```
function x_star = prob2_sd(f, x0, thresh, itmax)
    if nargin < 4
        itmax = 1000;
    end
    if nargin < 3
        thresh = 1e-8;
    end

    x = transpose(symvar(f));
    g = gradient(f);


    compute_g = @(x_val) double(subs(g, x, x_val));

    x_val = x0;
    numit = 1;

    syms alph

    loop_values = {};


    while prob2_thresh(f, x_val, thresh) && numit <= itmax

        direction = -compute_g(x_val);

        % line search
        phi = subs(f, x, x_val + alph * direction);
        gphi = gradient(phi);
        compute_phi = @(alpha_val) double(subs(phi, alph, alpha_val));
        compute_gphi = @(alpha_val) double(subs(gphi, alph, alpha_val));

        mu = 1e-4; alpha_val = 1; eta = 2;

        while compute_phi(alpha_val) > compute_phi(0) + mu * compute_gphi(0) * alpha_va
            alpha_val = alpha_val / eta;
        end
        % line search finished, optimal alpha found

        x_val = x_val + alpha_val * direction;

        % assign to records
        loop_values{numit, 1} = numit;
        loop_values{numit, 2} = x_val;
        loop_values{numit, 3} = direction;
        loop_values{numit, 4} = alpha_val;
        numit = numit + 1;

    end


    if numit > 15
        loop_values = [loop_values(1:10, :); loop_values(numit - 5:numit, :)];
```

```
    end

    [num_loop_values, ~] = size(loop_values);

    for i = 1:num_loop_values
        fprintf("Iteration %d\n", loop_values{i, 1})
        fprintf("Direction:")
        disp(loop_values{i, 3})
        fprintf("alpha")
        disp(loop_values{i, 4})
        fprintf("x_{k+1}")
        disp(loop_values{i, 2})
    end

    x_star = x_val;

end
```