```
% falsepos_falling_mass

%{
This program uses the false position root finding method to
solve for the mass of a falling object with air resistance.
%}

% Author: Calvin Sprouse
% Date: 2024 February 05

% Version 2: Adapted to use false position method

% workspace init
clear;
close all;

% define constants
% acceleration of gravity [m/s^2]
g = 9.8;
% drag coefficient [kg/m]
cd = 0.25;
% velocity at t=4s [m/s]
v = 36;
% time [s]
t = 4;

% define the drag function
func = @(x) sqrt(g.*x./cd) .* tanh( sqrt(g.*cd./x) .* t ) - v;

% plot the function to make sure there is a root
% plt = fplot(func, [0 500], "-k");
% plt.Parent.FontWeight = "normal";
% plt.Parent.FontName = "Times New Roman";
% plt.Parent.FontSize = 16;
% ylim([-10, 10]);
% yline(0, "--k");
% xlabel("Mass [kg]");
% ylabel("Drag Force [N]");
% title("Drag force for an object with known velocity", FontWeight="normal");
% exportgraphics(gcf, "root_plot.pdf");

% define the questioning strings
xl_query = 'Enter the lower bound for mass in kg? ';
xu_query = 'Enter the upper bound for mass in kg? ';

% ask the user for the initial guess
xl = input(xl_query);
xu = input(xu_query);

% evaluate at the initial guesses to make sure they surround the root
while func(xl)*func(xu) > 0
    disp('Supplied bounds do not surround the root.')
    xl = input(xl_query);
    xu = input(xu_query);
end

% define initial values for while loop
approxerr = 1;
count = 0;
oldxr = 0;

% define stop conditions
err_thresh = 1e-6;
iter_max = 1e3;

% iterate with bisection method until error is less than 0.05 percent
while (approxerr > err_thresh) && (count < iter_max)
    % calculate xr based on straight line slope
    xr = (func(xl)*xu - func(xu)*xl) / (func(xl) - func(xu));

    % figure out what side of the midpoint the root is on
    if func(xr)*func(xu) < 0
        % replace the lower bound with the midpoint
        xl = xr;
    else
        % replace the upper bound with the midpoint
        xu = xr;
    end

    % calculate the approximate error
    approxerr = abs((xr - oldxr) / xr);

    % store old guess for root
    oldxr = xr;

    % increase count value
    count = count + 1;
end

% output the results
out_str = 'The root is %5.2f kg. It was found after %i iterations. The approximate error is %6.4f%%.\n';
fprintf(out_str, xr, count, approxerr*100);

% use built in fuction (fzero) to find the root
fprintf('The root found with fzero is %5.2f. \n', fzero(func, xl));

```