ARTIFICIAL INTELLIGENCE LAB EXPERIMENTs

**AND-Gate:**

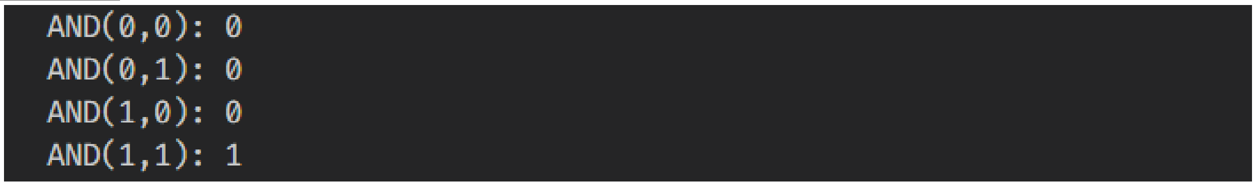
```
def activation(v):
   if v<=1.5:
      return 0
   else:
      return 1

def perceptron(x,w,b):
   r=np.dot(x,w)+b
   return activation(r)

def AND_gate(x):
   w=np.array([1,1])
   b=0.5
   return perceptron(x,w,b)

print("AND(0,0):",AND_gate(np.array([0,0])))
print("AND(0,1):",AND_gate(np.array([0,1])))
print("AND(1,0):",AND_gate(np.array([1,0])))
print("AND(1,1):",AND_gate(np.array([1,1])))
```

**OUTPUT :**



```
AND(0,0): 0
AND(0,1): 0
AND(1,0): 0
AND(1,1): 1
```

**NAND-Gate :**

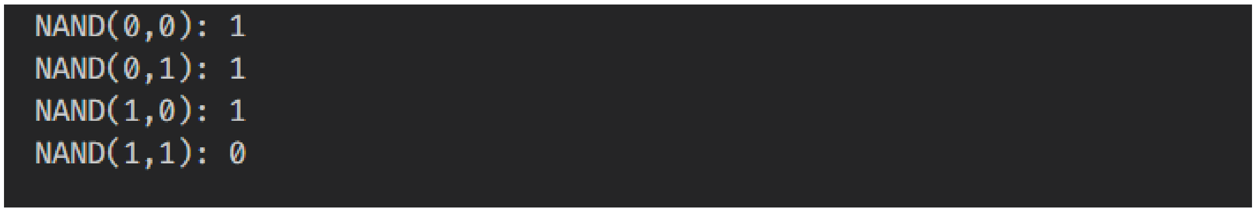
```
import numpy as np
def activation(v):
   if v<=1.5:
      return 1
   else:
      return 0

def perceptron(x,w,b):
   r=np.dot(x,w)+b
   return activation(r)

def NAND_gate(x):
   w=np.array([1,1])
   b=0.5
   return perceptron(x,w,b)

print("NAND(0,0):",NAND_gate(np.array([0,0])))
print("NAND(0,1):",NAND_gate(np.array([0,1])))
print("NAND(1,0):",NAND_gate(np.array([1,0])))
print("NAND(1,1):",NAND_gate(np.array([1,1])))
```

**OUTPUT** :



```
NAND(0,0): 1
NAND(0,1): 1
NAND(1,0): 1
NAND(1,1): 0
```

**OR-Gate** :

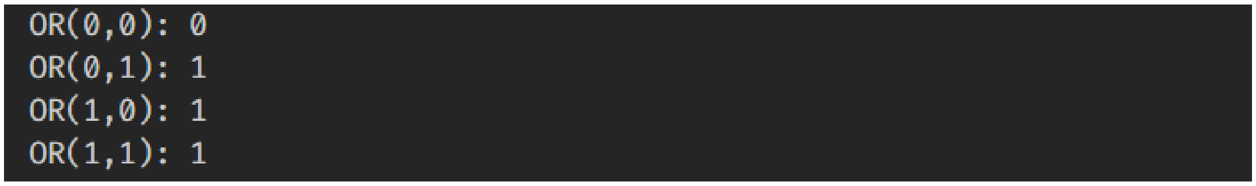
```
import numpy as np
def activation(v):
    if v<=0.5:
        return 0
    else:
        return 1

def perceptron(x,w,b):
    r=np.dot(x,w)+b
    return activation(r)

def OR_gate(x):
    w=np.array([1,1])
    b=0.5
    return perceptron(x,w,b)

print("OR(0,0):",OR_gate(np.array([0,0])))
print("OR(0,1):",OR_gate(np.array([0,1])))
print("OR(1,0):",OR_gate(np.array([1,0])))
print("OR(1,1):",OR_gate(np.array([1,1])))
```

**OUTPUT** :



```
OR(0,0): 0
OR(0,1): 1
OR(1,0): 1
OR(1,1): 1
```

**NOR-Gate** :

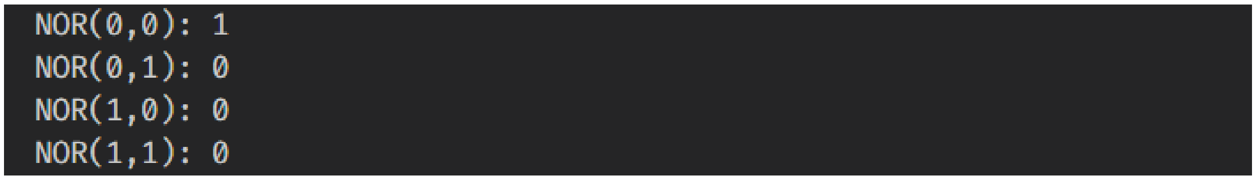
```
import numpy as np
def activation(v):
    if v<=0.5:
        return 1
    else:
        return 0

def perceptron(x,w,b):
    r=np.dot(x,w)+b
    return activation(r)

def NOR_gate(x):
    w=np.array([1,1])
    b=0.5
    return perceptron(x,w,b)

print("NOR(0,0):",NOR_gate(np.array([0,0])))
print("NOR(0,1):",NOR_gate(np.array([0,1])))
print("NOR(1,0):",NOR_gate(np.array([1,0])))
print("NOR(1,1):",NOR_gate(np.array([1,1])))
```

**OUTPUT** :

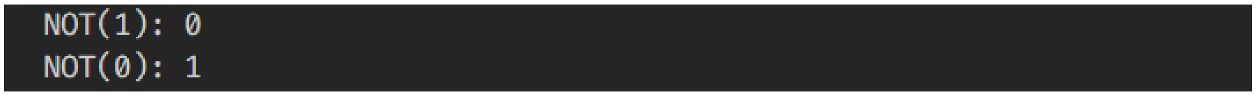


```
NOR(0,0): 1
NOR(0,1): 0
NOR(1,0): 0
NOR(1,1): 0
```

**NOT-Gate** :

```
import numpy as np
def activation(v):
    if v<=0.5:
        return 1
    else:
        return 0
def perceptron(x,w,b):
    r=np.dot(x,w)+b
    return activation(r)
def NOT_gate(x):
    w=1
    b=0.5
    return perceptron(x,w,b)
print("NOT(1):",NOT_gate(1))
print("NOT(0):",NOT_gate(0))
```

**OUTPUT** :



```
NOT(1): 0
NOT(0): 1
```