```
#part 1
import numpy as np
import matplotlib.pyplot as plt

# dinedefine ko po dito yung matrices
H = (1/np.pi) * np.array([[1, 1], [1, -1]])
Y = np.array([[0, -1], [1, 0]])
O = np.array([[1, 0], [1, 1]])



#part 1a

answer_1a = np.dot(H,Y)
magnitude1a = np.linalg.norm(answer_1a, 'fro')
eigenvalues_1a, _ =np.linalg.eig(answer_1a)
angle_1a = np.angle(eigenvalues_1a)
print(answer_1a)



#part 1b
answer_1b = np.dot(Y,H)
magnitude1b = np.linalg.norm(answer_1b)
eigenvalues_1b, _ =np.linalg.eig(answer_1b)
angle_1b = np.angle(eigenvalues_1b)
print("**********************************")
print(answer_1b)


#part1c
answer_1c = np.dot(H,H)
magnitude1c = np.linalg.norm(answer_1c)
eigenvalues_1c, _ =np.linalg.eig(answer_1c)
angle_1c = np.angle(eigenvalues_1c)
print("**********************************")
print(answer_1c)

#part1d
answer_1d= np.dot(answer_1a,O)
magnitude1d = np.linalg.norm(answer_1d)
eigenvalues_1d, _ =np.linalg.eig(answer_1d)
angle_1d = np.angle(eigenvalues_1d)
print("**********************************")
print(answer_1d)


#part1e
answer_2e= np.dot (H,O)
answer_1e= np.dot(answer_1a,answer_2e)
magnitude1e = np.linalg.norm(answer_1e)
eigenvalues_1e, _ =np.linalg.eig(answer_1e)
angle_1e = np.angle(eigenvalues_1e)
print("**********************************")
print(answer_1e)
```

```
[[0.+0.31830989j 0.-0.31830989j]
 [0.-0.31830989j 0.-0.31830989j]]
**********************************
[[0.-0.31830989j 0.+0.31830989j]
 [0.+0.31830989j 0.+0.31830989j]]
**********************************
[[2.02642367e-01 5.31624302e-18]
 [5.31624302e-18 2.02642367e-01]]
**********************************
[[0.+0.j         0.-0.31830989j]
 [0.-0.63661977j 0.-0.31830989j]]
**********************************
[[0.+2.02642367e-01j 0.+2.02642367e-01j]
 [0.-2.02642367e-01j 0.-5.31624302e-18j]]
```

```
#PLOTTING

vectors = np.array([[1, 0], [0, 1]])
```

```
transformed_HY = np.dot(answer_1a, vectors)
transformed_YH = np.dot(answer_1b, vectors)
transformed_HH = np.dot(answer_1c, vectors)
transformed_YHO = np.dot(answer_1d, vectors)
transformed_HYHO = np.dot(answer_1e, vectors)



# Plot the vectors
plt.figure(figsize=(18, 4))

# Transformed vectors for 1a
plt.subplot(151)
plt.quiver([0, 0], [0, 0], transformed_HY[:, 0], transformed_HY[:, 1], angles='xy', scale_units='xy', scale=1, color=['b', 'r'])
plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.title('Transformed Vectors for H · Y')

# Transformed vectors for part 1b
plt.subplot(152)
plt.quiver([0, 0], [0, 0], transformed_YH[:, 0], transformed_YH[:, 1], angles='xy', scale_units='xy', scale=1, color=['b', 'r'])
plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.title('Transformed Vectors for Y · H')

# Transformed vectors for part 1c
plt.subplot(153)
plt.quiver([0, 0], [0, 0], transformed_HH[:, 0], transformed_HH[:, 1], angles='xy', scale_units='xy', scale=1, color=['b', 'r'])
plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.title('Transformed Vectors for H · H')

# Transformed vectors for part 1d
plt.subplot(154)
plt.quiver([0, 0], [0, 0], transformed_YHO[:, 0], transformed_YHO[:, 1], angles='xy', scale_units='xy', scale=1, color=['b', 'r'])
plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.title('Transformed Vectors for Y · H · O')

# Transformed vectors for part 1e
plt.subplot(155)
plt.quiver([0, 0], [0, 0], transformed_HYHO[:, 0], transformed_HYHO[:, 1], angles='xy', scale_units='xy', scale=1, color=['b', 'r'])
plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.title('Transformed Vectors for H · Y · H · O')

plt.tight_layout()
plt.show()


#im having troubles po with this part of plotting sir pero all in all po na answer ko naman po im just having troubles plotting
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/IPython/core/formatters.py in __call__(self, obj)
    339                 pass
    340             else:
--> 341                 return printer(obj)
    342             # Finally look for special method names
    343             method = get_real_method(obj, self.print_method)

                                16 frames
/usr/local/lib/python3.10/dist-packages/matplotlib/transforms.py in transform_affine(self, points)
   1846             tpoints = affine_transform(points.data, mtx)
   1847             return np.ma.MaskedArray(tpoints, mask=np.ma.getmask(points))
-> 1848         return affine_transform(points, mtx)
   1849
```

wer using the internet po kase yung cell sa taas nag kaka error po whenver I try to plot

ıction from stack overflow po so that i can use small values instead of having to divide by zero po(yun po kase yung error na lumalabas saken)
ɔlot nagkakaroon po ng error sa matplotlib kaya i got this function po sa i can plot

```
l]

', angles='xy', scale=1, color=['b', 'r'])
```

```
l]

', angles='xy', scale=1, color=['b', 'r'])
```

```
l]

', angles='xy', scale=1, color=['b', 'r'])
```

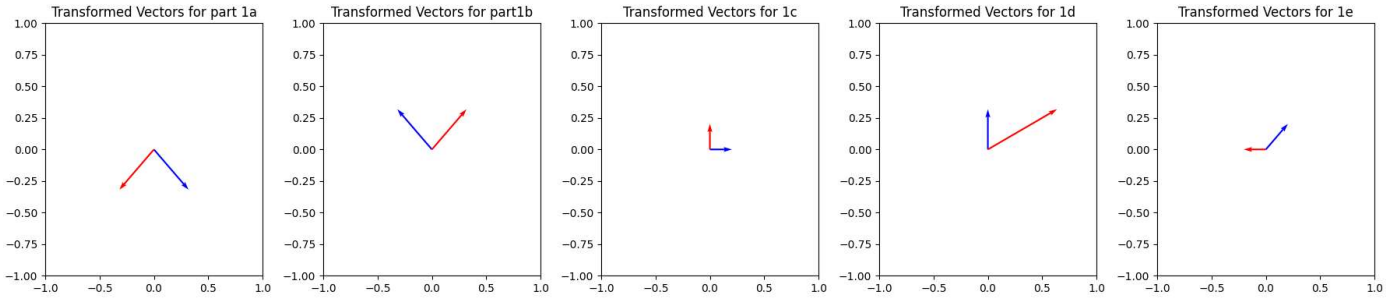
```
 1]

', angles='xy', scale=1, color=['b', 'r'])
```

```
:, 1]

', angles='xy', scale=1, color=['b', 'r'])
```

```
to just wanted to be honest po and show na I was having trouble with the questions in terms
 ko naman po ng maayos sa plotting lang po talaga
```




```
#part 2
# i used numpy po to get the determinants
determinant_H = np.linalg.det(H)
determinant_Y = np.linalg.det(Y)

print(f"Determinant of H: {determinant_H}")
print(f"Determinant of Y: {determinant_Y}")
```

```
Determinant of H: -0.20264236728467558
Determinant of Y: (-1+0j)
```

```
#part 3

# i defined po the matrices 1 as A and 2 as B
A = np.array([[5, 0, 0], [0, 5, 0], [0, 0, 5]]) * np.array([[1, 0, 5], [2, 7, 6], [6, 4, 7]])
B = np.array([[1, 2, 6], [3, 15, 4], [2, 10, 3]]) * np.array([[5, 2, 4], [6, 2, 4], [0, 1, 1]])

# i used numpy linalg.det po to find the determinant
det_3A = np.linalg.det(A)
det_3B = np.linalg.det(B)

#to print the answers
print(det_3A)
print("******************")
print(det_3B)
print("******************")
# Checking if the determinants are non-zero
if det_3A != 0 and det_3B != 0: #gumamit po ako sir ng if else statement to show if linearly independet or not po
    print("Both linear transformations A and B are linearly independent.")
else:
    print("At least one of the linear transformations A and B is linearly dependent.")
```

```
6124.999999999999
******************
3753.9999999999995
******************
Both linear transformations A and B are linearly independent.
```

```
#part 4

# dinefine ko po yung  time vector as T and amplitude vector as G by turning it po into an array using numpy
T = np.array([0, np.pi/4, np.pi/2, 3*np.pi/4, np.pi])#dito po ginamit ko lang din yung numpy.pi as pi based sa given
```

```
G = np.array([5, 3, 0, -3, 5])

# i got help po from youtube to plot the signal and plot it nicely po
plt.plot(T, G, marker='o', linestyle='-')
plt.xlabel('Time (T)')
plt.ylabel('Amplitude (G)')
plt.title('Signal Plot')
plt.grid(True)

# dito po nag Show ng answer na po ako
plt.show()
```