```
clear;
x = [0 0 ; 0 1 ; 1 0 ; 1 1];

t = zeros(4,16);
for i = 1:16
    t(:,i) = reshape(str2num(transpose(dec2bin(i-1,4))) ,4,1);
end
t(t==0) = -1;

weightVectorVariance = sqrt(1/2);
weightVectorMean = 0;
learningRate = 0.05;
thresholdValue = 0;
trainingEpoch = 20;
trainingResult = zeros(4,1);
resultNumber = zeros(2,1);

weightVector = weightVectorVariance.*randn(2,1) + weightVectorMean;

for i = 1:16
    for j = 1:trainingEpoch
        for k = 1:4
            output = O(x(k,:), weightVector, thresholdValue);
            deltaWeightVector = transpose(learningRate * (t(k,i) - output ) *
x(k,:));
            deltaThresholdValue = -learningRate * (t(k,i) - output );
            weightVector = weightVector + deltaWeightVector;
            thresholdValue = thresholdValue + deltaThresholdValue;
        end
    end
    for l = 1:4
        trainingResult(l) = O(x(l,:), weightVector, thresholdValue);
    end
    if trainingResult == t(:,i)
        resultNumber(1) = resultNumber(1) + 1;
    else
        resultNumber(2) = resultNumber(2) + 1;
    end
end
disp(resultNumber(1));
```

```
clear;
x = zeros(8,3);

for i = 1:8
    x(i,:) = reshape(str2num(transpose(dec2bin(i-1,3))) ,1,3);
end

t = zeros(8,256);
```

```
for i = 1:256
    t(:,i) = reshape(str2num(transpose(dec2bin(i-1,8))) ,8,1);
end
t(t==0) = -1;

weightVectorVariance = sqrt(1/2);
weightVectorMean = 0;
learningRate = 0.05;
thresholdValue = 0;
trainingEpoch = 20;
trainingResult = zeros(8,1);
resultNumber = zeros(2,1);

weightVector = weightVectorVariance.*randn(3,1) + weightVectorMean;

for i = 1:256
    for j = 1:trainingEpoch
        for k = 1:8
            output = O(x(k,:), weightVector, thresholdValue);
            deltaWeightVector = transpose(learningRate * (t(k,i) - output ) *
x(k,:));
            deltaThresholdValue = -learningRate * (t(k,i) - output );
            weightVector = weightVector + deltaWeightVector;
            thresholdValue = thresholdValue + deltaThresholdValue;
        end
    end
    for l = 1:8
        trainingResult(l) = O(x(l,:), weightVector, thresholdValue);
    end
    if trainingResult == t(:,i)
        resultNumber(1) = resultNumber(1) + 1;
    else
        resultNumber(2) = resultNumber(2) + 1;
    end
end
disp(resultNumber(1));
```

```
clear;
x = zeros(16,4);

for i = 1:16
    x(i,:) = reshape(str2num(transpose(dec2bin(i-1,4))) ,1,4);
end

t = zeros(16,10000);
tStore = randperm(65536, 10000);
for i = 1:10000
    t(:,i) = reshape(str2num(transpose(dec2bin(tStore(i)-1,16))) ,16,1);
end
```

```
t(t==0) = -1;

weightVectorVariance = sqrt(1/2);
weightVectorMean = 0;
learningRate = 0.05;
thresholdValue = 0;
trainingEpoch = 20;
trainingResult = zeros(16,1);
resultNumber = zeros(2,1);

weightVector = weightVectorVariance.*randn(4,1) + weightVectorMean;

for i = 1:10000
    for j = 1:trainingEpoch
        for k = 1:16
            output = O(x(k,:), weightVector, thresholdValue);
            deltaWeightVector = transpose(learningRate * (t(k,i) - output ) *
x(k,:));
            deltaThresholdValue = -learningRate * (t(k,i) - output );
            weightVector = weightVector + deltaWeightVector;
            thresholdValue = thresholdValue + deltaThresholdValue;
        end
    end
    for l = 1:16
        trainingResult(l) = O(x(l,:), weightVector, thresholdValue);
    end
    if trainingResult == t(:,i)
        resultNumber(1) = resultNumber(1) + 1;
    else
        resultNumber(2) = resultNumber(2) + 1;
    end
end
disp(resultNumber(1));
```

```
clear;
x = zeros(32,5);

for i = 1:32
    x(i,:) = reshape(str2num(transpose(dec2bin(i-1,5))) ,1,5);
end

t = zeros(32,10000);
tStore = randperm(4294967296, 10000);
for i = 1:10000
    t(:,i) = reshape(str2num(transpose(dec2bin(tStore(i)-1,32))) ,32,1);
end
t(t==0) = -1;

weightVectorVariance = sqrt(1/2);
```

```
weightVectorMean = 0;
learningRate = 0.05;
thresholdValue = 0;
trainingEpoch = 20;
trainingResult = zeros(32,1);
resultNumber = zeros(2,1);

weightVector = weightVectorVariance.*randn(5,1) + weightVectorMean;

for i = 1:10000
    for j = 1:trainingEpoch
        for k = 1:32
            output = O(x(k,:), weightVector, thresholdValue);
            deltaWeightVector = transpose(learningRate * (t(k,i) - output ) *
x(k,:));
            deltaThresholdValue = -learningRate * (t(k,i) - output );
            weightVector = weightVector + deltaWeightVector;
            thresholdValue = thresholdValue + deltaThresholdValue;
        end
    end
    for l = 1:32
        trainingResult(l) = O(x(l,:), weightVector, thresholdValue);
    end
    if trainingResult == t(:,i)
        resultNumber(1) = resultNumber(1) + 1;
    else
        resultNumber(2) = resultNumber(2) + 1;
    end
end
disp(resultNumber(1));
```

```
function output = O(x, weightVector, thresholdValue)

if (x * weightVector - thresholdValue) == 0
    output = 1;
else
    output = sign(x * weightVector - thresholdValue);
end

end
```