```
function x_return = newton_raphson(func,dfunc,guess,disp_iterations,tol)
%NEWTON_RAPHSON Single variable function newton-raphson solver
%   Inputs are:
%   func    :a function handle function of x
%   dfunc   :a function handle derivative of function of x
%   guess   :a numeric array of Mx1 initial guess for function
%   tol     :an optional scalar solver tolerance
%
%   Output is:
%   x      :a numeric array of Mx1 root of function for each guess

    arguments
        func
        dfunc
        guess (:,1) {mustBeNumeric, mustBeReal}
        disp_iterations {mustBeNumericOrLogical} = false;
        tol {mustBeNumeric, mustBeReal, mustBePositive} = 1e-8
    end

    x_return = zeros(1,length(guess));

%   Loop through each guess
    for i = 1:length(guess)
        x = guess(i);
        err = inf;
        count = 1;

        if disp_iterations
            fprintf('Initial Guess = %0.1f\n',x)
        end

%       Find values of x that satisfy tolerance
        while err > tol
            xnew = x - (func(x)./dfunc(x));
            err = abs(xnew-x);
            x = xnew;
            if disp_iterations
                fprintf('Iteration %d: x = %0.4f\n',count,x)
            end
            count = count + 1;
        end
        if disp_iterations && i ~= length(guess)
            disp('---')
        end
        x_return(i) = x;
    end
end
```