```
Name: Aarya Admane 22630 B2

import numpy as np

# Define 5x3 matrix representations for digits 0-9
digits = {
    0: [[1, 1, 1],
        [1, 0, 1],
        [1, 0, 1],
        [1, 0, 1],
        [1, 1, 1]],
    1: [[0, 1, 0],
        [1, 1, 0],
        [0, 1, 0],
        [0, 1, 0],
        [1, 1, 1]],
    2: [[1, 1, 1],
        [0, 0, 1],
        [1, 1, 1],
        [1, 0, 0],
        [1, 1, 1]],
    3: [[1, 1, 1],
        [0, 0, 1],
        [1, 1, 1],
        [0, 0, 1],
        [1, 1, 1]],
    4: [[1, 0, 1],
        [1, 0, 1],
        [1, 1, 1],
        [0, 0, 1],
        [0, 0, 1]],
    5: [[1, 1, 1],
        [1, 0, 0],
        [1, 1, 1],
        [0, 0, 1],
        [1, 1, 1]],
    6: [[1, 1, 1],
        [1, 0, 0],
        [1, 1, 1],
        [1, 0, 1],
        [1, 1, 1]],
    7: [[1, 1, 1],
        [0, 0, 1],
        [0, 1, 0],
        [1, 0, 0],
        [1, 0, 0]],
    8: [[1, 1, 1],
        [1, 0, 1],
        [1, 1, 1],
        [1, 0, 1],
```

```
        [1, 1, 1]],
    9: [[1, 1, 1],
        [1, 0, 1],
        [1, 1, 1],
        [0, 0, 1],
        [1, 1, 1]]
 }

 # Convert dictionary to a NumPy matrix (flattened 5x3 to 1D array)
 X = np.array([np.array(digits[i]).flatten() for i in range(10)])

# Identity matrix as labels (one-hot encoded)
Y = np.eye(10)

 # Train using Hebbian Learning (Weight Matrix W)
W = np.dot(Y.T, X)

 def recognize_number(x_test):
    """Recognize a given number based on the trained network"""
    x_test = np.array(x_test).flatten()  # Flatten 5x3 to 1D
    output = np.dot(W, x_test)  # Compute weighted sum
    recognized_digit = np.argmax(output)  # Find max index
# (corresponding to digit)
    return recognized_digit

 # Test Data (Example: Recognizing digit '3')
 test_digit = [[1, 1, 1],
               [0, 0, 1],
               [1, 1, 1],
               [0, 0, 1],
               [1, 1, 1]]
 recognized = recognize_number(test_digit)
 print("Recognized Digit:", recognized)
```

```
Recognized Digit: 3
```