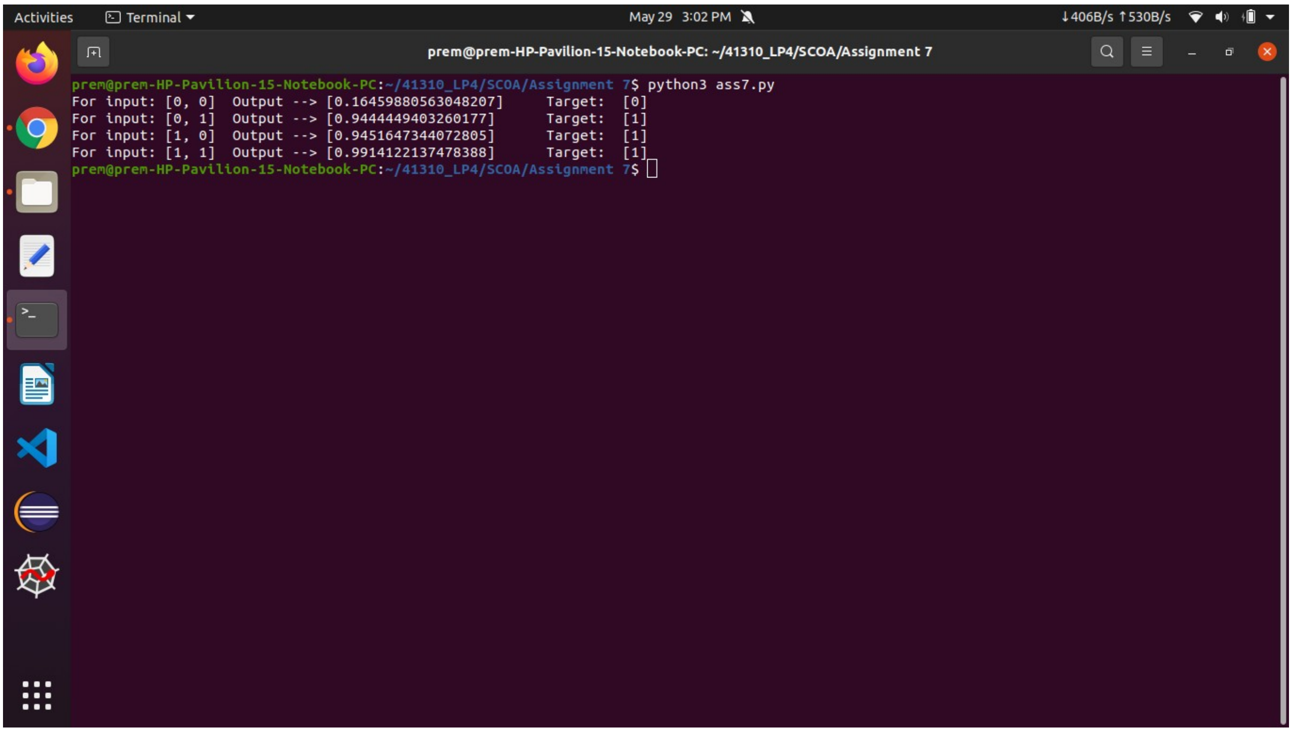
Name : Prem Vinod Bansod
Roll no : 41310
Assignment No : 07 (SCOA)

Code:

```
import string
import math
import random

class Neural:
	def __init__(self, pattern):
		self.ni=3
		self.nh=3
		self.no=1

		self.wih = []
		for i in range(self.ni):
			self.wih.append([0.0]*self.nh)

		self.who = []
		for j in range(self.nh):
			self.who.append([0.0]*self.no)


		self.ai, self.ah, self.ao = [],[],[]
		self.ai=[1.0]*self.ni
		self.ah=[1.0]*self.nh
		self.ao=[1.0]*self.no


		randomizeMatrix(self.wih,-0.2,0.2)
		randomizeMatrix(self.who,-2.0,2.0)


		self.cih = []
		self.cho = []
		for i in range(self.ni):
			self.cih.append([0.0]*self.nh)
		for j in range(self.nh):
			self.cho.append([0.0]*self.no)

	def backpropagate(self, inputs, expected, output, N=0.5, M=0.1):

		output_deltas = [0.0]*self.no
		for k in range(self.no):

			error = expected[k] - output[k]
			output_deltas[k]=error*dsigmoid(self.ao[k])


		for j in range(self.nh):
```

```
            for k in range(self.no):
                delta_weight = self.ah[j] * output_deltas[k]
                self.who[j][k]+= M*self.cho[j][k] + N*delta_weight
                self.cho[j][k]=delta_weight


        hidden_deltas = [0.0]*self.nh
        for j in range(self.nh):

            error=0.0
            for k in range(self.no):
                error+=self.who[j][k] * output_deltas[k]

            hidden_deltas[j]= error * dsigmoid(self.ah[j])

        for i in range(self.ni):
            for j in range(self.nh):
                delta_weight = hidden_deltas[j] * self.ai[i]
                self.wih[i][j]+= M*self.cih[i][j] + N*delta_weight
                self.cih[i][j]=delta_weight


    def test(self, patterns):
        for p in patterns:
            inputs = p[0]
            print('For input:', p[0], ' Output -->', self.runNetwork(inputs), '\tTarget: ', p[1])



    def runNetwork(self, feed):
        if(len(feed)!=self.ni-1):
            print('Error in number of input values.')


        for i in range(self.ni-1):
            self.ai[i]=feed[i]


        for j in range(self.nh):
            sum=0.0
            for i in range(self.ni):
                sum+=self.ai[i]*self.wih[i][j]

            self.ah[j]=sigmoid(sum)

        for k in range(self.no):
            sum=0.0
            for j in range(self.nh):
                sum+=self.ah[j]*self.wih[j][k]

            self.ao[k]=sigmoid(sum)
        return self.ao


    def trainNetwork(self, pattern):
        for i in range(500):

            for p in pattern:

                inputs = p[0]
                out = self.runNetwork(inputs)
                expected = p[1]
                self.backpropagate(inputs,expected,out)
        self.test(pattern)




def randomizeMatrix ( matrix, a, b):
    for i in range ( len (matrix) ):
        for j in range ( len (matrix[0]) ):

            matrix[i][j] = random.uniform(a,b)




def sigmoid(x):
    return 1 / (1 + math.exp(-x))




def dsigmoid(y):
    return y * (1 - y)



def main():

    pat = [
        [[0,0], [0]],
        [[0,1], [1]],
        [[1,0], [1]],
        [[1,1], [1]]
    ]
    newNeural = Neural(pat)
    newNeural.trainNetwork(pat)


if __name__ == "__main__":
    main()
```

Output:



Activities
Terminal
May 29 3:02 PM
↓406B/s ↑530B/s
prem@prem-HP-Pavilion-15-Notebook-PC: ~/41310_LP4/SCOA/Assignment 7

```
prem@prem-HP-Pavilion-15-Notebook-PC:~/41310_LP4/SCOA/Assignment 7$ python3 ass7.py
For input: [0, 0]  Output --> [0.16459880563048207]     Target:  [0]
For input: [0, 1]  Output --> [0.9444449403260177]      Target:  [1]
For input: [1, 0]  Output --> [0.9451647344072805]      Target:  [1]
For input: [1, 1]  Output --> [0.9914122137478388]      Target:  [1]
prem@prem-HP-Pavilion-15-Notebook-PC:~/41310_LP4/SCOA/Assignment 7$ 
```