Assignment 6

Experiment 12:

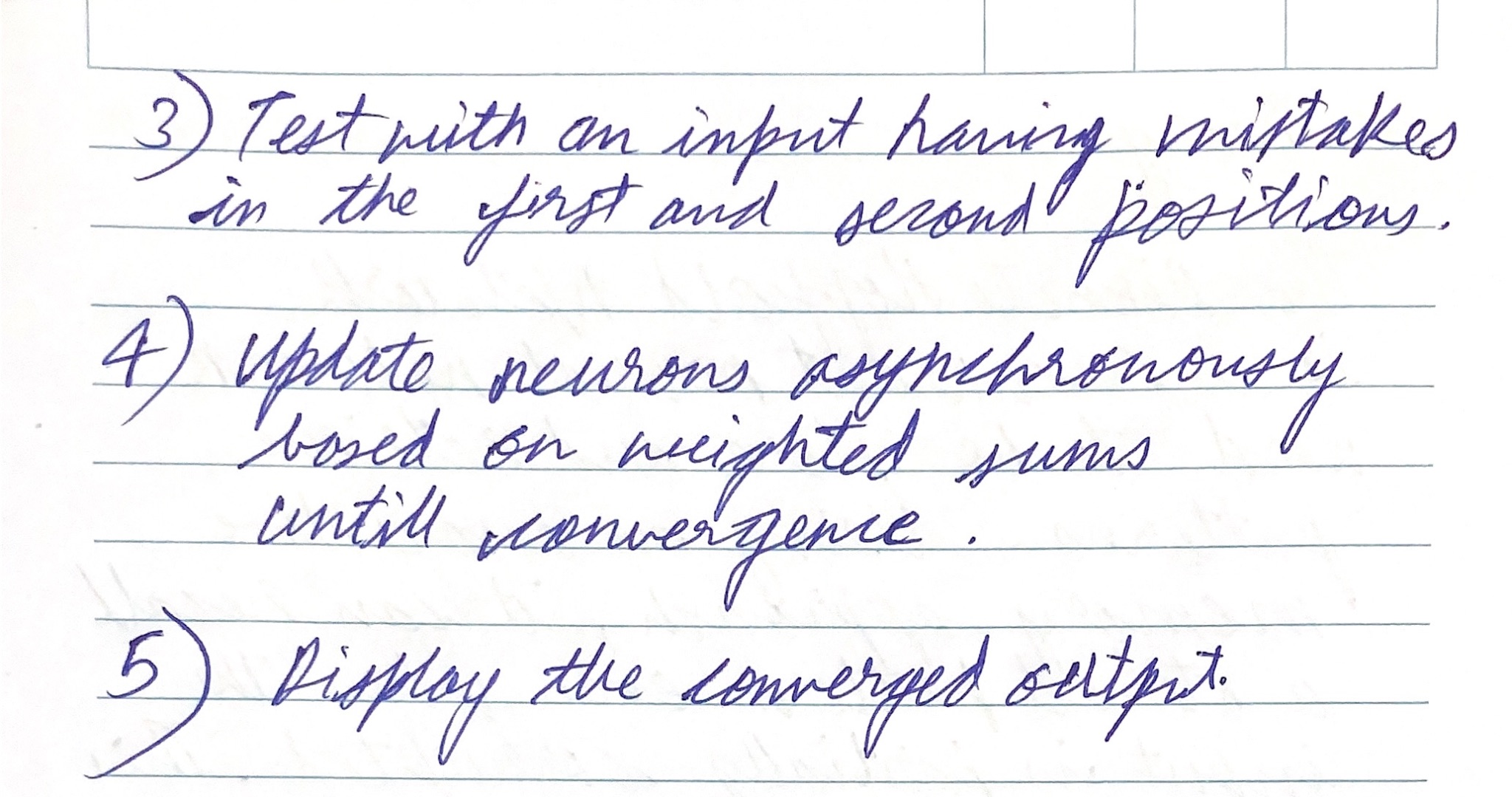
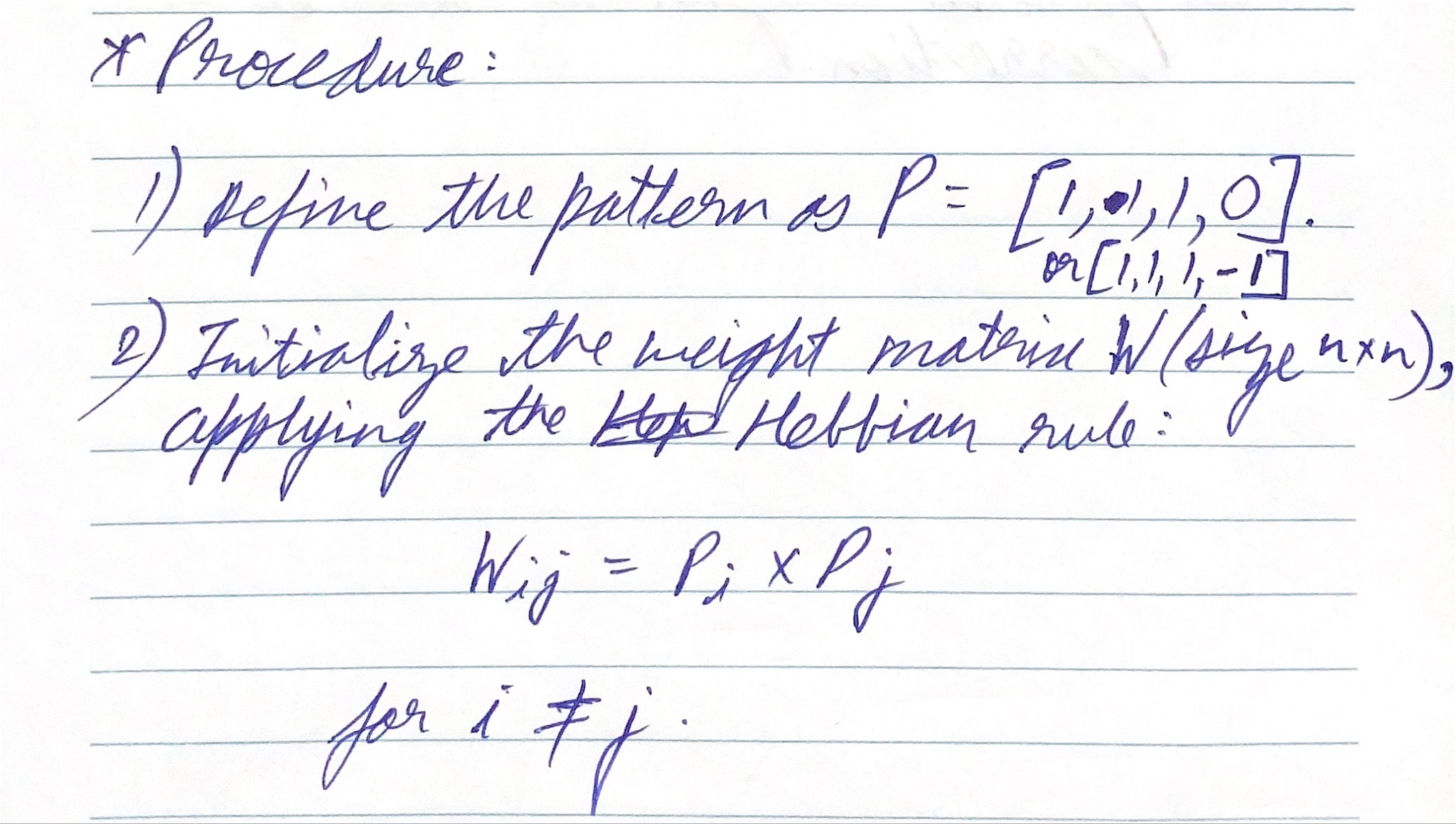
**Title:** Study of ANN Models

**Aim:** Implement a program to store a pattern (1 1 1 0). Test the network using Discrete Hopfield Net by giving the input with mistakes in First and Second position

**Objective:** Students will be able

* To understand practical aspects of Discrete Hopfield model
* To implement in solving pattern recognition problems

# 

****

**Expected Output:**

convergence has been obtained

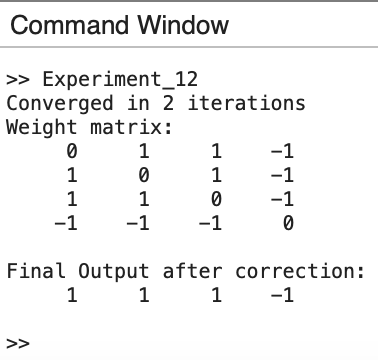
the converged output

1 1 1 - 1 (where, -1 represents 0)

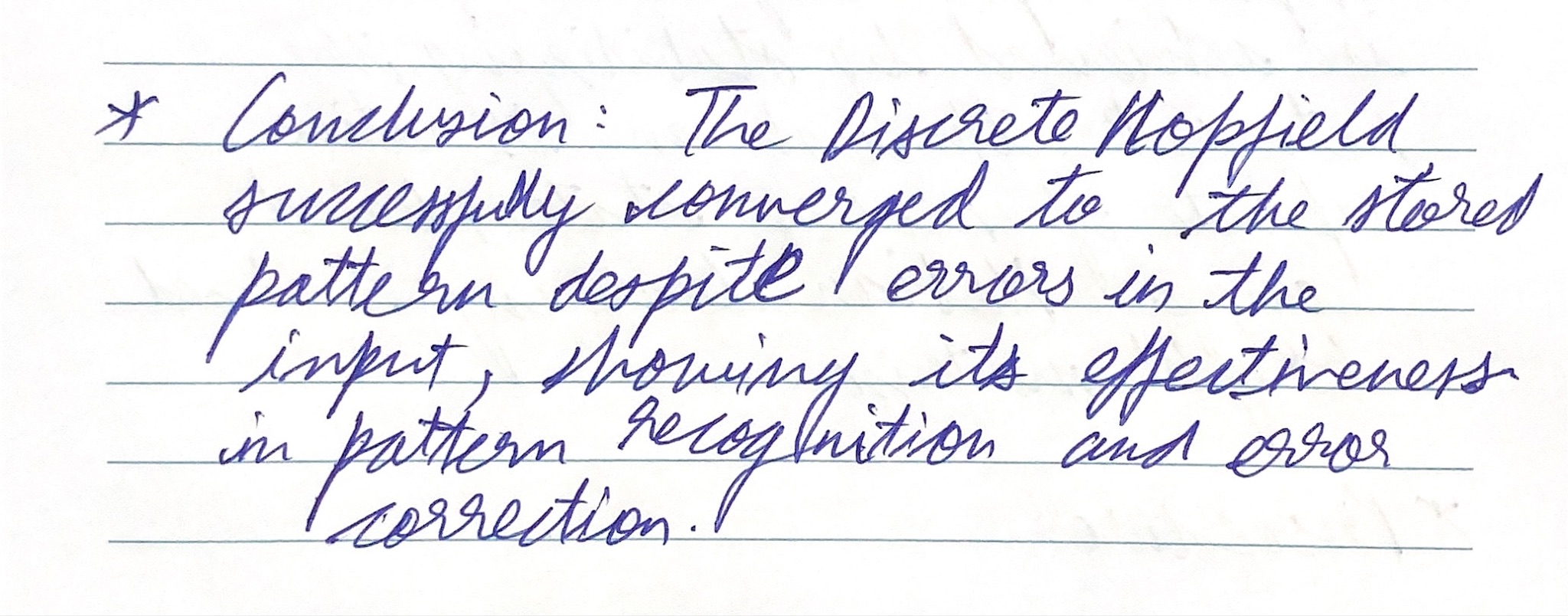
**Code:**

| % Define the pattern to be stored (P = [1 1 1 0])  P = [1 1 1 -1]; % Representing 0 as -1  % Number of neurons  n = length(P);  % Initialize weight matrix W (n x n) using Hebbian rule  W = P' \* P - eye(n); % Compute the outer product and subtract the identity matrix  % Test input with minor adjustments to encourage convergence  test\_input = [0 1 1 -1]; % Minor alteration to approach the target  % Define the asynchronous update rule for Hopfield Network  max\_iterations = 10; % Set max iterations to prevent infinite loops  S = test\_input; % Start with the test input  for iter = 1:max\_iterations  prev\_S = S; % Store the previous state  for i = 1:n  % Compute the weighted sum for neuron i  S(i) = sign(W(i, :) \* prev\_S'); % Update based on weighted sum  % Avoid any zero values by keeping previous state if result is zero  if S(i) == 0  S(i) = prev\_S(i);  end  end  % Check for convergence  if isequal(S, prev\_S)  disp(['Converged in ', num2str(iter), ' iterations']);  break;  end  end  disp('Weight matrix:')  disp(W)  % Display final converged output  disp('Final Output after correction:');  disp(S); |
| --- |

**Output:**



**Conclusion:**

****