Ex.No.2 **MATRICES IN R**

Date: 1-08-23

Aim

To implement the Matrices in R programming in the experiments and learn about them.

Procedure

1. To do programming in R, first install "RStudio" and "R" in the system. RStudio is an integrated development environment [IDE] for R and python.
2. Select the File in taskbar →open New file →R script or use shortcut "ctrl+shift+N"
3. Write the program in the script and save it using the extension R.
4. Run the program by clicking Run option or use the shortcut "ctrl+enter".
5. See the output in the console tab.

Concepts Applied

- Simple programs with Matrices

CREATING A MATRIX

A matrix is created using the matrix () function. The inputs taken are data, no. of rows, no. of columns … The data can be of numbers and strings too.

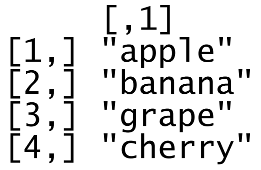
Script

#Creating a matrix

my_matrix <- matrix(c("apple","banana","grape","cherry"),nrow=4,ncol=1)

my_matrix

Output



```
     [,1]
[1,] "apple"
[2,] "banana"
[3,] "grape"
[4,] "cherry"
```

ACCESSING THE MATRIX ITEMS

The matrix items are accessed using the index or [ ] brackets. The 1st position specifies row position and 2nd position specifies the column position.

Script

#Accessing the matrix items

my_matrix <- matrix(c("apple","banana","grape","cherry"),nrow=4,ncol=1)

```
my_matrix[3,1]
```

Output

```
[1] "grape"
```

ACCESSING WHOLE ROWS OR COLUMNS

A whole row can be accessed if you specify a comma after the number in the brackets and likewise, a whole column can be accessed if you specify a comma before the number in the brackets.

Script

```
#accessing a whole row

my_matrix <- matrix(c("apple","banana","grape","cherry"),nrow=4,ncol=1)

my_matrix[1,]

#accessing a whole column

my_matrix <- matrix(c("apple","banana","grape","cherry"),nrow=4,ncol=1)

my_matrix[,1]
```

Output

```
[1] "apple"
[1] "apple"  "banana" "grape"  "cherry"
```

ACCESSING MORE THAN ONE ROW OR COLUMN

Accessing more than one row or column can be done by using c ().

Script

```
#accessing more than one row

my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=3,ncol=3)

my_matrix[c(1,2),]

##accessing more than one column

my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=3,ncol=3)

my_matrix[,c(1,2)]
```

Output

```
     [,1]     [,2]     [,3]    
[1,] "apple"  "cherry" "apple" 
[2,] "banana" "kiwi"   "banana"

     [,1]     [,2]    
[1,] "apple"  "cherry"
[2,] "banana" "kiwi"  
[3,] "grape"  "mango" 
```

ADD ROWS IN A MATRIX

Use the rbind () method to add more additional rows in the matrix. But, note that the number of cells should be in same length as the matrix.

Script

#Add rows

my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=2,ncol=3)

newmatrix <- rbind(my_matrix,c("guava","pineapple","sapota"))

newmatrix

Output

```
     [,1]     [,2]        [,3]    
[1,] "apple"  "grape"     "kiwi"  
[2,] "banana" "cherry"    "mango" 
[3,] "guava"  "pineapple" "sapota"
```

ADD COLUMNS IN A MATRIX

Use the cbind () method to add more additional colums in the matrix. But, note that the number of cells should be in same length as the matrix.

Script

#add columns

my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=3,ncol=2)

newmatrix <- cbind(my_matrix,c("guava","pineapple","sapota"))

newmatrix

Output

```
     [,1]     [,2]     [,3]
[1,] "apple"  "cherry" "guava"
[2,] "banana" "kiwi"   "pineapple"
[3,] "grape"  "mango"  "sapota"
```

REMOVING ROWS AND COLUMNS

Rows and columns of a matrix can be removed by using c () function with -ve symbol prefixing it.

Script

```
#removing rows and colums

my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=3,ncol=2)

#remove 3rd row and 2nd column

my_matrix <- my_matrix[-c(3),-c(2)]

my_matrix
```

Output

```
[1] "apple"  "banana"
```

CHECK IF AN ITEM EXISTS

To check if an item exists in the matrix or not, use the %in% operator.

Script

```
#check if item present

my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=3,ncol=2)

"apple" %in% my_matrix
```

Output

```
[1] TRUE
```

NUMBER OF ROWS AND COLUMNS

The number of rows and columns of a matrix can be found by using dim () function.

Script

```
#number of rows and columns of a matrix
```

```
my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=3,ncol=2)

dim(my_matrix)
```

Output

```
[1] 3 2
```

LENGTH OF A MATRIX

The length of a matrix can be found by using the length () function.

Script

```
#length of a matrix

my_matrix <- matrix(c("apple","banana","grape","cherry","kiwi","mango"),nrow=3,ncol=2)

length(my_matrix)
```

Output

```
[1] 6
```

COMBINE TWO MATRICES

Using rbind () and cbind () methods, we can combine two matrices by adding them as rows or columns.

Script

```
#combine two matrices

matrix1 <- matrix(c("apple","banana","grape","cherry"),nrow=2,ncol=2)

matrix2 <-matrix(c("guava","pineapple","sapota","kiwi"),nrow = 2, ncol = 2)

#adding them as rows

combined_matrix = rbind(matrix1,matrix2)

combined_matrix

#adding them as columns

combined_matrix = cbind(matrix1,matrix2)

combined_matrix
```

Output

```
     [,1]        [,2]    
[1,] "apple"     "grape" 
[2,] "banana"    "cherry"
[3,] "guava"     "sapota"
[4,] "pineapple" "kiwi"  
     [,1]     [,2]     [,3]        [,4]    
[1,] "apple"  "grape"  "guava"     "sapota"
[2,] "banana" "cherry" "pineapple" "kiwi"  
```

MATRIX ADDITION, SUBTRACTION, PRODUCT, DIVISION

The arithmetic operations on the matrix like Addition, Subtraction, Product, Division can be done.

Script

```
#Arithmetic oprations(+, -, * ,/) on matrices

m1 = matrix(c(1,2,3,4,5,6),nrow = 2, ncol = 3)

print("Matrix-1: ")

m1

m2 = matrix(c(0,2,1,3,4,5),nrow = 2, ncol = 3)

print("Matrix-2: ")

m2


#Addition

result = m1 + m2

print("Result of addition: ")

result


#Subtraction

result = m1 - m2

print("Result of Subtraction: ")

result
```

```
#Product
result = m1 * m2
print("Result of product: ")
result


#Division
result = m1 / m2
print("Result of division: ")
result
```

Output

```
[1] "Matrix-1: "
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
[1] "Matrix-2: "
     [,1] [,2] [,3]
[1,]    0    1    4
[2,]    2    3    5


[1] "Result of addition: "
     [,1] [,2] [,3]
[1,]    1    4    9
[2,]    4    7   11


[1] "Result of Subtraction: "
     [,1] [,2] [,3]
[1,]    1    2    1
[2,]    0    1    1


[1] "Result of product: "
     [,1] [,2] [,3]
[1,]    0    3   20
[2,]    4   12   30


[1] "Result of division: "
     [,1]     [,2] [,3]
[1,]  Inf 3.000000 1.25
[2,]    1 1.333333 1.20
```

MATRIX FROM A GIVEN LIST OF VECTORS

From the given list of vectors, a matrix can be formed using do.call () and rbind () methods.

Script

```
#matrix from a list of given vectors

l = list()

for (i in 1:6) l[[i]] <- c(i, 1:4)

print("List of vectors:")

print(l)

result = do.call(rbind, l)

print("New Matrix:")

print(result)
```

Output

```
[1] "List of vectors:"
[[1]]
[1] 1 1 2 3 4

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

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

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

[[5]]
[1] 5 1 2 3 4

[[6]]
[1] 6 1 2 3 4
[1] "New Matrix:"
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    1    2    3    4
[2,]    2    1    2    3    4
[3,]    3    1    2    3    4
[4,]    4    1    2    3    4
[5,]    5    1    2    3    4
[6,]    6    1    2    3    4
```

CONVERT A GIVEN MATRIX INTO A LIST OF COLUMN VECTORS

The given matrix can be converted into a list of column vectors again by using rep (), split ()…methods.

Script

```
#Convert a given matrix to a list of column vectors

x = matrix(1:12, ncol=3)

print("Original matrix:")

print(x)

print("list from the said matrix:")

l = split(x, rep(1:ncol(x), each = nrow(x)))

print(l)
```

Output

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

[1] "list from the said matrix:"

[1] 1 2 3 4

[1] 5 6 7 8

[1] 9 10 11 12
```

FINDING THE ROW AND COLUMN INDEX OF THE MINIMUM AND MAXIMUM VALUE IN A GIVEN MATRIX

Using the max () and min () functions to find the maximum and minimum values in the matrix respectively and using which and byrow== TRUE to find the index of those respective items in the matrix.

Script

```
# row and column index of maximum and minimum value in a given matrix

m = matrix(c(1:16), nrow = 4, byrow = TRUE)

print("Original Matrix:")
```

print(m)

result = which(m == max(m), arr.ind=TRUE)

print("Row and column of maximum value of the said matrix:")

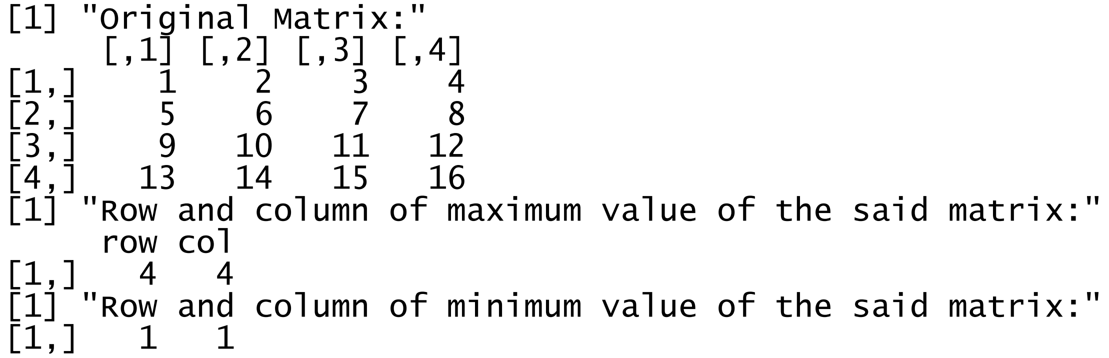
print(result)

result = which(m == min(m), arr.ind=TRUE)

print("Row and column of minimum value of the said matrix:")

print(result)

Output



```
[1] "Original Matrix:"
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
[1] "Row and column of maximum value of the said matrix:"
     row col
[1,]   4   4
[1] "Row and column of minimum value of the said matrix:"
[1,]   1   1
```

CONCATENATE TWO MATRICES OF DIFFERENT NUMBER OF ROWS

Using the matrix () to form matrices and concatenating them using rbind ().

Script

#Concatenate two given matrixes of same column but different rows

x = matrix(1:12, ncol=3)

y = matrix(13:24, ncol=3)

print("Matrix-1")

print(x)

print("Matrix-2")

print(y)

result = (rbind(x,y))

print("After concatenating two given matrices:")

print(result)

Output

```
[1] "Matrix-1"
    [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
[1] "Matrix-2"
     [,1] [,2] [,3]
[1,]   13   17   21
[2,]   14   18   22
[3,]   15   19   23
[4,]   16   20   24

[1] "After concatenating two given matrices:"
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
[5,]   13   17   21
[6,]   14   18   22
[7,]   15   19   23
[8,]   16   20   24
```

Result

Thus, the matrices have been successfully implemented in R programming.