HW3 Problem 1

```
fprintf('\n');
clearvars -except function_list hw_pub toolsPath
close all
CelestialConstants; % import useful constants

r = [-5633.9; -2644.9; 2834.4];
v = [2.425; -7.103; -1.800];

[a,e,i,RAAN,w,f] = cart2OE(r,v,Earth.mu);
fprintf('Semi-major axis: %.2f km\n',a);
fprintf('Eccentricity: %.5f \n',e);
fprintf('Inclination: %.2f deg\n',i * 180/pi);
fprintf('Right Ascention of Asc. Node: %.2f deg\n',RAAN * 180/pi);
fprintf('Argument of Periapse: %.2f deg\n',w * 180/pi);
fprintf('True anomaly: %.2f deg\n',f * 180/pi);
```

```
Semi-major axis: 6993.51 km
Eccentricity: 0.02212
Inclination: 28.40 deg
Right Ascention of Asc. Node: 82.52 deg
Argument of Periapse: 118.25 deg
True anomaly: 1.13 deg
```

Published with MATLAB® R2013b

HW3 Problem 2

```
fprintf('\n');
clearvars -except function_list hw_pub toolsPath
close all
CelestialConstants; % import useful constants

p = 11067.790;
e=0.83285;
a = p/(1-e*e);
i = 87.87*pi/180;
RAAN = 227.89*pi/180;
w = 53.38*pi/180;
f = 92.335*pi/180;
[r, v ] = OE2cart( a,e,i,RAAN,w,f,Earth.mu);

fprintf('r_ECI = %.2f\n',r(1));
fprintf('        %.2f km\n',r(2));
fprintf('        %.2f\n\n',r(3));
fprintf('v_ECI = %.3f\n',v(1));
fprintf('        %.3f km/s\n',v(2));
fprintf('        %.3f\n',v(3));
```

```
r_ECI = 6525.37
        6861.53 km
        6449.12

v_ECI = 4.902
        5.533 km/s
        -1.976
```

Published with MATLAB® R2013b

HW3 Problem 3

```
fprintf('\n');
clearvars -except function_list hw_pub toolsPath
close all
CelestialConstants; % import useful constants

n = 15.5918272 / day2sec *2*pi;
M = 231.8021 * pi/180;
w = 106.9025 * pi/180;
e = 0.0008148;
RAAN = 342.1053 * pi/180;
i = 51.6396 * pi/180;

M2 = M + 3600*n;
if M2 > 2*pi
    M2 = M2 - 2*pi;
end

f = E2f(M2E(M2,e),e);
a = (Earth.mu/n^2)^(1/3);

[r, v ] = OE2cart( a,e,i,RAAN,w,f,Earth.mu);

fprintf('r,v for ISS one hour after epoch:\n');
fprintf('r_ECI = %.2f\n',r(1));
fprintf('        %.2f km\n',r(2));
fprintf('        %.2f\n\n',r(3));
fprintf('v_ECI = %.3f\n',v(1));
fprintf('        %.3f km/s\n',v(2));
fprintf('        %.3f\n',v(3));
```

```
r,v for ISS one hour after epoch:
r_ECI = -6119.72
        -407.27 km
        -2865.53

v_ECI = 2.704
        -5.087 km/s
        -5.067
```

Published with MATLAB® R2013b

CelestialConstants

Table of Contents

Description

All sorts of constants for orbital mechanics purposes

```
fcnPrintQueue(mfilename('fullpath')) % Add this code to code app
```

Earth

```
Earth.mu = 3.986e5; %km3/s2
Earth.R = 6378; %km
```

Celestial units

```
au2km = 149597870.7;
```

Physical constants

```
day2sec = 86400; % sec/day
```

Published with MATLAB® R2013b

```matlab
function [a,e,i,RAAN,w,f] = cart2OE( r, v ,mu)
%cart2OE return classical orbital elements from cartesian coords
% Only valid for e < 1
% units in radians
fcnPrintQueue(mfilename('fullpath')) % Add this code to code app

h = cross(r,v);
n = cross([0;0;1],h);
ecc_vec = ((norm(v)*norm(v)-mu/norm(r))*r - dot(r,v)*v)/mu;

e = norm(ecc_vec);
a = 0;
if e < 1.0
    specific_energy = norm(v)*norm(v)/2-mu/norm(r);
    a = -mu/2/specific_energy;
end

i = acos(h(3)/norm(h));
RAAN = acos(n(1)/norm(n));
if n(2) < 0
    RAAN = 2*pi-RAAN;
end
w = acos(dot(n,ecc_vec)/(norm(n)*norm(ecc_vec)));
if ecc_vec(3) < 0
    w = 2*pi-w;
end
f = acos(dot(ecc_vec,r)/(norm(ecc_vec)*norm(r)));
if dot(r,v) < 0
    f = 2*pi-f;
end
```

Published with MATLAB® R2013b

```
function [r, v ] = OE2cart( a,e,i,RAAN,w,f,mu)
%cart2OE return classical orbital elements from cartesian coords
% Only valid for e < 1
% units in radians
fcnPrintQueue(mfilename('fullpath')) % Add this code to code app

% First find r,v in the perifocal coord system.
p = a*(1-e*e);
r_pqw = [p*cos(f);p*sin(f);0]/(1+e*cos(f));
v_pqw = [-sqrt(mu/p)*sin(f); sqrt(mu/p)*(e+cos(f));0];

r = Euler2DCM('313', -[w,i,RAAN])*r_pqw;
v = Euler2DCM('313', -[w,i,RAAN])*v_pqw;
```

Published with MATLAB® R2013b

```
function DCM = Euler2DCM( seq_string, angle_vector )
%Euler2DCM Turn an Euler Angle set into a DCM
%   Angle vector in radians
fcnPrintQueue(mfilename('fullpath'))

DCM = eye(3);
%get the trig functions
c = zeros(3,1);
s = zeros(3,1);
c(1) = cos(angle_vector(1));
s(1) = sin(angle_vector(1));
c(2) = cos(angle_vector(2));
s(2) = sin(angle_vector(2));
c(3) = cos(angle_vector(3));
s(3) = sin(angle_vector(3));

for idx = 3:-1:1
    if strcmp(seq_string(idx),'1')
        M = [1 0 0; 0 c(idx) s(idx); 0 -s(idx) c(idx)];
        DCM = DCM*M;
    elseif strcmp(seq_string(idx),'2')
        M = [c(idx) 0 -s(idx); 0 1 0; s(idx) 0 c(idx)];
        DCM = DCM*M;
    elseif strcmp(seq_string(idx),'3')
        M = [c(idx) s(idx) 0; -s(idx) c(idx) 0; 0 0 1];
        DCM = DCM*M;
    else
        fprintf('%s is not a valid axis\n', seq_string(idx))
    end
end


end
```

Published with MATLAB® R2013b

```matlab
function E = M2E( M, e )
%M2E Mean anom (M) to eccentric anom (E)
fcnPrintQueue(mfilename('fullpath')) % Add this code to code app

tol = 1e-5;

if (M < 0 && M > -pi) || M > pi
    E_1 = M - e;
else
    E_1 = M + e;
end

E = E_1 + tol + 1;
while abs(E_1-E) > tol
    E = E_1;
    E_1 = E - (E - e*sin(E) - M)/(1 - e*cos(E));
end
```

Published with MATLAB® R2013b

```
function f = E2f( E, e )
%E2f Eccentric anomaly (E) to true anomaly (f)
% Only valid for e < 1
% units in radians
fcnPrintQueue(mfilename('fullpath')) % Add this code to code app

% Vallado eqn 2-10
f = acos((cos(E) - e)/(1-e*cos(E)));
if E > pi
    f = 2*pi - f;
end
```

Published with MATLAB® R2013b

```
function fcnPrintQueue( filename )
global function_list;
if exist('function_list', 'var')
    file_in_list = 0;
    for idx = 1:length(function_list)
        if strcmp(function_list(idx), filename);
            file_in_list = 1;
            break
        end
    end
    if ~file_in_list
        function_list = [function_list, filename];
    end
end
end
```