```
function x = OrbitalElementsConverter(flag,x,mu,time)


if isequal(flag,'Cartesian')

    % Calculate Norms and Vectors of r and v:
    v = x(4:6);
    r = x(1:3);
    vmag = norm(v,2);
    rmag = norm(r,2);

    % Vectors:
    h = cross(r,v); % Angular Momentum
    hmag = norm(h,2);
    n = cross([0;0;1],h); % Node Vector
    nmag = norm(n,2)
    e = 1/mu*((vmag^2-mu/rmag)*r-dot(r,v)*v); % Eccentricity
    emag = norm(e,2);

    % Angles:
    i = acosd(dot(h,[0;0;1])/(hmag));
    Om = acosd(n(1)/nmag);
    om = acosd(dot(n,e)/(nmag*emag));
    if emag < 0
        om = 360-om;
    end

    % Other:
    p = norm(h,2)^2/mu;
    a = p/(1-emag^2);
    f = acos(dot(e,r)/(emag*rmag));
    M0 = 180/pi*(asin(sqrt(1-emag^2)*sin(f)/(1+emag*cos(f)))-
emag*sqrt(1-emag^2)*sin(f)/(1+emag*cos(f)));
    M = 180/pi*sqrt(mu/a^3)*time+M0;

    x = [a;emag;i;Om;om;M];
elseif isequal(flag,'Orbital')
    a = x(1); e = x(2); i = x(3); Om = x(4); om = x(5); M0 =
 x(6)*pi/180;
    p = a*(1-e^2);
    h = sqrt(p*mu);

    % Determine Mean Anomaly at time:
    M = sqrt(mu/a^3)*time+M0;

    %Determine eccentric anomaly, true anomaly, then r
    syms Es
    E = double(solve(Es == M+e*sin(Es)));
    v = 2*atan(sqrt((1+e)/(1-e))*tan(E/2));
    r = p/(1+e*cos(v));

    % Determine r, v in pqw frame:
```

```
    rpqw = [r*cos(v);r*sin(v);0];
    vpqw = [-mu/h*sin(v);mu/h*(e+cos(v));0];

    % Transfrom pqw to ijk:
    Tpqw2ijk = [cosd(Om)*cosd(om)-sind(Om)*sind(om)*cosd(i), -
cosd(Om)*sind(om)-sind(Om)*cosd(om)*cosd(i), sind(Om)*sind(i);
                sind(Om)*cosd(om)+cosd(Om)*sind(om)*cosd(i), -
sind(Om)*sind(om)+cosd(Om)*cosd(om)*cosd(i), -cosd(Om)*sind(i);
                sind(om)*sind(i), cosd(om)*sind(i),cosd(i)];
    rijk = Tpqw2ijk*rpqw;
    vijk = Tpqw2ijk*vpqw;

    x = [rijk; vijk];
else
    display('Invalid Flag');
end


end
```

Published with MATLAB® R2020b