```matlab
%% CHEN 320 - 202 Homework 2
% Problem: 1
% Author: Nathaniel Thomas
% Date: 09/16/2022
% Due:  09/18/2022

%% Problem 1(a):
% Reaction: 2CO + + O_2 -> 2CO_2

T = 2000;   % Temperature, K
P = 1;      % Pressure, atm
k = 62.4E6; % Equilibrium constant, atm
n_CO = 2;   % Basis of CO, gmoles
n_O2 = 1;   % Basis of O_2, gmoles
n_N2 = 3.76;% Basis of N_2, gmoles

syms x

% Function handle for final expression
k_func = ((x.^2).*(6.76 - 0.5.*x)) - (k.*(1 - 0.5.*x).*(2-x).^2);

% Function handle for derivative of final expression
k_func_diff = diff(k_func);

% Convert to Matlab Functions
k_func = matlabFunction(k_func);
k_func_diff = matlabFunction(k_func_diff);

x_i = 0.1; % Initial guess for moles of CO reacted, gmoles
x_i1 = x_i - k_func(x_i)./k_func_diff(x_i);

% Newton's Method
% Control loop. Stop after [tol] is reached.

tol=10^-4;  itr_max=100;           % Input tolerance and maximum iterations
% Input starting point
for i=1:itr_max                    % Begin iterative solution
    fx = k_func(x_i);              % Calculate function  value
    dfx=k_func_diff(x_i);          % Calculate df/fx
    x_i1=x_i-fx/dfx;                % Apply Newton's method
    Rerr=(x_i1-x_i)./x_i1;           % Calculate relative error
                                   % Print out intermediate results
    fprintf(' i=%d  x=%f  f(x)=%f   relative error=%f\n',i,x_i1,fx,Rerr)
    x_i=x_i1;                       % Tranfer xnew to x0 for next iteration
    if abs(Rerr)<tol               % Check for convergence
         break
    end
end

if i>itr_max-1                     % Check for maximum iteration limit
    fprintf('Maximum iterations exceeded without convergence\n')
else
end

fprintf("The mole fraction is: %.3f\n", x_i1);
```

```
% Newton's method cannot converge due to the extremely steep slope at the x
% = 0.1 point.

%% Problem  1(b)
% Illinois Method
tol=10^-4; itr_max=100;            % Input tolerance and maximum iterations
x0=0.1;                            % Input starting point
f = k_func;                        % Define function as an anonymous function
%
% Bound the root with a positive and negative function value
%
dx=0.1*x0;                        % Set step size dx for bounding the root
if dx<10^-8;dx=10^-8;end          % Clamp dx to 10^-8 if less than this
x1=x0+dx;                         % Increment x by dx
                                  % Move in the desired direction
if abs((x1))>abs(f(x0));dx=-dx;end
while f(x0)*f(x1)>0               % Check for a bounded root
    x0=x1;                        % Transfer x1 to x0
    x1=x0+dx;                     % If not bounded, continue incrementing
                                  %      x by dx
end
fprintf('The function has been bounded between x= %f and %f\n',x0,x1)
%
% Begin iterative solution of bounded root using Illinois method
%
fun=f(x0);                        % Initialize fun
for i=1:itr_max
                                  % Apply estimate of the root
    xnew=x1-f(x1)*(x1-x0)/(f(x1)-fun);
    Rerr=(xnew-x1)/xnew;          % Calculate relative error
                                  % Print out intermediate results
    fprintf(' i=%d  x=%f  f(x)=%f   relative error=%f\n',i,xnew,f(x1),Rerr)
                                  % Check for convergence
    if abs(Rerr)<tol,break,end
    if (f(x1)*f(xnew))<0          % Check for alternating sign of function
                                  %      values
        x0=x1;                    % Transfer value of xnew to x1 for next
                                  %      iteration
        x1=xnew;                  % Transfer value of xnew to x1 for next
                                  %      iteration
        fun=f(x0);                % Use use regula falsi method when the
                                  %      sign of the function alternates
    else
        x1=xnew;                  % Transfer value of xnew to x1 for next
                                  %      iteration
        fun=0.5*f(x0);            % Use Illinois method modification when
                                  %      the sign does not alternate
    end
end
if i>itr_max-1                     % Check for maximum iteration limit
    fprintf('Maximum iterations exceeded without convergence\n')
else
end
fprintf("The mole fraction is: %.3f\n", x1);
```

```
%% Problem 1(c)

% Use fzero() to find a solution:
xi = 0.1;
res = fzero(k_func, xi);
fprintf("The mole fraction is: %.3f\n", res);
```

Output for Problem 1:

i=1 x=0.733333 f(x)=-214000799.932900   relative error=0.863636

i=2 x=1.155556 f(x)=-63407641.412415   relative error=0.365385

i=3 x=1.437037 f(x)=-18787445.308173   relative error=0.195876

i=4 x=1.624691 f(x)=-5566645.804434   relative error=0.115501

i=5 x=1.749792 f(x)=-1649371.438717   relative error=0.071495

i=6 x=1.833192 f(x)=-488697.260161   relative error=0.045494

i=7 x=1.888786 f(x)=-144793.605028   relative error=0.029434

i=8 x=1.925839 f(x)=-42896.096288   relative error=0.019240

i=9 x=1.950517 f(x)=-12704.157057   relative error=0.012652

i=10 x=1.966914 f(x)=-3758.345885   relative error=0.008336

i=11 x=1.977722 f(x)=-1107.718269   relative error=0.005465

i=12 x=1.984659 f(x)=-322.394145   relative error=0.003495

i=13 x=1.988737 f(x)=-89.927391   relative error=0.002051

i=14 x=1.990568 f(x)=-21.768238   relative error=0.000919

i=15 x=1.990968 f(x)=-3.339993   relative error=0.000201

i=16 x=1.990986 f(x)=-0.139315   relative error=0.000009

The mole fraction is: 1.991

The function has been bounded between x= 1.990000 and 2.000000

i=1 x=1.992665 f(x)=23.040000   relative error=-0.003681

i=2 x=1.990756 f(x)=10.571923   relative error=-0.000959

i=3 x=1.991034 f(x)=-1.801762   relative error=0.000140

i=4 x=1.990987 f(x)=0.361375   relative error=-0.000023

The mole fraction is: 1.991

The mole fraction is: 1.991

```matlab
%% CHEN 320 - 202 Homework 2
% Problem: 2
% Author: Nathaniel Thomas
% Date: 09/16/2022
% Due:  09/18/2022

%% Problem 2
r = 15 / 2; % Radius of the spherical tank, ft
V = 500; % Volume of liquid in the tank, ft^3
% Function for determining height
f = @(h) V - (pi .* h.^2 * (3 .* r - h))./3;

% Start with a guess of the tank's radius
h = r;
h = fzero(f, h);

fprintf("The height of liquid in the tank is %.3f ft.\n", h);
```

Problem 2 output:

```
The height of liquid in the tank is 5.263 ft.
```

```matlab
%% CHEN 320 - 202 Homework 2

% Problem: 3
% Author: Nathaniel Thomas
% Date: 09/16/2022
% Due:  09/18/2022

%% Problem 3

Q = 850; % Desired heat flux, Btu/(h ft^2)
h = 1.0; % Convective heat transfer coefficient, Btu/(h ft^2 degF)
T = 100 + 459.67; % Surrounding temperature, F
e = 0.9; % Emmisivity, Dimensionless
o = 1.174E-9; % Btu/(h ft R^4)

% Function for determining the average surface temperature
f = @(Ts) h*(Ts - T) + e * o * (Ts^4 - T^4) - Q;

Ts = T*2; % Guess that the average surface is double the surroundings
Ts = fzero(f, Ts);
fprintf("The average surface temperature is %.3f degF \n", Ts - 459.67);
```

Problem 3 output:

```
The average surface temperature is 420.252 degF
```