**Aim:** Write a program to implement Back Propagation Network.

**Code:**

importnumpy as np

x1 = float(input("Enter X1: "))

print(x1)

x2 = float(input("Enter X2: "))

print(x2)

b1 = float(input("Enter bias 1: "))

b2 = float(input("Enter bias 2: "))

b3 = float(input("Enter bias 3: "))

alpha = float(input("Enter alpha: "))

t = float(input("Enter target: "))

a = [0.6,0.3,-0.1,-0.3,0.4,0.5,0.4,0.1,-0.2]

print('phase 1')

zin1 = float(b1\*a[1]+x1\*a[0]+x2\*a[2])

print('zin1=',zin1)

zp1 = 1/(1+np.exp(-zin1))

print('z1=',zp1)

fzin1= zp1\*(1-zp1)

print('fzin1=',fzin1)

zin2= float(a[3]\*x1+a[4]\*x2+a[5]\*b2)

print('zin2=',zin2)

zp2 = 1/(1+np.exp(-zin2))

print('z2=',zp2)

fzin2= zp2\*(1-zp2)

print('fzin2=',fzin2)

yin=float(zp1\*a[6]+zp2\*a[7]+b3\*a[8])

print('yin=',yin)

y = 1/(1+np.exp(-yin))

print('y=',y)

fyin= y\*(1-y)

print('fyin=',fyin)

print('phase 2')

dell1=(t-y)\*fyin

print('dell1=',dell1)

delta\_w11=alpha\*dell1\*zp1

print('delta\_w11=',delta\_w11)

delta\_w21=alpha\*dell1\*zp2

print('delta\_w21=',delta\_w21)

dellin1=dell1\*a[6]

print('dellin1=',dellin1)

dellin2 = dell1\*a[7]

print('dellin2=',dellin2)

delta1=dellin1\*fzin1

print('delta1=',delta1)

delta2=dellin2\*fzin2

print('delta2=',delta2)

delta\_w01=alpha\*dell1

print('delta\_w01=',delta\_w01)

print('phase 3')

delta\_v11=alpha\*delta1\*x1

print('delta\_v11=',delta\_v11)

delta\_v12=alpha\*delta2\*x1

print('delta\_v12=',delta\_v12)

delta\_v21=alpha\*delta1\*x2

print('delta\_v21=',delta\_v21)

delta\_v22=alpha\*delta2\*x2

print('delta\_v22=',delta\_v22)

delta\_v01 = alpha\*delta1

print('delta\_v01=',delta\_v01)

delta\_v02 = alpha\*delta2

print('delta\_v02=',delta\_v02)

**Output:**

