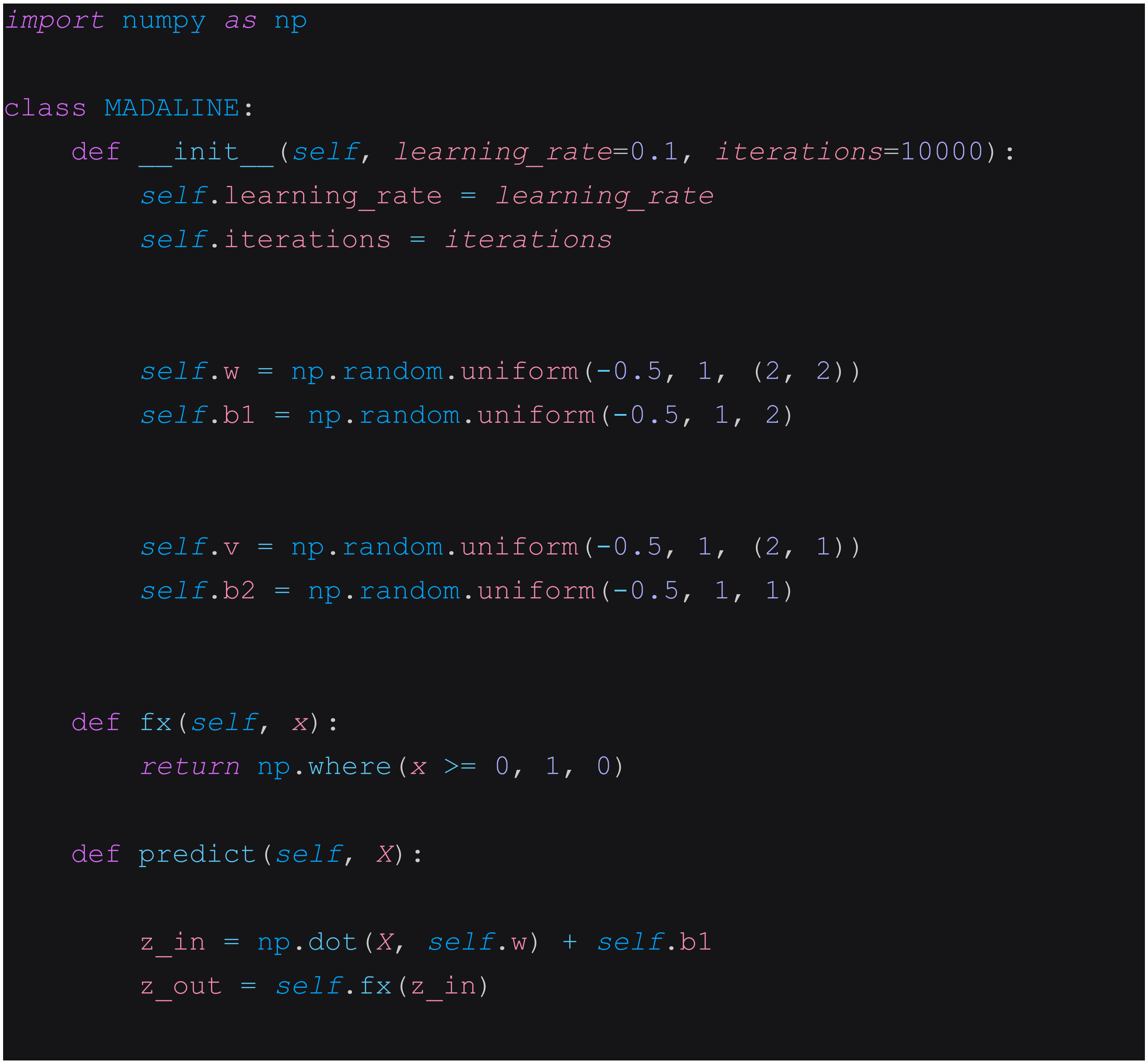
SOFT COMPUTING
ASSIGNMENT -5

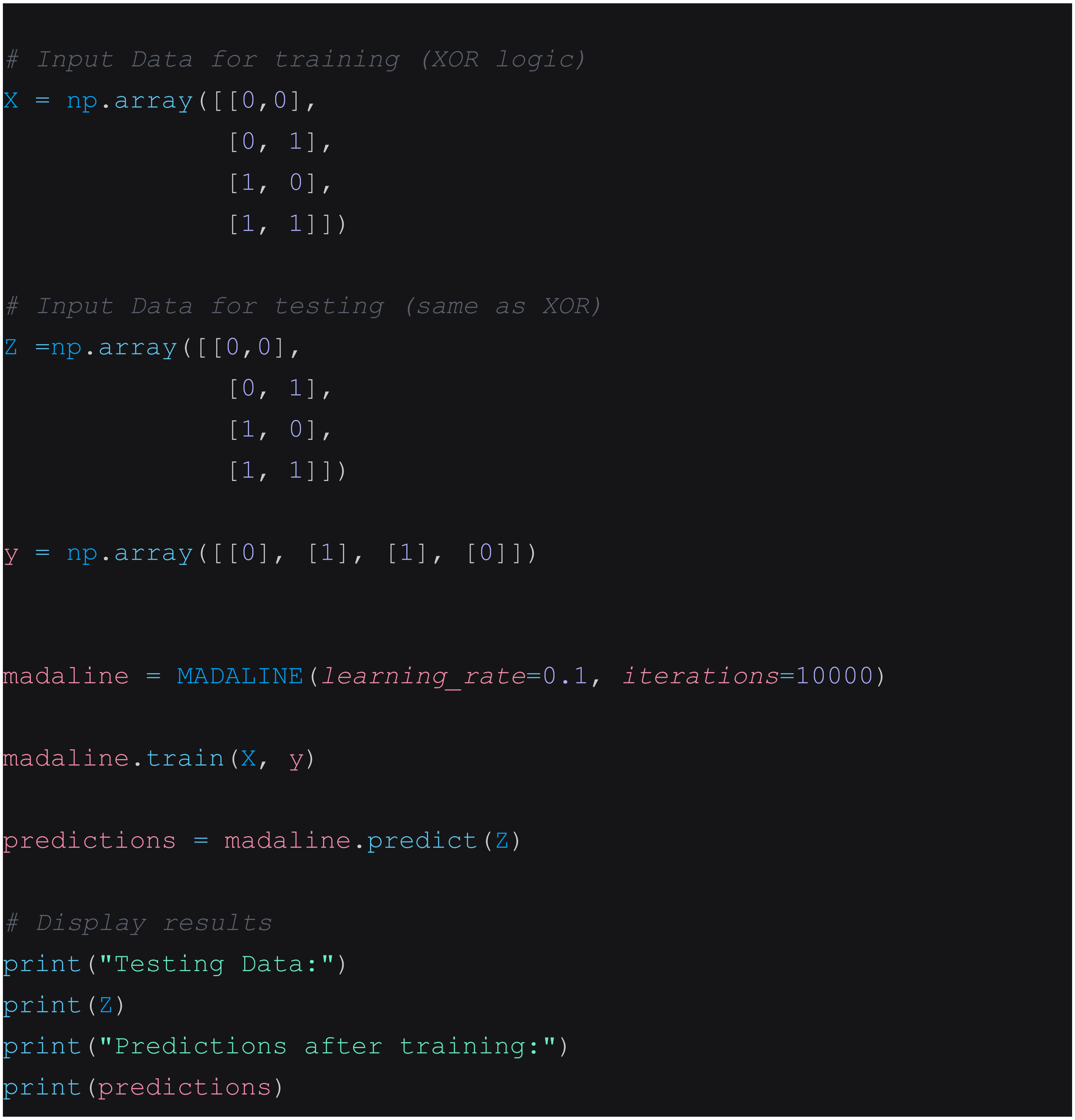
Perumalla Dharan
AP21110010201

Write a Python program to implement a MADALINE on logical XOR function. Take the binary inputs and outputs.

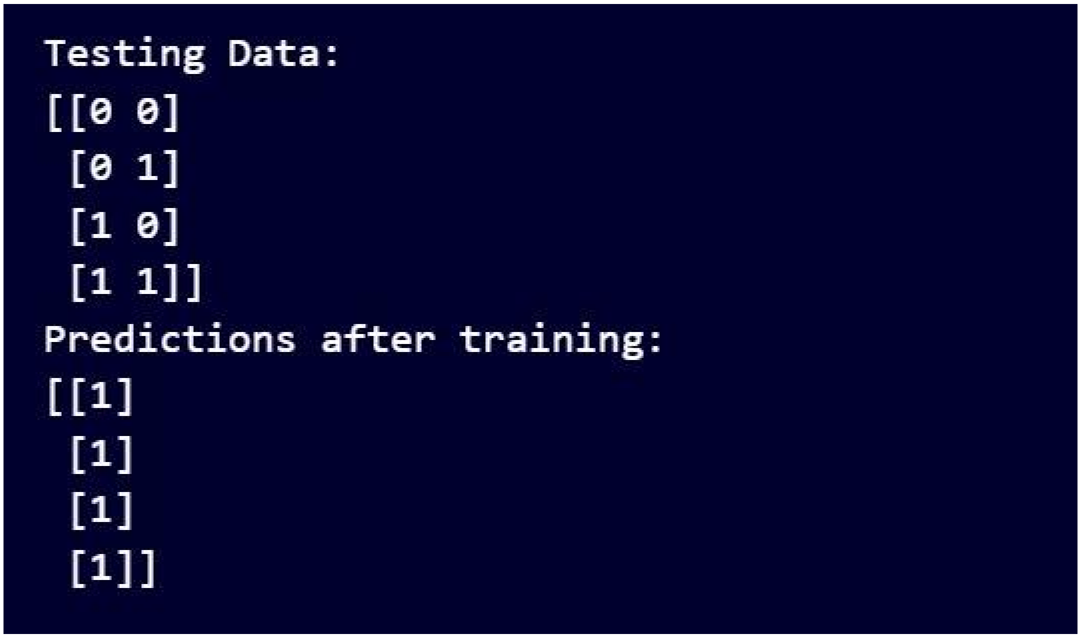
Note: Upload the source file of the Python program and a Word document file that contains the Python program along with the results.



```python
import numpy as np

class MADALINE:
    def __init__(self, learning_rate=0.1, iterations=10000):
        self.learning_rate = learning_rate
        self.iterations = iterations


        self.w = np.random.uniform(-0.5, 1, (2, 2))
        self.b1 = np.random.uniform(-0.5, 1, 2)


        self.v = np.random.uniform(-0.5, 1, (2, 1))
        self.b2 = np.random.uniform(-0.5, 1, 1)


    def fx(self, x):
        return np.where(x >= 0, 1, 0)

    def predict(self, X):

        z_in = np.dot(X, self.w) + self.b1
        z_out = self.fx(z_in)
        y_in = np.dot(z_out, self.v) + self.b2
        y_out = self.fx(y_in)

        return y_out

    def train(self, X, y):
        for epoch in range(self.iterations):
            for i in range(len(X)):

                z_in = np.dot(X[i], self.w) + self.b1
                z_out = self.fx(z_in)

                y_in = np.dot(z_out, self.v) + self.b2
                y_out = self.fx(y_in)

                if y_out != y[i]:
                    if y[i] == -1:
                        for j in range(len(z_in)):
                            if z_in[j] > 0:

                                self.w[:, j] += (self.learning_rate * (X[i]) *(y[i]-z_in[j].item()))
                                self.b1[j] += (self.learning_rate * (y[i]-z_in[j].item()))

                    elif y[i] == 1:

                                closest_to_zero_index = np.argmin(np.abs(z_in))

                                self.w[:, closest_to_zero_index] += (self.learning_rate * X[i] *(y[i]-z_in[j].item()))
                                self.b1[closest_to_zero_index] += (self.learning_rate *(y[i]-z_in[closest_to_zero_index].item()))
```



```
# Input Data for training (XOR logic)
X = np.array([[0,0],
              [0, 1],
              [1, 0],
              [1, 1]])

# Input Data for testing (same as XOR)
Z =np.array([[0,0],
              [0, 1],
              [1, 0],
              [1, 1]])

y = np.array([[0], [1], [1], [0]])


madaline = MADALINE(learning_rate=0.1, iterations=10000)

madaline.train(X, y)

predictions = madaline.predict(Z)

# Display results
print("Testing Data:")
print(Z)
print("Predictions after training:")
print(predictions)
```



```
Testing Data:
[[0 0]
 [0 1]
 [1 0]
 [1 1]]
Predictions after training:
[[1]
 [1]
 [1]
 [1]]
```