20MCA241DATA SCIENCE LAB

Lab Report SubmittedBy

JISHA CHACKO

Reg. No.:AJC20MCA-2044

In Partial fulfillment for the Award of the Degree Of

MASTER OF COMPUTER APPLICATIONS (2 Year)
(MCA)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2020-2022

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

CERTIFICATE

This is to certify that the Lab report, "**20MCA241 DATA SCIENCE LAB"** is the bonafide work of **JISHA CHACKO(Reg.No:AJC20MCA-2044)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2021-22.

Ms. Shelly Shiju George

Lab In-Charge

CONTENT

Date:24/11/2021

PROGRAM NO:1

AIM: Perform all matrix operations using python (using numpy).

PROGRAM

```
import numpy

x=numpy.array([[2,4],[7,5]])

y=numpy.array([[5,6],[4,7]])

print("Matrix Addition")

print(numpy.add(x,y))

print("Matrix Subraction")

print(numpy.subtract(x,y))

print("Matrix

multiplication")

print(numpy.multiply(x,y))

print("Matrix product")

print(numpy.dot(x,y))

print("Matrix square root")

print(numpy.sqrt(x))

print("Matrix divison")

print(numpy.divide(x,y))

print("Matrix sum of

element")

print(numpy.sum(x))
```

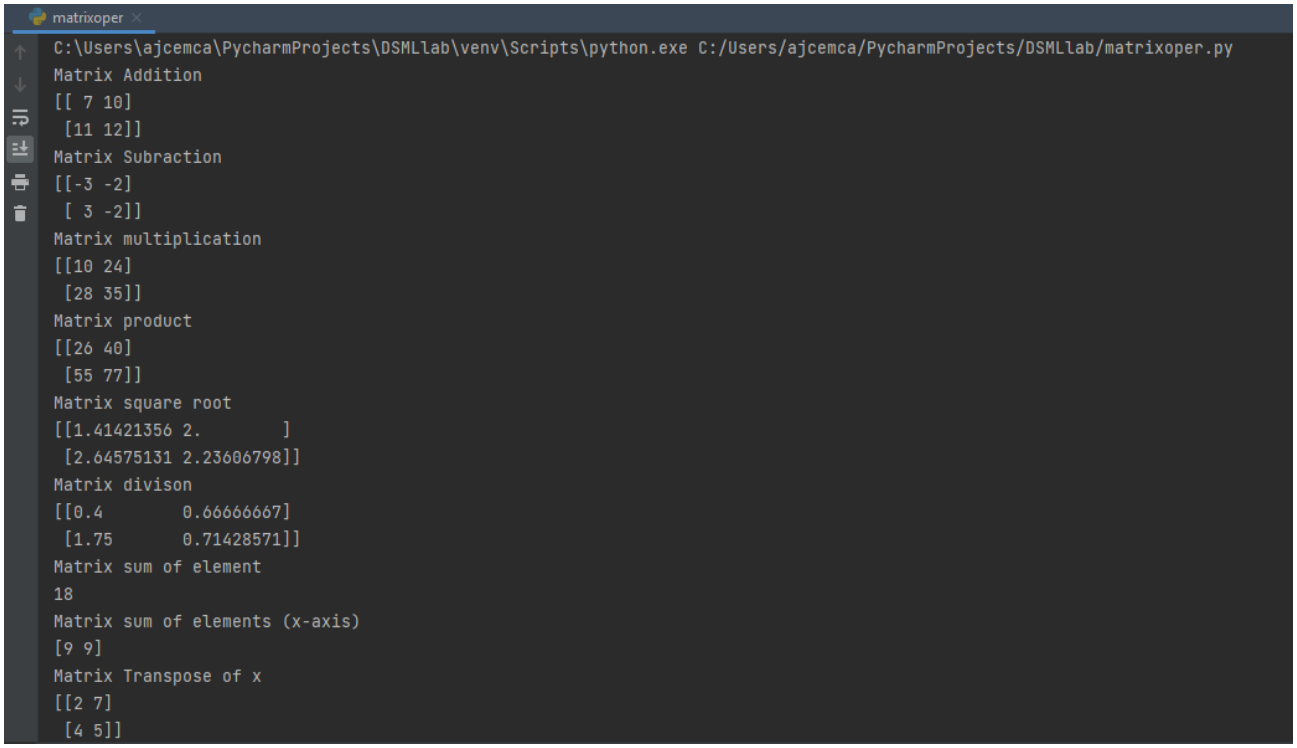
print("Matrix sum of elements (x-

axis)") print(numpy.sum(x,axis=0))

print("Matrix Transpose of

x") print(x.T)

OUTPUT



PROGRAM NO:2

Date:01/12/2021

AIM: Perform SVD(Singular Value Decomposition)

PROGRAM

```
from numpy import

array from

scipy.linalg import

svd

a=array([[1,2,3,4],[7,8,3,5],[4,6,9,10]])

print(a)

u,s,vt=svd(a)

print("Decompod

Matrix\n",u)

print("Inverse Matrix\n",s)

print("Transpose

matrix\n",vt)
```

OUTPUT

```
C:\Users\ajcemca\PycharmProjects\pythonProject\venv\Scrip
[[ 1  2  3  4]
 [ 7  8  3  5]
 [ 4  6  9 10]]
Decomposed Matrix
 [[-0.27122739  0.25018762  0.92943093]
 [-0.575834   -0.81593689  0.05159647]
 [-0.77126579  0.52120355 -0.36537097]]
Inverse Matrix
 [19.40153082  5.77253959  0.5083193 ]
Transpose matrix
 [[-0.38074978 -0.50391495 -0.48875402 -0.60184619]
 [-0.5849343  -0.50236097  0.5185905   0.36952567]
 [-0.336162    0.15621646 -0.67921184  0.63345308]
 [-0.63235795  0.68505445  0.17565499 -0.31617898]]

Process finished with exit code 0
```

PROGRAM NO:3

Date:01/12/2020

AIM:

Program to implement k-NN classification using any standard dataset available in the public domain and find the accuracy of the algorithm

PROGRAM

```
fromsklearn.neighbors  import KNeighborsClassifier
from sklearn.model_selection  import train_test_split
 from sklearn.datasets  import load_iris
from sklearn.metrics  import accuracy_score


idata=load_ir is() x=idata.data  y=idata.target
x_train,x_test,y_train,y_test=train_test_split(  x,y,test_size=0.3,random_state=55)
knn=KNeighborsClassifier(n_neighbors=3)  knn.fit(x_train,y_train)  y_p=knn.predict(x_test)
print(knn.predict(x_test))
print("Accuracy  score : ",accuracy_score(y_test,y_p))
```

OUTPUT

```
C:\Users\ajcemca\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/aj
[0 0 0 2 2 0 2 2 0 0 0 1 2 0 2 2 1 0 1 2 1 2 1 2 1 1 2 1 2 1 0 0 2 2 0 1 1
 0 2 1 2 0 1 0 1]
Accuracy score :  0.9555555555555556

Process finished with exit code 0
```

PROGRAM NO:4

Date:01/12/2021

AIM:

Program to implement k-NN classification using any random data set without using inbuilt packages.

PROGRAM

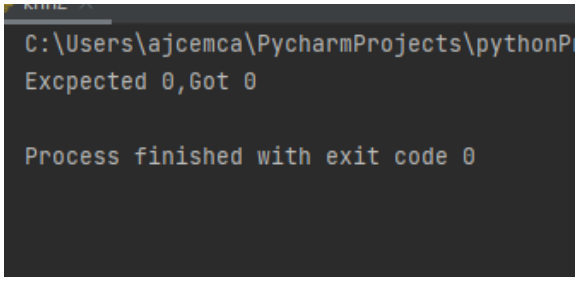
```
from math import
sqrt def
e_dis(r1,r2):
  dist=0.0
for i in
  range(len(r1)-
  1): dist+=(r1[i]-
  r2[i])**2
  return sqrt(dist)
def
 get_ne(train,test_row,num_n
 eig): distances=list()
for train_row in train:
  dist=e_dis(test_row,train_row)
  distances.append([test_row,train_row
  ]) distances.sort(key=lambda
  tup:tup[1])
neighbors=list() for i in
```

```
 range(num_neig):
    neighbors.append(distances[i
    ][0]) return neighbors
def
 predict_classif(train,test_row,num_ne
 ig): neighbors =
 get_ne(train,test_row,num_neig)

 out_val=[row[-1]  for row in neighbors]

    prediction=max(set(out_val),key=out_val.count
    ) return prediction

 dataset=[[2.734,
          2.5 5,0],
     [1.45,3.36,0],
       [2.334, 2.355, 0],
       [1.45, 3.36, 0],
       [2.334, 2.55, 0],
       [1.45, 3.336, 0],
       [3.334, 3.55, 1],
       [1.45, 3.36, 1],
       [3.734, 4.55, 1],
       [3.45, 4.36, 1],
       [4.734, 5.55, 1],
```

[3.45, 5.36, 1]]

prediction=predict_classif(dataset,dataset[0],3) print('Excpected %d,Got %d'%(dataset[0][-1],prediction))

OUTPUT



PROGRAM NO:5

Date:08/12/2021

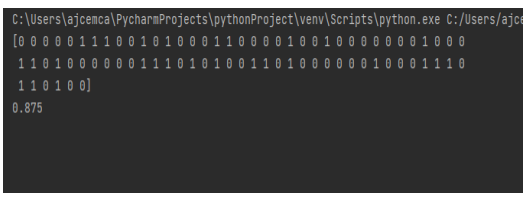
AIM:

Program to implement Naïve Bayes Algorithm using any standard dataset available in the public domain and find the accuracy of the algorithm

Program:

```
import pandas as pd
from sklearn.model_selection import
train_test_split from sklearn.preprocessing
import StandardScaler from
sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix,accuracy_score

dataset=pd.read_csv('Social_Network_Ad
s.csv') x=dataset.iloc[:,[2,3]].values
y=dataset.iloc[:,-1].values

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30)


sc=StandardScaler()
x_train=sc.fit_transform(x_trai
n) x_test=sc.transform(x_test)

classifier=GaussianNB()
classifier.fit(x_train,y_train)

y_pred=classifier.predict(x_test
) print(y_pred)

ac =
accuracy_score(y_test,y_pre
d) print(ac)
```

OUTPUT



PROGRAM NO:6

Date: 08/01/2022

AIM:

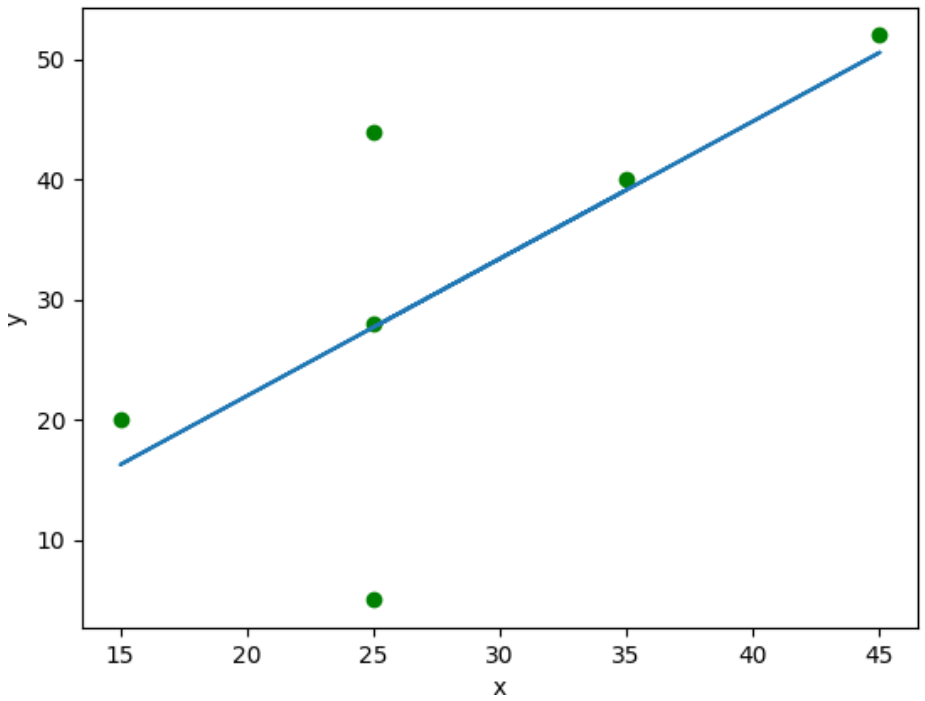
Program to implement linear regression techniques using any standard dataset available in the public domain and evaluate its performance.

PROGRAM (inbuilt):

```
import numpy as np

import matplotlib.pyplot as plt from sklearn.linear_model import LinearRegression
x=np.array([5,15,25,35,45,55]).reshape((-1,1)) y=np.array([5,20,14,32,22,38])
pri nt(x)
pri nt(y) model=LinearRegression() model.fit(x,y) r_sq=model.score(x,y)
print('coefficent of determination: ',r_sq) print('intercept: ',model.intercept_) print('slope :
',model.coef_) y_pred=model.predict(x) print('Predicted response: ',y_pred)


plt.scatter(x,y,color="g") plt.plot(x,y_pred) plt.xlabel('x')
plt.ylabel('y') plt.show()
```

OUTPUT

```
C:\Users\ajcemca\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/ajcemca/Pycharm
[[ 5]
 [15]
 [25]
 [35]
 [45]
 [55]]
[ 5 20 14 32 22 38]
coefficent of determination:  0.7158756137479542
intercept:  5.633333333333329
slope :  [0.54]
Predicted response:  [ 8.33333333 13.73333333 19.13333333 24.53333333 29.93333333 35.33333333]
```



PROGRAM NO:7

Date:15/01/2022

AIM:

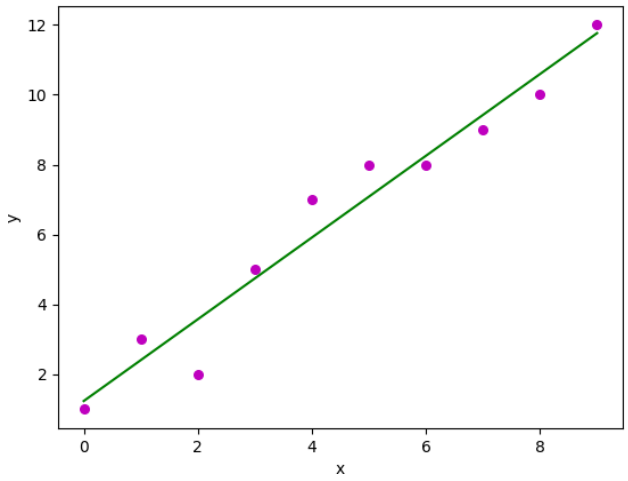
Program to implement linear regression techniques using any standard dataset available in the public domain and evaluate its performance.

PROGRAM (Without inbuilt):

```
import numpy as np
import matplotlib.pyplot as plt
def estimate_coef(x,y):
n=np.size(x) m_x=np.mean(x) m_y=np.mean(y)
SS_xy=np.sum(y*x) - n *m_y* m_x SS_xx=np.sum(x*x) - n
*m_x* m_x b_1=SS_xy / SS_xx b_0=m_y - b_1*
m_x return (b_0,b_1)

def plot_regr_line(x,y,b): plt.scatter(x,y,color="m",marker="o",s=30)
y_pred=b[0]+b[1]*x plt.plot(x,y_pred,color="g")
plt.xlabel('x')
plt.ylabel(' y')


plt.show()


def main():
x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])

b = estimate_coef(x, y)
print("Estimated coefficients:\nb_0 = {} \
\nb_1 = {}".format(b[0], b[1])) plot_regr_line(x, y, b)

if name ==" main ":
main()
```

OUTPUT

```
C:\Users\ajcemca\PycharmProjects\py
Estimated coefficients:
b_0 = 1.2363636363636363
b_1 = 1.1696969696969697
```



PROGRAM NO:8

Date:15-12-2021

AIM:

Program to implement multiple regression techniques using any standard dataset available in the public domain and evaluate its performance.

PROGRAM

```
import pandas df=pandas.read_csv("cars.csv")

x=df[['Weight','Volume']] y=df['CO2']

from sklearn import linear_model

regr=linear_model.LinearRegression()

regr.fit(x,y) predictedco2=regr.predict([[2300,1300]])

print(predictedco2)
```

OUTPUT

```
[107.2087328]
[0.00755095 0.00780526]
```

PROGRAM NO:9

Date:15-12-2021

AIM:

Program to implement multiple regression techniques using any standard dataset available in the public domain and evaluate its performance.

PROGRAM

```
import matplotlib.pyplot as plt

from sklearn import datasets,linear_model,metrics


boston=datasets.load_boston()


x=boston.data

y=boston.target


from sklearn.model_selection import

train_test_split

x_train,x_test,y_train,y_test=train_test_split(

x,y,test_size=0.4,random_state=1)


reg=linear_model.LinearRegression()

reg.fit(x_train,y_train)


pre=reg.predict(x_test

) print("Prediction :

",pre)

print('Coefficients:
```

',reg.coef_)

print('Variance Score:{}'.format(reg.score(x_test,y_test)))

OUTPUT

```
Prediction :  [32.65503184 28.0934953  18.02901829 21.47671576 18.8254387  19.87997758
 32.42014863 18.06597765 24.42277848 27.00977832 27.04081017 28.75196794
 21.15677699 26.85200196 23.38835945 20.66241266 17.33082198 38.24813601
 30.50550873  8.74436733 20.80203902 16.26328126 25.21805656 24.85175752
 31.384365   10.71311063 13.80434635 16.65930389 36.52625779 14.66750528
 21.12114902 13.95558618 43.16210242 17.97539649 21.80116017 20.58294808
 17.59938821 27.2212319   9.46139365 19.82963781 24.30751863 21.18528812
 29.57235682 16.3431752  19.31483171 14.56343172 39.20885479 18.10887551
 25.91223267 20.33018802 25.16282007 24.42921237 25.07123258 26.6603279
  4.56151258 24.0818735  10.88682673 26.88926656 16.85598381 35.88704363
 19.55733853 27.51928921 16.58436103 18.77551029 11.13872875 32.36392607
 36.72833773 21.95924582 24.57949647 25.14868695 23.42841301  6.90732017
 16.56298149 20.41940517 20.80403418 21.54219598 33.85383463 27.94645899
 25.17281456 34.65883942 18.62487738 23.97375565 34.6419296  13.34754896
 20.71097982 30.0803549  17.13421671 24.30528434 19.25576671 16.98006722
 27.00622638 41.85509074 14.11131512 23.25736073 14.66302672 21.86977175
 23.02527624 29.0899182  37.11937872 20.53271022 17.36840034 17.71399314]
Coefficients:  [-1.12386867e-01  5.80587074e-02  1.83593559e-02  2.12997760e+00
 -1.95811012e+01  3.09546166e+00  4.45265228e-03 -1.50047624e+00
  3.05358969e-01 -1.11230879e-02 -9.89007562e-01  7.32130017e-03
 -5.44644997e-01]
Variance Score:0.763417443213847
```

PROGRAM NO:10

Date:22-12-2021

AIM:

Program to implement decision trees using any standard dataset available in the public domain and find the accuracy of the algorithm

PROGRAM

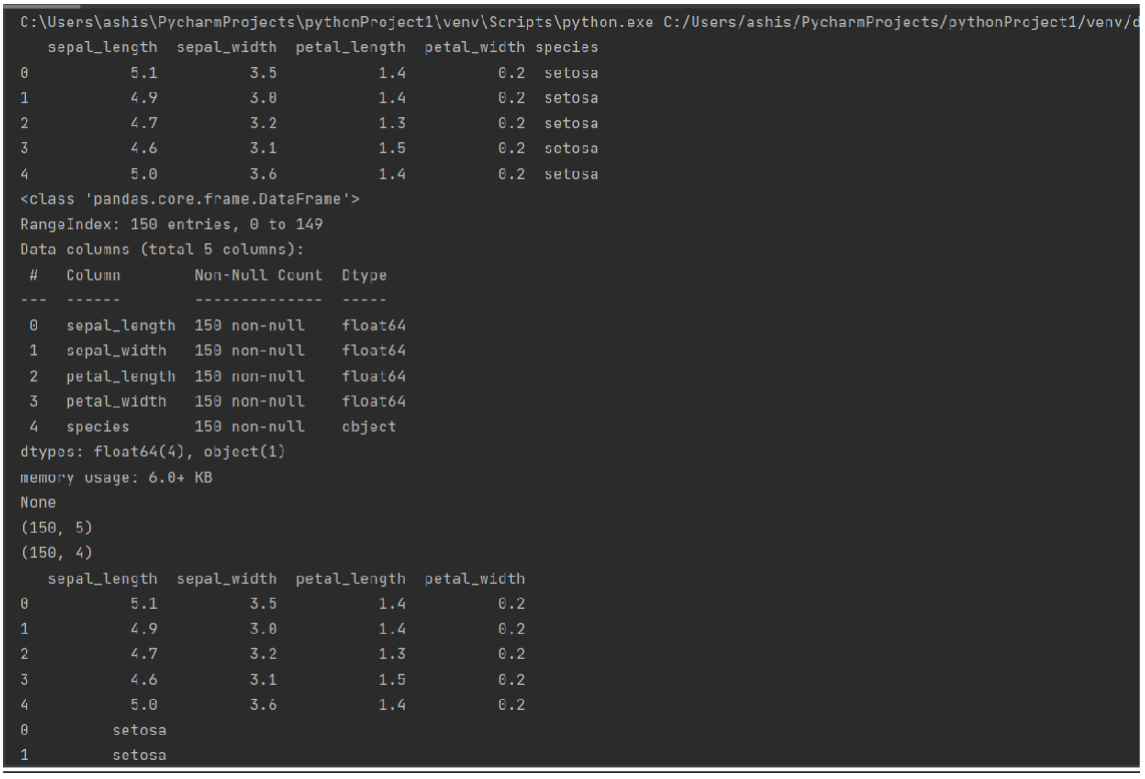
```
import pandas as

pd import numpy

as np import

seaborn as sns

import matplotlib.pyplot as plt


from sklearn.preprocessing import

LabelEncoder from sklearn.model_selection

import train_test_split


from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import classification_report,confusion_matrix


from sklearn.tree

import plot_tree

df=sns.load_dataset('iri

s') print(df.head())

print(df.info())

df.isnull().any()

print(df.shape)
```

```
sns.pairplot(data=df,hue='spe

cies') plt.savefig("pne.png")


sns.heatmap(df.corr())

plt.savefig("one.png")

target=df['species']

df1=df.copy()

df1=df1.drop('species',a

xis

=1)

print(df1.shape)

print(df1.head())


x=df1

print(target

)


le=LabelEncoder()

target=le.fit_transform(targ

et) print(target)

y=target


x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)


print("Training  split
```

```
input",x_train.shape)  print("Testing

split input",x_test.shape)


dtree=DecisionTreeClassifier()

dtree.fit(x_train,y_train)



print("Decision  tree classifier created")


y_pred=dtree.predict(x_test)

print("classsification  report

\n",classification_report(y_test,y_pred))

cm=confusion_matrix(y_test,y_pred)

plt.figure(figsize=(5,5))


sns.heatmap(data=cm,linewidth=5,annot=True,square=True,cmap='Blues')


plt.ylabel('Actual

label')

plt.xlabel('Predict

d label')


all_sample_title='Accuracy  Score:{0}'.format(dtree.score(x_test,y_test))


plt.savefig("two.png")
```

```
plt.figure(figsize=(20,20))
dec_tree=plot_tree(decision_tree=dtree,feature_names=df1.co
lumns,
    class_names=["setosa","vercicikor","verginica"],filled=True,precision=4,rounded=True)
plt.savefig("three.png")
```

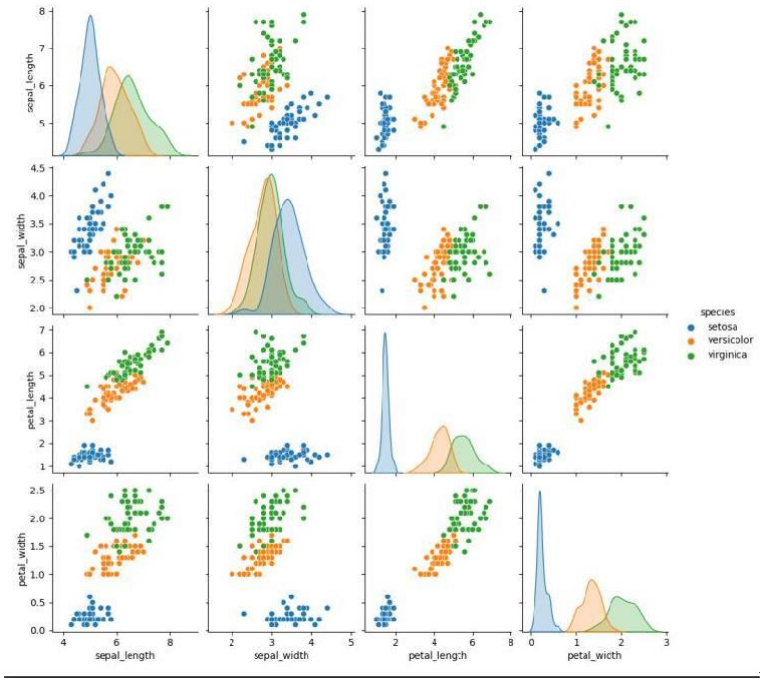
OUTPUT



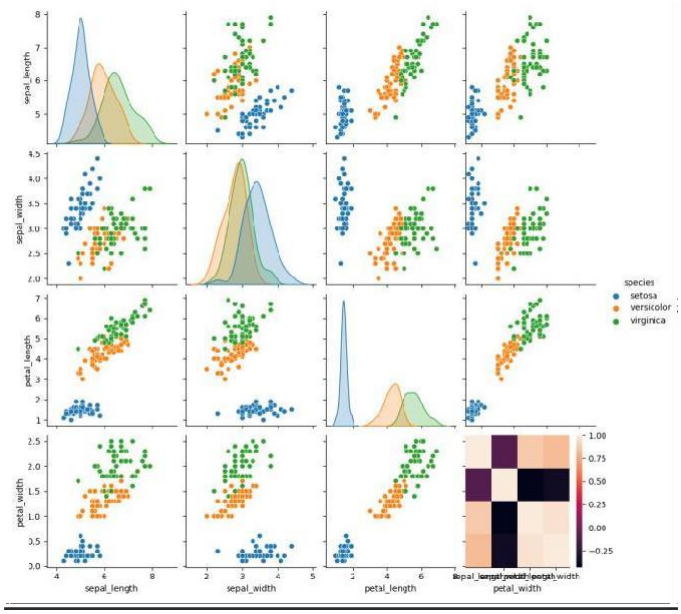
```
C:\Users\ashis\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/ashis/PycharmProjects/pythonProject1/venv/d
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
None
(150, 5)
(150, 4)
   sepal_length  sepal_width  petal_length  petal_width
0           5.1          3.5           1.4          0.2
1           4.9          3.0           1.4          0.2
2           4.7          3.2           1.3          0.2
3           4.6          3.1           1.5          0.2
4           5.0          3.6           1.4          0.2
0        setosa
1        setosa
```

```
2          setosa
3          setosa
4          setosa
          ...
145    virginica
146    virginica
147    virginica
148    virginica
149    virginica
Name: species, Length: 150, dtype: object
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
Training split input (120, 4)
Testing split input (30, 4)
Decision tree classifier created
classsification report
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

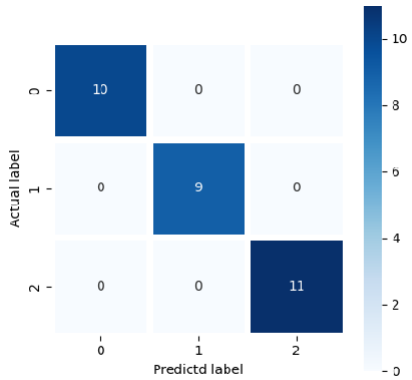
Pne.png



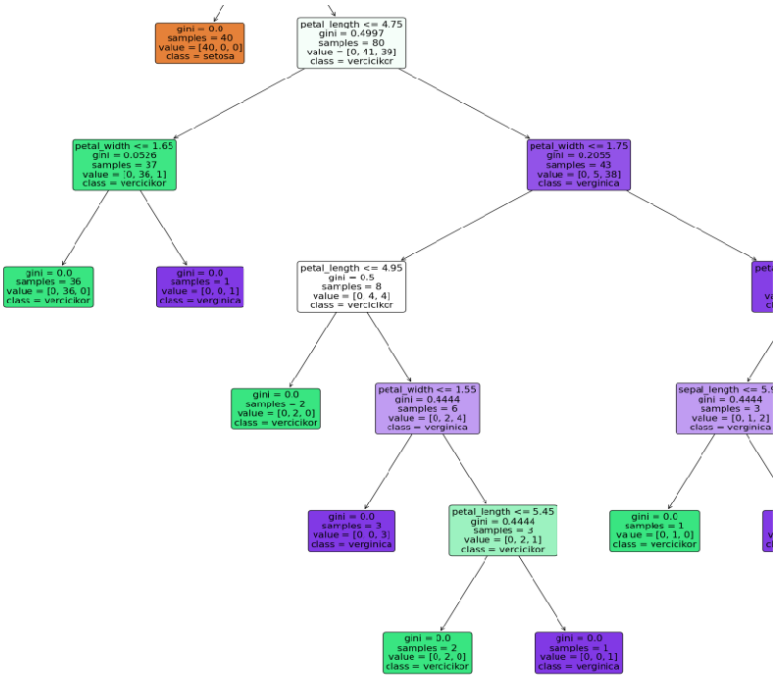
One.png



Two.png


10
0
0
0
9
0
0
0
11
Actual label
Predictd label
0
1
2
10
8
6
4
2
0

Three.png


gini = 0.0
samples = 40
value = [40, 0, 0]
class = setosa
petal_length <= 4.75
gini = 0.4997
samples = 80
value = [0, 11, 39]
class = vercicikor
petal_width <= 1.65
gini = 0.0526
samples = 37
value = [0, 36, 1]
class = vercicikor
petal_width <= 1.75
gini = 0.2055
samples = 43
value = [0, 5, 38]
class = verginica
gini = 0.0
samples = 36
value = [0, 36, 0]
class = vercicikor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = verginica
petal_length <= 4.95
gini = 0.5
samples = 8
value = [0, 4, 4]
class = vercicikor
gini = 0.0
samples = 2
value = [0, 2, 0]
class = vercicikor
petal_width <= 1.55
gini = 0.4444
samples = 6
value = [0, 2, 4]
class = verginica
gini = 0.0
samples = 3
value = [0, 0, 3]
class = verginica
petal_length <= 5.45
gini = 0.4444
samples = 3
value = [0, 2, 1]
class = vercicikor
gini = 0.0
samples = 2
value = [0, 2, 0]
class = vercicikor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = verginica

PROGRAM NO:11

Date:05-01-2022

AIM:

Program to implement k-means clustering technique using any standard dataset available in the public domain

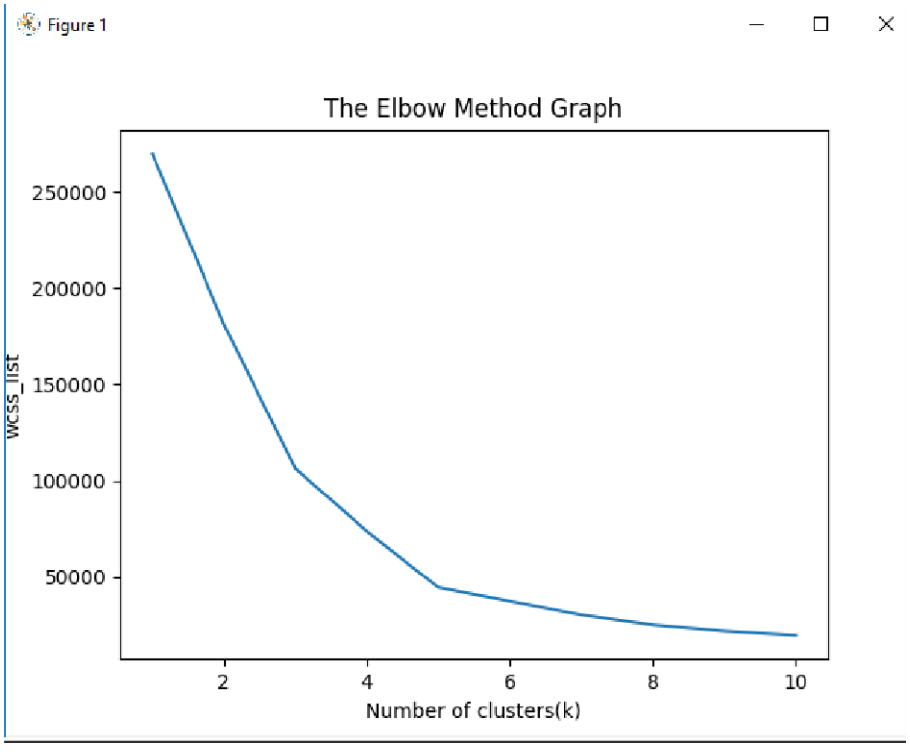
PROGRAM

```
import numpy as nm

import

matplotlib.pyplot as

mtp import pandas as

pd dataset =

pd.read_csv('Mall_Customers.cs

v')

x=dataset.iloc[:,[3,4]].values

print(x)

from sklearn.cluster import

KMeans wcss_list=[]

for i in range(1,11):

  kmeans=KMeans(n_clusters=i,init='k-

  means++',random_state=42) kmeans.fit(x)

  wcss_list.append(kmeans.in

erti a_)

mtp.plot(range(1,11),wcss_li

st) mtp.title('The Elbow

Method Graph')
```

```
mtp.xlabel('Number of
clusters(k)')
mtp.ylabel('wcss_list')
mtp.show()

kmeans=KMeans(n_clusters=5,init='k-
means++',random_state=42)
y_predict=kmeans.fit_predict(x)

print(y_predict)

mtp.scatter(x[y_predict ==0,0],x[y_predict ==0,1],s=100,c='blue',label='cluster 1')
mtp.scatter(x[y_predict ==1,0],x[y_predict ==1,1],s=100,c='green',label='cluster 2')
mtp.scatter(x[y_predict ==2,0],x[y_predict ==2,1],s=100,c='red',label='cluster 3')
mtp.scatter(x[y_predict ==3,0],x[y_predict ==3,1],s=100,c='cyan',label='cluster 4')
mtp.scatter(x[y_predict ==4,0],x[y_predict ==4,1],s=100,c='magenta',label='cluster 5')
mtp.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],s=300,c='black',label
='cluster'
) mtp.title('Clusters of customers')
mtp.xlabel('Annual Income (K$)')
mtp.ylabel('Spending Score(1-100)')
mtp.legend()
mtp.show()
```

OUTPUT

```
C:\Users\ajcemca\PycharmProje
[[ 15  39]
 [ 15  81]
 [ 16   6]
 [ 16  77]
 [ 17  40]
 [ 17  76]
 [ 18   6]
 [ 18  94]
 [ 19   3]
 [ 19  72]
 [ 19  14]
 [ 19  99]
 [ 20  15]
 [ 20  77]
 [ 20  13]
```



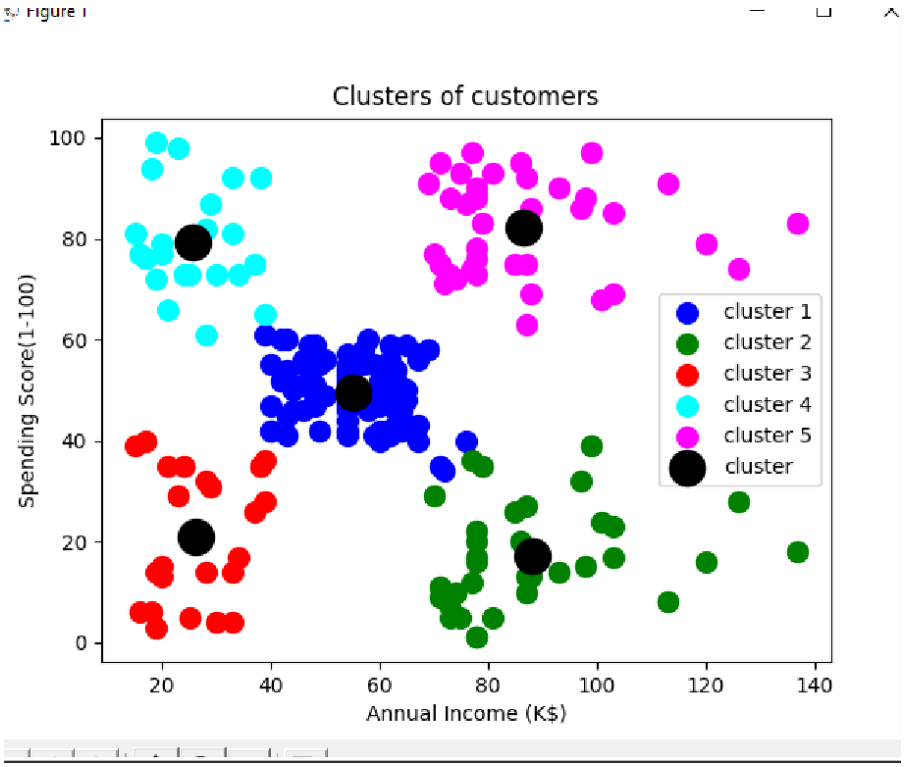


Figure 1
Clusters of customers
Spending Score(1-100)
Annual Income (K$)
0
20
40
60
80
100
20
40
60
80
100
120
140
cluster 1
cluster 2
cluster 3
cluster 4
cluster 5
cluster

PROGRAM NO:12

Date:05-01-2022

AIM:

Program to implement k-means clustering technique using any standard dataset available in the public domain (Using world_country_and_usa_states_latitude_and_longitude_values.csv)

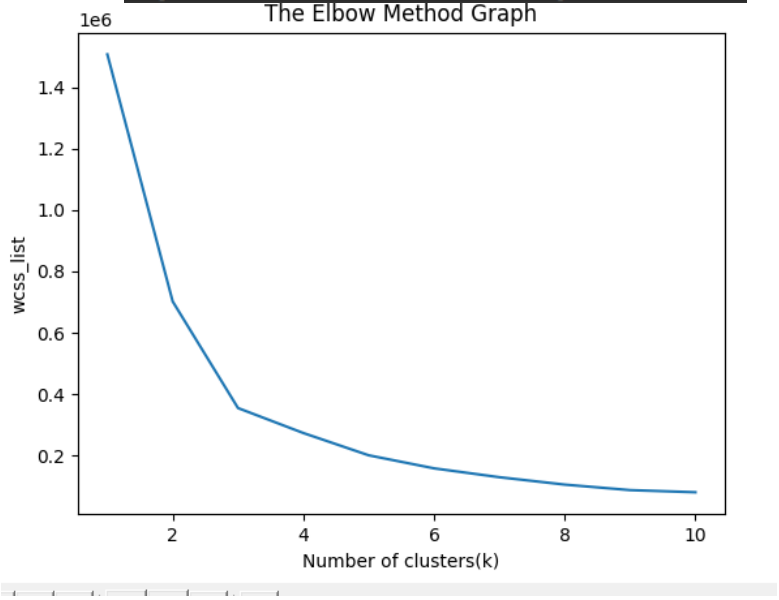
PROGRAM

```
import numpy as nm
import
matplotlib.pyplot as
mtp import pandas as
pd

dataset =
pd.read_csv('world_country_and_usa_states_latitude_and_longitude_values.cs
v') x=dataset.iloc[:,[1,2]].values
print(x)

from sklearn.cluster import
KMeans wcss_list=[]
for i in range(1,11):
  kmeans=KMeans(n_clusters=i,init='k-
  means++',random_state=42) kmeans.fit(x)
  wcss_list.append(kmeans.in
erti a_)
mtp.plot(range(1,11),wcss_li
st) mtp.title('The Elbow
Method Graph')
mtp.xlabel('Number of
clusters(k)')
mtp.ylabel('wcss_list')
mtp.show()

kmeans=KMeans(n_clusters=3,init='k-
means++',random_state=42)
y_predict=kmeans.fit_predict(x)
print(y_predict)

mtp.scatter(x[y_predict ==0,0],x[y_predict ==0,1],s=100,c='blue',label='cluster 1')
mtp.scatter(x[y_predict ==1,0],x[y_predict ==1,1],s=100,c='green',label='cluster 2')
mtp.scatter(x[y_predict ==2,0],x[y_predict ==2,1],s=100,c='red',label='cluster 3')
mtp.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],s=300,c='black',label
='cluster'
) mtp.title('Clusters of customers')
```

```
mtp.xlabel('Annual  Income  (K$)')
mtp.ylabel('Spending Score(1-100)')
mtp.legend()

mtp.show()
```

OUTPUT

```
C:\Users\ajcemca\PycharmProjects\Rmca_DLMLLab_28
[[ 4.25462450e+01  1.60155400e+00]
 [ 2.34240760e+01  5.38478180e+01]
 [ 3.39391100e+01  6.77099530e+01]
 [ 1.70608160e+01 -6.17964280e+01]
 [ 1.82205540e+01 -6.30686150e+01]
 [ 4.11533320e+01  2.01683310e+01]
 [ 4.00690990e+01  4.50381890e+01]
 [ 1.22260790e+01 -6.90600870e+01]
 [-1.12026920e+01  1.78738870e+01]
 [-7.52509730e+01 -7.13890000e-02]
 [-3.84160970e+01 -6.36166720e+01]
 [-1.42709720e+01 -1.70132217e+02]
 [ 4.75162310e+01  1.45500720e+01]
 [-2.52743980e+01  1.33775136e+02]
 [ 1.25211100e+01 -6.99683380e+01]
 [ 4.01431050e+01  4.75769270e+01]
 [ 4.39158860e+01  1.76790760e+01]
 [ 1.31938870e+01 -5.95431980e+01]
 [ 2.36849940e+01  9.03563310e+01]
```



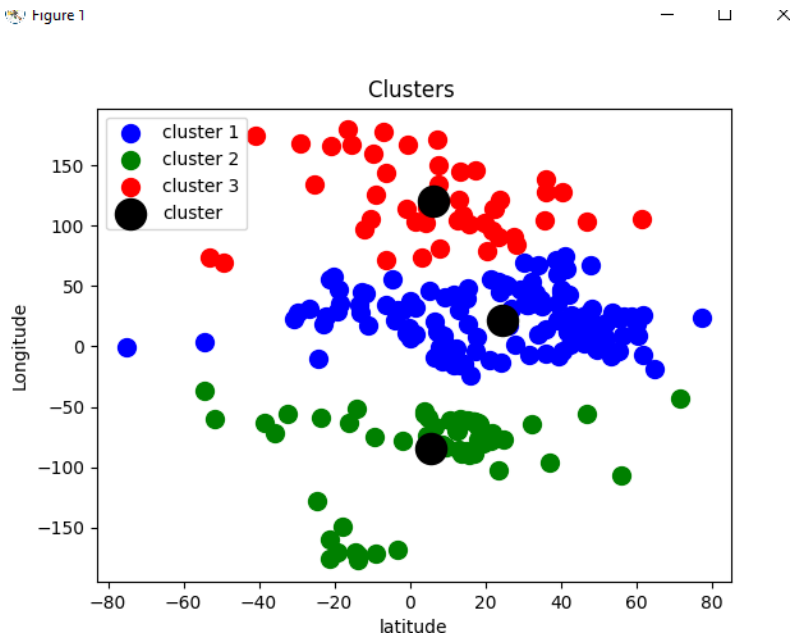


Figure 1
Clusters
cluster 1
cluster 2
cluster 3
cluster
Longitude
latitude
150
100
50
0
−50
−100
−150
−80
−60
−40
−20
0
20
40
60
80

PROGRAM NO:13

Date:2/02/2022

AIM:

Programs on convolutional neural network to classify images from any standard dataset in the public domain

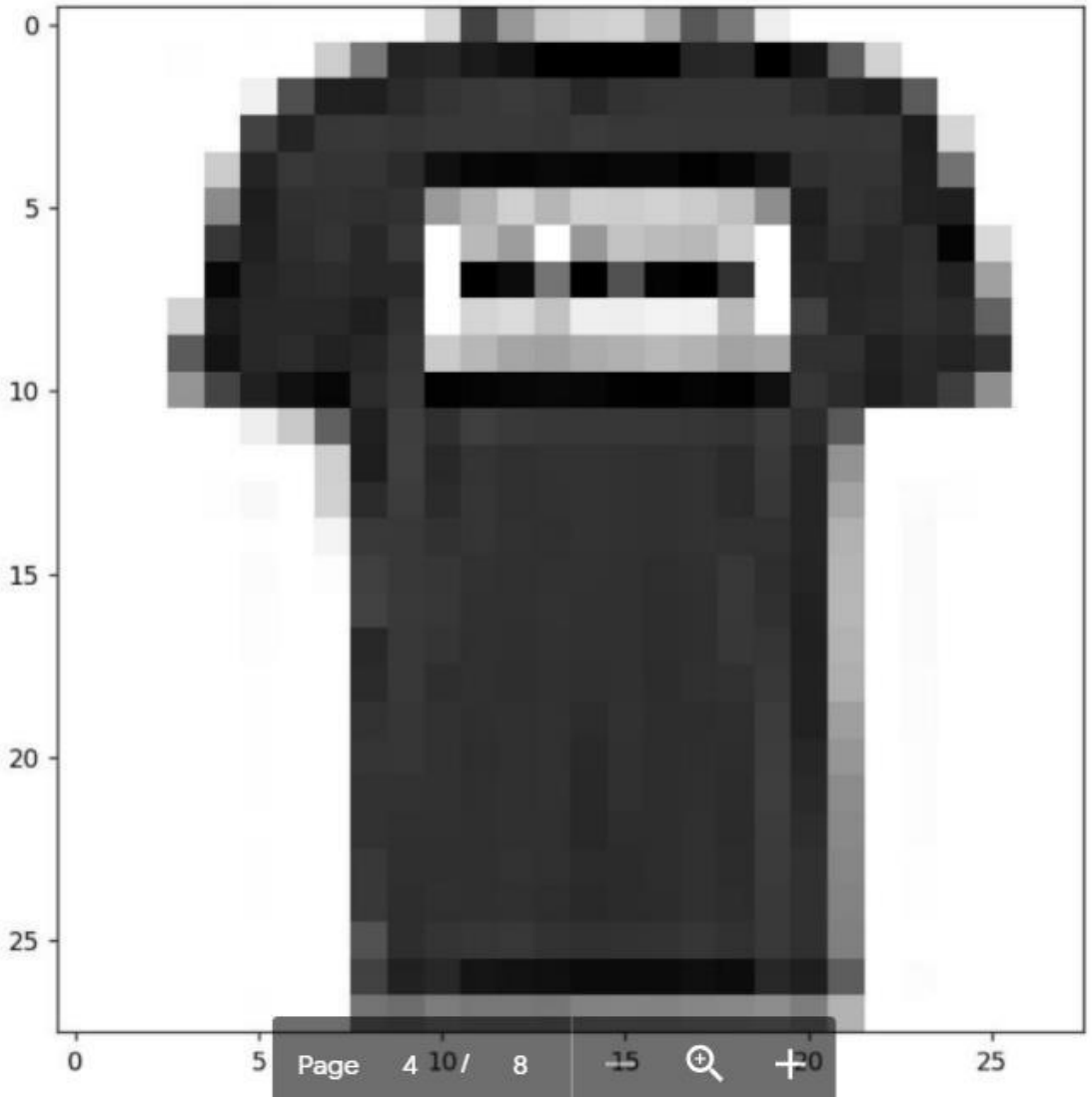
PROGRAM

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras

np.random.seed(42)
# tf.set.random. seed(42)
fashion_mnist = keras.datasets.fashion_mnist
(X_train, y_train), (X_test, y_test) = fashion_mnist.load_data()
print(X_train.shape, X_test.shape)
X_train = X_train / 255.0
X_test = X_test / 255.0
plt.imshow(X_train[1], cmap='binary')
plt.show()
np.unique(y_test)

class_names = ['T-Shirt/Top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', '8ag',
'Ankle Boot']
n_rows = 5
n_cols = 10
plt.figure(figsize=(n_cols * 1.4, n_rows * 1.6))
for row in range(n_rows):
    for col in range(n_cols):
        index = n_cols * row + col
        plt.subplot(n_rows, n_cols, index + 1)
        plt.imshow(X_train[index], cmap='binary', interpolation='nearest')
        plt.axis('off')
        plt.title(class_names[y_train[index]])
plt.show()

model_CNN = keras.models.Sequential()
model_CNN.add(keras.layers.Conv2D(filters=32, kernel_size=7, padding='same', activation='relu',
input_shape=[28, 28, 1]))
model_CNN.add(keras.layers.MaxPooling2D(pool_size=2))
model_CNN.add(keras.layers.Conv2D(filters=64, kernel_size=3, padding='same',
activation='relu'))
```

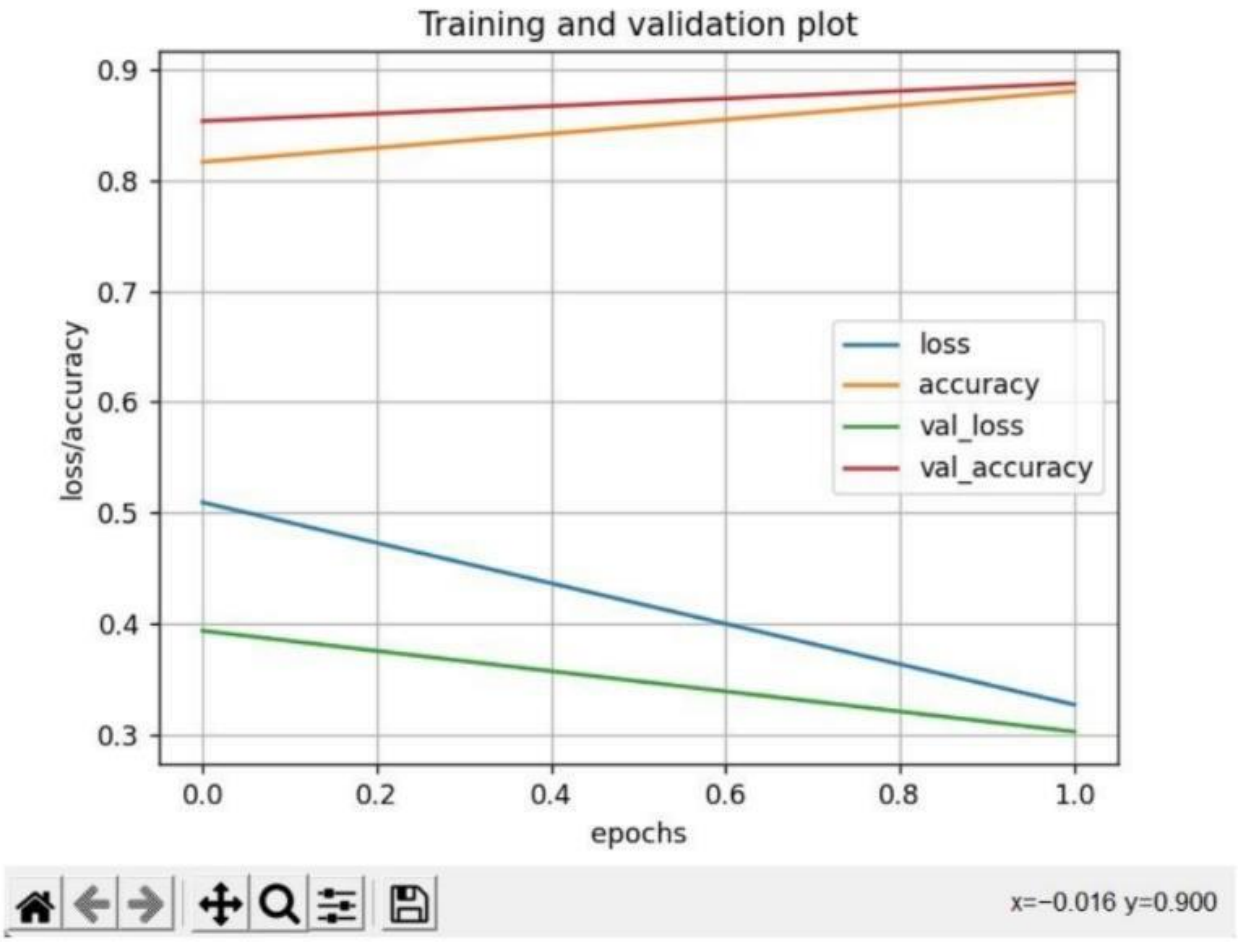
```
model_CNN.add(keras.layers.MaxPooling2D(pool_size=2))
model_CNN.add(keras.layers.Conv2D(filters=32, kernel_size=3, padding='same', activation='relu'))
model_CNN.add(keras.layers.MaxPooling2D(pool_size=2))

model_CNN.summary()

model_CNN.add(keras.layers.Flatten())
model_CNN.add(keras.layers.Dense(units=128, activation='relu'))
model_CNN.add(keras.layers.Dense(units=64, activation='relu'))
model_CNN.add(keras.layers.Dense(units=10, activation='softmax'))

model_CNN.summary()

model_CNN.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

X_train = X_train[..., np.newaxis]
X_test = X_test[..., np.newaxis]

history_CNN = model_CNN.fit(X_train, y_train, epochs=2, validation_split=0.1)
pd.DataFrame(history_CNN.history).plot()
plt.grid(True)
plt.xlabel('epochs')
plt.ylabel('loss/accuracy')
plt.title('Training and validation plot')
plt.show()
test_loss, test_accuracy = model_CNN.evaluate(X_test, y_test)
print(' Test Loss :{}, Test Accuracy : {}'.format(test_loss, test_accuracy))
```

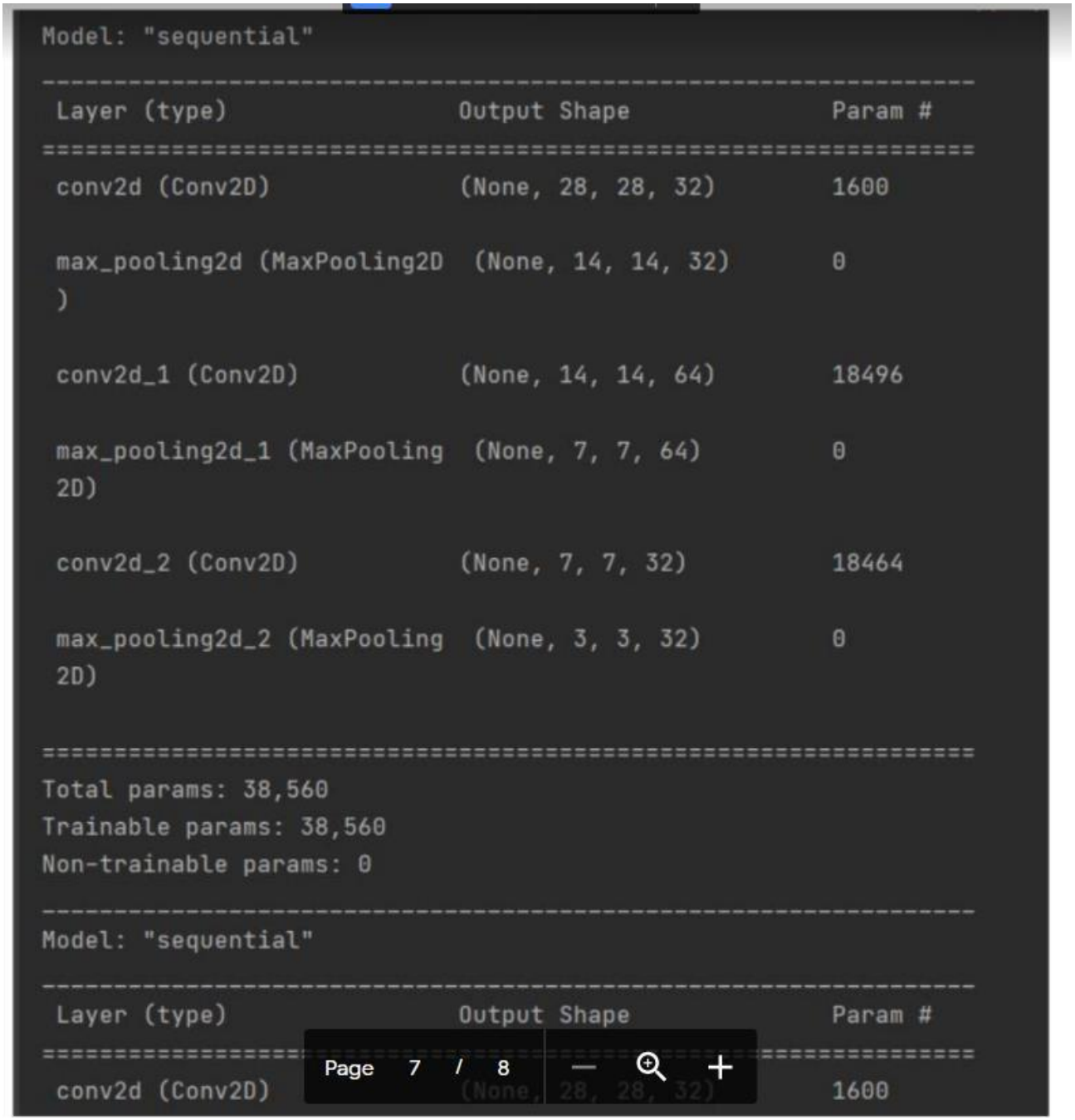
OUTPUT

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
32768/29515 [==============================] - 0s 2us/step
40960/29515 [==========================================] - 0s 2us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26427392/26421880 [==============================] - 12s 0us/step
26435584/26421880 [==============================] - 12s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
16384/5148 [================================================================================================] - 0s 0s/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4423680/4422102 [==============================] - 2s 0us/step
4431872/4422102 [==============================] - 2s 0us/step
(60000, 28, 28) (10000, 28, 28)
```


0
5
10
15
20
25
0
5
Page 4 10 / 8
15
20
25

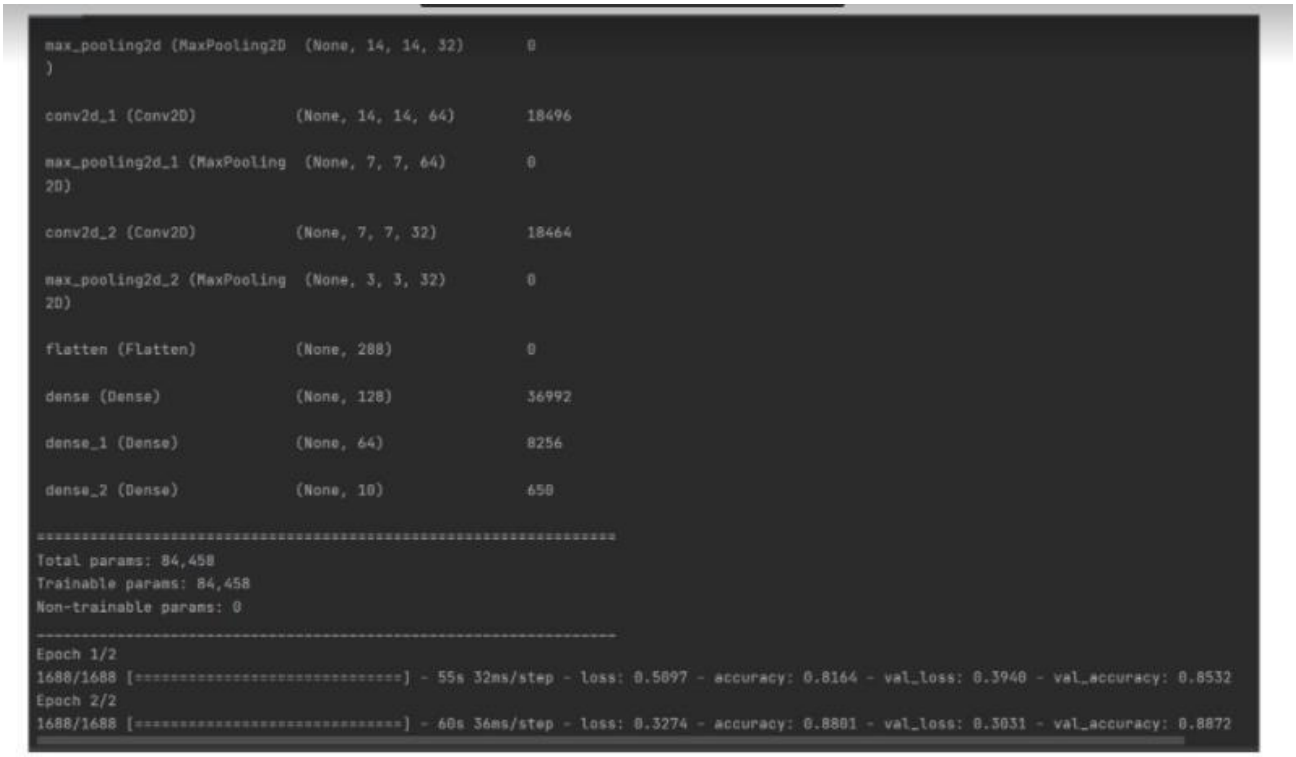

Ankle Boot T-Shirt/Top T-Shirt/Top Dress T-Shirt/Top Pullover Sneaker Pullover Sandal Sandal
T-Shirt/Top Ankle Boot Sandal Sandal Sneaker Ankle Boot Trouser T-Shirt/Top Shirt Coat
Dress Trouser Coat Bag Coat Dress T-Shirt/Top Pullover Coat Coat
Sandal Dress Shirt Shirt T-Shirt/Top Bag Sandal Pullover Trouser Shirt
Shirt Sneaker Ankle Boot Sandal Ankle Boot Pullover Sneaker Dress T-Shirt/Top Dress


Training and validation plot
loss/accuracy
epochs
loss
accuracy
val_loss
val_accuracy
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.0
0.2
0.4
0.6
0.8
1.0
x=−0.016 y=0.900



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 32)        1600

 max_pooling2d (MaxPooling2D  (None, 14, 14, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 14, 14, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 7, 7, 64)         0
 2D)

 conv2d_2 (Conv2D)           (None, 7, 7, 32)          18464

 max_pooling2d_2 (MaxPooling  (None, 3, 3, 32)         0
 2D)

=================================================================
Total params: 38,560
Trainable params: 38,560
Non-trainable params: 0
_________________________________________________________________
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 32)        1600
```

Page 7 / 8



```
max_pooling2d (MaxPooling2D  (None, 14, 14, 32)       0
)

conv2d_1 (Conv2D)            (None, 14, 14, 64)       18496

max_pooling2d_1 (MaxPooling  (None, 7, 7, 64)         0
2D)

conv2d_2 (Conv2D)            (None, 7, 7, 32)         18464

max_pooling2d_2 (MaxPooling  (None, 3, 3, 32)         0
2D)

flatten (Flatten)            (None, 288)              0

dense (Dense)                (None, 128)              36992

dense_1 (Dense)              (None, 64)               8256

dense_2 (Dense)              (None, 10)               650

=================================================================
Total params: 84,458
Trainable params: 84,458
Non-trainable params: 0
_________________________________________________________________
Epoch 1/2
1688/1688 [==============================] - 55s 32ms/step - loss: 0.5897 - accuracy: 0.8164 - val_loss: 0.3940 - val_accuracy: 0.8532
Epoch 2/2
1688/1688 [==============================] - 60s 36ms/step - loss: 0.3274 - accuracy: 0.8801 - val_loss: 0.3031 - val_accuracy: 0.8872
```

PROGRAM NO:14

Date: 16/02/2022

AIM:

Program to implement a simple web crawler using python

PROGRAM

```
import requests
from bs4 import
BeautifulSoup import csv
import lxml
URL = "http://www.values.com/inspirational-
quotes" r = requests.get(URL)
print(r.content)
soup = BeautifulSoup(r.content,'lxml')
print(soup.prettify())
quotes = []
table = soup.find('div',attrs={'id': 'all_quotes'})
for row in table.findAll('div',attrs={'class':'col-6 col-lg-3 text-center margin-30px-bottom sm-
margin -30px- top'}):
  quote = {}
  quote['theme']= row.h5.text
  quote['url']= row.a['href']
  quote['img'] = row.img['src']
  quote['lines'] =
  row.img['alt'].split("#")[0]
  quote['author'] =
  row.img['alt'].split("#")[1]
  quotes.append(quote)
  filename =
  'inspirational_quotes.csv' with
  open(filename,'w',newline='')as f:
    w=
    csv.DictWriter(f,['theme','url','img','lines','author'])
    w.writeheader()
    for quote in
      quotes:
      w.writerow(quote
      )
```

OUTPUT

```
<meta charset="utf-8"/>
<meta content="text/html; charset=utf-8" http-equiv="content-type"/>
<meta content="IE=edge" http-equiv="X-UA-Compatible"/>
<meta content="width=device-width,initial-scale=1.0" name="viewport"/>
<meta content="The Foundation for a Better Life | Pass It On.com" name="description"/>
<link href="/apple-touch-icon.png" rel="apple-touch-icon" sizes="180x180"/>
<link href="/favicon-32x32.png" rel="icon" sizes="32x32" type="image/png"/>
<link href="/favicon-16x16.png" rel="icon" sizes="16x16" type="image/png"/>
<link href="/site.webmanifest" rel="manifest"/>
<link color="#c8102e" href="/safari-pinned-tab.svg" rel="mask-icon"/>
<meta content="#c8102e" name="msapplication-TileColor"/>
<meta content="#ffffff" name="theme-color"/>
<link crossorigin="anonymous" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCb
<link href="/assets/application-2a7a8e6a1c3f620bac9efa66420f5579.css" media="all" rel="stylesheet"/>
<meta content="authenticity_token" name="csrf-param"/>
<meta content="NUSNlipO+WUPfi5GKrz5YpcXkt8KOjhiTNZhFCUexVVr6hlN9ohTFh3W1fwZk9+pE4VlLysU2oF6ipferI/7Wg==" name="csrf-token"/>
<!-- Global site tag (gtag.js) - Google Analytics -->
<script async="" src="https://www.googletagmanager.com/gtag/js?id=UA-1179606-29">
</script>
<script>
 window.dataLayer = window.dataLayer || [];
        function gtag(){dataLayer.push(arguments);}
        gtag('js', new Date());
        gtag('config', 'UA-1179606-29');
</script>
```

```
     </a>
     |
     <a href="/terms-of-use">
      Terms of Use
     </a>
    </div>
   </div>
  </div>
 </div>
</footer>
<a class="scroll-top-arrow" href="javascript:void(0);">
 <i class="ti-arrow-up">
 </i>
</a>
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/1.12.4/jquery.js">
</script>
<script crossorigin="anonymous" integrity="sha384-U02eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" src="https://cdnjs.cloudflare.com/ajax/li
</script>
<script crossorigin="anonymous" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" src="https://stackpath.bootstrapcdn.com/b
</script>
<script src="/assets/pofo-1a7dc0d92519266568dcfcc8a6e53534.js">
</script>
</body>
</html>
```

```
theme,url,img,lines,author
LOVE,/inspirational-quotes/7444-where-there-is-love-there-is-life,https://assets.passiton.com/quotes/quote_artwork/7444
LOVE,/inspirational-quotes/7439-at-the-touch-of-love-everyone-becomes-a-poet,https://assets.passiton.com/quotes/quote_a
FRIENDSHIP,/inspirational-quotes/8304-a-friend-may-be-waiting-behind-a-stranger-s-face,https://assets.passiton.com/quot
FRIENDSHIP,/inspirational-quotes/3331-wherever-we-are-it-is-our-friends-that-make,https://assets.passiton.com/quotes/qu
FRIENDSHIP,/inspirational-quotes/8303-find-a-group-of-people-who-challenge-and,https://assets.passiton.com/quotes/quote
FRIENDSHIP,/inspirational-quotes/8302-there-s-not-a-word-yet-for-old-friends-who-ve,https://assets.passiton.com/quotes/
FRIENDSHIP,/inspirational-quotes/7435-there-are-good-ships-and-wood-ships-ships-that,https://assets.passiton.com/quotes
PERSISTENCE,/inspirational-quotes/6377-at-211-degrees-water-is-hot-at-212-degrees,https://assets.passiton.com/quotes/qu
PERSISTENCE,/inspirational-quotes/8301-the-key-of-persistence-opens-all-doors-closed,https://assets.passiton.com/quotes
PERSISTENCE,/inspirational-quotes/7918-you-keep-putting-one-foot-in-front-of-the,https://assets.passiton.com/quotes/quo
PERSISTENCE,/inspirational-quotes/7919-to-persist-with-a-goal-you-must-treasure-the,https://assets.passiton.com/quotes/
PERSISTENCE,/inspirational-quotes/8300-failure-cannot-cope-with-persistence,https://assets.passiton.com/quotes/quote_ar
INSPIRATION,/inspirational-quotes/8298-though-no-one-can-go-back-and-make-a-brand-new,https://assets.passiton.com/quote
INSPIRATION,/inspirational-quotes/8297-a-highly-developed-values-system-is-like-a,https://assets.passiton.com/quotes/qu
INSPIRATION,/inspirational-quotes/7066-just-don-t-give-up-trying-what-you-really-want,https://assets.passiton.com/quote
INSPIRATION,/inspirational-quotes/8296-when-we-strive-to-become-better-than-we-are,https://assets.passiton.com/quotes/q
INSPIRATION,/inspirational-quotes/8299-the-most-important-thing-is-to-try-and-inspire,https://assets.passiton.com/quote
OVERCOMING,/inspirational-quotes/6828-bad-things-do-happen-how-i-respond-to-them,https://assets.passiton.com/quotes/quo
OVERCOMING,/inspirational-quotes/8294-show-me-someone-who-has-done-something,https://assets.passiton.com/quotes/quote_a
```

PROGRAMNO:15

Date: 16/02/2022

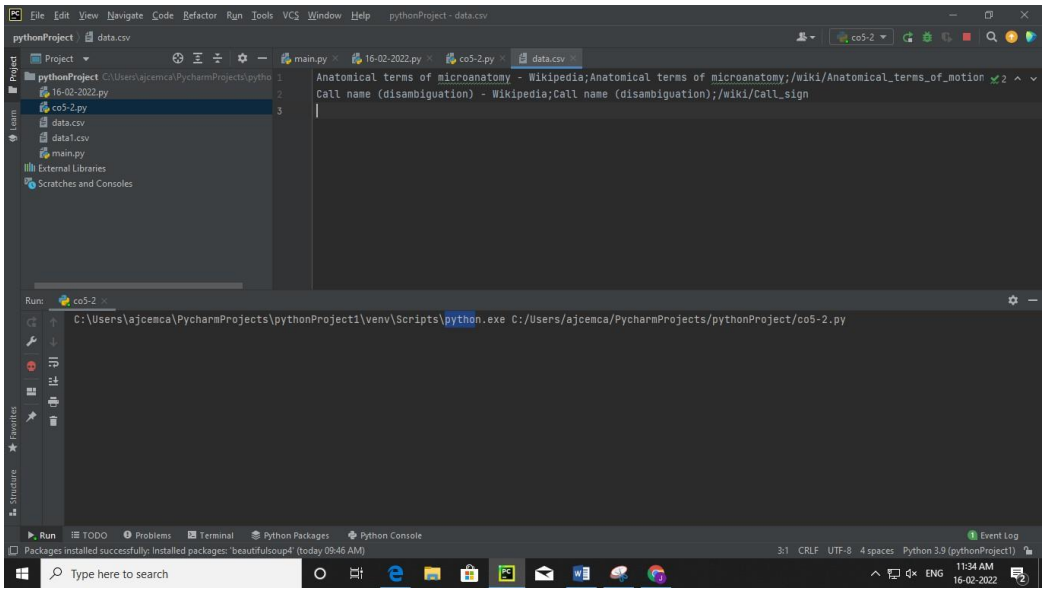
AIM:

Program to implement a simple web crawler using python

PROGRAM

```
import requests
from bs4 import BeautifulSoup
pages_crawled = []
def crawler(url):
   page = requests.get(url)
   soup= BeautifulSoup(page.text,
   'html.parser') links = soup.find_all('a')
   for link in links:
      if 'href' in link.attrs:
         if link['href'].startswith('/wiki') and ':' not in
            link['href']: if link['href'] not in
            pages_crawled:
               new_link =
               f"https://en.wikipedia.org{link['href']}"
               pages_crawled.append(link['href'])
               try:
                  with open('data.csv','a') as file:
                     file.write(f'{soup.title.text};{soup.h1.text};{link["href"]}\n')
                     crawler(new_link)
               except:
continue

crawler('https://en.wikipedia.

org')
```

OUTPUT


File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonProject - data.csv
pythonProject data.csv
Project
pythonProject C:\Users\ajcemca\PycharmProjects\pytho
16-02-2022.py
co5-2.py
data.csv
data1.csv
main.py
External Libraries
Scratches and Consoles
main.py
16-02-2022.py
co5-2.py
data.csv
Anatomical terms of microanatomy - Wikipedia;Anatomical terms of microanatomy;/wiki/Anatomical_terms_of_motion
Call name (disambiguation) - Wikipedia;Call name (disambiguation);/wiki/Call_sign
Run: co5-2
C:\Users\ajcemca\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/ajcemca/PycharmProjects/pythonProject/co5-2.py
Run TODO Problems Terminal Python Packages Python Console
Event Log
Packages installed successfully: Installed packages: 'beautifulsoup4' (today 09:46 AM)
3:1 CRLF UTF-8 4 spaces Python 3.9 (pythonProject1)
Type here to search
ENG
11:34 AM
16-02-2022

PROGRAM NO:16

Date: 16-02-2022

AIM:

Program to implement scrap of any website

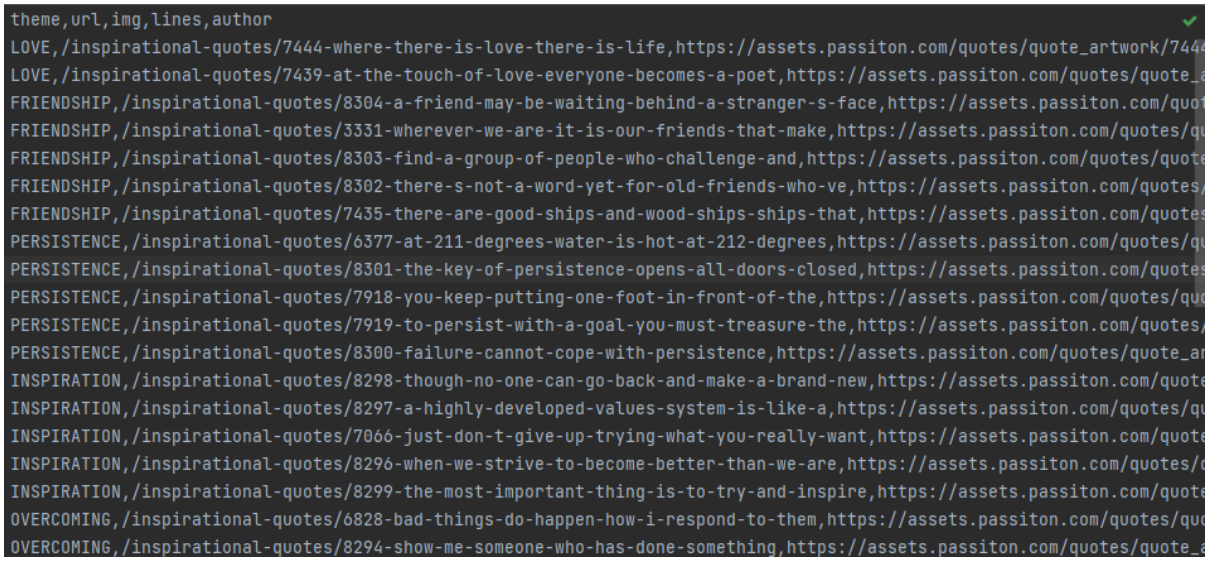
PROGRAM

```
import requests
from bs4 import
BeautifulSoup import csv
import lxml
URL = "http://www.values.com/inspirational-
quotes" r = requests.get(URL)
print(r.content)
soup = BeautifulSoup(r.content,'lxml')
print(soup.prettify())
quotes = []
table = soup.find('div',attrs={'id': 'all_quotes'})
for row in table.findAll('div',attrs={'class':'col-6 col-lg-3 text-center margin-30px-bottom sm-
margin -30px- top'}):
  quote = {}
  quote['theme']= row.h5.text
  quote['url']= row.a['href']
  quote['img'] = row.img['src']
  quote['lines'] =
  row.img['alt'].split("#")[0]
  quote['author'] =
  row.img['alt'].split("#")[1]
  quotes.append(quote)
  filename = 'inspirational_quotes.csv'
  with open(filename,'w',newline='')as f:
    w=
    csv.DictWriter(f,['theme','url','img','lines','autho
    r']) w.writeheader()
    for quote in
      quotes:
      w.writerow(quote
      )
```

OUTPUT

```
<meta charset="utf-8"/>
<meta content="text/html; charset=utf-8" http-equiv="content-type"/>
<meta content="IE=edge" http-equiv="X-UA-Compatible"/>
<meta content="width=device-width,initial-scale=1.0" name="viewport"/>
<meta content="The Foundation for a Better Life | Pass It On.com" name="description"/>
<link href="/apple-touch-icon.png" rel="apple-touch-icon" sizes="180x180"/>
<link href="/favicon-32x32.png" rel="icon" sizes="32x32" type="image/png"/>
<link href="/favicon-16x16.png" rel="icon" sizes="16x16" type="image/png"/>
<link href="/site.webmanifest" rel="manifest"/>
<link color="#c8102e" href="/safari-pinned-tab.svg" rel="mask-icon"/>
<meta content="#c8102e" name="msapplication-TileColor"/>
<meta content="#ffffff" name="theme-color"/>
<link crossorigin="anonymous" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCb
<link href="/assets/application-2a7a8e6a1c3f620bac9efa66420f5579.css" media="all" rel="stylesheet"/>
<meta content="authenticity_token" name="csrf-param"/>
<meta content="NUSNlipO+WUPfi5GKrz5YpcXkt8KOjhiTNZhFCUexVVr6hlN9ohTFh3W1fwZk9+pE4VlLysU2oF6ipferI/7Wg==" name="csrf-token"/>
<!-- Global site tag (gtag.js) - Google Analytics -->
<script async="" src="https://www.googletagmanager.com/gtag/js?id=UA-117960629">
</script>
<script>
 window.dataLayer = window.dataLayer || [];
        function gtag(){dataLayer.push(arguments);}
        gtag('js', new Date());
        gtag('config', 'UA-117960629');
</script>
<script>
```

```
      </a>
      |
      <a href="/terms-of-use">
       Terms of Use
      </a>
     </div>
    </div>
   </div>
  </div>
 </footer>
 <a class="scroll-top-arrow" href="javascript:void(0);">
  <i class="ti-arrow-up">
  </i>
 </a>
 <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/1.12.4/jquery.js">
 </script>
 <script crossorigin="anonymous" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" src="https://cdnjs.cloudflare.com/ajax/li
 </script>
 <script crossorigin="anonymous" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" src="https://stackpath.bootstrapcdn.com/b
 </script>
 <script src="/assets/pofo-1a7dc0d92519266568dcfcc8a6e53534.js">
 </script>
</body>
</html>
```



```
theme,url,img,lines,author
LOVE,/inspirational-quotes/7444-where-there-is-love-there-is-life,https://assets.passiton.com/quotes/quote_artwork/744
LOVE,/inspirational-quotes/7439-at-the-touch-of-love-everyone-becomes-a-poet,https://assets.passiton.com/quotes/quote_
FRIENDSHIP,/inspirational-quotes/8304-a-friend-may-be-waiting-behind-a-stranger-s-face,https://assets.passiton.com/quo
FRIENDSHIP,/inspirational-quotes/3331-wherever-we-are-it-is-our-friends-that-make,https://assets.passiton.com/quotes/q
FRIENDSHIP,/inspirational-quotes/8303-find-a-group-of-people-who-challenge-and,https://assets.passiton.com/quotes/quot
FRIENDSHIP,/inspirational-quotes/8302-there-s-not-a-word-yet-for-old-friends-who-ve,https://assets.passiton.com/quotes/
FRIENDSHIP,/inspirational-quotes/7435-there-are-good-ships-and-wood-ships-ships-that,https://assets.passiton.com/quotes
PERSISTENCE,/inspirational-quotes/6377-at-211-degrees-water-is-hot-at-212-degrees,https://assets.passiton.com/quotes/q
PERSISTENCE,/inspirational-quotes/8301-the-key-of-persistence-opens-all-doors-closed,https://assets.passiton.com/quotes
PERSISTENCE,/inspirational-quotes/7918-you-keep-putting-one-foot-in-front-of-the,https://assets.passiton.com/quotes/quo
PERSISTENCE,/inspirational-quotes/7919-to-persist-with-a-goal-you-must-treasure-the,https://assets.passiton.com/quotes/
PERSISTENCE,/inspirational-quotes/8300-failure-cannot-cope-with-persistence,https://assets.passiton.com/quotes/quote_ar
INSPIRATION,/inspirational-quotes/8298-though-no-one-can-go-back-and-make-a-brand-new,https://assets.passiton.com/quot
INSPIRATION,/inspirational-quotes/8297-a-highly-developed-values-system-is-like-a,https://assets.passiton.com/quotes/q
INSPIRATION,/inspirational-quotes/7066-just-don-t-give-up-trying-what-you-really-want,https://assets.passiton.com/quot
INSPIRATION,/inspirational-quotes/8296-when-we-strive-to-become-better-than-we-are,https://assets.passiton.com/quotes/
INSPIRATION,/inspirational-quotes/8299-the-most-important-thing-is-to-try-and-inspire,https://assets.passiton.com/quot
OVERCOMING,/inspirational-quotes/6828-bad-things-do-happen-how-i-respond-to-them,https://assets.passiton.com/quotes/quo
OVERCOMING,/inspirational-quotes/8294-show-me-someone-who-has-done-something,https://assets.passiton.com/quotes/quote_
```

PROGRAM NO:17

Date:16-02-2022

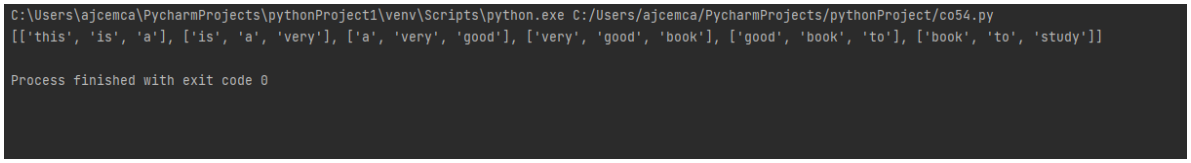
AIM:

Program for Natural Language Processing which performs n-grams

PROGRAM

```
def generate_ngrams(text,
   WordsToCombine):  words =
   text.split()
   output = []
   for i in range(len(words)  - WordsToCombine  + 1):
      output.append(words[i:i  + WordsToCombine])
   return output
x=generate_ngrams(text='this  is a very good book to study',
WordsToCombine=3)  print(x)
```

OUTPUT



PROGRMNO:18

Date:16-02-2022

AIM:

Program for Natural Language Processing which performs n-grams (Using in built functions)

PROGRAM

```
import nltk


from nltk.util import ngrams
samplText = 'this is a very good book to study'
NGRAMS =
ngrams(sequence=nltk.word_tokenize(samplText), n=2) for
grams in NGRAMS:
  print(grams)
```

OUTPUT

```
C:\Users\ajcemca\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/ajcemca/PycharmProjects/pythonProject/co5-5.py
('this', 'is')
('is', 'a')
('a', 'very')
('very', 'good')
('good', 'book')
('book', 'to')
('to', 'study')

Process finished with exit code 0
```

PROGRAM NO:19

Date: 16-02-2022

AIM:

Program for Natural Language Processing which performs speech tagging

PROGRAM

```
import nltk

from nltk.corpus import stopwords
from nltk.tokenize import
wordpunct_tokenize,sent_tokenize stop_words =
set(stopwords.words('english'))
txt = "Sukanya,Rajib and Naba are my good
   friends."\ "Sukanya is getting marreird next
   year."\
   "Marriage is a big step in one's life."\
   "it is both exciting and frightening."\
   "But frendship is a scared bond between
   people."\ "it is a special kind of love
   between us." \
   "many of you must have tried searching for a
friend ."\ "but never found the right one."
tokenized =
sent_tokenize(txt) for i in
tokenized:
   wordList = nltk.word_tokenize(i)
   wordList = [w for w in wordList if not w in
   stop_words] tagged = nltk.pos_tag(wordList)
   print(tagged)
```

OUTPUT

```
[('Sukanya', 'NNP'), (',', ','), ('Rajib', 'NNP'), ('Naba', 'NNP'), ('good', 'JJ'), ('friends.Sukanya', 'NN'), ('getting', 'VBG'), ('marreird', 'JJ'),
('next', 'JJ'), ('year.Marriage', 'NN'), ('big', 'JJ'), ('step', 'NN'), ('one', 'CD'), ("'s", 'POS'), ('life.it', 'NN'),

('exciting', 'VBG'), ('frightening.But', 'JJ'), ('frendship', 'NN'), ('scared', 'VBD'), ('bond', 'NN'), ('people.it', 'NN'), ('special', 'JJ'), ('kind', 'NN'),

('must', 'MD'), ('tried', 'VB'), ('searching', 'VBG'), ('friend', 'NN'), ('.but', 'NN'), ('never', 'RB'), ('found', 'VBD'), ('
```

PROGRAM NO:20

Date:1/03/2022

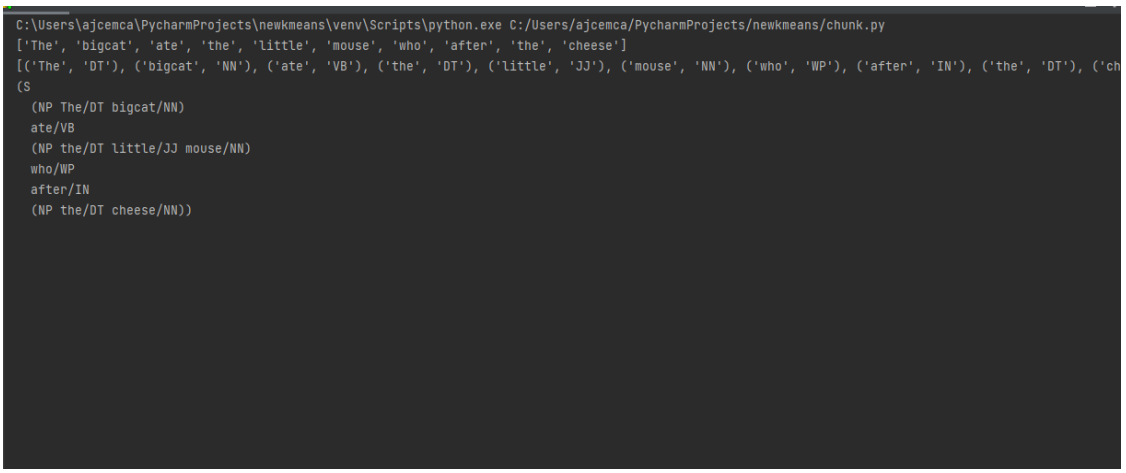
AIM:

program for Natural Language Processing which performs Chunking

PROGRAM

```
import nltk
new= "The bigcat ate the little mouse who after the cheese"
new_tokens=nltk.word_tokenize(new)
print(new_tokens)

new_tag = nltk.pos_tag(new_tokens)
print(new_tag)

grammer=r"NP: {<DT>?<JJ>*<NN>}"
chunkParser = nltk.RegexpParser(grammer)
chunked=chunkParser.parse(new_tag)
print(chunked)
chunked.draw()
```

OUTPUT



PROGRAM NO:21

Date:1/03/2022

AIM:

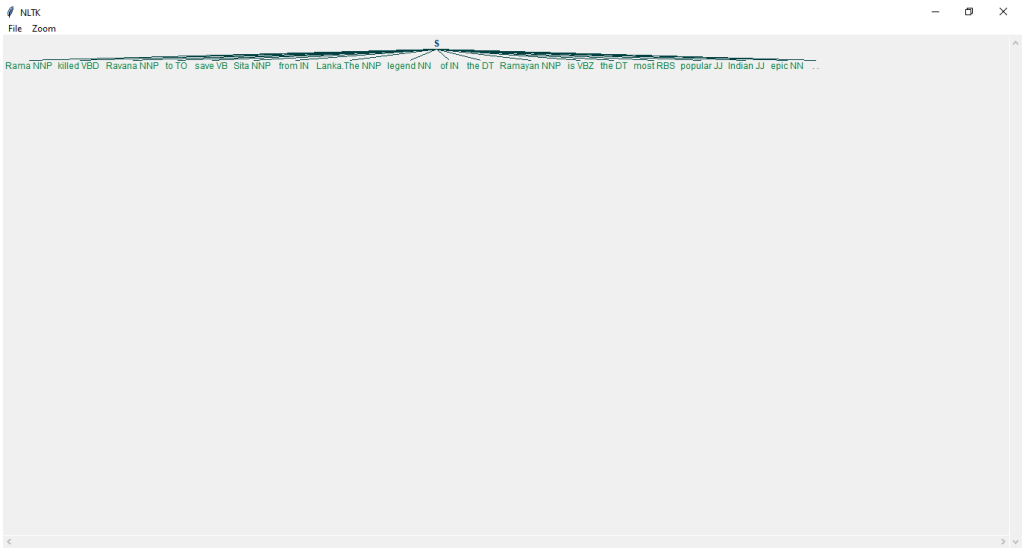
program for Natural Language Processing which performs Chunking

PROGRAM

```
import nltk
nltk.download('averaged_perception_tagger')
sample_text="""
Rama killed Ravana to save Sita from Lanka.The legend of the Ramayan is the most popular Indian epic.
A lot of Movies and
serials have have already been shot in several language here in India based on the Ramayana."""

tokenize= nltk.sent_tokenize(sample_text)
for i in tokenize:
words = nltk.word_tokenize(i)
tagged_words = nltk.pos_tag(words)
chunkGram=r"""VB: {}"""
chunkParser=nltk.RegexpParser(chunkGram)
chunked=chunkParser.parse(tagged_words)
print(chunked)
chunked.draw()
```

OUTPUT

```
(S
  Rama/NNP
  killed/VBD
  Ravana/NNP
  to/TO
  save/VB
  Sita/NNP
  from/IN
  Lanka.The/NNP
  legend/NN
  of/IN
  the/DT
  Ramayan/NNP
  is/VBZ
  the/DT
  most/RBS
  popular/JJ
  Indian/JJ
  epic/NN
  ./.)
```


NLTK
File Zoom
S
Rama NNP killed VBD Ravana NNP to TO save VB Sita NNP from IN Lanka.The NNP legend NN of IN the DT Ramayan NNP is VBZ the DT most RBS popular JJ Indian JJ epic NN .