**Program 5**

**Build an Artificial Neural Network by implementing the Back propagation algorithm and test the same using appropriate data sets**

import numpy as np

x=np.array(([2,9],[1,5],[3,6]),dtype=float)

y=np.array(([92],[86],[89]),dtype=float)

x=x/np.amax(x,axis=0)

y=y/100

def sigmoid(x):

return (1/(1+np.exp(-x)))

def derivatives\_sigmoid(x):

return x\*(1-x)

epoch=7000

lr=0.1

inputlayer\_neuron=2

hiddenlayer\_neuron=3

output\_neuron=1

wh=np.random.uniform(size=(inputlayer\_neuron,hiddenlayer\_neuron))

bh=np.random.uniform(size=(1,hiddenlayer\_neuron))

wout=np.random.uniform(size=(hiddenlayer\_neuron,output\_neuron))

bout=np.random.uniform(size=(1,output\_neuron))

for i in range(epoch):

hinp1=np.dot(x,wh)

hinp=hinp1+bh

hlayer\_act=sigmoid(hinp)

outinp1=np.dot(hlayer\_act,wout)

outinp=outinp1+bout

output=sigmoid(outinp)

EO=y-output

outgrad=derivatives\_sigmoid(output)

d\_output=EO\*outgrad

EH=d\_output.dot(wout.T)

hiddengrad=derivatives\_sigmoid(hlayer\_act)

d\_hiddenlayer=EH\*hiddengrad

wout+=hlayer\_act.T.dot(d\_output\*lr)

wh+=x.T.dot(d\_hiddenlayer)\*lr

print("input:\n"+str(x))

print("actual output:\n"+str(y))

print("predicted output:\n",output)