```
clear all, clc
format short

% Define the Objective function
f = @(x) x.^4 - 10*x.^3 + 40*x.^2 - 50*x;

% Set parameters
L = 0;          % Lower limit of the search range
R = 2;          % Upper limit of the search range
maxerr = 0.005; % Maximum error (stopping criteria)
maxiter = 100;  % Maximum number of iterations (safety parameter)

% Plot the function
t = linspace(L, R, 100);
plot(t, f(t), 'k', 'LineWidth', 2);
title('f(x) = x^4 - 10x^3 + 40x^2 - 50x');
xlabel('x');
ylabel('f(x)');
grid on;

% Golden section search
ratio = 0.618;  % Golden ratio
x2 = L + ratio * (R - L); % Compute x2
x1 = L + R - x2;          % Compute x1
err = R - L;              % Initial Error
iter = 1;                 % Set iteration counter initially

fprintf('Iteration\t   L\t\t   R\t\t   x1\t\t   x2\t\t  f(x1)\t\t  f(x2)\t\t↙
Error\n');
fprintf↙
('--------------------------------------------------------------------------------------↙
------------------------\n');

% Create a storage for results
rsl = [];

while err > maxerr
    % Compute Error
    err = R - L;

    % Compute function values
    fx1 = f(x1);
    fx2 = f(x2);

    % Display current iteration details
    fprintf('%4d\t\t%8.6f\t%8.6f\t%8.6f\t%8.6f\t%8.6f\t%8.6f\t%8.6f\n', ...
        iter, L, R, x1, x2, fx1, fx2, err);

    % Store results for this iteration
    rsl(iter,:) = [L, R, x1, x2, fx1, fx2, err];

    % Update interval based on comparison of function values
    if fx1 > fx2  % Look for "Minimum"
        L = x1;   % Update L
        x1 = x2;  % Update x1
        x2 = L + ratio * (R - L); % Compute new x2
    elseif fx1 < fx2
        R = x2;    % Update R
        x2 = x1;  % Update x2
        x1 = L + R - x2; % Compute new x1
    elseif fx1 == fx2
        if min(abs(x1), abs(L)) == abs(L)
            R = x2; % Update R
        else
            L = x1; % Update L
        end
        x1 = L + (1 - ratio) * (R - L);
        x2 = L + ratio * (R - L);
    end

    % Check if maximum iterations reached
    if iter == maxiter
        fprintf('Maximum number of iterations (%d) reached.\n', maxiter);
        break;
    else
        iter = iter + 1; % Update iteration counter
    end
end

% Display the termination condition
if iter < maxiter
    fprintf('Maximum error limit %.6f reached after %d iterations.\n', maxerr, iter);
end

% Display results as a table
Variables = {'L', 'R', 'x1', 'x2', 'fx1', 'fx2', 'Error'};
ResultTable = array2table(rsl);
ResultTable.Properties.VariableNames(1:size(ResultTable, 2)) = Variables;
disp(ResultTable);

% Compute & Print Optimal Result
xopt = (L + R) / 2;       % Optimal "x" (mid-point of final L & R)
fopt = f(xopt);           % Optimal value of f(x)
fprintf('\nOptimal value of x = %.6f\n', xopt);
fprintf('Optimal value of f(x) = %.6f\n', fopt);

% Mark the minimum point on the plot
hold on;
plot(xopt, fopt, 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');
text(xopt + 0.05, fopt, ['Minimum (' num2str(xopt, '%.6f') ', ' num2str(fopt, '%.6f')↙
')']);
hold off;
```