PRACTICAL No. 01

Aim : Write a program to find sum, mean & product of a vector ignore NA or NULL.

Theory : Sum (), mean (), prod () methods are available in R which are used to compute the specified operation over the arguments specified in the method.

- **Sum () :** is used to calculate sum
  Syntax : sum (x)
- **mean () :** function is used to calculate mean
  na.rm : Boolean value to ignore NA value.
  Syntax : mean (x, na.rm)
- **Prod () :** is used to calculate product
  Syntax : prod (x)

CODE & OUTPUT

```
> x <- c (10, 11, 12, 13, 14)
> print (x)
[1] 10 11 12 13 14
> sum (x)
[1] 60
> mean (x)
[1] 12
> prod (x)
[1] 240240
> x <- c (1, 2, NA, 5, 6, NA)
```

00

```
> Print (x)
[1] 1 2 NA 5 6 NA
> sum(x)
[1] NA
>Sum (x, na.rm = TRUE)
[1] 14
> mean (x, na.rm = TRUE)
[1] 3.5
> Prod (x, na.rm = TRUE)
[1] 60
```

Conclusion :- Successfully demonstrated the applicatio of removal of NA values from vector.

PRACTICAL NO. 02

Aim: Write a program to create a vector using colon (:) operator and sequence() function.

Theory: R has colon operator which makes it really easy to define a sequence of integers.
for eg: code 1:15 generates a vector of consisting of the integers 1 to 15.

Seq() function: Used to create a sequence of elements in a vector. It takes the length & difference between values as optional argument.

Syntax : Seq (from, to, by, length.out)

Parameters from:- Starting element of the sequence. **by:-** Difference between the elements
to:- Ending elements of sequence.
lengthout:- Maximum length of vector.

CODE & OUTPUT:

```
> x <- c(1:15)
> print(x)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

500

```
> y = seq (1, 5, by = 0.5)
> print (y)
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
> z = seq (1, 5, length.out = 5)
> print (z)
[1] 1 2 3 4 5
> a = seq (10, 20, from = 6)
> Print (a)
[1] 6 11 16.
>b = seq (1, 3, to = 15)
> print (b)
[1] 1 4 7 10 13.
> c = seq (from = 5, to = 13)
> Print (c)
[1] 5 6 7 8 9 10 11 12 13.
```

conclusion:-

Successfully demonstrated program to create a vector using colon(:) operator & sequence () function

PRACTICAL NO :- 3

AIM :- Write a program to create a list containing string, number, vector and a logical values perform any 3 operation on it.

THEORY :- A list is generated using list() function. It is basically a generic vector that contains different objects. R allows its users to perform various operations on lists which can be used to illustrate data in different form.

CODES and OUTPUT :-

- CREATING A LIST :- lists in R can be created by placing the sequence inside the list() function

```
> list1 <- list("ABC", "XYZ", 4, 5.6, TRUE)
> print(list1)
[[1]]
[1] "ABC"
[[2]]
[1] "XYZ"
[[3]]
[1] 4
[[4]]
[1] 5.6
[[5]]
[1] TRUE
```

• NAMING A LIST :- Names can be assigned to the elements of the list and these names can be used to access the elements.

```
> list_data <- list(c("Jan", "Feb", "Mar"), matrix(c(1,2,3,4,5,6
  nrow=2), list("red", "blue", "green"))
> names(list_data) <- c("week", "Matrix", "Months")
> print(list_data)
$ week
[1] "Jan"  "Feb"  "Mar"
$ Matrix
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
$ Months
$ Months [[1]]
[1] "red"
$ Months [[2]]
[1] "blue"
$ Months [[3]]
[1] "green"
```

• ACCESSING LIST ELEMENTS :- In order to access the elements, use the index number, and in case of named list, elements can be accesed using its name also

```
> print(list_data[2])
$ Matrix
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> print (list_data $week)
[1] "Jan" "Feb" "Mar"
```

CONCLUSION :-

```
> list_data [4] <- "New Element"
> print (list_data[4])
[[1]]
[1] "New Element"
```

- REMOVING THE ELEMENT FROM THE LIST :-

```
> list_data [4] = NULL
> print (list_data [4])
$(NA)
NULL
```

- MERGING THE LIST :-

```
> list1 <- c ("Mon", "TUE", "wed")
> list2 <- c ("Jan", "Feb", "Mar")
> list3 <- c (100, 200, 300)
>merge.list <-c (list1, list2, list3)
> print (merge.list)
[1] "Mon" "TUE" "wed" "Jan" "Feb" "Mar" "100
    200 300"
```

• CONVERTING THE LIST INTO VECTOR :-

```
> V1 <- unlist (list 1)
> V2 <- unlist (list 2)
> print (V1)
[1] 1 2 3 4 5 6 7 8 9 10
> print (V2)
[1] 11 12 13 14 15
> result <- V1 + V2
[1] 12 14 16 18 20 17 19 21 23 25
```

• CONCLUSION :- Successfully created a program to create a list containing strings, numbers, vectors and a logical values

PRACTICAL NO. 04.

Aim :- Write a program to create a dataframe. Access each elements of those dataframe from the list.

Theory :- A dataframe is a table or a two-dimensional array like structure in which each column contains value of one one variable and each row contains one set of values from each column.

Following are the characteristics of a dataframe

- The column names should be non-empty.
- The row names should be unique.
- The data stored in a data frame can be of numeric, factor or character type.
- Each column should contain same number of data items.

CODES AND OUTPUT :-

CREATING LIST OF DATA FRAME.

```
> df1 = data.frame (colour = c("Red", "Blue", Green"),
      colour_id = c(1,2,3))
> df2 = data.frame (colour = c("orange", "Black", "white")
      colour_id = c(4,5,6))
> list_dataframe = list (df1, df2)
  > print (list_dataframe).
```

300

[[1]]

	colour	colour-id
1	Red	1
2	Blue	2
3	Green	3

[[2]]

	colour	colour-id
1	orange	4
2	Black	5
3	White	6

NAMING DATA FRAME:

```
list_dataframe = list("colour-list 1"=df1,"colour_list 2"=df2)
> Print (list_dataframe)
```

$ colour_list 1

	colour	colour-id
1	Red	1
2	Blue	2
3	Green	3

$ colour_list 2

	colour	colour-id
1	orange	4
2	Black	5
3	White	6

ACCESSING COMPONENTS BY NAMES:

```
> Print (list - dataframe $ colour - list 2)
```

	colour	colour-id
1	orange	4
2	Black	5
3	White.	6.

ACCESSING TOP LEVEL COMPONENT.

```
> Print ( list - dataframe [[1]] )
```

	Colour	colour-id.
1	Red	1
2	Blue.	2
3.	Green.	3.

ACCESSING INNER LEVEL COMPONENT.

```
> Print ( list - dataframe [[1]] [2] )
```

	colour-id.
1	1
2	2
3	3

ACCESSING ANOTHER INNER LEVEL COMPONENT.

```
> print ( list - dataframe [[1]] [2,2] )
```

[1] 2.

308

MODYFYING COMPONENTS OF DATA FRAME.

```
>print list-dataframe[[1]] [2] = c(21,22,23)
  >print (list-dataframe)
```

```
$ colour-list 1
      colour   colour-id.
1     Red      21
2     Blue     22
3     Green    23
```

```
$ colour-list 2
      colour   colour_id.
1     orange   4
2     Black    5
3     white    6
```

CONCATENATING A NEW DATA FRAME.

```
>df3= data.frame (colour=c("sea Green", "Sky Blue","Peach"),
          Colour_id = c(41,42,43)
>newlist-dataframe = c(list-dataframe,newlist-dataframe)
 >Print (list-dataframe)
```

```
$  colour-list 1
      colour   colour_id.
1     Red      21
2     Blue     22
3     Green    23
```

$ colour-list 2

	colour	colour-id.
1	orange	4
2	Black	5
3	white	6

$ colour-list 3.

	Colour	colour-id.
1	Sea Green	41
2	sky Blue	42
3	Peach.	43.

DELETE OPERATIONS

> list - dataframe [[3]] <- NULL

> print (list - dataframe)

$ colour_list 1

	colour	colour_id
1	Red	21
2	Blue	22
3.	Green	23

$ colour_list 2.

	colour	colour_id
1	orange	4
2	Black	5
3	white	6.

:00:

CONCLUSION: Successfully demonstrated a program to create a dataframe.

PRACTICAL NO. 05.

Aim:- Write a program to create an ordered factor from data consisting of names of months.

Theory: Factors are data objects used to categorize data and store it as levels. They can store a string as well as an integer. They represent coloumns as they have a limited number of unique values. Factors in R can be created using factor() function. It takes a vector as input. c() function is used to create a vector with explicity provided values.

Ordered factors is an extension of factors. It arrange the levels in increasing order.

CODE and OUTPUT:-

```
>months_v=c("March","April","January","November",
  "January","september","october","April","July",
  "May")
>print("original vector")
Original vector
```

800

```
> print (months - v)
[1] "March" "April" "January" "November" "January"
"September" "October" "April" "July" "May".
> f = factor (months - v)
> print("ordered factor of the said vector")
ordered factor of the said vector.
> print (f)
```

CONCLUSION :- Succesfully demonstrated a program to create an ordered factor from data consisting of names of months.

PRACTICAL NO. 06.

Aim :- Write a program to create a dataframe from four vectors.

Theory :- To create a dataframe in R using the vector, we must first have a series of vectors containing data. The data.frame() function is used to create a dataframe from vector in R.

CODE and OUTPUT :-

```
> emp_id = (1, 2, 3, 4, 5, 6, 7)
> emp_name = c("Kanchan", "Insiya", "Pinky",
  "Suraj", "Alsiba", "Ankita", "Aadi")
> emp_location = c("Boisar", "Virar", "Kandivali",
  "Dadar", "Dahanu", "Umroli", "Palghar")
> emp_contact = c(123456, 908765, 456378,
  876345, 987654, 890765, 230967)
> df1 = data.frame(emp_id, emp_name, emp_
  location, emp_contact).
> Print(df1)
```

	emp_id	emp_name	emp_location	emp_contact
1	1	kanchan	Boisar	123456
2	2	Insiya	virar	908765
3	3	pinky	Kandivali	456378
4	4	Suraj	Dadar	876345
5	5	Alsiba	Dahanu	987654
6	6	Ankita	Umroli	890765
7	7	Aadi	Palghar	230967

PRACTICAL NO. 07

Aim :- Write a program to extract 3rd and 5th row with 1st and 3rd columns from a given dataframe.

Theory :- We are using a built-in function dataframe() for this. A data frame is used for storing data tables which has a list of vectors with equal length.
finally, extract the specified rows with specified columns from a given dataframe by calling like [c(3,5), c(1,3)]

CODES and OUTPUT :-

```
> new = df1[(3,5),(1,3)]
> print(new)
```

emp_id	emp_location
3	Bandra.
5	Nallasopara.

CONCLUSION :- Successfully demonstrated a program to extract 3rd & 5th row with 1st & 3rd colums from a given dataframe.

PRACTICAL NO. 08.

Aim :- Write a R program to drop row / column by number from a given dataframe.

Theory :- In this method, the user needs to provide the index of the particular row that is needed to be dropped from the dataframe. The '-' sign indicate dropping rows & columns.

CODES and OUTPUT :-

```
>df1 [-2]
>print (df1)
```

	emp_id	emp_location	emp_contact
1	1	Boisar	123456
2	2	Virar	908765
3	3	Kandivali	456378
4	4	Dadar	876345
5	5	Dahanu	987654
6	6	Umroli	890765
7	7	Palghar.	230967.

```
>df1 [-3,]
>print (df1)
```

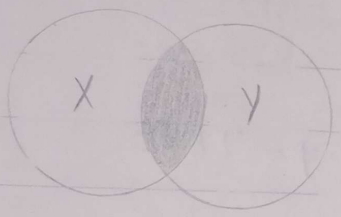
	emp_id	emp_location	emp_contact
1	1	Boisar	123456
2	2	Virar	908765
3	3	Kandivali	456378
4	4	Dadar	876345
5	5	Dahanu	987654
6	6	Umroli	890765
7	7	Palghar.	230967

PRACTICAL No. 09.

Aim :- Write a R program to create inner, outer, left, right join (merge) from given two data frames.

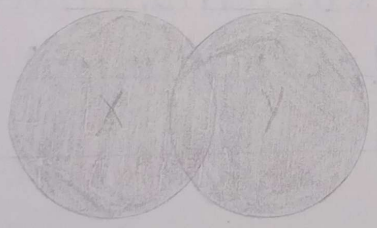
Theory :-

1] **Natural join or Inner join :-** To keep only rows that match from the data frames, specify the arguement all = FALSE.



all = FALSE

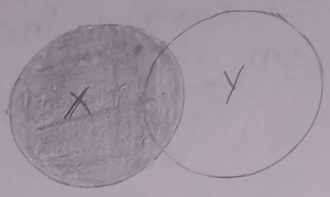
2] **Full outer join or outer join :-** To keep all rows from both data frames, specify all = TRUE.



all = TRUE

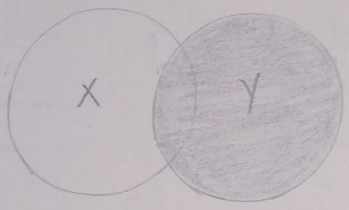
3] **Left outer join or left join :-** To include all the rows of your data from x and only those from y that match, specify x = TRUE

S.T.D.



all.x = TRUE

4] Right outer join or Right join :- To include all the rows of your data frame Y and only those from x that match, specify Y = TRUE.



all.y = TRUE

CODES and OUTPUT :-

```
> df1 = data.frame(numid = c(12,14,10,11)
> df2 = data.frame(numid = c(13,15,11,12))
> Print ("Left outer join")
 [1] " Left outer join.
> result (= merge (df1, df2, by = "numid", all.x = TRUE)
> Print (result)
```

```
   numid.
1    10
2    11
3    12
4    14
```

```
> print ("Right outer join")
  [1] Right outer join.
> result = merge (df1, df2, by = "numid", all.y = TRUE)
> print (result)
```

```
   numid.
1    11
2    12
3    13
4    15
```

```
> print ("Outer join")
  [1] Outer join.
> result = merge (df1, df2, by = "numid", all = TRUE)
> print (result)
```

```
   numid.
1    10
2    11
3    12
4    13
5    14
6    15
```

```
> print ("Inner join")
  [1] Inner join.
```

```
> result = merge (df1, df2, by = "numid", all = FALSE)
> print (result)
      numid
1        11
2        12
```

CONCLUSION :- Successfully demonstrated a program to create inner, outer, left, right join (merge) from given two dataframes.

PRACTICAL NO. 10.

Aim :- Write a program to create a data frame using two given vectors and display the duplicated elements and unique rows of the said dataframe.

Theory :-

Duplicate Data :- For identification, we will use duplicated () function which returns the count of duplicate rows.

Syntax :- duplicated (dataframe)

Using unique :- we use unique () to get rows having unique value in our data.

unique (dataframe)

CODES and OUTPUT :-

```
> emp_id = c(1,2,3,4,5,6,7)
> emp_name = c("Kanchan", "Insiya", "pinky",
  "Suraj", "Alsiba", "Ankita", "Aadi")
> emp_location = c("Boisar", "virar", "kandivali",
  "Dadar", "Dahanu", "Umroli", "palghar")
> emp_contact = c(123456, 908765, 456378, 816345,
  987654, 890765, 230967)
```

PRO

```
>df1 = data.frame(emp_id, emp_name, emp_location, emp_contact)
>print(df1)
```

	emp_id	emp_name	emp_location	emp_contact
1	1	kanchan	Boisar	123456
2	2	Insiya	virar	908765
3	3	pinky	Kandivali	456378
4	4	Suraj	Dadar	876345
5	5	Alsiba	Dahanu	987654
6	6	Ankita	Umroli	890765
7	7	Aadi	Palghar	230967

```
>print(duplicated(df1))
[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
>print(unique(df1))
```

	emp_id	emp_name	emp_location	emp_contact
1	1	kanchan	Boisar	123456
2	2	Insiya	virar	908765
3	3	pinky	Kandivali	456378
4	4	Suraj	Dadar	876345
5	5	Alsiba	Dahanu	987654
6	6	Ankita	Umroli	890765
7	7	Aadi	Palghar	230967

PRACTICAL NO. 11

Aim :- Implement matrices addition, substraction, Multiplication and division.

Theory :- Matrix in R are a bunch of values, either, real or complex numbers, arranged in a group of fixed numbers of rows and columns.

- **MATRICES ADDITION :-** The addition of two same ordered matrices Mr*c and Nr*c yields a matrix Rr*c where every element is the sum of corresponding elements of the inputs matrices.
- **MATRICES SUBTRACTION :-** The substraction of two same ordered matrices Mr*c and Nr*c yields a matrix Rr*c where every elements is the difference of corresponding elements of the second input matrix from the first.
- **MATRICES MULTIPLICATION :-** The multiplication of two same ordered matrices Mr*c and Nr*c yields a matrix Rr*c where every element is the product of corresponding elements of the input matrices
- **MATRICES DIVISION :-** The division of two same ordered matrices Mr*c and Nr*c yields a matrix Rr*c where every element is the quotient of corresponding elements of the first matrix element divided by the second.

10.

CODES and OUTPUT :-

```
> M1 = matrix (c(15,76,23,45,78,98,34,54,67),nrow=3)
> print (M1)
     [,1] [,2] [,3]
[1,]   15   45   34
[2,]   76   78   54
[3,]   23   98   67
> M2 = matrix (c(56,90,98,12,43,69,92,47,47),nrow=3)
> print (M2)
     [,1] [,2] [,3]
[1,]   56   12   92
[2,]   90   43   47
[3,]   98   69   47
> print ("Addition of Matrix")
Addition of Matrix.
> result = M1 + M2.
> print (result).
     [,1] [,2] [,3]
[1,]   71   57  126
[2,]  166  121  101
[3,]  121  167  114.
```

3)

```
> print ("subtraction of Matrix")
Subtraction of Matrix.
> result = M1 - M2
> print (result)
      [,1] [,2] [,3]
[1,]   -41   33   -58
[2,]   -14   35     1
[3,]   -75   29    20.
> print ("Multiplication of Matrix")
Multiplication of matrix.
> result = M1 * M2.
> print (result)
      [,1] [,2] [,3]
[1,]   840  540  3128
[2,]  6840 3354  2538
[3,]  2254 6762  3149.
> print ("Division of Matrix")
Division of Matrix.
> result = M1 / M2
> print (result)
      [,1]   [,2]   [,3]
[1,] 0.2678 3.7500 0.3965
[2,] 0.8444 1.8139 1.1489
[3,] 0.2346 1.4202 1.4255
```

PRACTICAL NO. 12.

Aim :- Demonstrate any five data visualization tools with suitable data.

Theory :-

- **BAR PLOT :-** There are two types of bar plots - horizontal and vertical bars of certain length proportional to the value of data item. By setting horiz = parameter to true and false, we can get horizontal & vertical bar plots respectively.
- **R - Pie charts :-** A pie chart is a circular statistical graphic, which is divided into slices to illustrate numerical proportions. A circular chart cuts in a form of radii into segments describing relative frequencies or magnitude also known as a circle graph.
- **Histogram :-** A histogram contains a rectangular area to display the statistical information which is proportional to the frequency of a variable and its width in successive numerical intervals.
- **Density chart :-** A density plot is a representation of the distribution of a numeric variable that uses a kernel density estimate to show the probability density function of the variable. In R language

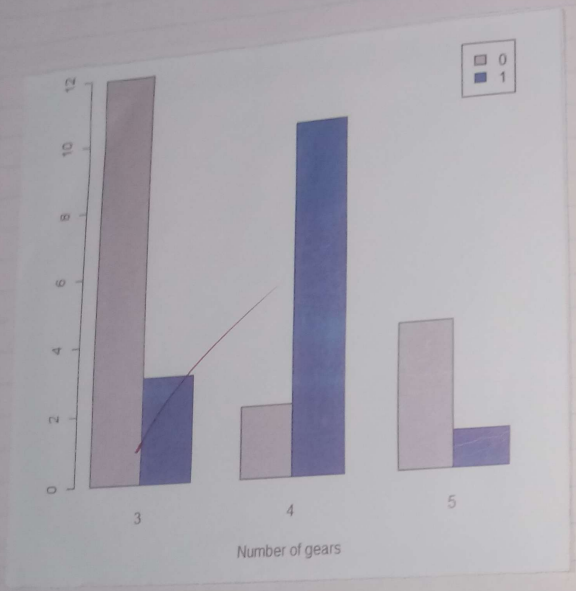
we use the density () function which helps to compute kernel estimates.

- LINE PLOT :- A line graph is a chart that is used to display information in the form of a series of data points It utilizes points and lines to represent change over time. The graph represents different values as it can move up and down based on the suitable variable.

INPUT / SOURCE CODE :-

```
> count ← table (mtcars $ gear)
> barplot (count)
> barplot (count, main horiz = TRUE)
> barplot (count, main = "simple Barplot")
    xlab = "Improvement", ylab = "fequency",
  legend = rownames (count), col = c("red", "Yellow",
          "Green")
> count <- table(mtcars $ vs, mtars $ gear)
> head (count)
> barplot (count, main = "for distribution gear & vs",
  xlab = "Number of Gears", legends = rownames (count),
col = c("grey", "blue", beside = TRUE)
```

Bar chart



- PIE CHART.

```
>slices <-c(10,12,14,16,18)
>lbls <-c("AA","BB","CC","DD","EE")
>pie (slices,labels=lbls,main="simple pie chart")
>pct <-round(slices/sum(slices)*100)
>lbls <-paste(lbls," ",pct,"%",sep=" ")
>pie (slices,labels=lbls,main="pie chart with %",
      col=rainbow)
```

- 3D REPRESENTATION OF PIE CHART.

```
install.packages("plotrix")
library(plotrix)
Pie3D(slices,labels=lbls,main="3D Pie chart",
      col=rainbow(5),explode=0.1)
```

- HISTOGRAM.

```
>mtcars$mpg
>hist(mtcars$mpg)
>hist(mtcars$mpg,breaks=15,col="green")
```

- DENSITY PLOT:

```
>density-data <-density(mtcars$mpg)
>plot(density-data,main="Density of miles per Gallon")
>polygon(density-data,col="skyblue",border=
        "black").
```

OSD

• LINE PLOT

```
weight = c(2.5, 2.8, 3.2, 4.8, 5.1, 5.9, 6.8, 7.1, 7.8, 8.1)
month = c(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
plot(month, weight, type = "b", main = "Baby
                    weight chart")
```

Output:-