SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA ENGINEERING AND TECHNOLOGY

CSE 240 Data Science with R

STUDENT WORK BOOK

Name	**:**	M.Sathishkumar
Unique ID	**:**	E0119052
Year	**:**	II
Quarter	**:**	Q6
Department	**:**	B.Tech CSE (CyS & IoT or AI &ML)
Faculty Name	**:**	Prof.B.Nirmala or Prof.N.Chiranjeevi
Academic Year	**:**	2020-2021

Date: 02/11/2020

Q.NO : Question 1

Program:

```
#Matrix creation
matrix<-matrix(c(c(9,10,11,12),c(13,14,15,16)),nrow = 4,ncol = 2)
#combining two vertices and converting into 4 by 2 matrix
print(matrix)
```

Output:

```
     [,1] [,2]
[1,]    9   13
[2,]   10   14
[3,]   11   15
[4,]   12   16
```

Explanation:

- Matrix is a two-dimensional data structure in R programming.
- Matrix can be created using the matrix () function
- Dimension of the matrix can be defined by passing appropriate value for arguments nrow and ncol

Date: 02/11/2020

Q.NO : Question 2

Program:

```
#UserInput
Id = readline("Enter your UserID:")
#Prompts user for input(id)
Branch = readline(prompt="Enter you Branch/Group:")
#Prompts user for input(branch)
cat("Your UserID is",Id,"and you belong to",Branch,"group")
#Displaying Values
```

Output:

```
Enter your UserID:E0119052
Enter you Branch/Group:AI & ML
Your UserID is E0119052 and you belong to AI & ML group
```

Explanation:

- ✶ readline reads a line from the terminal (in interactive use).
- ✶ readline() lets the user enter a one-line string at the terminal.
- ✶ The prompt argument is printed in front of the user input. It usually ends on ": ".

Date: 02/11/2020

Q.NO : Question 3

Program:

```
#Dataframe creation
Name <- c("Jhon", "Lee", "Suzan", "Abhinav", "Brain", "Emma", "David")
#Variables
Subject <- c("Data Science", "Machine Learning", "Deep Learning",
             "Data Structures", "Database Managemnt System",
             "Operating Systems", "Python Programming")

Score <- c(56, 76, 86, 96, 73, 87, 47)

Rank <- c(5,8,6,7,9,2,1)

df = data.frame(Name,Subject,Score,Rank)
print(df)
#Dataframe Created Successfully
```

Output:

```
     Name                   Subject Score Rank
1    Jhon              Data Science    56    5
2     Lee          Machine Learning    76    8
3   Suzan             Deep Learning    86    6
4 Abhinav           Data Structures    96    7
5   Brain Database Managemnt System    73    9
6    Emma          Operating Systems    87    2
7   David         Python Programming    47    1
```

Explanation:

- A **data frame** is used for storing data tables. It is a list of vectors of equal length.
- The top line of the table, called the **header**, contains the column names.
- Each horizontal line afterward denotes a **data row**, which begins with the name of the row, and then followed by the actual data.
- Each data member of a row is called a **cell**.

Date: 02/11/2020

Q.NO : Question 4

Program:

```
#Dataframe creation
Name <- c("Jhon", "Lee", "Suzan", "Abhinav", "Brain", "Emma", "David")
#Variables
Subject <- c("Data Science", "Machine Learning", "Deep Learning",
             "Data Structures", "Database Managemnt System",
             "Operating Systems", "Python Programming")
Score <- c(56, 76, 86, 96, 73, 87, 47)
Rank <- c(5,8,6,7,9,2,1)
df = data.frame(Name,Subject,Score,Rank)
print(df)
#Dataframe Created Successfully
print(summary(df))
#Dataframe Stastical summary
```

Output:

```
     Name                         Subject      Score           Rank
 Abhinav:1   Data Science              :1   Min.   :47.00   Min.   :1.000
 Brain  :1   Data Structures           :1   1st Qu.:64.50   1st Qu.:3.500
 David  :1   Database Managemnt System:1   Median :76.00   Median :6.000
 Emma   :1   Deep Learning             :1   Mean   :74.43   Mean   :5.429
 Jhon   :1   Machine Learning          :1   3rd Qu.:86.50   3rd Qu.:7.500
 Lee    :1   Operating Systems         :1   Max.   :96.00   Max.   :9.000
 Suzan  :1   Python Programming        :1
```

Explanation:

- ✶ Summary is a generic function used to produce result summaries of the results of various model fitting functions.
- ✶ The function invokes particular methods which depend on the class of the first argument.
- ✶ The form of the value returned by summary depends on the class of its argument

Date: 02/11/2020

Q.NO : Question 5

Program:

```
#Dataframe creation
Name <- c("Jhon", "Lee", "Suzan", "Abhinav", "Brain", "Emma", "David")
#Variables
Subject <- c("Data Science", "Machine Learning", "Deep Learning",
             "Data Structures", "Database Managemnt System",
             "Operating Systems", "Python Programming")
Score <- c(56, 76, 86, 96, 73, 87, 47)
Rank <- c(5,8,6,7,9,2,1)
df = data.frame(Name,Subject,Score,Rank)
print(df)
#Dataframe Created Successfully
print(df["Subject"])
#Sliced column with column name
```

Output:

```
                      Subject
1               Data Science
2           Machine Learning
3              Deep Learning
4             Data Structures
5 Database Managemnt System
6           Operating Systems
7          Python Programming
```

Explanation:

- We retrieve a data frame column slice with the *single square bracket* "[]" operator.
- Here We can retrieve the same column slice by its name.
- And we can pack the row names in an index vector in order to retrieve multiple rows.

Date: 02/11/2020

Q.NO : Question 6

Program:

```
#Dataframe creation
Name <- c("Jhon", "Lee", "Suzan", "Abhinav", "Brain", "Emma", "David")
#Variables
Subject <- c("Data Science", "Machine Learning", "Deep Learning",
             "Data Structures", "Database Managemnt System",
             "Operating Systems", "Python Programming")
Score <- c(56, 76, 86, 96, 73, 87, 47)
Rank <- c(5,8,6,7,9,2,1)
df = data.frame(Name,Subject,Score,Rank)
#Dataframe Created Successfully
print(df[1:2,])
#splicing Dataframe with respect to index
```

Output:

```
  Name           Subject Score Rank
1 Jhon      Data Science    56    5
2  Lee Machine Learning    76    8
```

Explanation:

- We retrieve rows from a data frame with the single square bracket operator, just like what we did with columns.
- However, in additional to an index vector of row positions, we append an extra comma character.
- This is important, as the extra comma signals a wildcard match for the second coordinate for column positions.