```
%Script to test fixed step size routines for Euler's method, the
%improved Euler method, RK4, and the implicit/backward Euler method.

clear;
close all;
clc;

%A scalar example for y'=f(t,y), y(t0)=y0 out to time t = T, step size h.
t0 = 0; y0 = 1; T = 1; h = 0.1;
f = @(t,y) -y+sin(t);

%Run each method, plot results
[t,y1] = euler_sys(f,[t0,T],y0,h);
[t,y2] = impeuler_sys(f,[t0,T],y0,h);
[t,y3] = rk4_sys(f,[t0,T],y0,h);
[t,y4] = backeuler_sys(f,[t0,T],y0,h);
figure;
plot(t,y1,'-k',t,y2,'-r',t,y3,'-b',t,y4,'-g')
fprintf('Time %f Euler %f Improved Euler %f RK4 %f Implicit Euler %f\n',...
t(end),y1(end),y2(end),y3(end),y4(end));

%A system y'=f(t,y), y(t0)=y0 out to time t = T, step size h. Here
%f(t,y) should accept scalar t, vector y input, return a row vector of correct
%size.
t0 = 0; y0 = [1,2]; T = 1; h = 0.1;
f = @(t,y) [-t*y(2) -y(1)+y(2)];

%Run each method, plot results parametrically
[t,y1] = euler_sys(f,[t0,T],y0,h);
[t,y2] = impeuler_sys(f,[t0,T],y0,h);
[t,y3] = rk4_sys(f,[t0,T],y0,h);
[t,y4] = backeuler_sys(f,[t0,T],y0,h);
figure
plot(y1(:,1),y1(:,2),'-r');
hold on
plot(y2(:,1),y2(:,2),'-b');
plot(y3(:,1),y3(:,2),'-k');
plot(y4(:,1),y4(:,2),'-g');
hold off
```