```
close all
clear
clc

%----question (c)--------------
%----test codes for validating functi ons in (a) and (b)
% declare a RPY matrix
RPY_angles_init = [pi/8;pi/6;pi/2.8];
% compute the Rotation Matrix
R = RPY_to_Rot(RPY_angles_init);
disp("Rotation Matrix obtained from Function (a): ");
disp(R);
% input the Rotation Matrix obtained above to Rot_to_RPY function to check
% if we can get the same values as declared for RPY_angles
RPY_angles_end = Rot_to_RPY(R);
disp("RPY angles obtained from Function (b): ");
disp(RPY_angles_end);
disp("Initial RPY angles for comparison: ");
disp(RPY_angles_init);
% check equality
equaltiy = false;
criteria = 1E-10;
for i=1:3
    % check if each angle difference in two RPY matrices are within the criteria (a very small number!)
    if (RPY_angles_end(i)-RPY_angles_init(i))<criteria
        % assign true to equality if current angle difference is within
        % criteria!
        equality = true;
    else
        % if there exists inequality in a certain pair of angles, assign
        % false to equaltiy and then break the for loop to quit
        equality = false;
        break;
    end
end

% check equality and display corresponding messages.
if equality == true
    disp("The codes are validated to be correct because the input RPY angles are equal to the RPY angles that went through two functions!!");
else
    disp("The codes has error!!");
end
%----question (c) ends---------



%------question (a)------------
function R = RPY_to_Rot(RPY_angles)
% assign the RPY angles for convenience
% Roll angle
gamma = RPY_angles(1);
% Pitch angle
beta = RPY_angles(2);
% Yaw angle
alpha = RPY_angles(3);
% Compute each elementary rotation matrices for RPY.
% Roll Matrix
R1 = [1,0,0;0,cos(gamma),-sin(gamma);0,sin(gamma),cos(gamma)];
% Pitch Matrix
R2 = [cos(beta),0,sin(beta);0,1,0;-sin(beta),0,cos(beta)];
% Yaw Matrix
R3 = [cos(alpha),-sin(alpha),0;sin(alpha),cos(alpha),0;0,0,1];
% Total Rotation Matrix
R = R3*R2*R1;
end
%------question (a) ends--------



%------question (b)-------------
function RPY_angles = Rot_to_RPY(R)
% assign the Rotation Matrix variables for convenience
r11 = R(1,1);
r12 = R(1,2);
r13 = R(1,3);
r21 = R(2,1);
r22 = R(2,2);
r23 = R(2,3);
r31 = R(3,1);
r32 = R(3,2);
r33 = R(3,3);
% Compute the cos(beta), sin(beta) and pitch angle beta
cbeta = sqrt(r11^2+r21^2);
sbeta = -r31;
beta = atan2(sbeta,cbeta);
% Compute the cos(alpha), sin(alpha) and yaw angle alpha
calpha = r11/cbeta;
salpha = r21/cbeta;
alpha = atan2(salpha,calpha);
% Compute the cos(gamma), sin(gamma) and roll angle gamma
cgamma = r33/cbeta;
sgamma = r32/cbeta;
gamma = atan2(sgamma,cgamma);
```

```
% assign the rpy angles to return function
RPY_angles = [gamma; beta; alpha];
end
%------question (b) ends--------
```

```
Rotation Matrix obtained from Function (a):
    0.3758   -0.7494    0.5452
    0.7803    0.5732    0.2502
   -0.5000    0.3314    0.8001

RPY angles obtained from Function (b):
    0.3927
    0.5236
    1.1220

Initial RPY angles for comparison:
    0.3927
    0.5236
    1.1220

The codes are validated to be correct because the input RPY angles are equal to the RPY angles that went through two functions!!
```

Published with MATLAB® R2018b