Program:

```
# AIM: To perform fundamental rotation.
# rotate mobile frame with an angle pi/4 about first unit vector of F.
# calculate final value of q wrt fixed frame.

from math import sin, cos
from matplotlib import pyplot as plt


def algo(q_m, t, axis):
      # finding the value of q_f given the value of q_m.
      # q_m: initial coordinates of the mobile frame.
      # t is angle in radians.

      rotation_matrices = {
      1: lambda t1, x, y, z: [x, y*cos(t1)-z*sin(t1), sin(t1)*y+cos(t1)*z],
      2: lambda t2, x, y, z: [x*cos(t1)+z*sin(t2), y, -sin(t2)*x+cos(t2)*z],
      3: lambda t3, x, y, z: [x*cos(t3)-y*sin(t3), x*sin(t3)+y*cos(t3), z]
      }

      # assert len(q_m) == 3, "Input dimension should be 3."
      assert axis in range(1, 4), "axis value should be between [1-3]"

      return rotation_matrices[axis](t, *q_m)


def main():
   q_m = map(int, input("Enter the coordinates in spaced integers: ").split())
   angle = float(input("Enter the angle of rotation(in radians): "))
   axis = int(input("Enter the axis along which you want to rotate: "))
   transformed_coords = algo(q_m, angle, axis)
   plt.plot(range(3), )


if __name__ == "__main__":
      main()
# AIM: To perform fundamental rotation.

# rotate mobile frame with an angle pi/4 about first unit vector of F.
# calculate final value of q wrt fixed frame.

from math import sin, cos
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.patches import FancyArrowPatch
from mpl_toolkits.mplot3d import proj3d
```

```
def algo(q_m, t, axis):
     # finding the value of q_f given the value of q_m.
     # q_m: initial coordinates of the mobile frame.
     # t is angle in radians.

     rotation_matrices = {
     1: lambda t1, x, y, z: [x, y*cos(t1)-z*sin(t1), sin(t1)*y+cos(t1)*z],
     2: lambda t2, x, y, z: [x*cos(t2)+z*sin(t2), y, -sin(t2)*x+cos(t2)*z],
     3: lambda t3, x, y, z: [x*cos(t3)-y*sin(t3), x*sin(t3)+y*cos(t3), z]
     }

     # assert len(q_m) == 3, "Input dimension should be 3."
     assert axis in range(1, 4), "axis value should be between [1-3]"

     return rotation_matrices[axis](t, *q_m)


def main():
   coords = list(map(int, "1 2 3".split())) # input("Enter the coordinates in
spaced integers: ").split()))
   angle = 0.7854 #float(input("Enter the angle of rotation(in radians): "))
   axis = 1 # int(input("Enter the axis along which you want to rotate: "))
   print("initial.")
   transformed_coords = algo(coords, angle, axis)
   print(transformed_coords)
   fig = plt.figure()
   ax = fig.add_subplot(111, projection='3d')
   ax.scatter3D(*zip(coords, transformed_coords))
   ax.plot(*zip(coords, transformed_coords), marker="<")
   ax.plot(*zip((0, 0, 0), transformed_coords))
   ax.plot(*zip((0, 0, 0), coords))


if __name__ == "__main__":
     main()
```

OUTPUT:

Enter the coordinates in spaced integers: 4 2 6
Enter the angle of rotation(in radians): 0.758398
Enter the axis along which you want to rotate: 2
initial: [1, 2, 3]
final: [1, -0.7071132745559481, 3.535532607252656]