```
# Single Layer Perceptron Implementation
import numpy as np

# Step 1: Define the Activation Function
def activation_function(x):
    return np.where(x >= 0, 1, 0)

# Step 2: Define the Perceptron Training Function
def perceptron_training(X, y, learning_rate=0.1, epochs=10):
    X = np.insert(X, 0, 1, axis=1)  # Add bias term
    weights = np.zeros(X.shape[1])

    print("Initial Weights:", weights)
    print("\nTraining started...\n")

    for epoch in range(epochs):
        print(f"Epoch {epoch + 1}")
        for i in range(len(X)):
            net_input = np.dot(X[i], weights)
            prediction = activation_function(net_input)
            error = y[i] - prediction
            weights += learning_rate * error * X[i]
            print(f"Input: {X[i][1:]}, Target: {y[i]}, Prediction: {prediction}, Updated Weights: {weights}")
        print("-" * 50)

    print("\nTraining completed.")
    print("Final Weights:", weights)
    return weights

# Step 3: Define the Prediction Function
def perceptron_predict(X, weights):
```

```
    X = np.insert(X, 0, 1, axis=1)
    output = activation_function(np.dot(X, weights))
    return output

# Step 4: Main Program
if __name__ == "__main__":
    # Input dataset for AND logic gate
    X = np.array([
        [0, 0],
        [0, 1],
        [1, 0],
        [1, 1]
    ])

    # Target output for AND gate
    y = np.array([0, 0, 0, 1])

    # Training
    learning_rate = 0.1
    epochs = 10
    weights = perceptron_training(X, y, learning_rate, epochs)

    # Prediction
    print("\nTesting perceptron after training:")
    predictions = perceptron_predict(X, weights)
    for i in range(len(X)):
        print(f"Input: {X[i]} → Predicted Output: {predictions[i]}")
```

```
Initial Weights: [0. 0. 0.]

Training started...

Epoch 1
Input: [0 0], Target: 0, Prediction: 1, Updated Weights: [-0.1  0.   0. ]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.1  0.   0. ]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.1  0.   0. ]
Input: [1 1], Target: 1, Prediction: 0, Updated Weights: [0.  0.1 0.1]
--------------------------------------------------
Epoch 2
Input: [0 0], Target: 0, Prediction: 1, Updated Weights: [-0.1  0.1  0.1]
Input: [0 1], Target: 0, Prediction: 1, Updated Weights: [-0.2  0.1  0. ]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.1  0. ]
Input: [1 1], Target: 1, Prediction: 0, Updated Weights: [-0.1  0.2  0.1]
--------------------------------------------------
Epoch 3
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.1  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 1, Updated Weights: [-0.2  0.2  0. ]
Input: [1 0], Target: 0, Prediction: 1, Updated Weights: [-0.3  0.1  0. ]
Input: [1 1], Target: 1, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
--------------------------------------------------
Epoch 4
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 1], Target: 1, Prediction: 1, Updated Weights: [-0.2  0.2  0.1]
```

```
Epoch 5
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 1], Target: 1, Prediction: 1, Updated Weights: [-0.2  0.2  0.1]
--------------------------------------------------
Epoch 6
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 1], Target: 1, Prediction: 1, Updated Weights: [-0.2  0.2  0.1]
--------------------------------------------------
Epoch 7
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 1], Target: 1, Prediction: 1, Updated Weights: [-0.2  0.2  0.1]
--------------------------------------------------
Epoch 8
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 1], Target: 1, Prediction: 1, Updated Weights: [-0.2  0.2  0.1]
--------------------------------------------------
Epoch 9
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 1], Target: 1, Prediction: 1, Updated Weights: [-0.2  0.2  0.1]
--------------------------------------------------
Epoch 10
Input: [0 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [0 1], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 0], Target: 0, Prediction: 0, Updated Weights: [-0.2  0.2  0.1]
Input: [1 1], Target: 1, Prediction: 1, Updated Weights: [-0.2  0.2  0.1]
--------------------------------------------------

Training completed.
Final Weights: [-0.2  0.2  0.1]

Testing perceptron after training:
Input: [0 0] → Predicted Output: 0
Input: [0 1] → Predicted Output: 0
Input: [1 0] → Predicted Output: 0
Input: [1 1] → Predicted Output: 1
```