The following exercise tells you how to compute the Essential matrix using the 8 point method.

E = q2.transpose * [tx] * q1

1. Generate 3d 50 points using uniform distribution
   np.random.uniform()
   In the range 0 to 1
   The matrix can be 3 X 50. = Q

   Solution:
   ```
   p = np.random.uniform(0,1,(50,3))
   ```

2. Add [1 4 3] to all the points -> 1 to x, 4 to y and 3 to z.

```
p = p + b = np.repeat([[1,4,3]],50,axis=0)
```

This will give p => 50 X 3

-- You need to put the homogenous component “1” at the end of each 3d vector.
FInal matrix will be of size 4X 50 or 50 X 4.

```
o1 = np.ones(50,1)
p = np.hstack((p,o1))
```

3. Project the points into two cameras using the two projection matrices.
P1 = [I 0]

`P1 = [ [1 0 0 0], [0 1 0 0], [0 0 1 0]]` -> **first camera**

```
q = np.dot(P1, Q.T)
q = q.T

u1 = q[:,0]/q[:,2]
v1 = q[:,1]/q[:,2]
```

For each point,
q = P1 * Q
The output is in homogenous coordinates, divide by the last component to get [u1, v1, 1]

Generate a rotation and translation
R =[0.8660   0.2500   0.4330
        0   0.8660   -0.5000

-0.5000   0.4330   0.7500]

T = [-0.4 0.3 0.6]

q = P2 * Q

```
q = np.dot(P2, Q.T)
q = q.T

u2 = q[:,0]/q[:,2]
v2 = q[:,1]/q[:,2]
```

P2 = [[R t]] -> **second camera**

q2 = P2 * Q
The output is in homogenous coordinates, divide by the last component of each point to get [u2, v2, 1]

4. Use corresponding points in the image to generate 8 point equations

A = [u2 * u1, u2 * v1 , u2 , v2 * u1 , v2 * v1 , v2 , u1 , v1 , 1]

5. Formulate two equations:

Solve Ax = 0

```
u, s, vh = np.linalg.svd(A, full_matrices=True)
```

F = vh[

Ax = 0 and
Ax = b

6. Solve using svd and pseudoinverse

7. What are the values?

8. Use python-opencv to decompose the essential matrix:

R1, R2, T = cv2.decomposeEssentialMat(E)