ASSIGNMENT 4- GROUP B

```
import numpy as np
class HopfieldNetwork:
    def __init__(self, n_neurons):
        self.n_neurons = n_neurons
        self.weights = np.zeros((n_neurons, n_neurons))
    def train(self, patterns):
        for pattern in patterns:
            self.weights += np.outer(pattern, pattern)
        self.weights /= self.n_neurons
        np.fill_diagonal(self.weights, 0)
    def predict(self, pattern, max_iterations=10):
        for _ in range(max_iterations):
            pattern = np.sign(np.dot(self.weights, pattern))
        return pattern
    def calculate_energy(self, pattern):
        return -0.5 * np.dot(pattern, np.dot(self.weights, pattern))

if __name__ == '__main__':
    patterns = np.array([
        [-1, 1, -1, -1],
        [-1, -1,- 1,- 1],
        [-1, -1, 1, -1],
        [-1, 1, -1, 1]
    ])

    n_neurons = patterns.shape[1]
    network = HopfieldNetwork(n_neurons)
    network.train(patterns)
```

```
    for pattern in patterns:
        prediction = network.predict(pattern)
        energy = network.calculate_energy(prediction)
        print('Input pattern:', pattern)
        print('Predicted pattern:', prediction)
        print('Energy:', energy)
        print('---')
```

OUTPUT:

```
Input pattern: [-1  1 -1 -1]
Predicted pattern: [-1.  1. -1. -1.]
Energy: -1.0
---
Input pattern: [-1 -1 -1 -1]
Predicted pattern: [-1. -1. -1. -1.]
Energy: -1.0
---
Input pattern: [-1 -1  1 -1]
Predicted pattern: [-1. -1.  1. -1.]
Energy: -1.0
---
Input pattern: [-1  1 -1  1]
Predicted pattern: [-1.  1. -1.  1.]
Energy: -1.0
```