Problem 2

a)

Changes done to `MainKinematics`:

```
 7 %%%%%% MODIFY. Initial state values and parameter values
 8 state = [0;0;0];  % euler angles
 9 omega_ab_in_b = 2 * [1; 1; 1];
10
11 % Simulate dynamics
12 try
13     %%%%%% MODIFY THE FUNCTION "Kinematics" TO PRODUCE SIMULATIONS OF THE SOLID ORIENTATION
14     %%%%%%
15     %%%%%% Hints:
16     %%%%%% - "parameters" allows you to pass some parameters to the "Kinematic" function.
17     %%%%%% - "state" will contain representations of the solid orientation (SO(3)).
18     %%%%%% - use the "reshape" function to turn a matrix into a vector or vice-versa.
19
20     [time,statetraj] = ode45(@(t,x)Kinematics(t, x, omega_ab_in_b),[0, time_final],state);
```

```
46     omega = omega_ab_in_b;
47     R = Rotations(state_animate.');  % .' to avoid complex conjugates
```

b)

Changes done to `MainKinematicsDCM`:

```
 7 %%%%%% MODIFY. Initial state values and parameter values
 8 state = reshape(eye(3), [9,1]);
 9 omega_ab_in_b = 2 * [1; 1; 1];
10
11 % Simulate dynamics
12 try
13     %%%%%% MODIFY THE FUNCTION "Kinematics" TO PRODUCE SIMULATIONS OF THE SOLID ORIENTATION
14     %%%%%%
15     %%%%%% Hints:
16     %%%%%% - "parameters" allows you to pass some parameters to the "Kinematic" function.
17     %%%%%% - "state" will contain representations of the solid orientation (SO(3)).
18     %%%%%% - use the "reshape" function to turn a matrix into a vector or vice-versa.
19
20     [time,statetraj] = ode45(@(t,x)KinematicsDCM(t, x, omega_ab_in_b),[0, time_final],state);
```

```
46     omega = omega_ab_in_b;
47     R = reshape(state_animate, [3,3]);
```