**1)AND Gate with 2 inputs  
 Program**

import numpy as np

def binary\_treshhold(result):

if result>=2:

return 1

else:

return 0

#x-input

#w-weight

#b-bias

def perceptron(x,w,b) :

temp=x.dot(w)+b

return binary\_treshhold(temp)

def ANDgate(x):

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

b=0.5

return perceptron(x,w,b)

example1=np.array([0,0])

example2=np.array([0,1])

example3=np.array([1,0])

example4=np.array([1,1])

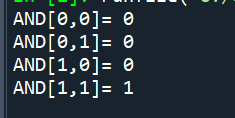
print("AND[0,0]=",ANDgate(example1))

print("AND[0,1]=",ANDgate(example2))

print("AND[1,0]=",ANDgate(example3))

print("AND[1,1]=",ANDgate(example4))

**Output**

****

**2)OR gate with 2 inputs**

**Program**

import numpy as np

def binary\_treshhold(result):

if result>1:

return 1

else:

return 0

#x-input

#w-weight

#b-bias

def perceptron(x,w,b) :

temp=x.dot(w)+b

return binary\_treshhold(temp)

def ORgate(x):

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

b=1

return perceptron(x,w,b)

example1=np.array([0,0])

example2=np.array([0,1])

example3=np.array([1,0])

example4=np.array([1,1])

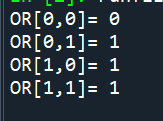
print("OR[0,0]=",ORgate(example1))

print("OR[0,1]=",ORgate(example2))

print("OR[1,0]=",ORgate(example3))

print("OR[1,1]=",ORgate(example4))

**Output**

****

**3)NAND Gate with 2 inputs**

**Program**

import numpy as np

def binary\_treshhold(result):

if result<2:

return 1

else:

return 0

#x-input

#w-weight

#b-bias

def perceptron(x,w,b) :

temp=x.dot(w)+b

return binary\_treshhold(temp)

def NANDgate(x):

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

b=0.5

return perceptron(x,w,b)

example1=np.array([0,0])

example2=np.array([0,1])

example3=np.array([1,0])

example4=np.array([1,1])

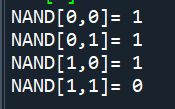
print("NAND[0,0]=",NANDgate(example1))

print("NAND[0,1]=",NANDgate(example2))

print("NAND[1,0]=",NANDgate(example3))

print("NAND[1,1]=",NANDgate(example4))

**Output**

****

**4)NOR Gate with 2 inputs**

**Program**

import numpy as np

def binary\_treshhold(result):

if result<2:

return 1

else:

return 0

#x-input

#w-weight

#b-bias

def perceptron(x,w,b) :

temp=x.dot(w)+b

return binary\_treshhold(temp)

def NORgate(x):

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

b=1

return perceptron(x,w,b)

example1=np.array([0,0])

example2=np.array([0,1])

example3=np.array([1,0])

example4=np.array([1,1])

print("NOR[0,0]=",NORgate(example1))

print("NOR[0,1]=",NORgate(example2))

print("NOR[1,0]=",NORgate(example3))

print("NOR[1,1]=",NORgate(example4))

**Output**

**A screenshot of a computer program

Description automatically generated with low confidence**

**5)XOR Gate with 2 inputs**

**Program**

import numpy as np

def binarythreshhold(result):

if result>=0:

return 1

else:

return 0

def neuron(x\_inputs,wt\_values,bias):

sum1=np.dot(x\_inputs,wt\_values)+bias

return binarythreshhold(sum1)

def xorfn(x):

#Assign wts for input of each of 3 neurons

wts\_gt1=np.array([-1,-1])

wts\_gt2=np.array([-1,-1])

wts\_gt3=np.array([1,-1])

#assign bias value for each og gates

bias1=1.5

bias2=0.5

bias3=-0.5

#Obtain reslt for gate1

gate1\_op=neuron(x,wts\_gt1,bias1)

#Obtain reslt for gate2

gate2\_op=neuron(x,wts\_gt2,bias2)

#pass gate1 and gate2 op as input to gate3

gate\_3\_input=np.array([gate1\_op,gate2\_op])

return neuron(gate\_3\_input,wts\_gt3,bias3)

inps=[]

n=int(input("Enter no of test cases: "))

print("Enter 2 digits 0 or 1")

for i in range(n):

i1=int(input(""))

i2=int(input(""))

eg1=np.array([i1,i2])

inps.append(eg1)

for ins in inps:

print("XOR (",ins[0]," , ",ins[1],") : ",xorfn(ins))

**Output**

**A screenshot of a computer program

Description automatically generated with medium confidence**

**6)NOT gate**

**Program**

def binary\_treshhold(result):

if result<2:

return 1

else:

return 0

#x-input

#w-weight

#b-bias

def perceptron(x,w,b) :

temp=x\*w+b

return binary\_treshhold(temp)

def NOTgate(x):

w=1

b=1

return perceptron(x,w,b)

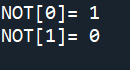
example1=0

example2=1

print("NOT[0]=",NOTgate(example1))

print("NOT[1]=",NOTgate(example2))

**Output**

****

**7)AND Gate with 3 inputs**

**Program**

import numpy as np

def binary\_treshhold(result):

if result>=4:

return 1

else:

return 0

#x-input

#w-weight

#b-bias

def perceptron(x,w,b) :

temp=x.dot(w)+b

return binary\_treshhold(temp)

def ANDgate(x):

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

b=1

return perceptron(x,w,b)

example1=np.array([0,0,0])

example2=np.array([0,0,1])

example3=np.array([0,1,0])

example4=np.array([0,1,1])

example5=np.array([1,0,0])

example6=np.array([1,0,1])

example7=np.array([1,1,0])

example8=np.array([1,1,1])

print("AND[0,0,0]=",ANDgate(example1))

print("AND[0,0,1]=",ANDgate(example2))

print("AND[0,1,0]=",ANDgate(example3))

print("AND[0,1,1]=",ANDgate(example4))

print("AND[1,0,0]=",ANDgate(example5))

print("AND[1,0,1]=",ANDgate(example6))

print("AND[1,1,0]=",ANDgate(example7))

print("AND[1,1,1]=",ANDgate(example8))

**Output**

**A picture containing text, font, screenshot, typography

Description automatically generated**

**8)OR Gate with 3 inputs**

**Program**

import numpy as np

def binary\_treshhold(result):

if result>1:

return 1

else:

return 0

#x-input

#w-weight

#b-bias

def perceptron(x,w,b) :

temp=x.dot(w)+b

return binary\_treshhold(temp)

def ORgate(x):

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

b=1

return perceptron(x,w,b)

example1=np.array([0,0,0])

example2=np.array([0,0,1])

example3=np.array([0,1,0])

example4=np.array([0,1,1])

example5=np.array([1,0,0])

example6=np.array([1,0,1])

example7=np.array([1,1,0])

example8=np.array([1,1,1])

print("OR[0,0,0]=",ORgate(example1))

print("OR[0,0,1]=",ORgate(example2))

print("OR[0,1,0]=",ORgate(example3))

print("OR[0,1,1]=",ORgate(example4))

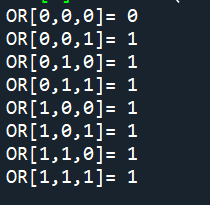
print("OR[1,0,0]=",ORgate(example5))

print("OR[1,0,1]=",ORgate(example6))

print("OR[1,1,0]=",ORgate(example7))

print("OR[1,1,1]=",ORgate(example8))

**Output**

****