```matlab
function [y] = g(x)
% the first half of f, used for splitting f in the product rule
y = sin(x)+1;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [y] = h(x)
% the second half of f, used for splitting f in the product rule
result1 = 10*x - x.^2 - 25;
y = exp(result1);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [y] = f(x)
y = g(x).*h(x);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [y] = gd1(x)
% the first derivative of g
y = cos(x);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [y] = hd1(x)
% the first derivative of h
result1 = 10-2.*x;
y = result1.*h(x);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [y] = fd1(x)
% the first derivative of f in terms of g, g', h, and h'
result1 = gd1(x).*h(x);
result2 = g(x).*hd1(x);
y = result1 + result2;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [y] = fd2(x)
% the second derivative of h in terms of g and h
% f'' = -sin(x)*h(x) + cos(x)*h'(x) + (10-2x)*f'(x) - 2f(x)
result1 = -sin(x).*h(x);
result2 = cos(x).*hd1(x);
result3 = (10-2*x).*fd1(x);
y = result1 + result2 + result3 - 2*f(x);
end
```

```

function [extrema, results] = ...
    newtonsMethod(f,fd1,x0,maxIterations,tolerance)
%This function takes a function f, its derivative fd1, an ...
    initial guess x0,
% and optional parameters for max number of iterations and ...
    tolerance,
% then uses Newton's method

% initialize the results list
results = [0 0];
% currentIn = x_n
currentIn = x0;
extrema = x0;

% if maximum number of iterations is not provided, default to 20
if ¬exist('maxIterations','var')
    maxIterations = 20;
end

% if tolerance not provided, default to 10^-6
if ¬exist('tolerance','var')
    tolerance = 10^-6;
end

% iteratively apply Newton's method at most maxIterations times
% the 0th application of Newton's method is index 1
for i = 1:maxIterations+1
    % add the pair (x_n, f(x_n)) to the results list
    results(i,1) = currentIn;
    results(i,2) = f(currentIn);
    % extrema stores the best guess so far
    extrema = currentIn;
    % if the newest iteration made a sufficiently small change ...
        in x_n,
    % return the current results
    if i > 1 && abs(results(i,2) - results(i-1,2)) < tolerance
        break;
    end
    % x_n+1 = x_n - f(x_n)/f'(x_n)
    currentIn = currentIn - f(currentIn)/fd1(currentIn);
end

end
```

```
format long
% apply Newton's Method on f' and f'' to find where f'=0
[maximizer, results] = newtonsMethod(@fd1,@fd2,5.8,20,10^-6);
% results are in the form (x_n, f'(x_n)) starting at n=0
results
% print x* and f(x*)
[maximizer, f(maximizer)]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Output:

>> homework3

results =

   5.800000000000000   0.015229789194561
   5.813958283700808   0.000072431508262
   5.814025326199298   0.000000001885060
   5.814025327944197   0.000000000000000


ans =

   5.814025327944197   0.282417839514858
```