UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FEELT – FACULDADE DE ENGENHARIA ELÉTRICA
ENGENHARIA DE COMPUTAÇÃO

LUCAS ALBINO MARTINS
12011ECP022

ROBÓTICA: TRABALHO 08 (Transformações Homogêneas) – Lista de Exercicios 02

UBERLÂNDIA
2021

1) Um sistema de referência móvel UVW foi rotacionado de 90 graus em torno do eixo Z do sistema de referênciaglobal XYZ. Calcular as coordenadas do vetor rXYZ quando rUVW = [-3, 4, -11].

Rot(Z,90º)= $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ x $\begin{pmatrix} -3 \\ 4 \\ -11 \\ 1 \end{pmatrix}$ = $\begin{pmatrix} -3 \\ 4 \\ -11 \\ 1 \end{pmatrix}$

2) Calcular o vetor resultante da translação do vetor rXYZ = [4, 4, 11] segundo o vetor p=[6, 3, 8].

V = trans(6,3,8) = $\begin{pmatrix} 1 & 0 & 0 & 6 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ x $\begin{pmatrix} 4 \\ 4 \\ 11 \\ 1 \end{pmatrix}$ = $\begin{pmatrix} 10 \\ 7 \\ 19 \\ 1 \end{pmatrix}$

3) Um sistema de referência móvel UVW foi transladado um vetor p=[8, -4, 5] e, posteriormente, rotacionado de 90 graus em torno do eixo X do sistema de referênciaglobal XYZ. Calcular as coordenadas do vetor rXYZ sendoque rUVW = [7, 3,12].

T1= trans(8,-4,5) = $\begin{pmatrix} 1 & 0 & 0 & 8 \\ 0 & 1 & 0 & -4 \\ 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

T2 = Rot(90º)= $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

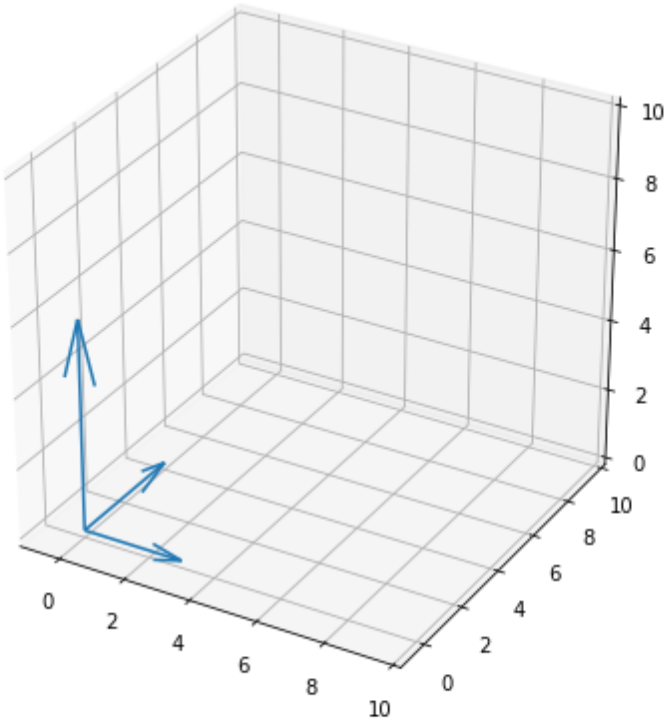
T1.T2 = trans(8,-4,5).Rot(90º)

T1.T2 = $\begin{pmatrix} 1 & 0 & 0 & 8 \\ 0 & 1 & 0 & -4 \\ 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ x $\begin{pmatrix} 7 \\ 3 \\ 12 \\ 1 \end{pmatrix}$ = $\begin{pmatrix} 15 \\ -1 \\ 17 \\ 1 \end{pmatrix}$

4) Translação: escrever um programa de computador querecebe as coordenadas do ponto P a ser transladado. A translação deve ser de 2 unidades em x, 3 unidades em y, e 5 unidades em z. Apresentar a matriz de transformação e plotar as novas coordenadas.

Saída:

```
Digite o valor da coordenado do ponto para x: 1
Digite o valor da coordenado do ponto para y: 1
Digite o valor da coordenado do ponto para z: 1
Matriz transformação:  [[1 0 0 1]
 [0 1 0 1]
 [0 0 1 1]
 [0 0 0 1]]
Posição a ser transladada:  [[2]
 [3]
 [5]]
Nova posição: [[3]
 [4]
 [6]]
```



Código programa em Python utilizando google colab.

```
import numpy as np
import math
import matplotlib.pyplot as plt
```

```
# Translação: escrever um programa de computador querecebe as coord
enadas
#  do ponto P a ser transladado. A translação deve ser de 2 unidade
s em x,
#  3 unidades  em y, e 5 unidades em z. Apresentar a matriz de tran
sformação
# e  plotar as novas coordenadas.



# Usuário entra com os valores

p0x = int(input('Digite o valor da coordenado do ponto para x: '))
p0y = int(input('Digite o valor da coordenado do ponto para y: '))
p0z = int(input('Digite o valor da coordenado do ponto para z: '))

# Matriz transformação homogênea

T = np.array([[1,0,0,p0x],[0,1,0,p0y],[0,0,1,p0z],[0,0,0,1]], dtype
=int)
print('Matriz transformação: ',T)

# Deslocamento em vetor(matriz)

P1 = np.array([[2],[3],[5],[1]], dtype=int)

# Multiplicação das matrizes
P2 = np.dot(T,P1)
P1 = P1[0:3]
print('Posição a ser transladada: ',P1)
P2
P2 = P2[0:3]
# Imprime a nova posição vetorial
print('Nova posição:',P2)

# Plotar coordenadas
x = 1
y = 2
z = 3
x,y = np.mgrid[-1:1:500J,-1:1:500J]
z = 1 + 23

fig = plt.figure(figsize=(7,7))
ax = plt.axes(projection = '3d')
ax.set_xlim([-1,10])
ax.set_ylim([-1,10])
ax.set_zlim([0,10])

inicio = [0,0,0]
```

```
ax.quiver(inicio[0],inicio[1],inicio[2], P2[0],0,0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,P2[1],0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,0,P2[2])
plt.show()
```

5) Translações sucessivas: escrever um programa que recebe x0, y0 e z0 da matriz de transformação T0, x1, y1 e z1 da matriz de transformação T1. Realizar a transformação sucessiva T0.T1 e a transformação T1.T0 sobre o ponto P=(3,4,7). Apresentar os resultados destas duas transformações sucessivas.

Saída:

```
Digite o valor da coordenado do ponto para x0: 1
Digite o valor da coordenado do ponto para y0: 1
Digite o valor da coordenado do ponto para z0: 1
Matriz transformação:  [[1 0 0 1]
 [0 1 0 1]
 [0 0 1 1]
 [0 0 0 1]]
Digite o valor da coordenado do ponto para x1: 2
Digite o valor da coordenado do ponto para y2: 2
Digite o valor da coordenado do ponto para z3: 2
Matriz transformação:  [[1 0 0 2]
 [0 1 0 2]
 [0 0 1 2]
 [0 0 0 1]]
Ponto de deslocamento: [[3]
 [4]
 [5]
 [7]]
Transformação sucessiva T0.T1:  [[1 0 0 3]
 [0 1 0 3]
 [0 0 1 3]
 [0 0 0 1]]
Transformação sucessiva (T1.T0) no ponto P=[3,4,7]:  [[24]
 [25]
 [26]
 [ 7]]
```

Código programa em Python utilizando google colab.

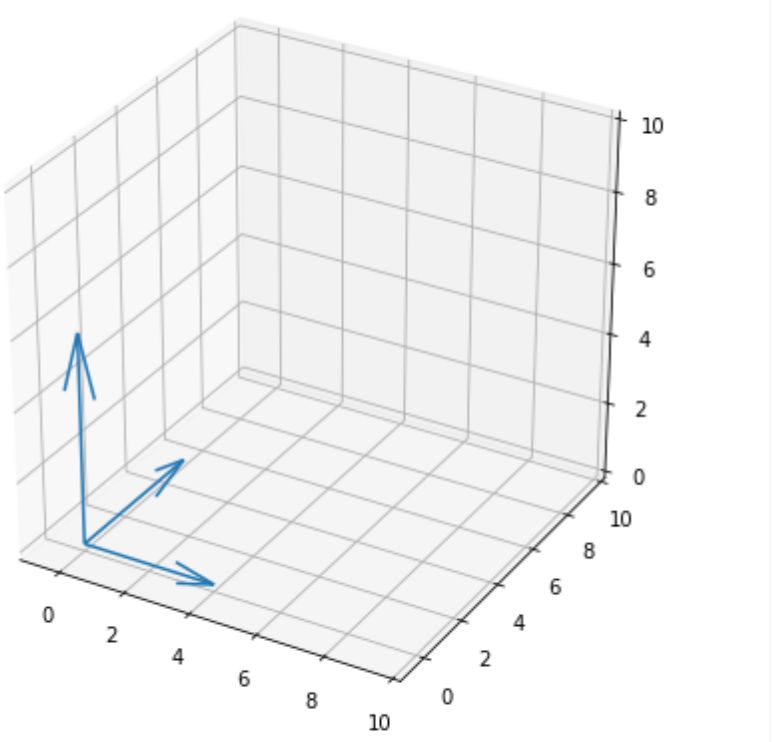
```
import numpy as np
import math
import matplotlib.pyplot as plt



# Translações sucessivas: escrever um programa que recebe x0,  y0
e z0
# da matriz de transformação T0, x1, y1 e z1  da matriz de transfor
mação T1.
```

```
# Realizar a transformação sucessiva T0.T1 e a transformação T1.T0
sobre o ponto  P=(3,4,7).
# Apresentar os resultados destas duas  transformações sucessiva.


# Usuário entra com os valores

# Matriz T0

x0 = int(input('Digite o valor da coordenado do ponto para x0: '))
y0 = int(input('Digite o valor da coordenado do ponto para y0: '))
z0 = int(input('Digite o valor da coordenado do ponto para z0: '))

# Matriz transformação homogênea T0

T0 = np.array([[1,0,0,x0],[0,1,0,y0],[0,0,1,z0],[0,0,0,1]], dtype=i
nt)
print('Matriz transformação: ',T0)

# Matriz T1

x1 = int(input('Digite o valor da coordenado do ponto para x1: '))
y1 = int(input('Digite o valor da coordenado do ponto para y2: '))
z1 = int(input('Digite o valor da coordenado do ponto para z3: '))

# Matriz transformação homogênea T1

T1 = np.array([[1,0,0,x1],[0,1,0,y1],[0,0,1,z1],[0,0,0,1]], dtype=i
nt)
print('Matriz transformação: ',T1)

# Deslocamento em vetor(matriz)

P1 = np.array([[3],[4],[5],[7]], dtype=int)
print('Ponto de deslocamento:',P1)

# Multiplicação das matrizes
T2 = np.dot(T0,T1)
print('Transformação sucessiva T0.T1: ',T2)
T2 = np.dot(T1,T0)
T3 = np.dot(T2,P1)

# Imprime a nova posição vetorial
print('Transformação sucessiva (T1.T0) no ponto P=[3,4,7]: ',T3)
```

6) Rotação: Escrever um programa que recebe o ângulo teta a ser rotacionado em torno do eixo z e calcula as novas coordenadas do ponto P=(4,5,6) após sua rotação. Plotar as novas coordenadas.

Saída:

```
Digite o valor de teta: 0
Matriz de rotação z:  [[ 1. -0.  0.  0.]
 [ 0.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]
Posição a ser transladada:  [[4]
 [5]
 [6]]
Nova posição: [[4.]
 [5.]
 [6.]]
```



Código programa em Python utilizando google colab.

```
import numpy as np
import math
import matplotlib.pyplot as plt



# Rotação: Escrever um programa que recebe o ângulo teta a ser rota
cionado em torno do eixo z
# e  calcula as novas coordenadas do  ponto P=(4,5,6) após sua rota
ção. Plotar as novas  coordenadas.
```

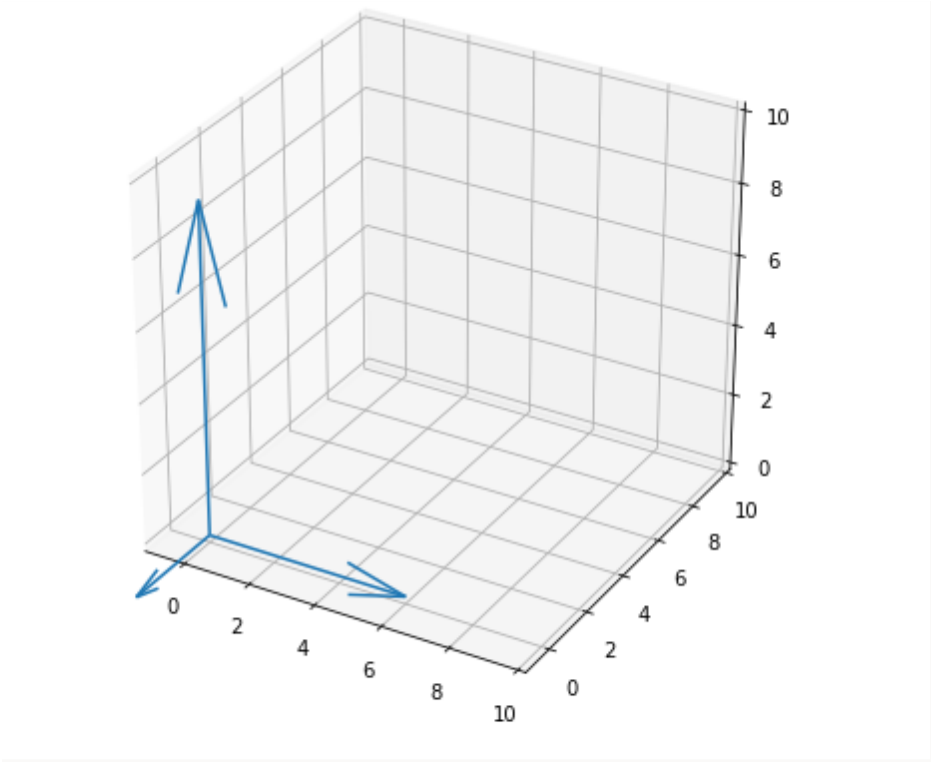
```
# Usuário entra com os valores

teta = float(input('Digite o valor de teta: '))


# Matriz de rotação z

T = np.array([[np.cos(teta),-
np.sin(teta),0,0],[np.sin(teta),np.cos(teta),0,0],[0,0,1,0],[0,0,0,
1]], dtype=float)
print('Matriz de rotação z: ',T)

# Vetor de deslocamento

P1 = np.array([[4],[5],[6],[1]], dtype=int)

# Multiplicação das matrizes
P2 = np.dot(T,P1)
P1 = P1[0:3]
print('Posição a ser transladada: ',P1)
P2
P2 = P2[0:3]
# Imprime a nova posição vetorial
print('Nova posição:',P2)

# Plotar coordenadas
x = 1
y = 2
z = 3
x,y = np.mgrid[-1:1:500J,-1:1:500J]
z = 1 + 23

fig = plt.figure(figsize=(7,7))
ax = plt.axes(projection = '3d')
ax.set_xlim([-1,10])
ax.set_ylim([-1,10])
ax.set_zlim([0,10])

inicio = [0,0,0]
ax.quiver(inicio[0],inicio[1],inicio[2], P2[0],0,0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,P2[1],0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,0,P2[2])
plt.show()
```

7) Rotações sucessivas: escrever um programa que recebe o ângulo teta da primeira rotação R1, e o ângulo alfa da segunda rotação R2 a serem aplicados sobre o ponto P=(6,6,8). Plotar as novas coordenadas.

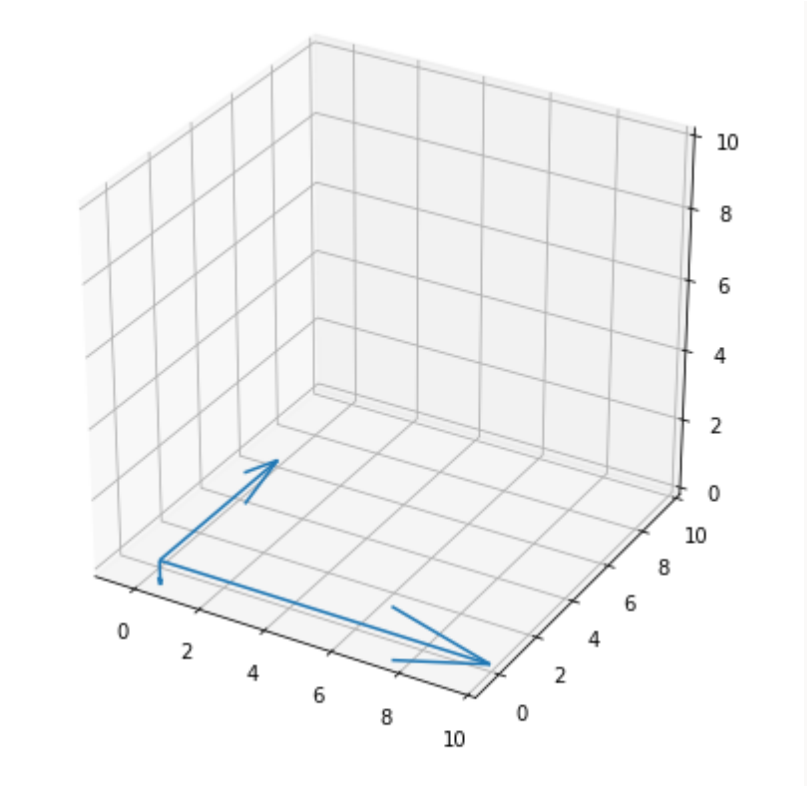
Saída:

```
Digite o valor de teta para R1: 0
Digite o valor de teta para R2: 1
Matriz de rotação x:  [[ 1.  0.  0.  0.]
 [ 0.  1. -0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]
Matriz de rotação x:  [[ 1.          0.          0.          0.
]
 [ 0.          0.54030231 -0.84147098  0.        ]
 [ 0.          0.84147098  0.54030231  0.        ]
 [ 0.          0.          0.          1.        ]]
Matriz de rotação y:  [[ 1.  0.  0.  0.]
 [ 0.  1.  0.  0.]
 [-0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]
Matriz de rotação y:  [[ 0.54030231  0.          0.84147098  0.
]
 [ 0.          1.          0.          0.        ]
 [-0.84147098  0.          0.54030231  0.        ]
 [ 0.          0.          0.          1.        ]]
Matriz de rotação z:  [[ 1. -0.  0.  0.]
 [ 0.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]
Matriz de rotação z:  [[ 0.54030231 -0.84147098  0.          0.
]
 [ 0.84147098  0.54030231  0.          0.        ]
 [ 0.          0.          1.          0.        ]
 [ 0.          0.          0.          1.        ]]
Posição a ser transladada:  [[6]
 [6]
 [8]]
Nova posição para rotação em x [[ 6.        ]
 [-3.48995404]
 [ 9.37124436]]
Nova posição para rotação em y: [[ 9.97358171]
 [ 6.        ]
 [-0.72640746]]
Nova posição para rotação em z [[-1.80701207]
 [ 8.29063974]
 [ 8.        ]]
```

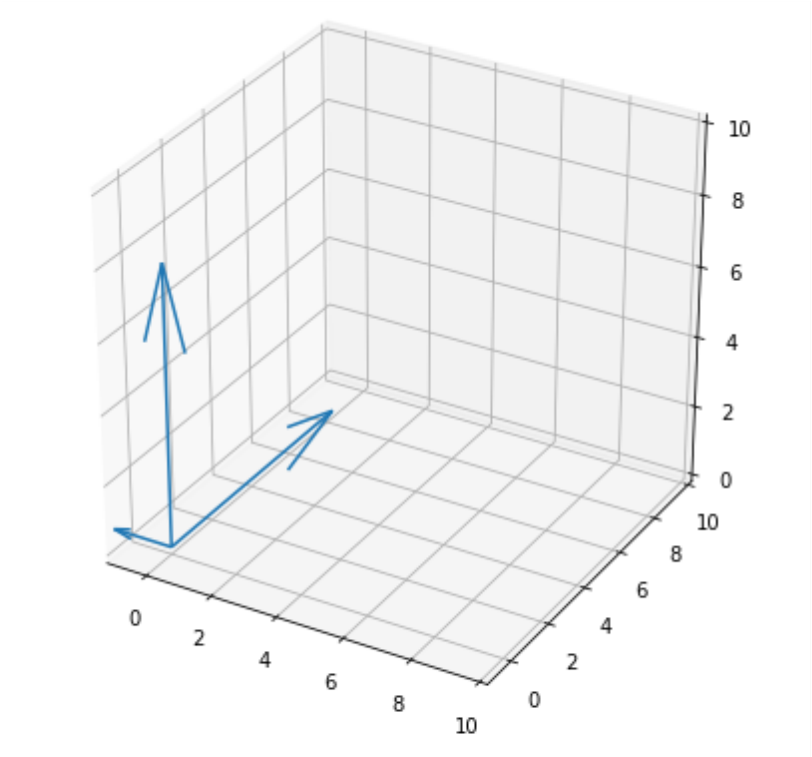
Rotação em X



Rotação em Y:



Rotação em Z:



Código programa em Python utilizando google colab.

```
import numpy as np
import math
import matplotlib.pyplot as plt



# Rotações sucessivas: escrever um programa que recebe o ângulo tet
a da  primeira rotação R1,
# e o ângulo alfa da segunda rotação  R2 a serem aplicados  sobre o
 ponto P=(6,6,8). Plotar as novas coordenadas.



# Usuário entra com os valores

teta1 = float(input('Digite o valor de teta para R1: '))
teta2 = float(input('Digite o valor de teta para R2: '))

# Matriz de rotação x
```

```
R1 = np.array([[1,0,0,0],[0,np.cos(teta1),-
np.sin(teta1),0],[0,np.sin(teta1),np.cos(teta1),0],[0,0,0,1]], dtyp
e=float)
print('Matriz de rotação x: ',R1)

R2 = np.array([[1,0,0,0],[0,np.cos(teta2),-
np.sin(teta2),0],[0,np.sin(teta2),np.cos(teta2),0],[0,0,0,1]], dtyp
e=float)
print('Matriz de rotação x: ',R2)


# Matriz de rotação y

R3 = np.array([[np.cos(teta1),0,np.sin(teta1),0],[0,1,0,0],[-
np.sin(teta1),0,np.cos(teta1),0,],[0,0,0,1]], dtype=float)
print('Matriz de rotação y: ',R3)

R4 = np.array([[np.cos(teta2),0,np.sin(teta2),0],[0,1,0,0],[-
np.sin(teta2),0,np.cos(teta2),0,],[0,0,0,1]], dtype=float)
print('Matriz de rotação y: ',R4)


# Matriz de rotação z

R5 = np.array([[np.cos(teta1),-
np.sin(teta1),0,0],[np.sin(teta1),np.cos(teta1),0,0],[0,0,1,0],[0,0
,0,1]], dtype=float)
print('Matriz de rotação z: ',R5)

R6 = np.array([[np.cos(teta2),-
np.sin(teta2),0,0],[np.sin(teta2),np.cos(teta2),0,0],[0,0,1,0],[0,0
,0,1]], dtype=float)
print('Matriz de rotação z: ',R6)

# Vetor de deslocamento

P1 = np.array([[6],[6],[8],[1]], dtype=int)

# Multiplicação das matrizes

P2 = np.dot(R1,R2)
P3 = np.dot(R3,R4)
P4 = np.dot(R5,R6)
P5 = np.dot(P2,P1)
P6 = np.dot(P3,P1)
P7 = np.dot(P4,P1)
P1 = P1[0:3]
print('Posição a ser transladada: ',P1)
P5
```

```
P5 = P5[0:3]
# Imprime a nova posição vetorial
print('Nova posição para rotação em x',P5)

P6
P6 = P6[0:3]
# Imprime a nova posição vetorial
print('Nova posição para rotação em y:',P6)

P7
P7 = P7[0:3]
# Imprime a nova posição vetorial
print('Nova posição para rotação em z',P7)

# Plotar coordenadas rotação em x
x = 1
y = 2
z = 3
x,y = np.mgrid[-1:1:500J,-1:1:500J]
z = 1 + 23

fig = plt.figure(figsize=(7,7))
ax = plt.axes(projection = '3d')
ax.set_xlim([-1,10])
ax.set_ylim([-1,10])
ax.set_zlim([0,10])

inicio = [0,0,0]
ax.quiver(inicio[0],inicio[1],inicio[2], P5[0],0,0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,P5[1],0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,0,P5[2])
plt.show()

# Plotar coordenadas rotação em y
x = 1
y = 2
z = 3
x,y = np.mgrid[-1:1:500J,-1:1:500J]
z = 1 + 23

fig = plt.figure(figsize=(7,7))
ax = plt.axes(projection = '3d')
ax.set_xlim([-1,10])
ax.set_ylim([-1,10])
ax.set_zlim([0,10])

inicio = [0,0,0]
ax.quiver(inicio[0],inicio[1],inicio[2], P6[0],0,0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,P6[1],0)
```

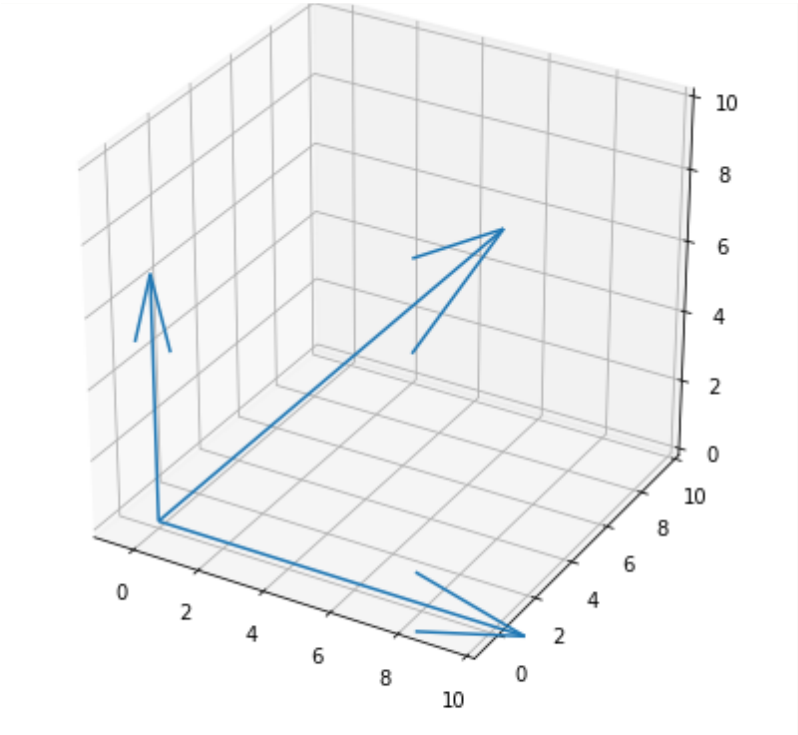
```
ax.quiver(inicio[0],inicio[1],inicio[2], 0,0,P6[2])
plt.show()

# Plotar coordenadas rotação em z
x = 1
y = 2
z = 3
x,y = np.mgrid[-1:1:500J,-1:1:500J]
z = 1 + 23

fig = plt.figure(figsize=(7,7))
ax = plt.axes(projection = '3d')
ax.set_xlim([-1,10])
ax.set_ylim([-1,10])
ax.set_zlim([0,10])

inicio = [0,0,0]
ax.quiver(inicio[0],inicio[1],inicio[2], P7[0],0,0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,P7[1],0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,0,P7[2])
plt.show()
```

8) Translação e rotação sucessivas: escrever um programa que recebe x e y da matriz de translação T e o ângulo teta da matriz de rotação R a serem aplicados sobre o ponto P=(4,5,7). Plotar o resultado da transformação sucessiva T.R e o resultado da transformação sucessiva R.T.

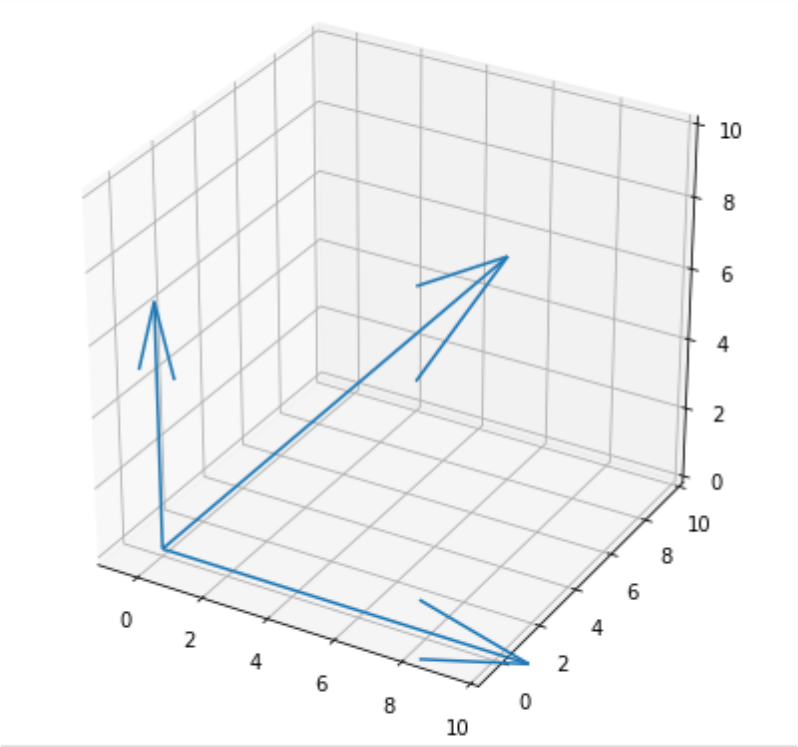
Saída:

```
Digite o valor da coordenado do ponto para x: 1
Digite o valor da coordenado do ponto para y: 2
Digite o valor de teta: 0
Matriz transformação:  [[1 0 1]
 [0 1 2]
 [0 0 1]]
Matriz de rotação:  [[ 1. -0.  0.]
 [ 0.  1.  0.]
 [ 0.  0.  1.]]
Posição a ser transladada:  [[4]
 [5]
 [7]]
Nova posição: [[11.]
 [19.]
 [ 7.]]
Nova posição: [[11.]
 [19.]
    [ 7.]]
```

T.R no ponto:



R.T no ponto:



Código programa em Python utilizando google colab.

```
import numpy as np
import math
```

```
import matplotlib.pyplot as plt


# Translação e rotação sucessivas: escrever um programa que recebe
x e y  da matriz de translação T
# e o ângulo teta da matriz de rotação R a serem  aplicados sobre
o ponto P=(4,5,7).
# Plotar o resultado da transformação  sucessiva T.R e o resultado
da transformação sucessiva R.T



# Usuário entra com os valores

p0x = int(input('Digite o valor da coordenado do ponto para x: '))
p0y = int(input('Digite o valor da coordenado do ponto para y: '))
teta = float(input('Digite o valor de teta: '))



# Matriz transformação homogênea

T0 = np.array([[1,0,p0x],[0,1,p0y],[0,0,1]], dtype=int)
print('Matriz transformação: ',T0)

# Matriz de rotação z

T1 = np.array([[np.cos(teta),-
np.sin(teta),0],[np.sin(teta),np.cos(teta),0],[0,0,1]], dtype=float
)
print('Matriz de rotação: ',T1)


# Deslocamento em vetor(matriz)

P1 = np.array([[4],[5],[7]], dtype=int)

# Multiplicação das matrizes
P2 = np.dot(T0,T1)
P3 = np.dot(P2,P1)
P4 = np.dot(T1,T0)
P5 = np.dot(P4,P1)

P1 = P1[0:3]
print('Posição a ser transladada: ',P1)
P3

# Imprime a nova posição vetorial
print('Nova posição:',P3)
```

```
P5

# Imprime a nova posição vetorial
print('Nova posição:',P5)

# Plotar coordenadas T.R no ponto
x = 1
y = 2
z = 3
x,y = np.mgrid[-1:1:500J,-1:1:500J]
z = 1 + 23

fig = plt.figure(figsize=(7,7))
ax = plt.axes(projection = '3d')
ax.set_xlim([-1,10])
ax.set_ylim([-1,10])
ax.set_zlim([0,10])

inicio = [0,0,0]
ax.quiver(inicio[0],inicio[1],inicio[2], P3[0],0,0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,P3[1],0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,0,P3[2])
plt.show()

# Plotar coordenadas R.T no ponto
x = 1
y = 2
z = 3
x,y = np.mgrid[-1:1:500J,-1:1:500J]
z = 1 + 23

fig = plt.figure(figsize=(7,7))
ax = plt.axes(projection = '3d')
ax.set_xlim([-1,10])
ax.set_ylim([-1,10])
ax.set_zlim([0,10])

inicio = [0,0,0]
ax.quiver(inicio[0],inicio[1],inicio[2], P5[0],0,0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,P5[1],0)
ax.quiver(inicio[0],inicio[1],inicio[2], 0,0,P5[2])
plt.show()
```