.464102

$vectors
          [,1]          [,2]       [,3]
[1,] 0.3005322 -7.071068e-01 -0.2002878
[2,] 0.6010643 -3.513989e-16 -0.4005756
[3,] 0.7405418  7.071068e-01  0.8941051
```