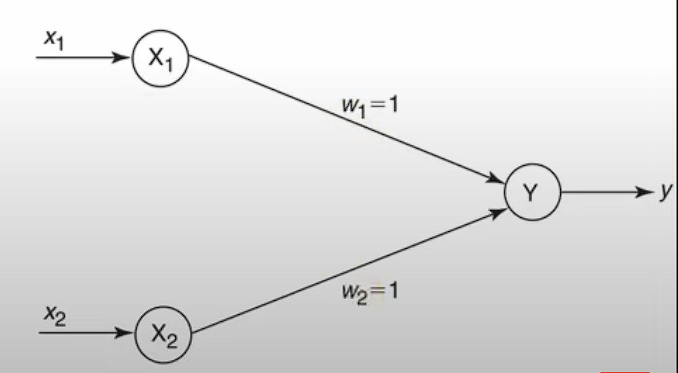
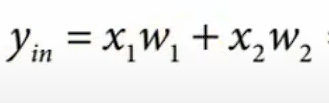
**Experiment No.: 01**

**Aim:** Write a program to implement logical OR using McCulloch Pitts neuron model.



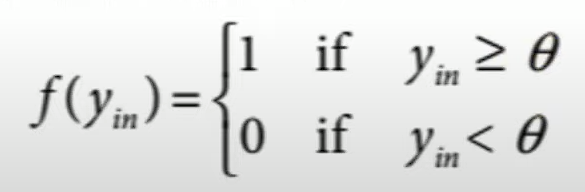
**Steps:**

1. **Define Parameters:**
   * Set the weights (**weight1** and **weight2**), bias, and threshold.
   * In this code, **weight1** and **weight2** are set to 1, bias is set to 0, and the threshold is a parameter with a default value of 1.
2. **Define Test Inputs:**
   * Create a list of test inputs (binary pairs) that you want to evaluate. In your code, **test\_inputs** contains the pairs (0,0), (0,1), (1,0), and (1,1).
3. **Calculate Linear Combination:**



* + For each test input, calculate the linear combination of inputs, weights, and bias using the formula: **linear\_combination = weight1 \* input1 + weight2 \* input2 + bias**.

1. **Apply Activation Function:**



* + Use a threshold-based activation function to determine the output. In your case, the output is set to 1 if the linear combination is greater than or equal to the threshold, and 0 otherwise.

1. **Store Results:**
   * Store the results, including input values, linear combination, and activation output, in a data structure (e.g., a list or DataFrame).

**Code:**

import numpy as np

import pandas as pd

def cal\_output\_and(threshold=1):

weight1 = 1

weight2 = 1

bias = 0

test\_inputs = [(0, 0), (0, 1), (1, 0), (1, 1)]

outputs = []

for test\_input in test\_inputs:

linear\_combination = weight1 \* test\_input[0] + weight2 \* test\_input[1] + bias

output = int(linear\_combination >= threshold)

outputs.append([test\_input[0], test\_input[1], linear\_combination, output])

output\_frame = pd.DataFrame(outputs, columns=['Input 1', 'Input 2', 'Linear Combination', 'Activation Output'])

print(output\_frame.to\_string(index=False))

print('Results for threshold {}: \n'.format(threshold))

return threshold

t = cal\_output\_and()