```
clear; clc;
disp("ADALINE NETWORK FOR OR
FUNCTION BIPOLAR INPUTS AND TARGET");

% Input vectors
i1 = [1 1 -1 -1];
i2 = [1 -1 1 -1];
i3 = [1 1 1 1]; % Bias input

% Target vector
t = [1 1 1 -1];

% Initial network weights and bias
w1 = 0.1;
w2 = 0.1;
b = 0.1;

% Initialize learning rate and error convergence
alpha = 0.1;
e = 0;
epoch = 0;

% Start training
while (e < 0.5)
    epoch = epoch + 1;
    e = 0;

    for j = 1:4
        % Compute net input and final output
        finaly(j) = w1 * i1(j) + w2 * i2(j) + b;
        nt = [finaly(j) t(j)];

        % Update weights and bias
        delwl = alpha * (t(j) - finaly(j)) * i1(j);
        delw2 = alpha * (t(j) - finaly(j)) * i2(j);
        delb = alpha * (t(j) - finaly(j)) * i3(j);

        % Update weights
        w1 = w1 + delwl;
        w2 = w2 + delw2;
        b = b + delb;

        % Print output
        out = [i1(j) i2(j) i3(j) nt delwl delw2 delb w1 w2 b];
        disp(out);
    end

    % Compute error
    for k = 1:4
        finaly(k) = w1 * i1(k) + w2 * i2(k) + b;
        e = e + (t(k) - finaly(k)) ^ 2;
    end
end

% Final output after training
for i = 1:4
    nety(i) = w1 * i1(i) + w2 * i2(i) + b;
    e = e + (t(i) - nety(i)) ^ 2; end
```