```
function Orientation

% Clear all input and output from the comand window
clc

% Time scale for ode45 solver
% time step
dt=0.001;
% upper time limit
ut=100; % seconds
% time interval
t=0:dt:ut;

% Row index corresponding to the upper time limit (+1 in order to account
% for the row at t=0)
riut=(ut/dt)+1



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
% Solving differential equation (function - DCMdiffEq) that relates time
% derivative of the general Directoin Cosine Matrix (DCM) to angular velocity
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%

% Initial conditions at t=0 sec: R(0)=I - rotating spacecraft frame of
% reference (o) coincides with the fixed frame (b)
% the elements of the indentity matrix are arranged into column vector,
% since ode45 requires a vector of initial values as an input
Rinit=[1;0;0;0;1;0;0;0;1];

% ode45 integrator or differential equation solver
% the output is a matrix where each row includes a value of time followed by
% nine elements of the rotation matrix corresponding to that time
[t,R]=ode45(@DCMdiffEq,t,Rinit)

% Display rotation matrix at time t=1000s;
R11=R(riut,1)
R12=R(riut,2);
R13=R(riut,3);
R21=R(riut,4);
R22=R(riut,5);
R23=R(riut,6);
R31=R(riut,7);
R32=R(riut,8);
R33=R(riut,9);

% Rotation matrix at time t=1000s
R=[R11,R12,R13;R21,R22,R23;R31,R32,R33]

% As assumed the spacecraft reference frame (o) initially
% coincides with the with the fixed frame (b) (R(0)=I),
% assuming the spacecraft lies along x axis in frame o, the position vector
% of the spacecraft in frame o is given by
ro=[1;0;0];
% the initial orientation of the spacecraft in the fixed frame (b) at t=0s
% is given by the following position vector
rbinit=eye(3)*ro;
```

```
% the final orientation of the spacecraft in the fixed frame (b) at t=1000s
% is given by the following position vector
rbfin=R*ro;



% Differential equation that relates time derivative of the general
% Directoin Cosine Matrix (DCM) to angular velocity
function dR=DCMdiffEq(t,R)

    omg1=-4; % rad/s
    omg2=1; % rad/s

    % Angular velocity of the spacecraft
    % x angular velocity component
    wx=-0.10*cos(omg1*t);
    % y angular velocity component
    wy=0.10*sin(omg1*t);
    % z angular velocity component
    wz=omg2;

    % Rotation angles
    % alpha: rotation angle about x axis = x angular velocity component
    % multiplied by time
    a=wx*t;
    % beta: rotation angle about y axis = y angular velocity component
    % multiplied by time
    b=wy*t;
    % gamma: rotation angle about z axis = z angular velocity component
    % multiplied by time
    g=wz*t;

    % Directional Cosine Matrices [notes]
    % rotation about x axis by angle alpha
    Rxa=[1,0,0;0,cos(a),-sin(a);0,sin(a),cos(a)];
    % rotation about y axis by angle beta
    Ryb=[cos(b),0,sin(b);0,1,0;-sin(b),0,cos(b)];
    % rotation about z axis by angle gamma
    Rzg=[cos(g),-sin(g),0;sin(g),cos(g),0;0,0,1];

    % General DCM
    R=Rzg*Ryb*Rxa;

    % Omega cross: skew-symmetric matrix constructed using the spacecraft
    % angular velocity components
    omgCross=[0,-wz,wy;wz,0,-wx;-wy,wx,0];

    % Differential equation that relates time derivative of the general
    % DCM (dR) to angular velocity [notes]
    dRdt=omgCross*R;

    % splitting dRdt matrix into separate elements (ode45 needs a vector
    % of differential elements)
    dR11=dRdt(1,1);
    dR12=dRdt(1,2);
    dR13=dRdt(1,3);
    dR21=dRdt(2,1);
    dR22=dRdt(2,2);
    dR23=dRdt(2,3);
    dR31=dRdt(3,1);
    dR32=dRdt(3,2);
```

```
    dR33=dRdt(3,3);

    dR=[dR11;dR12;dR13;dR21;dR22;dR23;dR31;dR32;dR33];

end
```