DATE:	**BRANCHING STATEMENTS**
EX NO:01	

AIM:

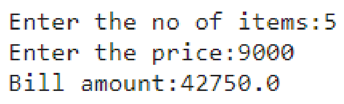
ALGORITHM:

CODING:

a. Program to get price and quantity of a product, calculate bill amount and calculate 10% discount for the bill amount 5000 and above

```
noi=int(input("Enter the no of items:"))
price=float(input("Enter the price:"))
amount=noi*price
if(amount>=5000):
   discount=amount*0.05
   amount=amount-discount
print("Bill amount:{}".format(amount))
```

OUTPUT:



```
Enter the no of items:5
Enter the price:9000
Bill amount:42750.0
```

b. Program to get a number num and check whether num is odd or even

```
num1=int(input("Enter your number:"))
if(num1%2==0):
            print("{} is even".format(num1))
else:
            print("{} is odd".format(num1))
```

OUTPUT:

```
Enter your number:5
5 is odd
```

c. Program to get a number num and check whether the last digit of num is divisible by 3

```
num=int(input("Enter a number:"))
last_digit=num%10
if(last_digit%3==0):
	print("{} is divisible by 3".format(last_digit))
else:
	print("{} is not divisible by 3".format(last_digit))
```

OUTPUT:

```
Enter a number:5
5 is not divisible by 3
```

d. Program to perform basic calculation operation

```
num1=int(input("Enter first number:"))
op=input("Enter operation:")
num2=int(input("Enter second number:"))
if op=='+':
	result=num1+num2
elif op=='-':
	result=num1-num2
elif op=='*':
	result=num1*num2
elif op=='/':
	result=num1/num2
elif op=='%':
	result=num1%num2
else:
	result="invalid operation"
print(result)
```

OUTPUT:

```
Enter first number:5
Enter operation:+
Enter second number:5
10
```

e. Program to get a number num and check whether num is prime or composite

```
n=int(input("Enter a number:"))
c=0
for i in range(1,n+1):
        if n%i==0:
                c=c+1
        if n==0:
                print("neither prime nor composite")
        elif c<=2:
                print("{} is prime".format(n))
        else:
                print("{} is composite".format(n))
```

OUTPUT:

```
Enter a number:5
5 is prime
```

RESULT:

DATE:	**LOOPING STATEMENTS**
EX NO:02	

AIM:

ALGORITHM:

CODING:

a. Program to find the factors of a number

```
def print_factors(x):
print("The factors of",x,"are:")
        for i in range(1,x+1):
                if x%i==0:
        print(i)
num=int(input("Enter a number:"))
print_factors(num)
```

OUTPUT:

```
Enter a number:9
The factors of 9 are:
1
3
9
```

b. Program to sort alphabetically the words from a string provided by the user

```
my_str=input("Enter  a string:")
words=my_str.split()
words.sort()
print("The sorted words are:")
for word in words:
        print(word)
```

OUTPUT:

```
Enter  a string:s t a y positive
The sorted words are:
a
positive
s
t
y
```

c. Python program to check if the number provided by the user is an Armstrong number or not

```
num=int(input("Enter a number:"))
sum=0
temp=num
        while temp>0:
                digit=temp%10
                sum+=digit**3
                temp//=10
if num==sum:
print(num,"is an armstrong number")
else:
print(num,"is not an armstrong number")
```

OUTPUT:

```
Enter a number:5
5 is not an armstrong number
```

d. Program to add two matrices using nested loop

```
x=[[12,7,3],[4,5,6],[7,8,9]]

y=[[5,8,1],[6,7,3],[4,5,9]]

result=[[0,0,0],[0,0,0],[0,0,0]]

for i in range(len(x)):

  for j in range(len(x[0])):

    result[i][j]=x[i][j]+y[i][j]

    for r in result:

      print(r)
```

OUTPUT:

```
[17, 0, 0]
[0, 0, 0]
```

```
[0, 0, 0]
[17, 15, 0]
[0, 0, 0]
[0, 0, 0]
[17, 15, 4]
[0, 0, 0]
[0, 0, 0]
[17, 15, 4]
[10, 0, 0]
[0, 0, 0]
[17, 15, 4]
[10, 12, 0]
[0, 0, 0]
[17, 15, 4]
[10, 12, 9]
[0, 0, 0]
[17, 15, 4]
[10, 12, 9]
[11, 0, 0]
[17, 15, 4]
[10, 12, 9]
[11, 13, 0]
[17, 15, 4]
[10, 12, 9]
[11, 13, 18]
```

e. Program to reverse a given number

```
n=int(input("Enter number:"))

rev=0

while(n>0):

    dig=n%10

    rev=rev*10+dig

    n=n//10

print("Reverse of the number:",rev)
```

OUTPUT:

```
Enter number:64
Reverse of the number: 46
```

RESULT:

DATE:	**FUNCTIONS AND STRINGS**
EX NO:03	

AIM:

ALGORITHM:

CODING:

a. Program to replace all occurrences of ‘a’ with $ in a String

```
string=input("Enter string:")

string=string.replace('a','$')

string=string.replace('A','$')

print("Modified string:")

print(string)
```

OUTPUT:

```
Enter string:Ahead
Modified string:
$he$d
```

b. Program to remove the nth index character from a non-empty string

```
def remove(string,n):

    first=string[:n]

    last=string[n+1:]

    return first+last

string=input("Enter the string:")

n=int(input("Enter the index of the character to remove:"))

print("Modified string:")

print(remove(string,n))
```

OUTPUT:

```
Enter the string:hi hello
Enter the index of the character to remove:3
Modified string:
hi ello
```

c. Program to count number of lowercase characters in a string

```
string=input("Enter string:")

count=0

for i in string:

    if(i.islower()):

        count=count+1

print("The number of lowercase characters is:")

print(count)
```

OUTPUT:

```
Enter string:Hi Hello
The number of lowercase characters is:
5
```

d. Program to calculate the number of uppercase letters and lowercase letters in a string

```
string=input("Enter string:")

count1=0

count2=0

for i in string:

    if(i.islower()):

        count1=count1+1

    elif(i.isupper()):

        count2=count2+1

print("The number of lowercase characters is:")

print(count1)

print("The number of uppercase characters is:")

print(count2)
```

OUTPUT:

```
Enter string:Hi Hello
The number of lowercase characters is:
5
The number of uppercase characters is:
2
```

e. Python program to calculate the number of words and the number of characters present in a string

```
string=input("Enter string:")

char=0

word=1

for i in string:

    char=char+1

    if(i==' '):

        word=word+1

print("Number of words in the string:")

print(word)

print("Number of characters in the string:")

print(char)
```

OUTPUT:

```
Enter string:this is an example
Number of words in the string:
4
Number of characters in the string:
18
```

f. String Capitalize

```
str="this is string example....wow!!!";

print("str.capitalize():",str.capitalize())

print("str.title():",str.title())

print("str.upper():",str.upper())
```

OUTPUT:

```
str.capitalize(): This is string example....wow!!!
str.title(): This Is String Example....Wow!!!
str.upper(): THIS IS STRING EXAMPLE....WOW!!!
```

g. Count

```
str="th is is string example...wow!!!"

sub="i"

print("str.count(sub,4,40):",str.count(sub,4,40))

sub="is"

print("str.count(sub):",str.count(sub))
```

OUTPUT:

```
str.count(sub,4,40): 2
str.count(sub): 2
```

h. isalnum()

```
str="this2009";

print(str.isalnum())

str="this is string example...wow!!!"

print(str.isalnum())
```

OUTPUT:

```
True
False
```

i. isdigit()

```
str="123456";
print(str.isdigit())
str="this is string example...wow!!!"
print(str.isdigit())
```

OUTPUT:

```
True
False
```

j. Join

s="*"

seq=("a","b","c")

print(s.join(seq))

OUTPUT:

```
a*b*c
```

k. istitle()

str="This Is Good"

print(str.istitle())

str="This is good"

print(str.istitle())

OUTPUT:

```
True
False
```

l. Max()

str="this is really a string example...wow!!!"

print("Max Character:"+max(str))

str="this is a string example...wow!!!"

print("Max Character:"+max(str))

OUTPUT:

```
Max Character:y
Max Character:x
```

RESULT:

DATE:	**LIST**
EX NO:04	

AIM:

ALGORITHM:

CODING

a. Program to find the Largest Number in a list

```
a=[]

n=int(input("Enter number of elements:"))

for i in range(1,n+1):

    b=int(input("Enter element:"))

    a.append(b)

    a.sort()

print("Largest element is:",a[n-1])
```

OUTPUT:

```
Enter number of elements:3
Enter element:2
Enter element:7
Enter element:2
Largest element is: 7
```

b. Program to Merge Two Lists and Sort it

```
a=[]
c=[]
n1=int(input("Enter number of elements:"))
for i in range(1,n1+1):
    b=int(input("Enter element:"))
    a.append(b)
n2=int(input("Enter number of elements:"))
for i in range(1,n2+1):
    d=int(input("Enter element:"))
    c.append(d)
new=a+c
new.sort()
print("Sorted list is:",new)
```

OUTPUT:

```
Enter number of elements:2
Enter element:4
Enter element:3
Enter number of elements:2
Enter element:2
Enter element:1
Sorted list is: [1, 2, 3, 4]
```

c. Program to swap the first and last value of a List

```
a=[]
n=int(input("Enter the number of elements in list:"))
for x in range(0,n):
    element=int(input("Enter element"+str(x+1)+":"))
    a.append(element)
temp=a[0]
a[0]=a[n-1]
a[n-1]=temp
print("New list is:")
print(a)
```

OUTPUT:

```
Enter the number of elements in list:2
Enter element1:100
Enter element2:90
New list is:
[90, 100]
```

d. Program to remove the duplicate items from a list

```
a=[]
n=int(input("Enter the number of elements in list:"))
for x in range(0,n):
    element=int(input("Enter element"+str(x+1)+":"))
    a.append(element)
b=set()
unique=[]
for x in a:
    if x not in b:
        unique.append(x)
        b.add(x)
print("Non-duplicate items:")
print(unique)
```

OUTPUT:

```
Enter the number of elements in list:2
Enter element1:2
Enter element2:2
Non-duplicate items:
[2]
```

e. Program to read a list of words and return the length of the longest one

```
a=[]
n=int(input("Enter the number of elements in list:"))
for x in range(0,n):
    element=input("Enter element"+str(x+1)+":")
    a.append(element)
max1=len(a[0])
temp=a[0]
for i in a:
    if(len(i)>max1):
        max1=len(i)
        temp=i
print("The word with the longest length is:")
print(temp)
```

OUTPUT:

```
Enter the number of elements in list:3
Enter element1:welcome
Enter element2:to
Enter element3:python
The word with the longest length is:
welcome
```

RESULT:

DATE:	**TUPLE**
EX NO:05	

AIM:

ALGORITHM:

CODING

a. Write a Python program to add an item in a tuple

```
tuplex=(4,6,2,8,3,1)

print(tuplex)

tuplex=tuplex+(9,)

print(tuplex)

tuplex=tuplex[:5]+(15,20,25)+tuplex[5:]

print(tuplex)

listx=list(tuplex)

listx.append(30)

tuplex=tuple(listx)

print(tuplex)
```

OUTPUT:

```
(4, 6, 2, 8, 3, 1)
(4, 6, 2, 8, 3, 1, 9)
(4, 6, 2, 8, 3, 15, 20, 25, 1, 9)
(4, 6, 2, 8, 3, 15, 20, 25, 1, 9, 30)
```

b. Write a Python program to get the 4th element from last of a tuple

```
tuplex=("w",3,"r","e","s","o","u","r","c","e")

print(tuplex)

item=tuplex[3]

print(item)

item1=tuplex[-4]

print(item1)
```

OUTPUT:

```
('w', 3, 'r', 'e', 's', 'o', 'u', 'r', 'c', 'e')
e
u
```

c. Write a Python program to count the elements in a list until an element is a tuple

```
num=[10,20,30,(10,20),40]

ctr=0

for n in num:

    if isinstance(n,tuple):

        break

    ctr+=1

print(ctr)
```

OUTPUT:

```
3
```

d. Write a Python program to convert a list of tuples into a dictionary

```
lt=[("x",1),("x",2),("x",3),("y",1),("y",2),("z",1)]

d={}

for a,b in lt:

    d.setdefault(a,[]).append(b)

print(d)
```

OUTPUT:

```
{'x': [1, 2, 3], 'y': [1, 2], 'z': [1]}
```

e. Program to count the occurrences of each element in a tuple

```
elements=('apple','banana','cherry','apple','cherry','cherry','banana')

element_count={}

for item in elements:

    if item in element_count:

        element_count[item]+=1

    else:

         element_count[item]=1

print("Count of each element in the tuple:")

for item, count in element_count.items():

    print(f"{item}:{count}")
```

OUTPUT:

```
Count of each element in the tuple:
apple:2
banana:2
cherry:3
```

RESULT:

DATE:	**DICTIONARY**
EX NO:06	

AIM:

ALGORITHM:

CODING:

a. Python Program to add a key-value pair to the dictionary

```
key=int(input("Enter the key (int) to be added:"))
value=int(input("Enter the value for the key to be added:"))
d={}
d.update({key:value})
print("Updated dictionary is:")
print(d)
```

OUTPUT:

```
Enter the key (int) to be added:5
Enter the value for the key to be added:4
Updated dictionary is:
{5: 4}
```

b. Python Program to concatenate two Dictionaries into one

```
d1={'A':1,'B':2}
d2={'C':3}
d1.update(d2)
print("Concatenated dictionary is:")
print(d1)
```

OUTPUT:

```
Concatenated dictionary is:
{'A': 1, 'B': 2, 'C': 3}
```

c. Python program to generate a dictionary that contains numbers (between 1 to n) in the form (x, x*x)

```
n=int(input("Enter a number:"))
d={x:x*x for x in range(1,n+1)}
print(d)
```

OUTPUT:

```
Enter a number:5
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

d. Python Program to sum all the items in a dictionary

```
d={'A':100,'B':540,'C':239}
print("Total sum of values in the dictionary:")
print(sum(d.values()))
```

OUTPUT:

```
Total sum of values in the dictionary:
879
```

e. Python program to map two lists into a dictionary

```
keys=[]
values=[]
n=int(input("Enter number of elements for dictionary:"))
print("For keys:")
for x in range(0,n):
    element=int(input("Enter element"+str(x+1)+":"))
keys.append(element)
print("For values:")
for x in range(0,n):
    element=int(input("Enter element"+str(x+1)+":"))
values.append(element)
d=dict(zip(keys,values))
print("The dictionary is:")
print(d)
```

OUTPUT:

```
Enter number of elements for dictionary:4
For keys:
Enter element1:5
Enter element2:4
Enter element3:3
Enter element4:2
For values:
Enter element1:2
Enter element2:3
Enter element3:4
Enter element4:5
The dictionary is:
{2: 5}
```

f. Python program to create a Dictionary with key as first character and values as works starting with the character:

```
test_string=input("Enter string:")
I=test_string.split()
d={}
for word in I:
    if(word[0]not in d.keys()):
        d[word[0]]=[]
        d[word[0]].append(word)
    else:
        if(word not in d[word[0]]):
            d[word[0]].append(word)
    for k,v in d.items():
        print(k,":",v)
```

OUTPUT:

```
Enter string:Apple
A : ['Apple']
```

RESULT:

DATE:	**VISUALIZING THE DATA FOR SAMPLE DATA SET USING MATPLOTLIB LIBRARY**
EX NO:07	

AIM:

ALGORITHM:

CODING:

```
### Sample Dataset
import pandas as pd
# Sample dataset
data={
    'Month':['January','February','March','April','May','June'],
    'Sales':[200,220,250,300,280,320],
    'Profit':[50,60,70,80,90,100]
}
df=pd.DataFrame(data)
### 1. Line Plot
import matplotlib.pyplot as plt
plt.figure(figsize=(10,6))
plt.plot(df['Month'],df['Sales'],marker='o',color='b',label='Sales')
plt.title('Sales Over Months')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.grid(True)
plt.legend()
plt.show()
### 2. Bar Chart
plt.figure(figsize=(10,6))
plt.bar(df['Month'],df['Sales'],color='orange')
plt.title('Monthly Sales')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.xticks(rotation=45)
plt.show()
### 3. Pie Chart
plt.figure(figsize=(8,8))
plt.pie(df['Sales'],labels=df['Month'],autopct='%1.1f%%',startangle=140)
plt.title('Sales Distribution by Month')
plt.axis('equal')
plt.show()
```

### 4. Scatter Plot

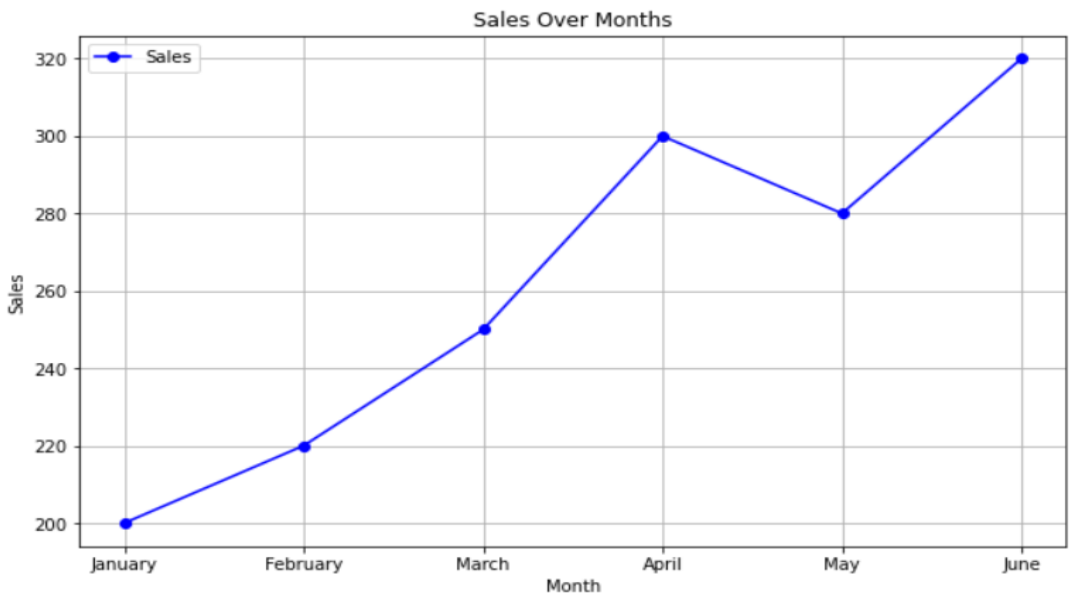
```
plt.figure(figsize=(10,6))
plt.scatter(df['Sales'],df['Profit'],color='green',s=100)
plt.title('Sales vs Profit')
plt.xlabel('Sales')
plt.ylabel('Profit')
plt.grid(True)
plt.show()
```

### 5. Histogram

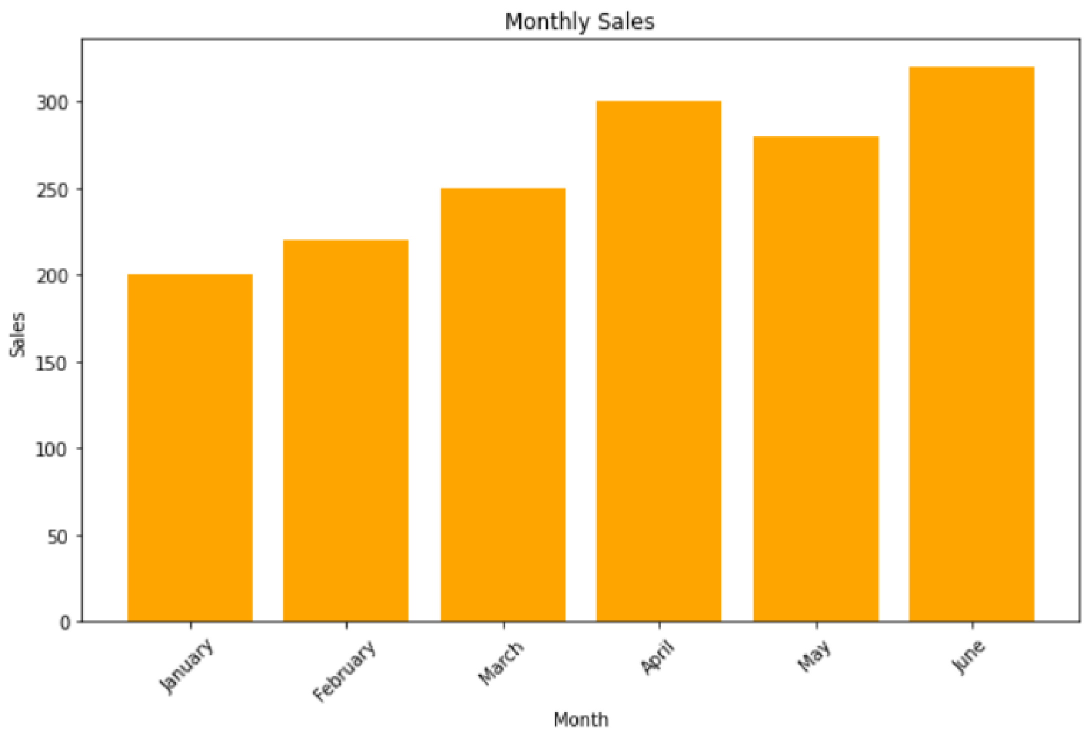
```
plt.figure(figsize=(10,6))
plt.hist(df['Sales'],bins=5,color='purple',alpha=0.7)
plt.title('Sales Distribution')
plt.xlabel('Sales')
plt.ylabel('Frequency')
plt.grid(axis='y')
plt.show()
```

OUTPUT:

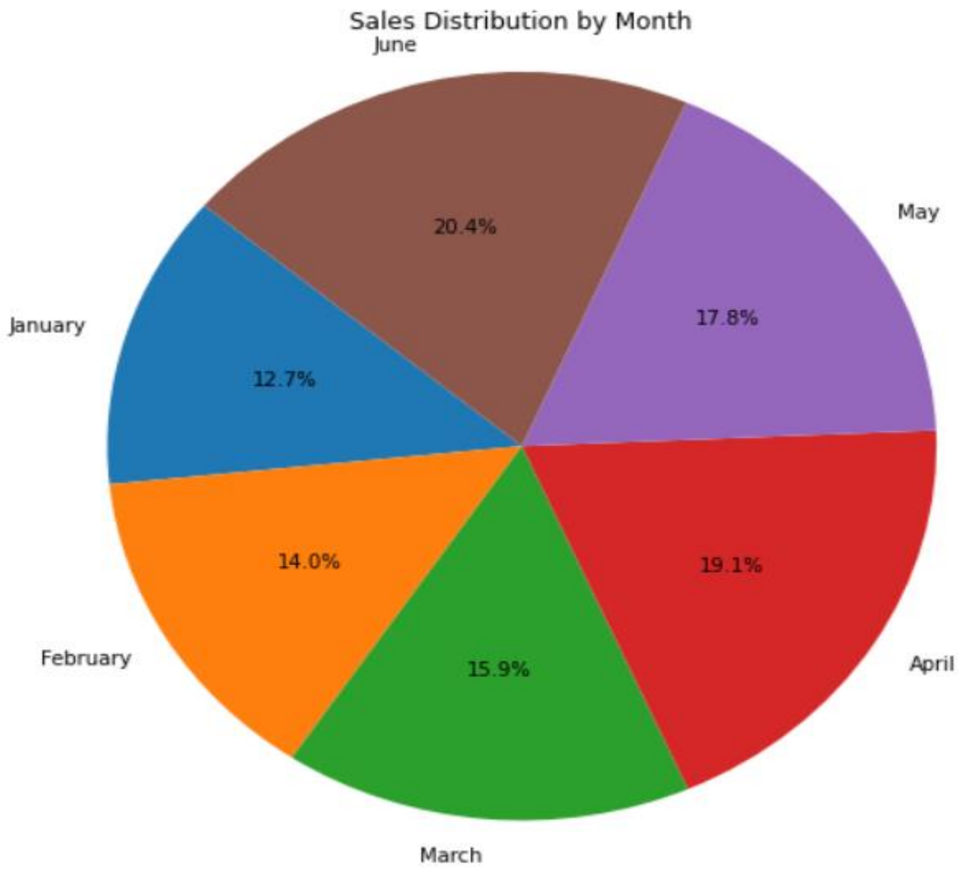
Line Plot



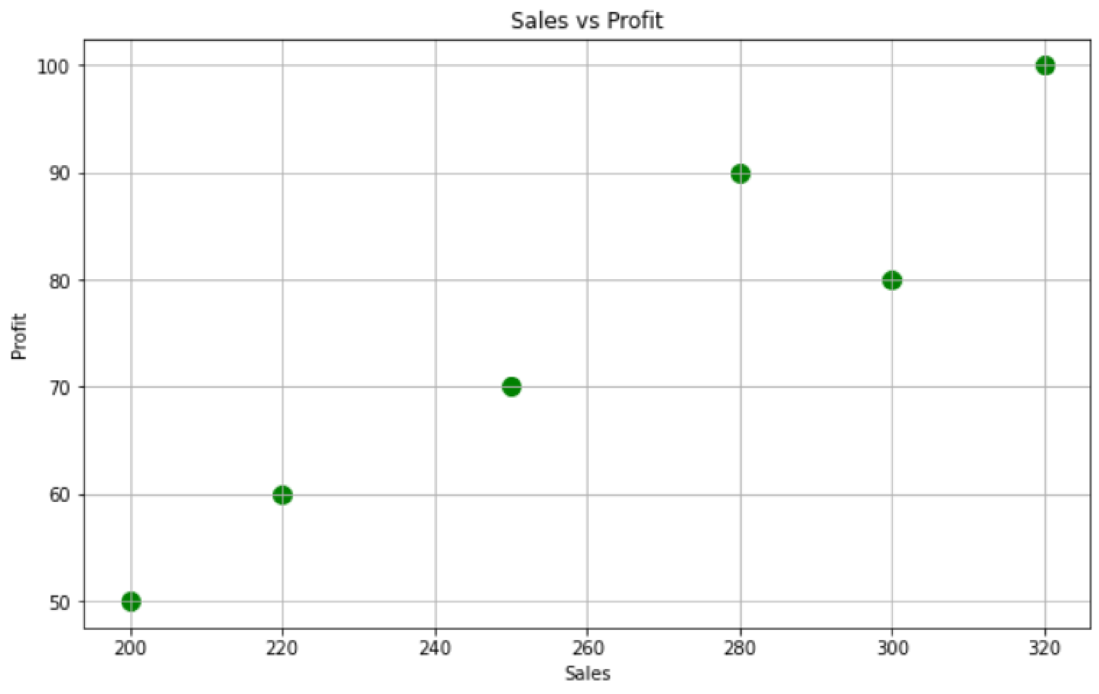
Bar Chart



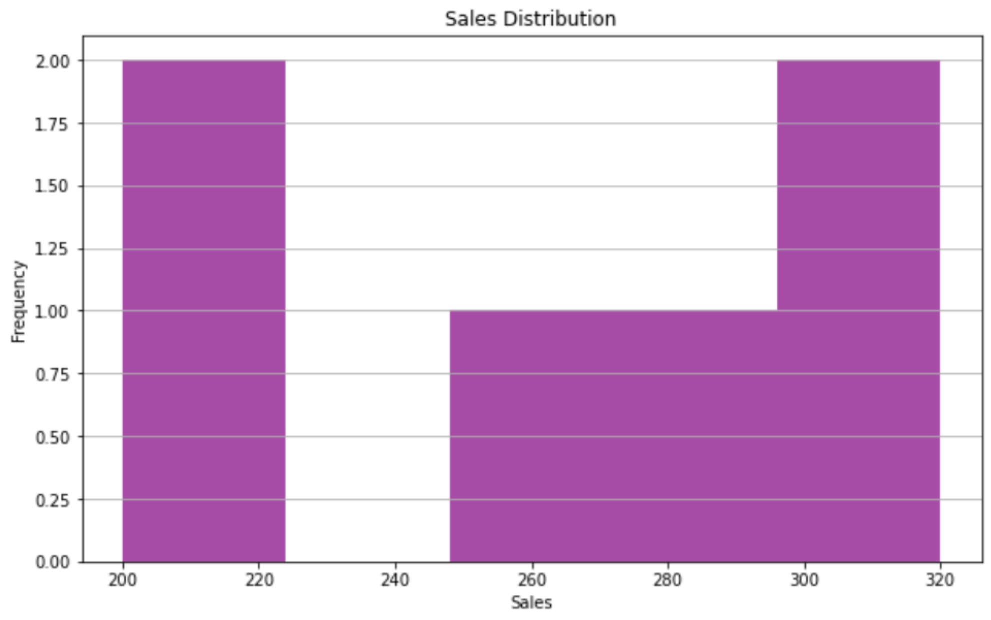
Pie Chart



Scatter Plot



Histogram



RESULT:

DATE:	**SIMPLE LINEAR REGRESSION MODEL USING SAMPLE DATA SET**
EX NO:08	

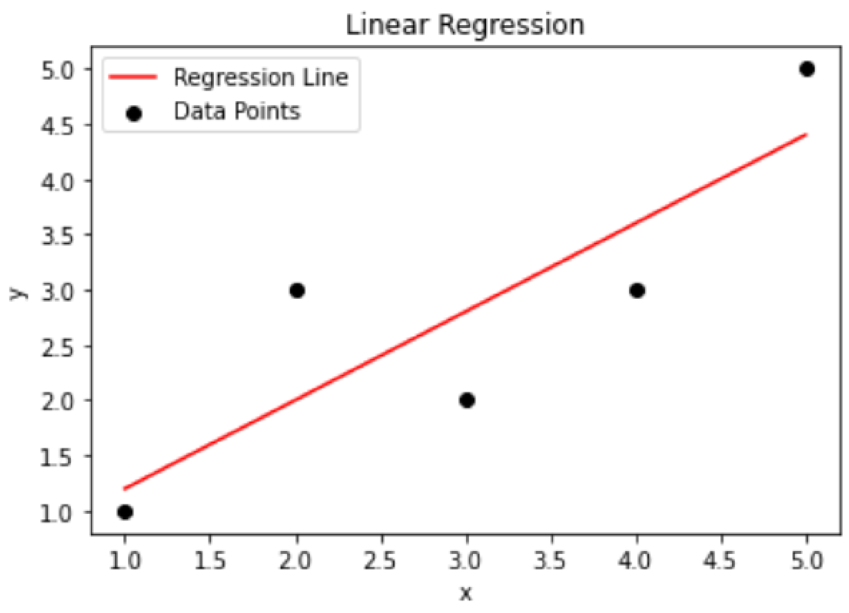
AIM:

ALGORITHM:

CODING

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
# Generate some sample data
x=np.array([1,2,3,4,5])
y=np.array([1,3,2,3,5])
# Save data to a CSV file
data=pd.DataFrame({'x':x,'y':y})
data.to_csv('data.csv',index=False)
# Load data
data=pd.read_csv('data.csv')
X=data['x'].values.reshape(-1,1)
Y=data['y'].values.reshape(-1,1)
# Fit model
model=LinearRegression()
model.fit(X,y)
# Predict on data
y_pred=model.predict(X)
# Visualize
plt.scatter(X,y,color='k',label='Data Points')
plt.plot(X,y_pred,color='r',label='Regression Line')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Linear Regression')
plt.legend()
plt.show()
# R-squared value
r2=model.score(X,y)
print("R-squared:",r2)
```

OUTPUT:



```
R-squared: 0.7272727272727273
```

RESULT:

DATE:	**CLASSIFICATION MODEL FOR A SAMPLE TRAINING DATA USING PYTHON STANDARD LIBRARIES**
EX NO:09	

AIM:

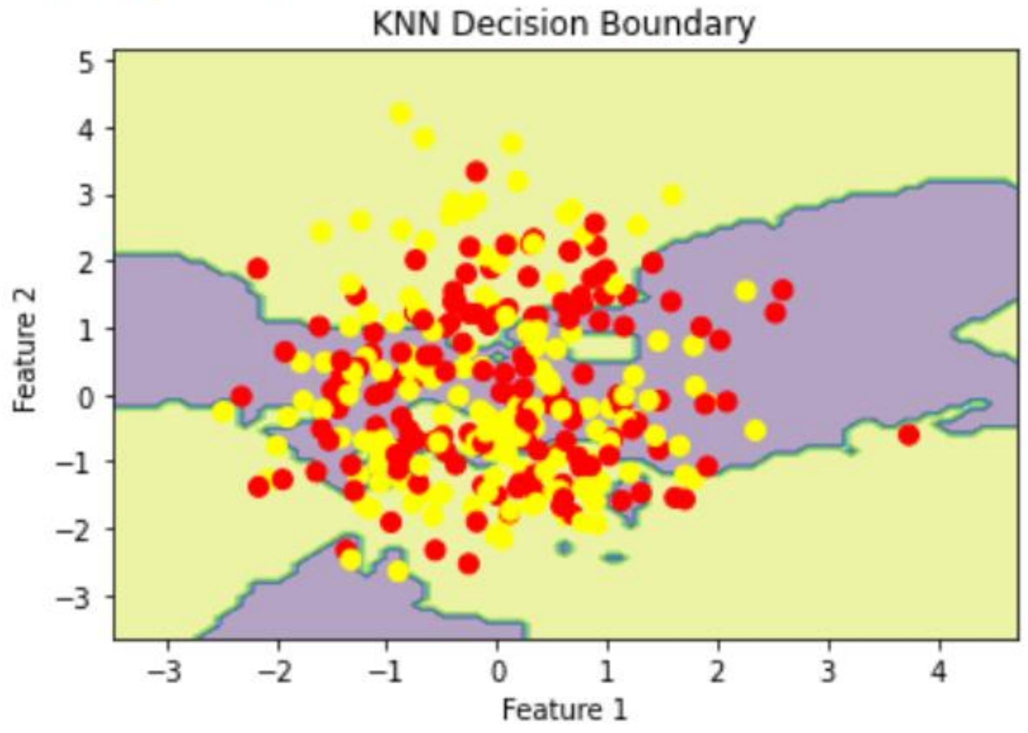
ALGORITHM:

CODING

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier
# Generate sample data
X,y=make_classification(n_samples=1000,n_features=4,n_informative=2,n_redundant=0,ran
dom_state=1)
# Split data
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.30,random_state=42)
# Train model
knn=KNeighborsClassifier()
knn.fit(X_train,y_train)
# Evaluate model
y_pred=knn.predict(X_test)
print("Accuracy:",accuracy_score(y_test,y_pred))
# Decision boundary visualization(using onlt the first two features for visualization)
x_min,x_max=X_test[:,0].min()-1,X_test[:,0].max()+1
y_min,y_max=X_test[:,1].min()-1,X_test[:,1].max()+1
xx,yy=np.meshgrid(np.arange(x_min,x_max,0.1),np.arange(y_min,y_max,0.1))
# Predict on the grid
Z=knn.predict(np.c_[xx.ravel(),yy.ravel(),np.zeros_like(xx.ravel()),np.zeros_like(xx.ravel())]
)
Z=Z.reshape(xx.shape)
# Plot decision boundary
plt.contourf(xx,yy,Z,alpha=0.4)
plt.scatter(X_test[:,0],X_test[:,1],c=y_test,s=50,cmap='autumn')
plt.title('KNN Decision Boundary')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()
```

OUTPUT:

```
Accuracy: 0.86
```



RESULT:

DATE:	**WORKING OF DECISION TREE USING AN APPROPRIATE DATA SET FOR BUILDING THE DECISION TREE AND APPLY THIS KNOWLEDGE TO CLASSIFY A NEW SAMPLE**
EX NO:10	

AIM:

ALGORITHM:

CODING

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
# Load iris dataset
iris=load_iris()
X,y=iris.data,iris.target
# Split into train-test
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=42)
# Train Decision tree
dt=DecisionTreeClassifier()
dt.fit(X_train,y_train)
# Evaluate on test set
y_pred=dt.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
print("Accuracy:",accuracy)
# Classify new sample
new_sample=[[5.1,3.5,1.4,0.2]]
new_pred=dt.predict(new_sample)
print("New Prediction:",new_pred)
```

OUTPUT:

```
Accuracy: 1.0
New Prediction: [0]
```

RESULT: