In [1]:
```
import numpy as np
# Define the sigmoid activation function
def sigmoid(x):
    return 1 / (1 + np.exp(-x))
# Define the derivative of the sigmoid function
def sigmoid_derivative(x):
    return x * (1 - x)
```

In [2]:
```
# Creating a dataset
# Each row represents a time step, and each column represents a feature
X = np.array([[0, 0, 1],
 [0, 1, 1],
 [1, 0, 1],
 [1, 1, 1]])
# Target labels
y = np.array([[1],
 [0],
 [1],
 [0]])
```

In [3]:
```
# Set the seed for reproducibility
np.random.seed(1)
```

In [4]:
```
# Initialize hyperparameters
epochs = 10000
learning_rate = 0.1
# Initialize weights randomly
input_size = 3
hidden_size = 4
output_size = 1
```

In [5]:
```
weights_input_hidden = np.random.uniform(-1, 1, (input_size, hidden_size))
weights_hidden_output = np.random.uniform(-1, 1, (hidden_size, output_size)
```

In [6]:
```
# Training the RNN
for epoch in range(epochs):

    # Forward propagation
    hidden_layer_input = np.dot(X, weights_input_hidden)
    hidden_layer_output = sigmoid(hidden_layer_input)
    output_layer_input = np.dot(hidden_layer_output, weights_hidden_output)
    output_layer_output = sigmoid(output_layer_input)

    # Calculate the loss
    loss = y - output_layer_output

    # Backpropagation
    d_output = loss * sigmoid_derivative(output_layer_output)
    error_hidden_layer = d_output.dot(weights_hidden_output.T)
    d_hidden_layer = error_hidden_layer * sigmoid_derivative(hidden_layer_o

    # Update weights
    weights_hidden_output += hidden_layer_output.T.dot(d_output) * learning
    weights_input_hidden += X.T.dot(d_hidden_layer) * learning_rate
    if epoch % 1000 == 0:
        print(f"Epoch {epoch}: Error {np.mean(np.abs(loss))}")
```

```
Epoch 0: Error 0.5040237243821001
Epoch 1000: Error 0.09112543351136504
Epoch 2000: Error 0.05085783957414422
Epoch 3000: Error 0.03785155334720379
Epoch 4000: Error 0.03110704285474844
Epoch 5000: Error 0.026870428063396862
Epoch 6000: Error 0.023914799439357523
Epoch 7000: Error 0.02171141194298185
Epoch 8000: Error 0.019991895757390896
Epoch 9000: Error 0.018604307728137825
```

In [7]:
```
# Test the trained RNN
test_input = np.array([[1, 0, 0]]) # Test input
hidden_layer_input = np.dot(test_input, weights_input_hidden)
hidden_layer_output = sigmoid(hidden_layer_input)
output_layer_input = np.dot(hidden_layer_output, weights_hidden_output)
output_layer_output = sigmoid(output_layer_input)
print(f"Test output: {output_layer_output}")
```

```
Test output: [[0.81500116]]
```