Name:-Pragati Anigiree

URN:-18132007

Roll\_no:-428

**Experiment No:1**

**Implementation of Perceptron program for AND Logic Gate**

**Program:**

import numpy as np

def unitStep(v):

if v >= 0:

return 1

else:

return 0

def perceptronModel(x, w, b):

v = np.dot(w, x) + b

y = unitStep(v)

return y

def AND\_Fun(x):

w = np.array([1, 1])

b = -1.5

return perceptronModel(x, w, b)

t1 = np.array([0, 1])

t2 = np.array([1, 1])

t3 = np.array([0, 0])

t4 = np.array([1, 0])

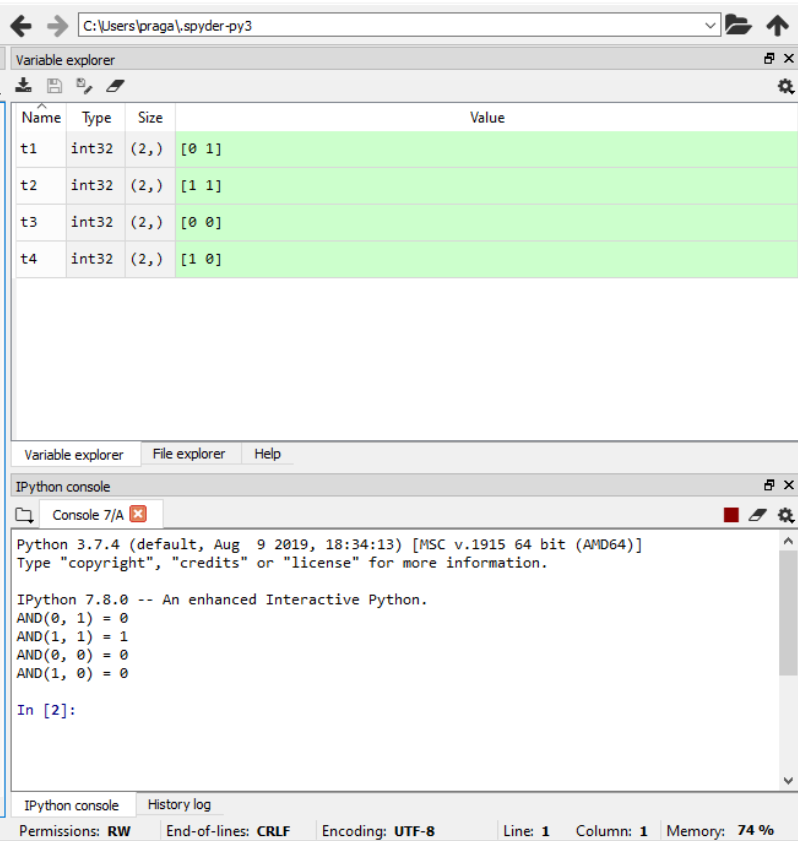
print("AND({}, {}) = {}".format(0, 1, AND\_Fun(t1)))

print("AND({}, {}) = {}".format(1, 1, AND\_Fun(t2)))

print("AND({}, {}) = {}".format(0, 0, AND\_Fun(t3)))

print("AND({}, {}) = {}".format(1, 0, AND\_Fun(t4)))

**OUTPUT:**



**Implementation of Perceptron program for OR Logic Gate**

**Program:**

import numpy as np

def unitStep(v):

if v >= 0:

return 1

else:

return 0

def perceptronModel(x, w, b):

v = np.dot(w, x) + b

y = unitStep(v)

return y

def OR\_Fun(x):

w = np.array([1, 1])

b = -0.5

return perceptronModel(x, w, b)

t1 = np.array([0, 1])

t2 = np.array([1, 1])

t3 = np.array([0, 0])

t4 = np.array([1, 0])

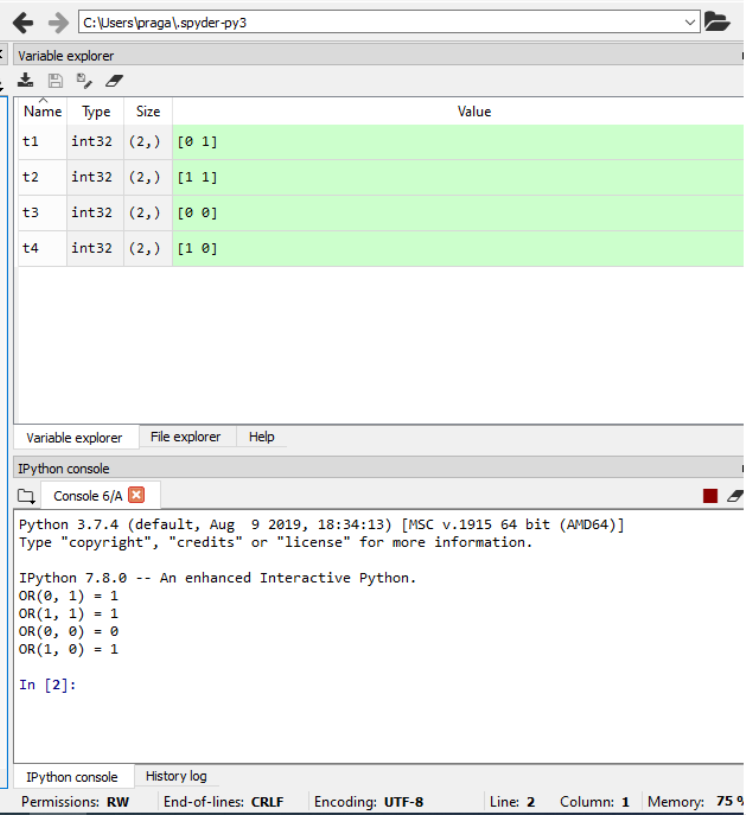
print("OR({}, {}) = {}".format(0, 1, OR\_Fun(t1)))

print("OR({}, {}) = {}".format(1, 1, OR\_Fun(t2)))

print("OR({}, {}) = {}".format(0, 0, OR\_Fun(t3)))

print("OR({}, {}) = {}".format(1, 0, OR\_Fun(t4)))

**OUTPUT:**



**Implementation of Perceptron program for XOR Logic Gate**

**Program:**

import numpy as np

def unitStep(v):

if v >= 0:

return 1

else:

return 0

def perceptronModel(x, w, b):

v = np.dot(w, x) + b

y = unitStep(v)

return y

def NOT\_Fun(x):

wNOT = -1

bNOT = 0.5

return perceptronModel(x, wNOT, bNOT)

def AND\_Fun(x):

w = np.array([1, 1])

bAND = -1.5

return perceptronModel(x, w, bAND)

def OR\_Fun(x):

w = np.array([1, 1])

bOR = -0.5

return perceptronModel(x, w, bOR)

def XOR\_Fun(x):

y1 = AND\_Fun(x)

y2 = OR\_Fun(x)

y3 = NOT\_Fun(y1)

final\_x = np.array([y2, y3])

finalOutput = AND\_Fun(final\_x)

return finalOutput

t1 = np.array([0, 1])

t2 = np.array([1, 1])

t3 = np.array([0, 0])

t4 = np.array([1, 0])

print("XOR({}, {}) = {}".format(0, 1, XOR\_Fun(t1)))

print("XOR({}, {}) = {}".format(1, 1, XOR\_Fun(t2)))

print("XOR({}, {}) = {}".format(0, 0, XOR\_Fun(t3)))

print("XOR({}, {}) = {}".format(1, 0, XOR\_Fun(t4)))

**OUTPUT:**

