* 1. **Aim:**

LAB Manual PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.02**

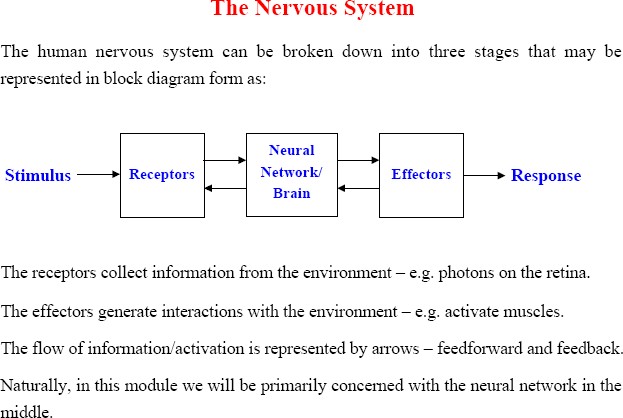
Implementation of logic gate (AND, OR, NOT, NAND, NOR ) using Mc-Culloch Pitts (MCP) model.

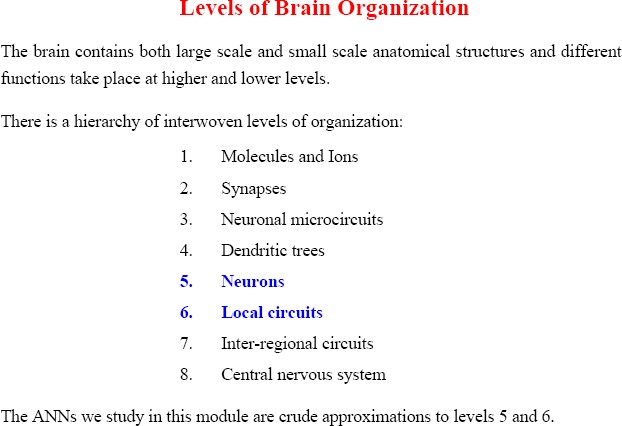
* 1. **Prerequisite:**
     1. Theoretical knowledge of Mc-Culloch Pitts model of neural network.
     2. Knowledge of logic gates.
     3. Different programming language structure overview.
  2. **Outcome:**

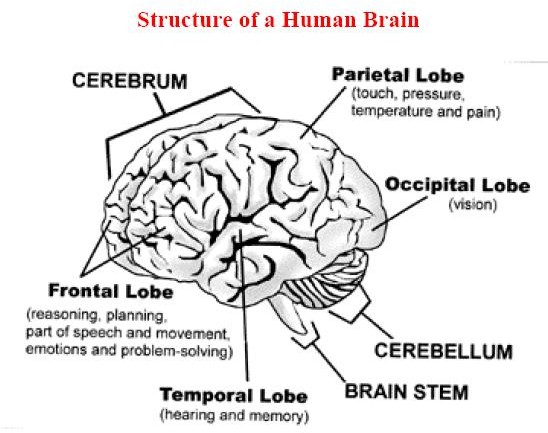
**After successful completion of this experiment students will be able to**

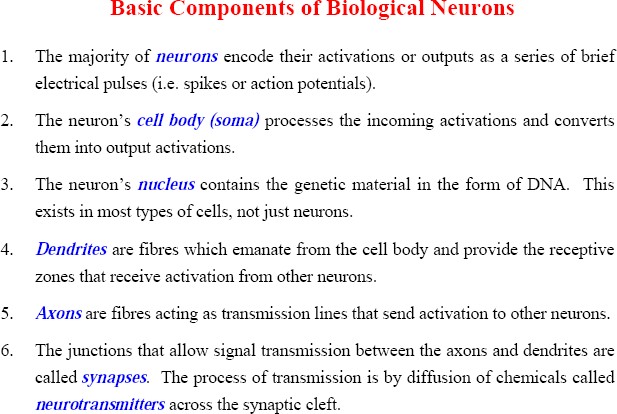
* + 1. Apply MCP Neuron Model to solve simple logic examples.
    2. Design neural network by making use of MCP neuron model.
  1. **Theory:**

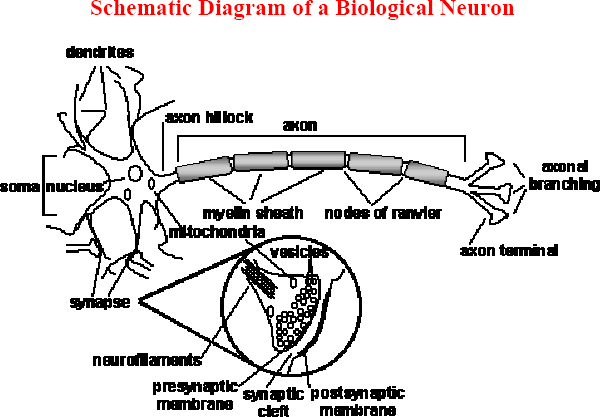
**A.4.1. Biological Neuron**:

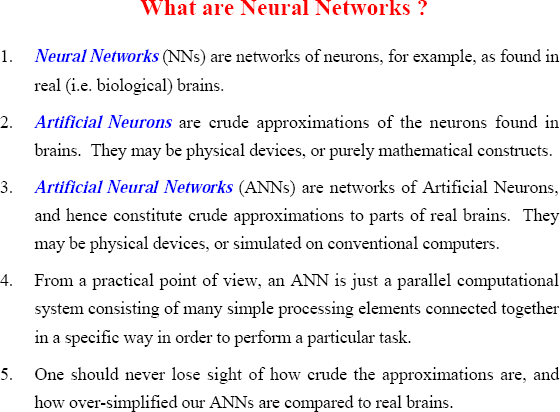


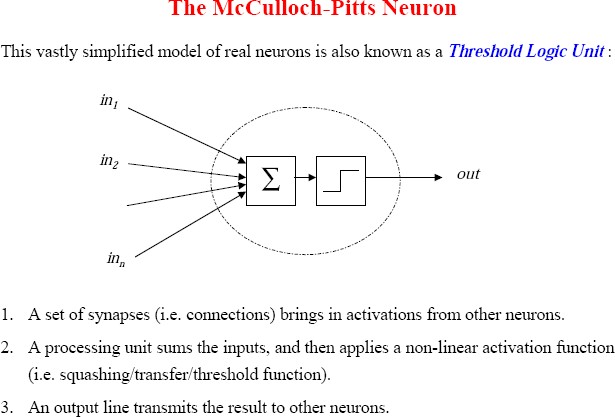


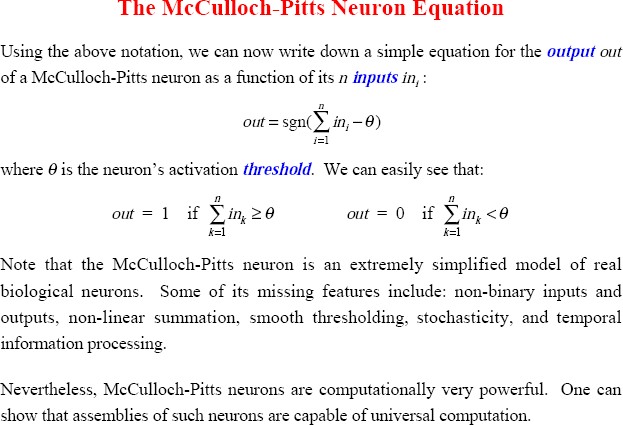


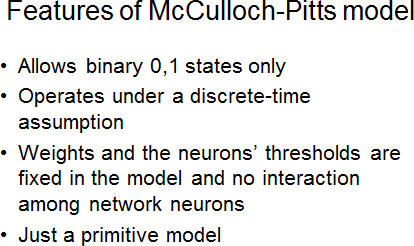


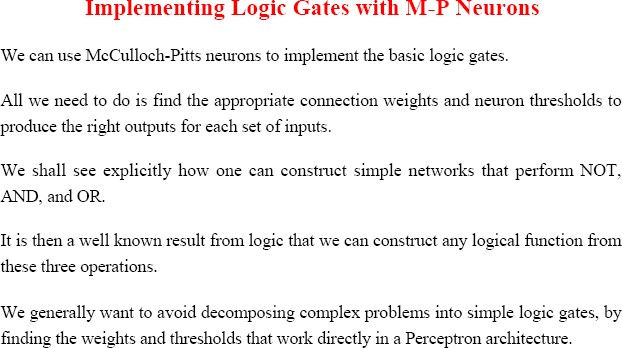


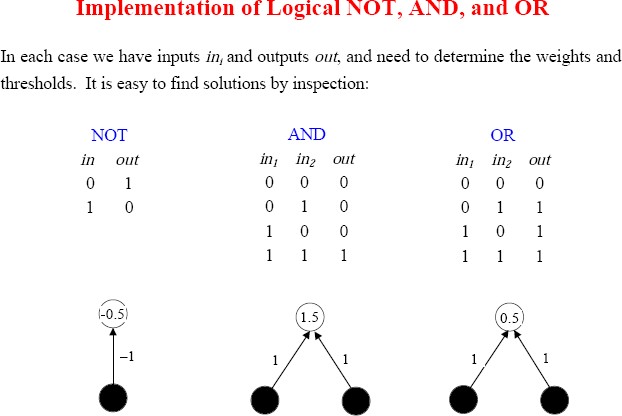


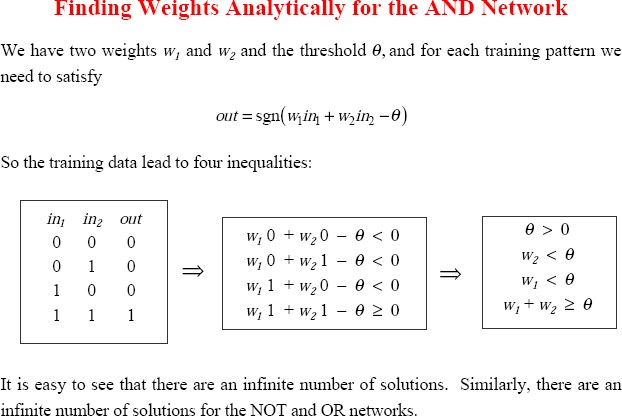












# Procedure/Algorithm:

* + 1. **:**
       1. **Rad the inputs for a logic gate**
       2. **Read weights for each input**
       3. **Read threshold value for the logic gate**
       4. **Determine Yin; total input to the logical gate.**
       5. **Compare Yin with threshold value.**
       6. **Generate the output as 1 if Yin ≥ threshold value; else 0.**
       7. **Repeat the same procedure (Step 1 to 6) for all logic gates.**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No. C043 | Name: Om Kamath |
| Class : B | Batch : B2 |
| Date of Experiment: 9/8/23 | Date of Submission: 10/8/23 |
| Grade : | Time of Submission: |
| Date of Grading: |  |

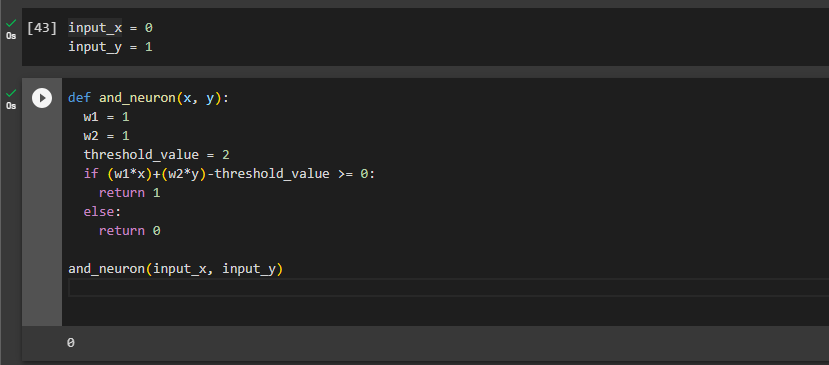
# Software Code written by student:

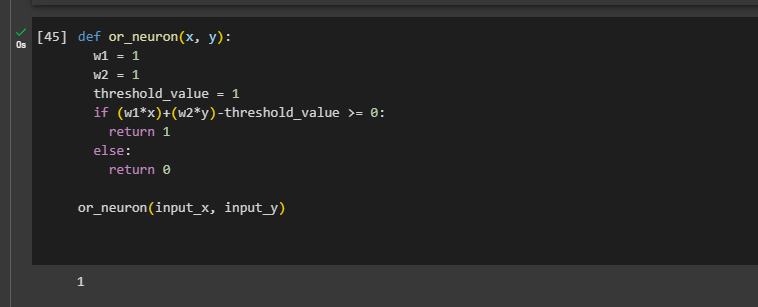
(Paste your Python/Matlab code completed during the 2 hours of practical in the lab here)

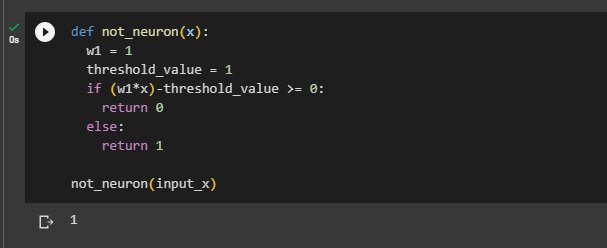
# -\*- coding: utf-8 -\*-

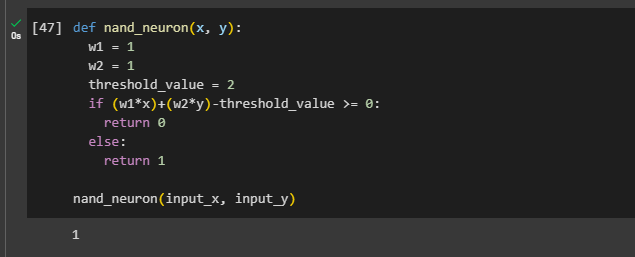
# Input and Output:

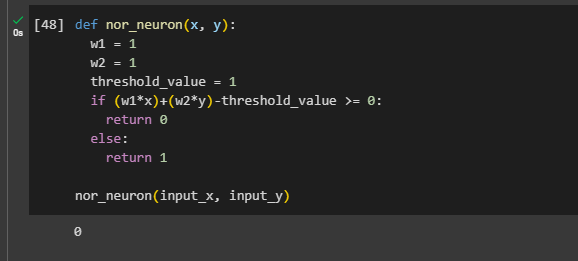
(Paste your program input and output in following format, If there is error then paste the specific error in the output part. In case of error with due permission of the faculty extension can be given to submit the error free code with output in due course of time. Students will be graded accordingly.)

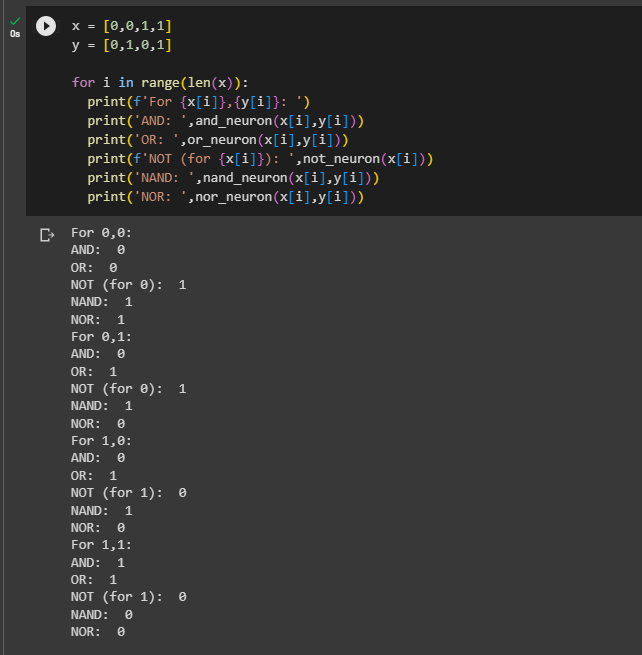












* 1. **Observations and learning:**

(Students are expected to comment on the output obtained with clear observations and learning for each task/ sub part assigned)

Understood how to create an MCP neuron using different weights, summation function and threshold value.

# Conclusion:

(Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.3)

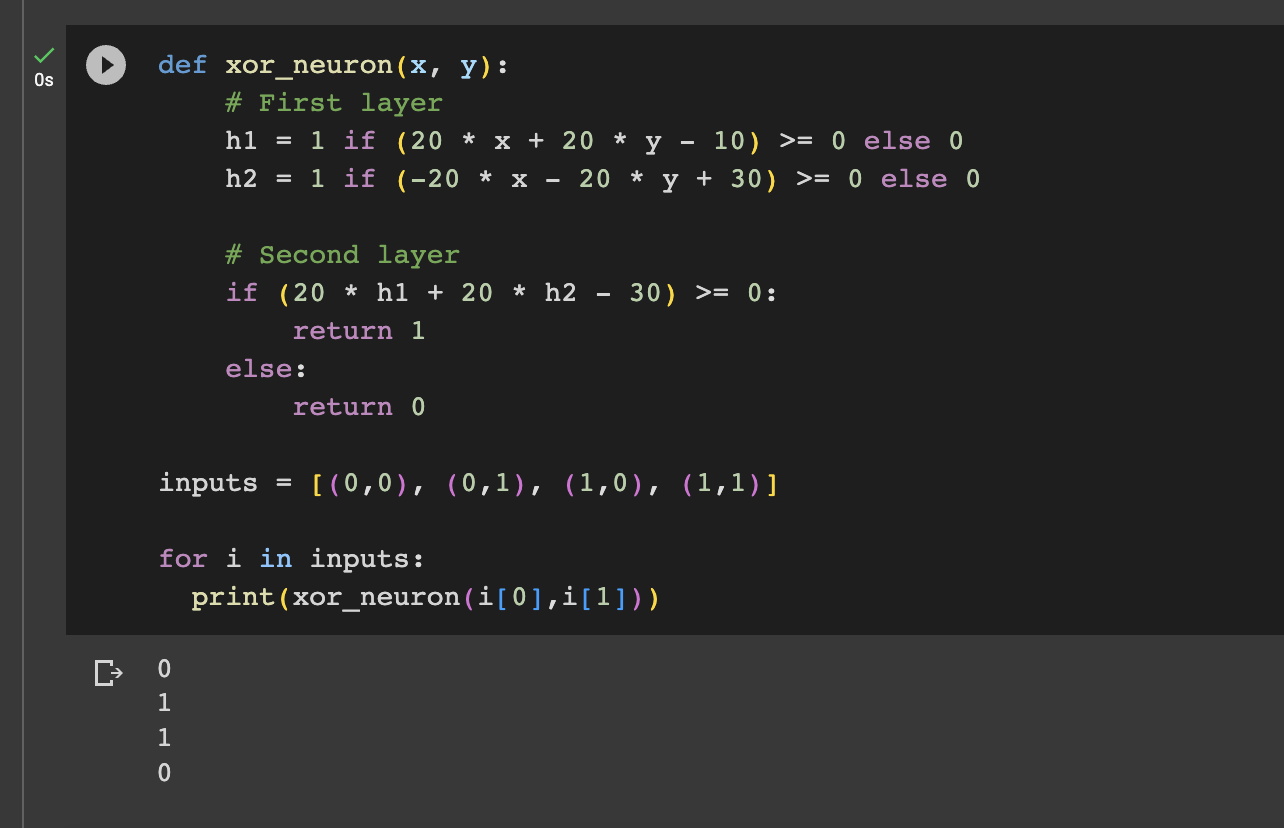
Explored MCP neural network and its similarity with biological neuron.

# Question of Curiosity

* + 1. **Implement ANDNOT Function using Mc-Culloch Pitts neuron model.**

**Same as above**

* + 1. **Implement XOR logic using Mc-Culloch Pitts neuron model.**

****

* + 1. **What are the limitations of Mc-Culloch Pitts neuron model?**

1. Binary Activation: The MCP neuron model works on binary activation, meaning it can only produce outputs of 0 or 1. This restricts its representation power, especially when modeling real-world continuous data.
2. Linear Separability: The MCP neuron can only solve linearly separable problems. For example, it can't solve the XOR problem with a single neuron, as XOR isn't linearly separable.
3. Lack of Learning Algorithm: The original MCP model didn't come with a learning algorithm. Weights had to be set manually, and there was no mechanism to automatically adjust the weights based on training data.
4. Simple Threshold Activation: The threshold activation function is very basic. In practice, more complex and smooth activation functions, like the sigmoid or ReLU, are used because they have better properties for training, especially when using gradient-based methods.
5. No Concept of Backpropagation: The MCP model predates the backpropagation algorithm, which is essential for training multi-layer neural networks. As such, MCP networks would be very limited in their depth and complexity.
6. Absence of Hidden Layers: The basic MCP model doesn't account for hidden layers, which are crucial for capturing complex patterns and relationships in data.
7. Absence of Bias: The original model doesn't incorporate bias, which is essential for shifting the activation function and allowing for more flexible decision boundaries.
8. Simplicity: While its simplicity can be seen as a strength when introducing neural networks, the MCP model is overly simplistic for practical applications in modern machine learning.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*