```
import numpy as np
class mcp:
    def __init__(self,X):
        self.X=X
        #self.y=y

    def g(self):
        return np.sum(self.X,axis=1)

    def f(self,op):
        y=[]
        b=self.X.shape[1] if op=='And' else 1
        for i in range(X.shape[0]):
            #print(1 if self.g()[i] >= b else 0)
            if self.g()[i] >= b:
                y.append([1])
            else:
                y.append([0])
        return np.array(y)

    #def loss_cal(y_pred,y):
    #   return ((y-y_pred)**2).mean()


n=int(input('Enter no of instances: '))
m=int(input('Enter no of features: '))
X=np.random.randint(2,size=(n,m))
print('\nTruth table:\n',X)
neuron=mcp(X)
print('\nOuput of And operation:\n',neuron.f('And'))
print('\nOuput of Or operation:\n',neuron.f('Or'))
```

```
Enter no of instances: 4
Enter no of features: 3

Truth table:
 [[0 1 0]
 [1 1 1]
 [0 1 0]
 [0 1 0]]

Ouput of And operation:
 [[0]
 [1]
 [0]
 [0]]

Ouput of Or operation:
 [[1]
 [1]
 [1]
 [1]]
```

✓ 0s completed at 11:22 AM