```matlab
% orbit - Program to compute the orbit of a comet.
clear all;

%* Set initial position and velocity of the comet.
r0 = 1; % AU
v0 = pi/2; % AU/yr
r = [r0 0];  v = [0 v0];
state = [ r(1) r(2) v(1) v(2) ];   % Used by R-K routines

%* Set physical parameters (mass, G*M)
GM = 4*pi^2;      % Grav. const. * Mass of Sun (au^3/yr^2)
mass = 1.;        % Mass of comet
adaptErr = 1.e-3; % Error parameter used by adaptive Runge-Kutta
time = 0;
alpha = 0.02;

%* Loop over desired number of steps using specified
%  numerical method.
nStep = 300;
tau = 0.1;
NumericalMethod = 4;
for iStep=1:nStep

  %* Record position and energy for plotting.
  rplot(iStep) = norm(r);           % Record position for polar plot
  thplot(iStep) = atan2(r(2),r(1));
  tplot(iStep) = time;
  kinetic(iStep) = .5*mass*norm(v)^2;   % Record energies
  potential(iStep) = - GM*mass/norm(r);

  %* Calculate new position and velocity using desired method.
  if( NumericalMethod == 1 )
    accel = -GM*r/norm(r)^3;
    r = r + tau*v;             % Euler step
    v = v + tau*accel;
    time = time + tau;
  elseif( NumericalMethod == 2 )
    accel = -GM*r/norm(r)^3;
    v = v + tau*accel;
    r = r + tau*v;             % Euler-Cromer step
    time = time + tau;
  elseif( NumericalMethod == 3 )
    state = rk4(state,time,tau,'gravrk',GM);
    r = [state(1) state(2)];   % 4th order Runge-Kutta
    v = [state(3) state(4)];
    time = time + tau;
  else
    param = [GM, alpha];
    [state time tau] = rka(state,time,tau,adaptErr,'grav_prec',param);
    r = [state(1) state(2)];   % Adaptive Runge-Kutta
    v = [state(3) state(4)];
  end
```

```
end

% find precession angles
% find angle at all maxima
th_prec = []; % accumulator for precession angles
for i = 2:length(rplot)
    if rplot(i) > rplot(i-1) && rplot(i) > rplot(i+1)
        th_prec = [th_prec, thplot(i)];
    end
end
% convert to degrees
th_prec = rad2deg(th_prec);
% find differences in precession angles
th_diff = diff(th_prec);
dth_exp = abs(mean(th_diff));

% prove precession angle difference in book
% find value of a
L = r0 * v0;
a = sqrt(1 + (GM * mass^2 * alpha) / L^2);
dth = 360 * (1-a) / a;

% print error in precession angle
err = abs(dth_exp - dth) / dth;
fprintf("The error in the precession angle approximation is %f degrees.\n", err)
```

```
The error in the precession angle approximation is -1.035810 degrees.
```