CODE FOR GALAXY 1

```
function bodies = ConstructGalaxy(rp,cm,pos,vel)

persistent bs;
numberOfBodies = 350;

if isempty(bs);
   rng('default');
   bs = ConstructGalaxy0(rp,cm,pos,vel,numberOfBodies);
end

bodies = bs;

function bodies = ConstructGalaxy0(rp,cm,pos,vel,n)

SolarMass = 1.9891e+30; % In kg
G = 6.672E-11; % Nm^2/kg^2 (Gravitational constant)
SpeedOfLight = 299792458; % in m/s
YearInSeconds = 365*24*60*60;
LightYear = SpeedOfLight*YearInSeconds;
Parsec = 3.26*LightYear;

radiusOuter = rp*Parsec;
radiusInner = (rp/3)*Parsec;

% Each star has X,Y,Z,VX,VY,VZ
% X,Y,Z position in cartesian coordinates
% VX,VY,VZ velocity in cartesian coordinates
cm = cm*SolarMass;

bodies = zeros(n,8);
bodies(1,1) = cm;
bodies(1,2) = pos(1)*Parsec;
bodies(1,3) = pos(2)*Parsec;
bodies(1,4) = pos(3)*Parsec;
bodies(1,5) = vel(1);
```

```
bodies(1,6) = vel(2);
bodies(1,7) = vel(3);
bodies(1,8) = 'r';

if n > 1
   for i = 2:n,
      m0 = rand*20+4;
      m = m0*SolarMass;
      r = rand*(radiusOuter - radiusInner) + radiusInner;
      arg = rand*(2*pi);
      x = r*cos(arg);
      y = r*sin(arg);
      z = 0;
      dx = cos(arg+pi/2);
      dy = sin(arg+pi/2);
      dz = 0;
      % Compute free fall velocity
      v = sqrt(G*cm/r);
      bodies(i,1) = m;
      bodies(i,2) = x+pos(1)*Parsec;
      bodies(i,3) = y+pos(2)*Parsec;
      bodies(i,4) = z+pos(3)*Parsec;
      bodies(i,5) = dx*v+vel(1);
      bodies(i,6) = dy*v+vel(2);
      bodies(i,7) = dz*v+vel(3);
      bodies(i,8) = 'r';
   end
end
```

CODE FOR GALAXY 2

```
function bodies = ConstructGalaxy(rp,cm,pos,vel)

persistent bs;
numberOfBodies = 350;

if isempty(bs);
```

```
    rng('default');
    bs = ConstructGalaxy0(rp,cm,pos,vel,numberOfBodies);
end

bodies = bs;

function bodies = ConstructGalaxy0(rp,cm,pos,vel,n)

SolarMass = 1.9891e+30; % In kg
G = 6.672E-11; % Nm^2/kg^2 (Gravitational constant)
SpeedOfLight = 299792458; % in m/s
YearInSeconds = 365*24*60*60;
LightYear = SpeedOfLight*YearInSeconds;
Parsec = 3.26*LightYear;

radiusOuter = rp*Parsec;
radiusInner = (rp/3)*Parsec;

% Each star has X,Y,Z,VX,VY,VZ
% X,Y,Z position in certesian coordinates
% VX,VY,VZ velocity in certesian coordinates
cm = cm*SolarMass;

bodies = zeros(n,8);
bodies(1,1) = cm;
bodies(1,2) = pos(1)*Parsec;
bodies(1,3) = pos(2)*Parsec;
bodies(1,4) = pos(3)*Parsec;
bodies(1,5) = vel(1);
bodies(1,6) = vel(2);
bodies(1,7) = vel(3);
bodies(1,8) = 'y';

if n > 1
    for i = 2:n,
        m0 = rand*20+4;
        m = m0*SolarMass;
```

```
        r = rand*(radiusOuter - radiusInner) + radiusInner;
        arg = rand*(2*pi);
        x = r*cos(arg);
        y = r*sin(arg);
        z = 0;
        dx = cos(arg+pi/2);
        dy = sin(arg+pi/2);
        dz = 0;
        % Compute free fall velocity
        v = sqrt(G*cm/r);
        bodies(i,1) = m;
        bodies(i,2) = x+pos(1)*Parsec;
        bodies(i,3) = y+pos(2)*Parsec;
        bodies(i,4) = z+pos(3)*Parsec;
        bodies(i,5) = dx*v+vel(1);
        bodies(i,6) = dy*v+vel(2);
        bodies(i,7) = dz*v+vel(3);
        bodies(i,8) = 'y';
    end
end
```

Code for partition block:

```
function [heavy,light] = Partition(bodies)

SolarMass = 1.9891e+30; % kg
Limit = 100*SolarMass;

n = size(bodies,1);
props = size(bodies,2);
heavy = zeros(n,props);
light = zeros(n,props);

lightIndex = 1;
heavyIndex = 1;

for i = 1:n
    m = bodies(i,1);
```

```
    if m < Limit
        light(lightIndex,:) = bodies(i,:);
        lightIndex = lightIndex + 1;
    else
        heavy(heavyIndex,:) = bodies(i,:);
        heavyIndex = heavyIndex + 1;
    end
end
```

Code for Gravity Block

```
function [heavy1,light1] = ApplyGravity(light,heavy)

G = 6.672E-11; % Nm^2/kg^2 (Gravitational constant)

YearInSeconds = 365*24*60*60;
timeStep = 2000000*YearInSeconds;

n = size(heavy,1);

heavy1 = heavy;
light1 = light;

for i = 1:n,
    mi = heavy(i,1);
    if mi == 0
        break;
    end
    xi = heavy(i,2);
    yi = heavy(i,3);
    zi = heavy(i,4);
    ar = [0 0 0];
    for j = 1:n,
        if i ~= j,
            mj = heavy(j,1);
            if mj == 0
                break;
            end
```

```
        xj = heavy(j,2);
        yj = heavy(j,3);
        zj = heavy(j,4);
        d = [xj yj zj] - [xi yi zi];
        dr2 = d(1)*d(1)+d(2)*d(2)+d(3)*d(3);
        ar = ar + (d/sqrt(dr2))*((G*mj)/dr2);
      end
    end
    for k = 1:3
      heavy1(i,4+k) = heavy(i,4+k) + ar(k)*timeStep;
    end
end

for i = 1:n,
    mi = light(i,1);
    if mi == 0
      break;
    end
    xi = light(i,2);
    yi = light(i,3);
    zi = light(i,4);
    ar = [0 0 0];
    for j = 1:n,
      mj = heavy(j,1);
      if mj == 0
        break;
      end
      xj = heavy(j,2);
      yj = heavy(j,3);
      zj = heavy(j,4);
      d = [xj yj zj] - [xi yi zi];
      dr2 = d(1)*d(1)+ d(2)*d(2) + d(3)*d(3);
      ar = ar + (d/sqrt(dr2))*((G*mj)/dr2);
    end
    for k = 1:3
      light1(i,4+k) = light(i,4+k) + ar(k)*timeStep;
    end
```

```
end

for i = 1:n
   for k = 1:3
      heavy1(i,k+1) = heavy1(i,k+1) + timeStep*heavy1(i,k+4);
   end
   for k = 1:3
      light1(i,k+1) = light1(i,k+1) + timeStep*light1(i,k+4);
   end
end
```

Code for Merge Block

```
function M = combine(A,B)

n = size(A,1);
nProps = size(A,2);
M = zeros(n,nProps);
for i = 1:n
   if A(i,1) == 0
      break
   end
   M(i,:) = A(i,:);
end
n1 = n - i + 1;
for j = 1:n1
   M(j+i-1,:) = B(j,:);
   if B(i,1) == 0
      break
   end
end
```

Code for Plot Block

```
function PlotAll(bodies)

persistent fig;
persistent oldPlot;
```

```
coder.extrinsic('findobj','get','set','figure','clf','hold','text','delete','plot3','drawnow');

n = size(bodies,1);

SpeedOfLight = 299792458; % in m/s
YearInSeconds = 365*24*60*60;
LightYear = SpeedOfLight*YearInSeconds;
Parsec = 3.26*LightYear;
foundFig = findobj('Tag','galaxyScreen');

if isempty(fig)||isempty(foundFig)
   if isempty(foundFig)
      fig = figure;
   else
      fig = figure(foundFig);
   end;
   clf(fig);
   set(fig, 'Name', 'Galaxy');
   set(fig, 'Tag', 'galaxyScreen');
   set(fig, 'Renderer', 'painters');
   set(fig, 'Color', 'black');
   hold('on');
   fig_axes = get(fig, 'CurrentAxes');
   init_axes(fig_axes);
   text(0,3.5*30000*Parsec,0,...
      ['Spiral galaxy formation on close encounters', ...
      char(10),'(based on Toomre & Toomre, 1972)'],...
      'Color','green','FontSize',12, 'HorizontalAlignment','center');
end

points_x = zeros(1,n);
points_y = zeros(1,n);
points_z = zeros(1,n);
points_col = zeros(1,n);
for i = 1:n,
   points_x(i) = bodies(i,2);
```

```
    points_y(i) = bodies(i,3);
    points_z(i) = bodies(i,4);
    points_col(i) = bodies(i,8);
end

%
% Remove the old plot.
%
if isempty(oldPlot)
    oldPlot = fig;
elseif ~isempty(foundFig)
    delete(oldPlot);
end

oldPlot = plot3(points_x,points_y,points_z,'w.');
drawnow;

function init_axes(a)

coder.extrinsic('set');

SpeedOfLight = 299792458; % in m/s
YearInSeconds = 365*24*60*60;
LightYear = SpeedOfLight*YearInSeconds;
Parsec = 3.26*LightYear;
set(a, 'CameraTarget', [0,0,0] );
set(a, 'CameraPosition', [0,22000*Parsec*3,18000*Parsec*3]);
set(a, 'CameraViewAngle', 80 );
set(a, 'CameraUpVector', [0,1,0]);
set(a, 'Visible', 'off' );
set(a, 'XLim', [-25000*Parsec*8,30000*Parsec*8]);
set(a, 'YLim', [-25000*Parsec*8,30000*Parsec*8]);
set(a, 'ZLim', [-30000*Parsec*8,30000*Parsec*8]);
```