```
import numpy as np

# Sigmoid activation and its derivative
def sigmoid(x):
    return 1 / (1 + np.exp(-x))

def sigmoid_derivative(x):
    return x * (1 - x)

# XOR input and output
# Inputs: 4 combinations of 2 binary inputs
X = np.array([
    [0, 0],
    [0, 1],
    [1, 0],
    [1, 1]
])

# Output: XOR results
# 0 ⊕ 0 = 0
# 0 ⊕ 1 = 1
# 1 ⊕ 0 = 1
# 1 ⊕ 1 = 0
y = np.array([[0], [1], [1], [0]])

# Seed for reproducibility
np.random.seed(42)

# Network architecture
input_layer_neurons = 2
hidden_layer_neurons = 2
output_neurons = 1

# Random weights and bias initialization
wh = np.random.uniform(size=(input_layer_neurons,
hidden_layer_neurons))
bh = np.random.uniform(size=(1, hidden_layer_neurons))
wo = np.random.uniform(size=(hidden_layer_neurons, output_neurons))
bo = np.random.uniform(size=(1, output_neurons))

# Training algorithm
epochs = 100
lr = 0.1  # learning rate

for epoch in range(epochs):
    # Forward pass
    hidden_input = np.dot(X, wh) + bh
    hidden_output = sigmoid(hidden_input)

    final_input = np.dot(hidden_output, wo) + bo
    final_output = sigmoid(final_input)
```

```
    # Error computation
    error = y - final_output

    # Backpropagation
    d_output = error * sigmoid_derivative(final_output)

    error_hidden = d_output.dot(wo.T)
    d_hidden = error_hidden * sigmoid_derivative(hidden_output)

    # Update weights and biases
    wo += hidden_output.T.dot(d_output) * lr
    bo += np.sum(d_output, axis=0, keepdims=True) * lr
    wh += X.T.dot(d_hidden) * lr
    bh += np.sum(d_hidden, axis=0, keepdims=True) * lr

    if epoch % 1 == 0:
        loss = np.mean(np.square(error))
        print(f"Epoch {epoch}, Loss: {loss}")
```

```
Epoch 0, Loss: 0.0005373448621307079
Epoch 1, Loss: 0.0005373228463382441
Epoch 2, Loss: 0.0005373008323023121
Epoch 3, Loss: 0.0005372788200227128
Epoch 4, Loss: 0.0005372568094992278
Epoch 5, Loss: 0.0005372348007316567
Epoch 6, Loss: 0.0005372127937197928
Epoch 7, Loss: 0.0005371907884634243
Epoch 8, Loss: 0.000537168784962347
Epoch 9, Loss: 0.0005371467832163535
Epoch 10, Loss: 0.0005371247832252337
Epoch 11, Loss: 0.0005371027849887817
Epoch 12, Loss: 0.0005370807885067904
Epoch 13, Loss: 0.0005370587937790558
Epoch 14, Loss: 0.0005370368008053618
Epoch 15, Loss: 0.0005370148095855109
Epoch 16, Loss: 0.0005369928201192932
Epoch 17, Loss: 0.0005369708324064992
Epoch 18, Loss: 0.0005369488464469246
Epoch 19, Loss: 0.0005369268622403606
Epoch 20, Loss: 0.0005369048797865994
Epoch 21, Loss: 0.0005368828990854386
Epoch 22, Loss: 0.0005368609201366675
Epoch 23, Loss: 0.0005368389429400795
Epoch 24, Loss: 0.0005368169674954673
Epoch 25, Loss: 0.0005367949938026253
Epoch 26, Loss: 0.0005367730218613482
Epoch 27, Loss: 0.0005367510516714286
Epoch 28, Loss: 0.0005367290832326563
Epoch 29, Loss: 0.0005367071165448274
```

```
Epoch 30, Loss: 0.0005366851516077342
Epoch 31, Loss: 0.0005366631884211749
Epoch 32, Loss: 0.0005366412269849353
Epoch 33, Loss: 0.0005366192672988132
Epoch 34, Loss: 0.0005365973093626013
Epoch 35, Loss: 0.0005365753531760939
Epoch 36, Loss: 0.0005365533987390844
Epoch 37, Loss: 0.0005365314460513632
Epoch 38, Loss: 0.0005365094951127281
Epoch 39, Loss: 0.000536487545922972
Epoch 40, Loss: 0.0005364655984818886
Epoch 41, Loss: 0.0005364436527892672
Epoch 42, Loss: 0.0005364217088449066
Epoch 43, Loss: 0.0005363997666485992
Epoch 44, Loss: 0.0005363778262001395
Epoch 45, Loss: 0.0005363558874993209
Epoch 46, Loss: 0.000536333950545937
Epoch 47, Loss: 0.0005363120153397782
Epoch 48, Loss: 0.0005362900818806473
Epoch 49, Loss: 0.0005362681501683308
Epoch 50, Loss: 0.0005362462202026241
Epoch 51, Loss: 0.0005362242919833224
Epoch 52, Loss: 0.0005362023655102193
Epoch 53, Loss: 0.0005361804407831085
Epoch 54, Loss: 0.0005361585178017851
Epoch 55, Loss: 0.000536136596566041
Epoch 56, Loss: 0.0005361146770756733
Epoch 57, Loss: 0.0005360927593304744
Epoch 58, Loss: 0.0005360708433302401
Epoch 59, Loss: 0.0005360489290747628
Epoch 60, Loss: 0.0005360270165638405
Epoch 61, Loss: 0.0005360051057972613
Epoch 62, Loss: 0.0005359831967748255
Epoch 63, Loss: 0.0005359612894963249
Epoch 64, Loss: 0.0005359393839615511
Epoch 65, Loss: 0.0005359174801703049
Epoch 66, Loss: 0.0005358955781223777
Epoch 67, Loss: 0.0005358736778175625
Epoch 68, Loss: 0.0005358517792556583
Epoch 69, Loss: 0.0005358298824364549
Epoch 70, Loss: 0.0005358079873597474
Epoch 71, Loss: 0.0005357860940253335
Epoch 72, Loss: 0.0005357642024330051
Epoch 73, Loss: 0.0005357423125825618
Epoch 74, Loss: 0.0005357204244737919
Epoch 75, Loss: 0.0005356985381064947
Epoch 76, Loss: 0.0005356766534804582
Epoch 77, Loss: 0.0005356547705954848
Epoch 78, Loss: 0.0005356328894513722
```

```
Epoch 79, Loss: 0.0005356110100479046
Epoch 80, Loss: 0.0005355891323848842
Epoch 81, Loss: 0.000535567256462107
Epoch 82, Loss: 0.0005355453822793621
Epoch 83, Loss: 0.0005355235098364487
Epoch 84, Loss: 0.0005355016391331624
Epoch 85, Loss: 0.0005354797701692947
Epoch 86, Loss: 0.0005354579029446447
Epoch 87, Loss: 0.000535436037459005
Epoch 88, Loss: 0.0005354141737121702
Epoch 89, Loss: 0.0005353923117039393
Epoch 90, Loss: 0.0005353704514341059
Epoch 91, Loss: 0.0005353485929024643
Epoch 92, Loss: 0.0005353267361088088
Epoch 93, Loss: 0.0005353048810529371
Epoch 94, Loss: 0.0005352830277346416
Epoch 95, Loss: 0.0005352611761537256
Epoch 96, Loss: 0.000535239326309974
Epoch 97, Loss: 0.0005352174782031907
Epoch 98, Loss: 0.0005351956318331675
Epoch 99, Loss: 0.0005351737871996966

# Final prediction after training
print("Final Output after Training:")
print(np.round(final_output))

Final Output after Training:
[[0.]
 [1.]
 [1.]
 [0.]]
```