

About R Programming:

- Open-source Language - R is freely available to the public. You can make improvements on the language and access hundreds of useful packages created by others for free.
- Domain-specific Language - R is a domain-specific language and its specialty lies in statistical and data analysis.
- Strong Graphical Capabilities - R can also be used for data visualization. It provides extended libraries that will help you produce high quality interactive graphics.
- Interpreted Language - R is an interpreted language like Python.
- Integration with Other Technologies

R is one of the most popular statistical programming languages for data scientists. It is heavily used in the field of machine learning, scientific computing, and statistical analysis.

➤ To install r-base

Step 1: -

- 1) Sudo apt-get update
- 2) Sudo apt-get install r-base

➤ First Program

```
# My first program in R Programming
myString <- "Hello, World!"
print (myString)
```

➤ Creating Variables in R

```
> name= "Sachin Tendulkar"
> age <- 49
> name
[1] "Sachin Tendulkar"
> print(age)
[1] 49
> |
```

Variable Name	Validity	Reason
var_name2.	valid	Has letters, numbers, dot and underscore
var_name%	Invalid	Has the character '%'. Only dot(.) and underscore allowed.
2var_name	invalid	Starts with a number
.var_name, var.name	valid	Can start with a dot(.) but the dot(.)should not be followed by a number.
.2var_name	invalid	The starting dot is followed by a number making it invalid.
_var_name	invalid	Starts with _ which is not valid

➤ Data Types: -

Basic data types in R can be divided into the following types:

numeric - (16.5, 59, 645)

integer - (11L, 5L, 12L, where the letter "L" declares this as an integer)

complex - (7 + 4i, where "i" is the imaginary part)

character (or string) - ("APSIT", "Data Mining", "11", "TRUE")

logical (or Boolean) - (TRUE or FALSE)

We can use the **class ()** function to check the data type of a variable:

```
R Console
> v <- TRUE
> print (class(v))
[1] "logical"
> print (v)
[1] TRUE
>
> # Numeric data type
> v <- 25.5
> print (class(v))
[1] "numeric"
> print (v)
[1] 25.5
>
> #Integer Data type
> v <- 35L
> print (class(v))
[1] "integer"
>
> #Complex data type
> v <- 5+3i
> print (class(v))
[1] "complex"
> print (v)
[1] 5+3i
> |
```

➤ R Arithmetic Operators

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
^	Exponent	x ^ y
%%	Modulus (Remainder from division)	x %% y

➤ R Operator

Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y

>	Greater than	$x > y$
<	Less than	$x < y$
>=	Greater than or equal to	$x \geq y$
<=	Less than or equal to	$x \leq y$

➤ R logical Operator

Operator	Description
&	Element-wise Logical AND operator. It returns TRUE if both elements are TRUE
&&	Logical AND operator - Returns TRUE if both statements are TRUE
	Elementwise- Logical OR operator. It returns TRUE if one of the statement is TRUE
	Logical OR operator. It returns TRUE if one of the statement is TRUE.
!	Logical NOT - returns FALSE if statement is TRUE

➤ Data structures in R :-

The frequently used ones are –

- Vectors
- Lists
- Matrices
- Arrays
- Factors
- Data Frames

1) Vector

When you want to create vector with more than one element, you should use `c()` function which means to combine the elements into a vector.

```
> #create a vector
> states <- c("Maharashtra", "Goa", "Karnataka")
> #print vector
> print (states)
[1] "Maharashtra" "Goa"          "Karnataka"
> #print class/type of the vector
> print (class(states))
[1] "character"
> |
```

2) List :-

A list in R can contain many different data types inside it. A list is a collection of data which is ordered and changeable. To create a list, use the `list()` function:

```

> # Create a list.
> list1 <- list(c(10,15,20),41.3,"APSIT")
>
> # Print the list.
> print(list1)
[[1]]
[1] 10 15 20

[[2]]
[1] 41.3

[[3]]
[1] "APSIT"

> #individual values
> print (list1[1])
[[1]]
[1] 10 15 20

> print (list1[2])
[[1]]
[1] 41.3

```

3) Matrix :-

A matrix is a two-dimensional rectangular data set. It can be created using a vector input to the matrix function.

```

> # Create a matrix.
> M = matrix( c('ab','ab','ac','ac','ab','aa'), nrow = 2, ncol = 3, byrow = TRUE)
> print(M)
      [,1] [,2] [,3]
[1,] "ab" "ab" "ac"
[2,] "ac" "ab" "aa"
> print (M[1])
[1] "ab"
> print (M[1,2])
[1] "ab"
> print (M[1,3])
[1] "ac"
> print (M[2,3])
[1] "aa"
> |

```

4) Arrays :-

While matrices are confined to two dimensions, arrays can be of any number of dimensions. The array function takes a dim attribute which creates the required number of dimensions. In the below example we create an array with two elements which are 3x3 matrices each.

```

> # Create an array.
> a <- array(c('green','yellow'),dim = c(3,3,2))
> print(a)
, , 1

      [,1]      [,2]      [,3]
[1,] "green"  "yellow" "green"
[2,] "yellow" "green"  "yellow"
[3,] "green"  "yellow" "green"

, , 2

      [,1]      [,2]      [,3]
[1,] "yellow" "green"  "yellow"
[2,] "green"  "yellow" "green"
[3,] "yellow" "green"  "yellow"

```

5) Data Frame :-

Create a Data frame

```

> # Create the data frame.
> student <- data.frame(
+   name = c("aa", "ab", "ac"),
+   gender = c("Male", "Male", "Female"),
+   height = c(152, 171.5, 165),
+   branch = c("comp", "comp", "IT"),
+   Age = c(22, 20, 21),
+   Marks = c(90, 75, 85)
+ )
> print(student)
  name gender height branch Age Marks
1  aa   Male   152.0   comp   22    90
2  ab   Male   171.5   comp   20    75
3  ac Female   165.0    IT    21    85
> |

```

DataFrames are generic data objects of R which are used to store the tabular data. Data frames are considered to be the most popular data objects in R programming because it is more comfortable to analyse the data in the tabular form.

Creating a data frame by reading CSV file

```

> print (getwd())
[1] "C:/Users/Archana/Documents"
> #then copy your .csv file into this location.
> data <- read.csv("testexample.csv")
> print(data)
  day outlook temp humidity  wind playball
1   1   sunny  hot     high   weak      no
2   2   sunny  hot     high strong      no
3   3 overcast hot     high   weak     yes
4   4    rain mild    high   weak     yes
5   5    rain cool  normal   weak     yes
6   6    rain cool  normal strong     no
7   7 overcast cool  normal strong    yes
8   8   sunny mild    high   weak     no
9   9   sunny cool  normal   weak     yes
10  10    rain mild  normal   weak     yes
11  11   sunny mild  normal strong    yes
12  12 overcast mild    high strong    yes
13  13 overcast hot    normal   weak    yes
14  14    rain mild    high strong     no
> |

```

Selecting a subset of data frame

```

> data <- read.csv("newexample.csv")
> print(data)
  id  name salary start_date dept
1  1  Rohan  6230 2012-01-01  IT
2  2  Danish  5150 2013-09-23 Operations
3  3 Michelle  6116 2014-11-15  IT
4  4   Ryan   7290 2014-05-11  HR
5  5   Gauri  8430 2015-03-27  Finance
6  6   Nina   5780 2013-05-21  IT
7  7  Simona  6320 2013-07-30 Operations
8  8   Guru   7220 2014-06-17  Finance
> #Get the maximum salary
> # Create a data frame.
> data <- read.csv("newexample.csv")
>
> # Get the max salary from data frame.
> sal <- max(data$salary)
> print(sal)
[1] 8430
> #Get the details of the person with max salary
> # Create a data frame.
> data <- read.csv("newexample.csv")
>
> # Get the max salary from data frame.
> sal <- max(data$salary)
>
> # Get the person detail having max salary.
> max_sal <- subset(data, salary == max(salary))
> print(max_sal)
  id name salary start_date dept
5  5 Gauri   8430 2015-03-27 Finance
> |

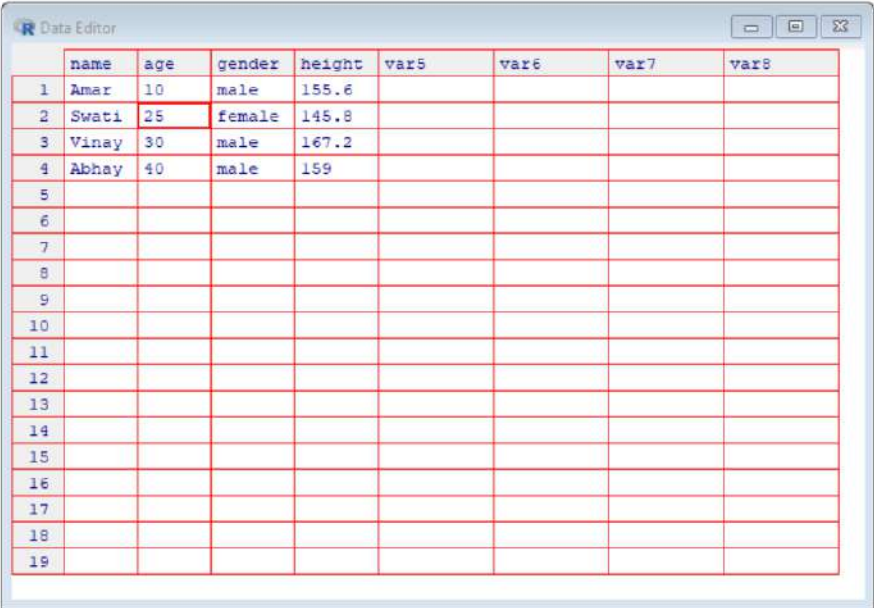
```

Editing the data frame


```

> student
  name age gender height
1 Amar  10   male  155.6
2 Swati 20 female  145.8
3 Vinay 30   male  167.2
4 Abhay 40   male  159.0
>
> mytable=data.frame(student)
> mytable=edit(mytable)

```



	name	age	gender	height	var5	var6	var7	var8
1	Amar	10	male	155.6				
2	Swati	25	female	145.8				
3	Vinay	30	male	167.2				
4	Abhay	40	male	159				
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								

```

> student
  name age gender height
1 Amar  10   male  155.6
2 Swati 20 female  145.8
3 Vinay 30   male  167.2
4 Abhay 40   male  159.0
>
> mytable=data.frame(student)
> mytable=edit(mytable)
>
> print(mytable)
  name age gender height
1 Amar  10   male  155.6
2 Swati 25 female  145.8
3 Vinay 30   male  167.2
4 Abhay 40   male  159.0
> |

```

➤ R Graphics

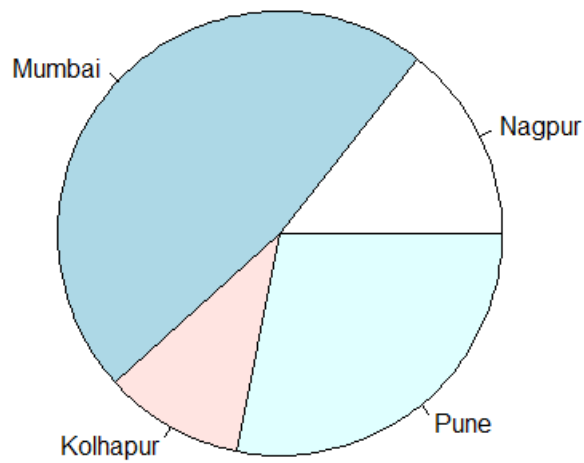
1) Pie chart: -

A pie chart is a circular graphical view of data. Use the pie() function to draw pie charts:

```

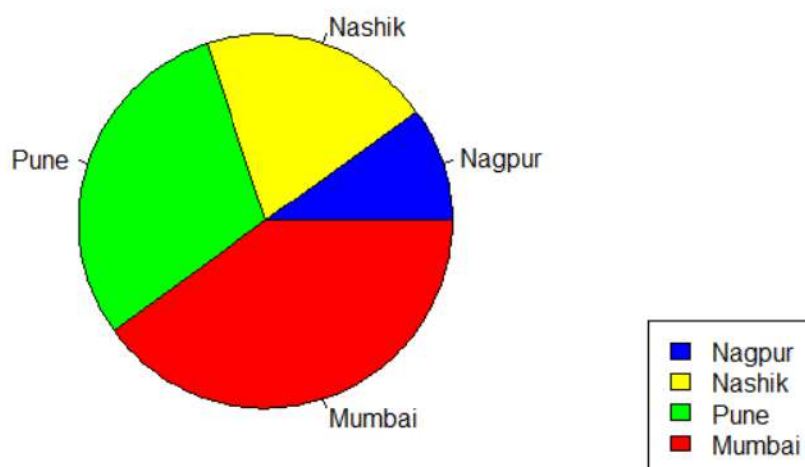
> # Create data for the graph.
> x <- c(21, 69, 15, 41)
> labels <- c("Nagpur", "Mumbai", "Kolhapur", "Pune")
>
> # Plot the chart.
> pie(x, labels)
> |

```



```
> # Create a vector of pies
> x <- c(10,20,30,40)
>
> # Create a vector of labels
> mylabel <- c("Nagpur", "Nashik", "Pune", "Mumbai")
>
>
> # Create a vector of colors
> colors <- c("blue", "yellow", "green", "red")
>
> # Display the pie chart with colors
> pie(x, label = mylabel, main = "City Population", col = colors)
>
> # Display the explanation box
> legend("bottomright", mylabel, fill = colors)
> |
```

City Population



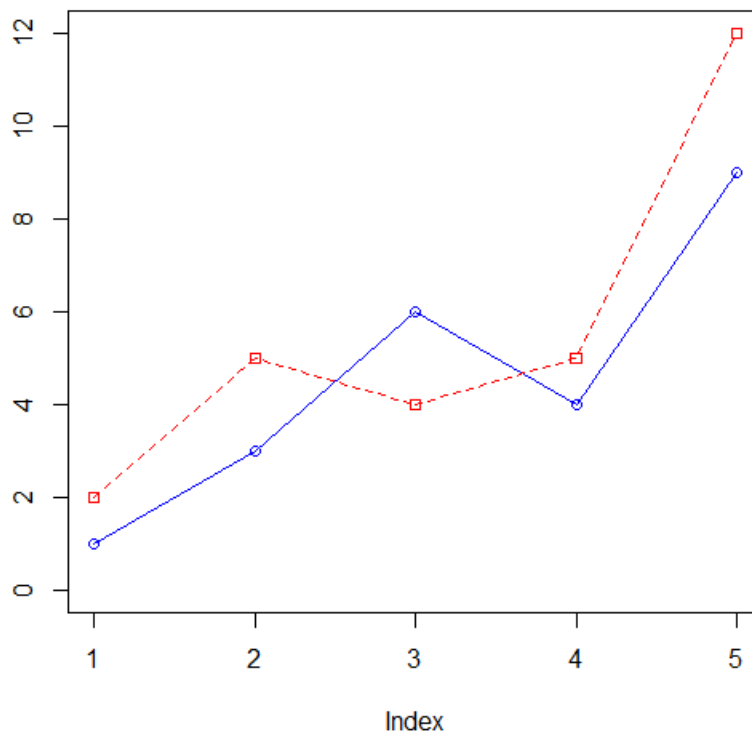
2) Line Charts :-

```
> # Define the cars vector with 5 values
> cars <- c(1, 3, 6, 4, 9)
>
> # Graph the cars vector with all defaults
> plot(cars)
> |
```

```
> # Define the cars vector with 5 values
> cars <- c(1, 3, 6, 4, 9)
>
> # Graph cars using blue points overlayed by a line
> plot(cars, type="o", col="blue")
> |
```

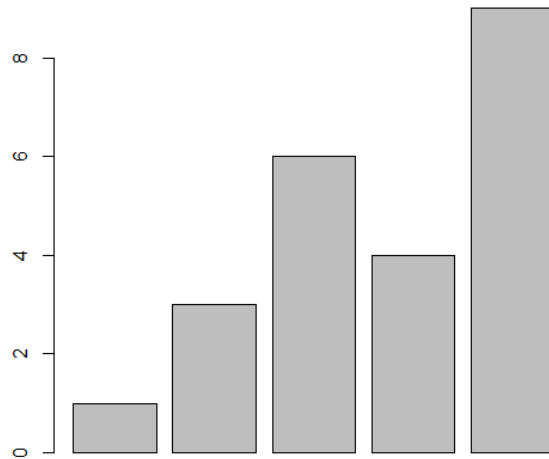
Two vectors for comparison: -

```
> # Define 2 vectors
> cars <- c(1, 3, 6, 4, 9)
> trucks <- c(2, 5, 4, 5, 12)
>
> # Graph cars using a y axis that ranges from 0 to 12
> plot(cars, type="o", col="blue", ylim=c(0,12))
>
> # Graph trucks with red dashed line and square points
> lines(trucks, type="o", pch=22, lty=2, col="red")
> |
```



3) Bar Charts :-

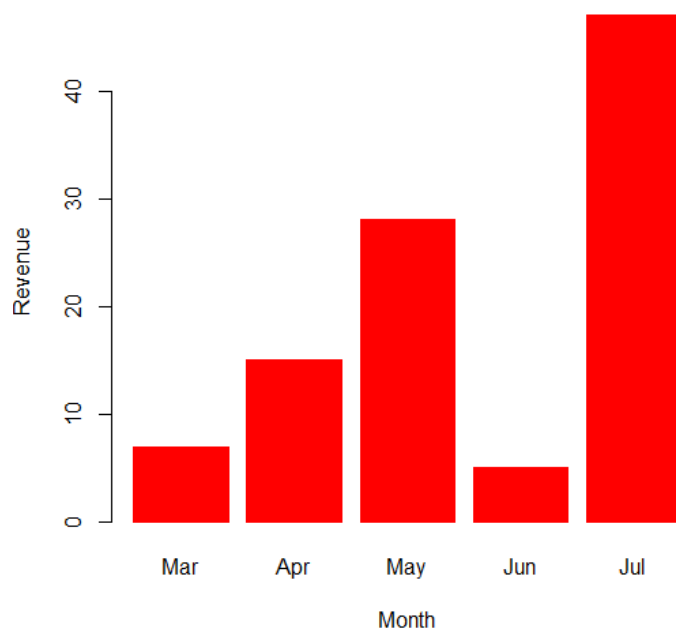
```
> # Define the cars vector with 5 values  
> cars <- c(1, 3, 6, 4, 9)  
>  
> # Graph cars  
> barplot(cars)  
> |
```



Colours in Bar chart

```
> # Create the data for the chart  
> H <- c(7,15,28,5,47)  
> M <- c("Mar", "Apr", "May", "Jun", "Jul")  
>  
> # Plot the bar chart  
> barplot(H,names.arg=M,xlab="Month",ylab="Revenue",col="red",  
+ main="Revenue chart",border="red")  
> |
```

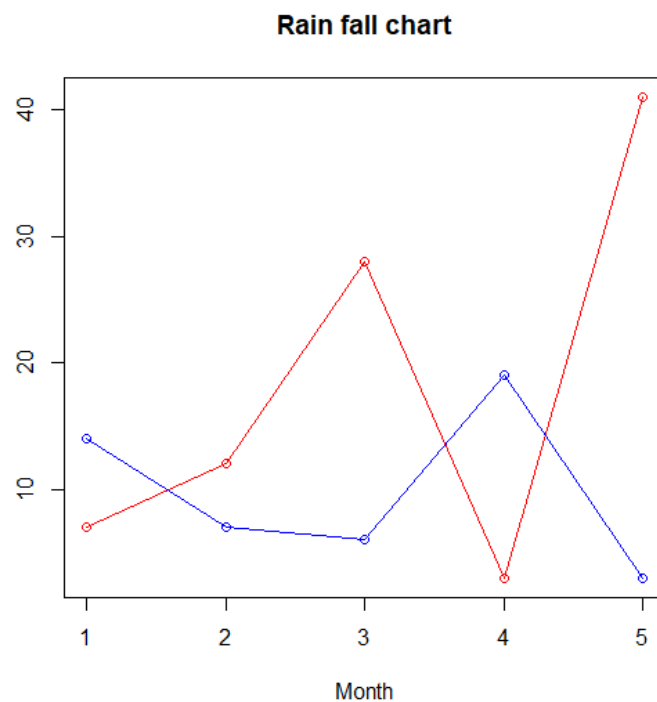
Revenue chart



```

> # Create the data for the chart.
> v <- c(7,12,28,3,41)
> t <- c(14,7,6,19,3)
>
>
>
> # Plot the bar chart.
> plot(v,type = "o",col = "red", xlab = "Month", ylab = "Rain fall",
+      main = "Rain fall chart")
>
> lines(t, type = "o", col = "blue")
>
>

```



Make Bar Plot Horizontal in R

```

temperatures <- c(22, 27, 26, 24, 23, 26, 28)

result <- barplot(temperatures,
  main = "Maximum Temperatures in a Week",
  xlab = "Degree Celsius",
  ylab = "Day",
  names.arg = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"),
  col = "blue",
  density = 20,
  horiz = TRUE
)

print(result)

```

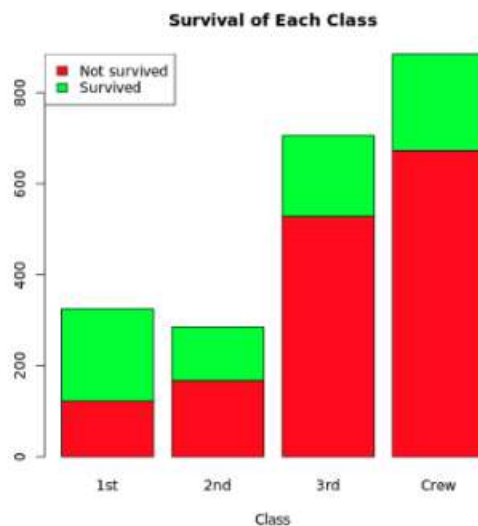
Stacked Bar Plot in R

```
# create a matrix
titanic_data <- matrix(c(122, 203, 167, 118, 528, 178, 673, 212),
  nrow = 2, ncol = 4)

result <- barplot(titanic_data,
  main = "Survival of Each Class",
  xlab = "Class",
  names.arg = c("1st", "2nd", "3rd", "Crew"),
  col = c("red", "green")
)

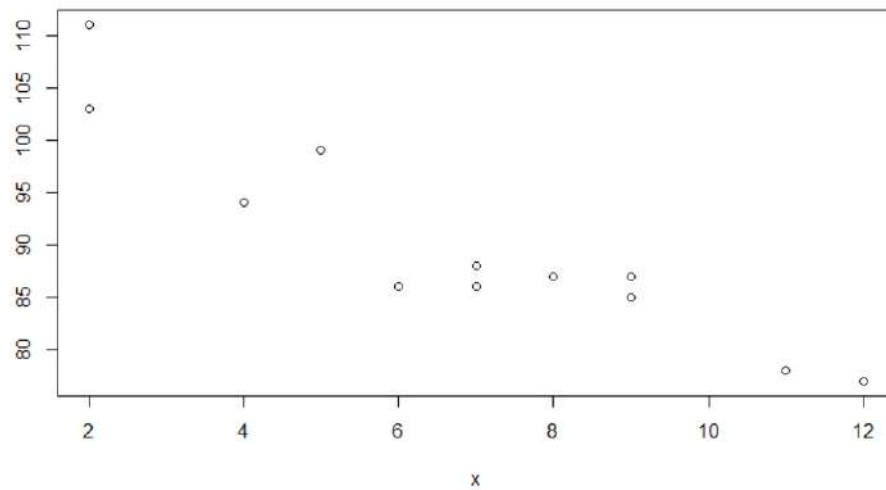
legend("topleft",
  c("Not survived", "Survived"),
  fill = c("red", "green")
)

print(result)
```

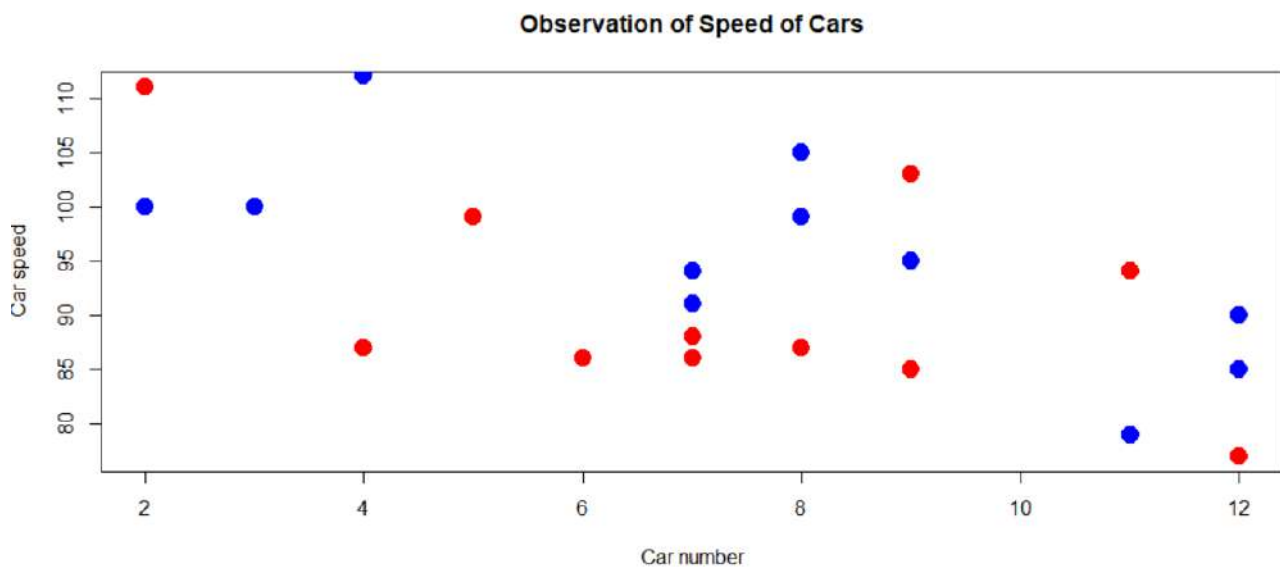


4) Scatter Plot :-

```
> x <- c(5,7,8,7,2,2,9,4,11,12,9,6)
> y <- c(99,86,87,88,111,103,87,94,78,77,85,86)
>
> plot(x, y)|
```



```
> # day one, the carno and speed of 12 cars:
> x1 <- c(5,7,8,7,2,9,4,11,12,9,6)
> y1 <- c(99,86,87,88,111,103,87,94,77,85,86)
>
> # day two, the carno and speed of 15 cars:
> x2 <- c(2,8,1,15,8,12,9,7,3,11,4,7,14,12)
> y2 <- c(100,105,84,90,99,90,95,94,100,79,112,91,80,85)
>
> plot(x1, y1, main="Observation of Speed of Cars", xlab="Car number", ylab="Car speed", col="red", cex=2, pch=19)
> points(x2, y2, col="blue", cex=2, pch=19)
> 
```



5. Box (Whisker) Plot

```
1 v=c(13,15,16,16,19,20,20,21,22,22,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)
2
3 boxplot(v)
```

Add Title, Label, New Colour to a Boxplot in R

```

1 v=c(13,15,16,16,19,20,20,21,22,22,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)
2
3 #boxplot(v)
4
5 # add title, label, new color to boxplot
6 boxplot(v,
7   main="Mileage Data Boxplot",
8   ylab="Miles Per Gallon(mpg)",
9   xlab="No. of Cylinders",
10  col="orange")

```

6.Histogram

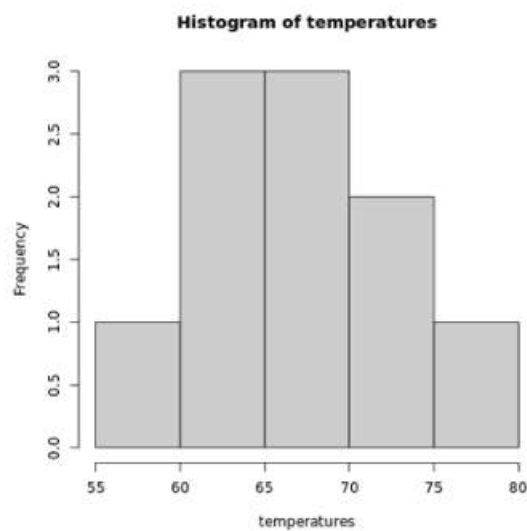
A histogram is a graphical display of data using bars of different heights.

Histogram is used to summarize discrete or continuous data that are measured on an interval scale.

```

1 temperatures <- c(67 ,72 ,74 ,62 ,76 ,66 ,65 ,59 ,61 ,69 )
2
3 # histogram of temperatures vector
4 result <- hist(temperatures)
5
6 print(result)

```



Histogram with title and label

```
1 temperatures <- c(67 ,72 ,74 ,62 ,76 ,66 ,65 ,59 ,61 ,69 )
2
3 # histogram of temperatures vector
4 result <- hist(temperatures,
5   main = "Histogram of Temperature",
6   xlab = "Temperature in degrees Fahrenheit"
7 )
8
9 print(result)
```

Range and colour to the histogram

```
1 temperatures <- c(67 ,72 ,74 ,62 ,76 ,66 ,65 ,59 ,61 ,69 )
2
3 # histogram of temperatures vector
4 result <- hist(temperatures,
5   main = "Histogram of Temperature",
6   xlab = "Temperature in degrees Fahrenheit",
7   col = "red",
8   xlim = c(50,100),
9   ylim = c(0, 5))
0
1 print(result)
```