**Week7:-**

**Aim:-b)Reading Excel data sheet in R**

**Program:-**

#install advanced version R4.2.0 for this packages

install.packages("readxl")

library("readxl")

read\_excel("C:/Desktop/II year.xlsx")

**Output:-**

# A tibble: 13 × 3

SNO NAME ROLLNO

<dbl> <chr> <chr>

1 1 SANJANA 18ND1A0501

2 2 KAVITHA 18ND1A0502

3 3 HARIKA 18ND1A0503

4 4 VINUTHNA 18ND1A0504

5 5 SWAPNA 18ND1A0505

6 6 HARITHA 18ND1A0506

**7)**

**Aim:-c)Reading Xml dataset in R**(It is executed in R studio)

**Program:-**

Input.xml:-

<RECORDS>

  <STUDENT>

      <ID>1</ID>

      <NAME>Alia</NAME>

      <MARKS>620</MARKS>

      <BRANCH>IT</BRANCH>

  </STUDENT>

  <STUDENT>

      <ID>2</ID>

      <NAME>Brijesh</NAME>

      <MARKS>440</MARKS>

      <BRANCH>Commerce</BRANCH>

   </STUDENT>

  <STUDENT>

      <ID>3</ID>

      <NAME>Yash</NAME>

      <MARKS>600</MARKS>

      <BRANCH>Humanities</BRANCH>

   </STUDENT>

  <STUDENT>

      <ID>4</ID>

      <NAME>Mallika</NAME>

      <MARKS>660</MARKS>

      <BRANCH>IT</BRANCH>

   </STUDENT>

  <STUDENT>

      <ID>5</ID>

      <NAME>Zayn</NAME>

      <MARKS>560</MARKS>

      <BRANCH>IT</BRANCH>

   </STUDENT>

</RECORDS>

install.packages("XML")

# Load the package required to read XML files.

library("XML")

# Also load the other required package.

library("methods")

# Give the input file name to the function.

result<-xmlParse(file = "C:/Dektop/input.xml")

# Print the result.

print(result)

**output:-**

1

Alia

620

IT

2

Brijesh

440

Commerce

3

Yash

600

Humanities

4

Mallika

660

IT

5

Zayn

560

IT

**Week8:-**

**Aim:-a) Implement R script to create a pie chart, Bar chat, scatter plot and Histogram (introduction to ggplot2 graphics)**

**Program:-**

print("\*\*\*piechart\*\*\*")

x<-c(10,15,13,16,19)

lbls<-c("january","feb","mar","apr","may")

pie(x,labels=lbls,main="piechart of months sales")

print("\*\*\*barchart\*\*\*")

install.apackages(“barplot”)

# Creating the data for Bar chart

H <- c(12,35,54,3,41)

M<- c("Feb","Mar","Apr","May","Jun")

# Giving the chart file a name

png(file = "bar\_properties.png")

#0000000000000000000000000000000000000000000000000000000000000000000000000 Plotting the bar chart

barplot(H,names.arg=M,xlab="Month",ylab="Revenue",col="Green",

main="Revenue Bar chart",border="red")

# Saving the file

dev.off()

print("\*\*\*scatterplot\*\*\*")

# Get the input values.

input <- mtcars[,c('wt','mpg')]

# Give the chart file a name.

png(file = "scatterplot.png")

# Plot the chart for cars with weight between 2.5 to 5 and mileage between 15 and 30.

plot(x = input$wt,y = input$mpg,

xlab = "Weight",

ylab = "Milage",

xlim = c(2.5,5),

ylim = c(15,30),

main = "Weight vs Milage"

)

# Save the file.

dev.off()

# Create data for the graph.

v <- c(9,13,21,8,36,22,12,41,31,33,19)

# Give the chart file a name.

png(file = "histogram.png")

# Create the histogram.

hist(v,xlab = "Weight",col = "yellow",border = "blue")

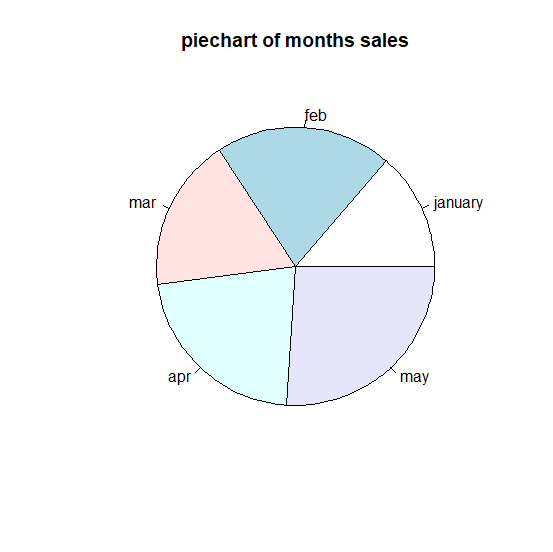
# Save the file.

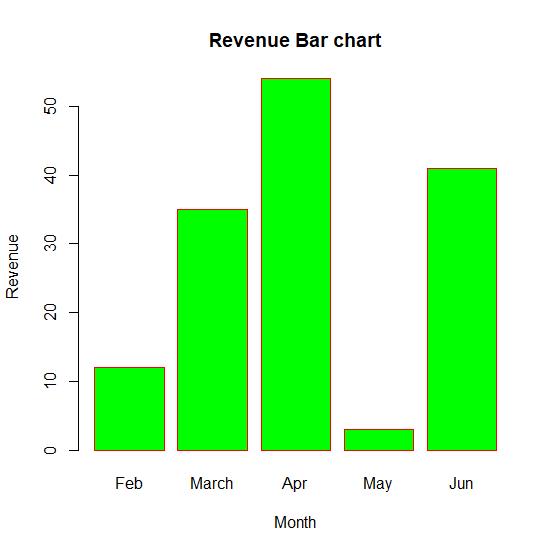
dev.off()

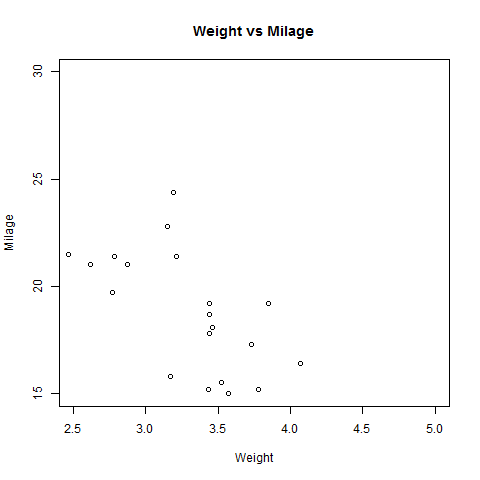
install.package(“ggplot2”)

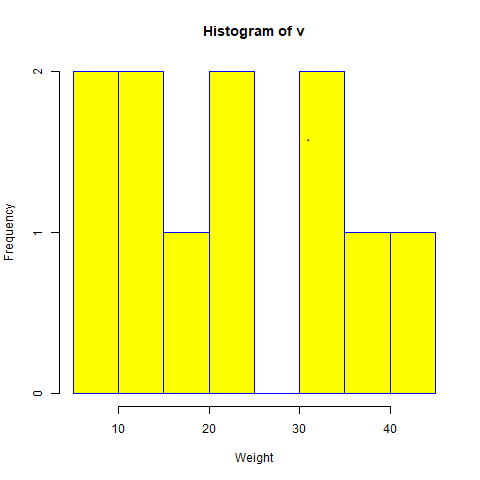
library(ggplot2)

**output:-**

****







**8)**

**Aim:-b) Implement R script to perform mean, median, mode, range, summary, variance, standard deviation operations.**

**Programs:-**

#mean

mean = mean(18,19,19,19,19,20,20)

print(mean)

#median

median = median(18,19,19,19,19,20,20)

print(median)

#mode

find\_mode<-function(x){

u<-unique(x)

tab<-tabulate(match(x,u))

u[tab==max(tab)]

}

data<-c(1,2,2,3,4,4,4,4,5,6)

find\_mode(data)

#range

Range=range(18,19,19,19,19,20,20)

print(Range)

#summary

Summary=summary(18,19,19,19,19,20,20)

print(Summary)

#variance

x=c(18,19,19,19,19,20,20)

V=var(x)

print(V)

#standard devation

x=c(18,19,19,19,19,20,20)

S=sd(x)

print(S)

**Output:-**

[1] 18

[1] 18

[1] 4

[1] 18 20

[1] 0.4761905

[1] 0.6900656

**Week9:-**

**Aim:-a) Implement R script to perform Normal, Binomial distributions.**

**Program:-**

Print(“Nomal distribution using dnorm”)

# creating a sequence of values

# between -15 to 15 with a difference of 0.1

x = seq(-15, 15, by=0.1)

y = dnorm(x, mean(x), sd(x))

# output to be present as PNG file

png(file="dnormExample.png")

# Plot the graph.

plot(x, y)

# saving the file

dev.off()

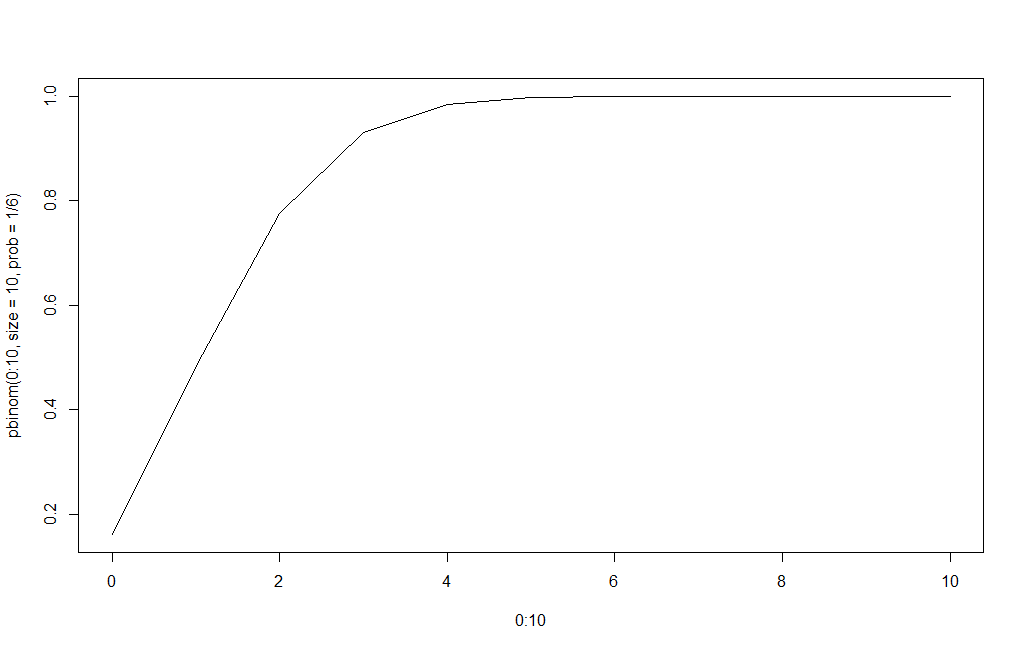
print("binominal distribution using pbinom")

pbinom(3, size = 13, prob = 1 / 6)

plot(0:10, pbinom(0:10, size = 10, prob = 1 / 6), type = "l")

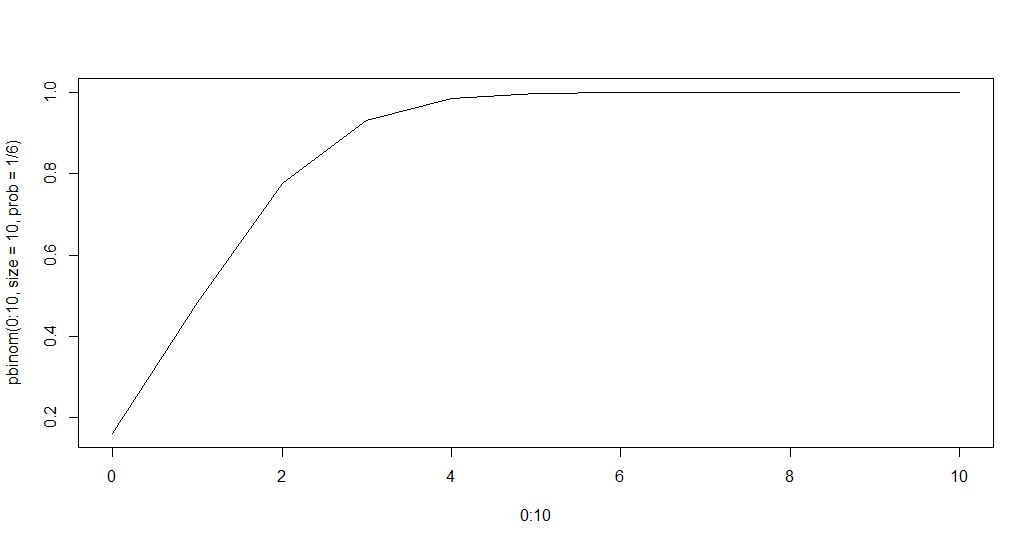
**Output:-**

“Nomal distribution using dnorm”

****

"binominal distribution using pbinom"

[1] 0.8419226

**9)**

**b) Implement R Script to perform correlations, Linear and multiple regression.**

**Correlation:-**

Correlation is one of the most common statistics. Using one single value, it describes the "degree of relationship" between two variables. Correlation ranges from -1 to +1.

**Linear regression:-**

The general mathematical equation for a linear regression is −

y = ax + b

Following is the description of the parameters used −

* **y** is the response variable.
* **x** is the predictor variable.
* **a** and **b** are constants which are called the coefficients.

The basic syntax for **lm()** function in linear regression is −

lm(formula,data)

Following is the description of the parameters used −

* **formula** is a symbol presenting the relation between x and y.
* **data** is the vector on which the formula will be applied.

**Multiple regression:-**

The general mathematical equation for multiple regression is −

y = a + b1x1 + b2x2 +...bnxn

Following is the description of the parameters used −

* **y** is the response variable.
* **a, b1, b2...bn** are the coefficients.
* **x1, x2, ...xn** are the predictor variables.

lm(y ~ x1+x2+x3...,data)

**program:-**

**print("corelation with example")**

x<-seq(-10,10, 1)

y<-x\*x

plot(x,y)

cor(x,y)

**print("understand linear regression")**

# Create the predictor and response variable.

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

relation <- lm(y~x)

# Give the chart file a name.

png(file = "linearregression.png")

# Plot the chart.

plot(y,x,col = "blue",main = "Height & Weight Regression",

abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")

# Save the file.

dev.off()

**print("understand multiple regression")**

input <- mtcars[,c("mpg","disp","hp","wt")]

print(head(input))

input <- mtcars[,c("mpg","disp","hp","wt")]

# Create the relationship model.

model <- lm(mpg~disp+hp+wt, data = input)

# Show the model.

print(model)

# Get the Intercept and coefficients as vector elements.

cat("# # # # The Coefficient Values # # # ","\n")

a <- coef(model)[1]

print(a)

Xdisp <- coef(model)[2]

Xhp <- coef(model)[3]

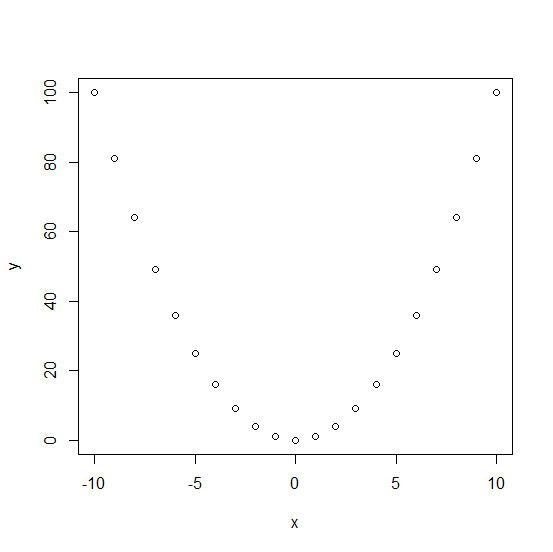
Xwt <- coef(model)[4]

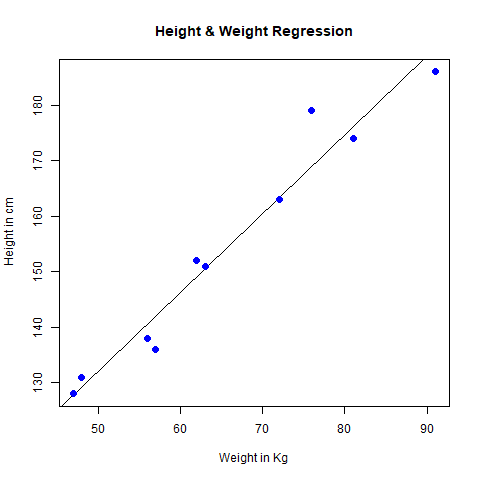
print(Xdisp)

print(Xhp)

print(Xwt)

**output:-**

****

****

lm(formula = mpg ~ disp + hp + wt, data = input)

Coefficients:

(Intercept) disp hp wt

37.105505 -0.000937 -0.031157 -3.800891

# # # # The Coefficient Values # # #

(Intercept)

37.10551

disp

-0.0009370091

hp

-0.03115655

Wt -3.800891

**Week10:-**

**Aim:- Introduction to Non tabular Data Types: Time series, spatial data, Network data**

**Nontabular datatype:-**

When referring to data or computer output, nontabular output refers to any data that is not formatted in a table.

**Creating a time series**:-  
  
**The ts() function will convert a numeric vector into an R time series object**. The format is ts(vector, start=, end=, frequency=) where start and end are the times of the first and last observation and frequency is the number of observations per unit time (1=annual, 4=quartly, 12=monthly, etc.).

**These four components are:**

* Secular trend, which describe the movement along the term;
* Seasonal variations, which represent seasonal changes;
* Cyclical fluctuations, which correspond to periodical but not seasonal variations;
* Irregular variations, which are other nonrandom sources of variations of series.

**Spatial data:-**

* Spatial data is **any type of data that directly or indirectly references a specific geographical area or location**. Sometimes called geospatial data or geographic information, spatial data can also numerically represent a physical object in a geographic coordinate system.

**[](https://www.google.com/search?rlz=1C1CHZN_enIN1004IN1004&q=What+is+spatial+dataset?&tbm=isch&source=iu&ictx=1&vet=1&fir=ODa01R0vAyuT5M,hVajSBvjCLY0SM,_&usg=AI4_-kST6iwHgakPiIhZCXYrBCxd-QQHBg&sa=X&ved=2ahUKEwiQv86n0sr3AhUWTGwGHVHHCIAQ9QF6BAgaEAE#imgrc=ODa01R0vAyuT5M)**

**Program:-**

Print(“time series”)

# Get the data points in form of a R vector.

rainfall <- c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,998.6,784.2,985,882.8,1071)

# Convert it to a time series object.

rainfall.timeseries <- ts(rainfall,start = c(2012,1),frequency = 12)

# Print the timeseries data.

print(rainfall.timeseries)

# Give the chart file a name.

png(file = "rainfall.png")

# Plot a graph of the time series.

plot(rainfall.timeseries)

# Save the file.

dev.off()

#Network data

## Download and install the package

install.packages("igraph") ## Load package

library(igraph)

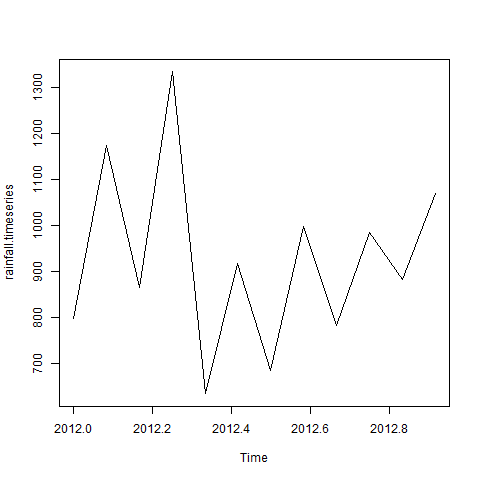
## Create undirected graphs

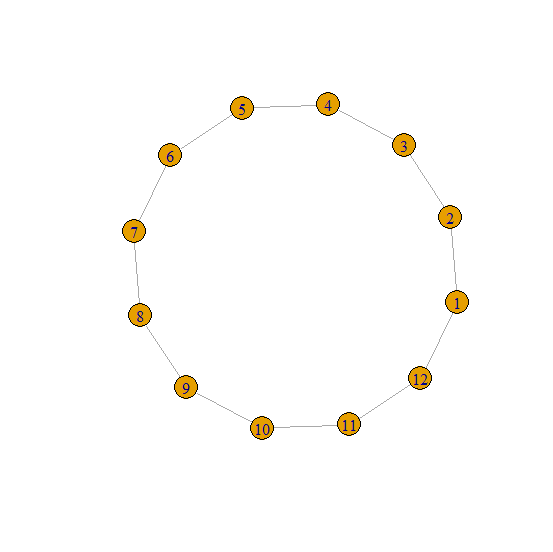
g <- graph\_from\_literal(1-2, 1-3, 1-7, 3-4, 2-3, 2-4, 3-5, 4-5, 4-6, 4-7, 5-6, 5-8, 6-7, 7-8)

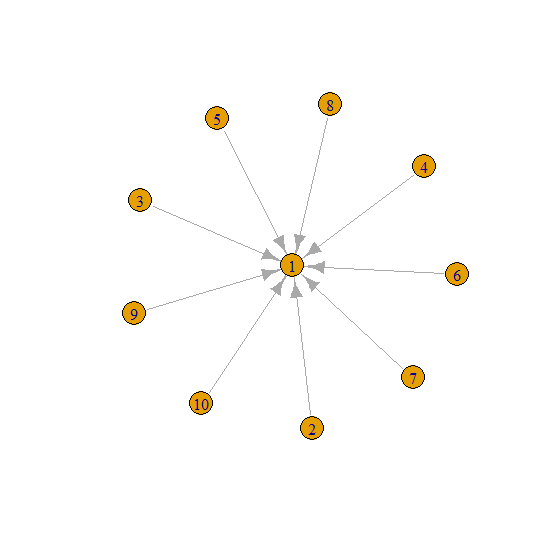
## Create directed graphs using addition or substraction operators

dg <- graph\_from\_literal(JFK-+PEK, JFK-+CDG, PEK++CDG)

**output:-**

****

****

****

IGRAPH cc50df2 UN-- 8 14 --

+ attr: name (v/c)

+ edges from cc50df2 (vertex names):

[1] 1--2 1--3 1--7 2--3 2--4 3--4 3--5 7--4 7--6 7--8 4--5 4--6 5--6 5--8

IGRAPH cc53417 DN-- 3 4 --

+ attr: name (v/c)

+ edges from cc53417 (vertex names):

[1] JFK->PEK JFK->CDG PEK->CDG CDG->PEK