Lab 4

1. Calculate the Fourier matrix for n = 4 in Python using the fast Fourier transform.

```
import numpy as np
from numpy.fft import fft, ifft
```

```
n = 4
F_n_4 = fft(np.eye(n))
print("Fourier matrix F_n_4:\n", F_n_4)
```

```
Fourier matrix F_n_4:
 [[ 1.+0.j  1.+0.j  1.+0.j  1.+0.j]
 [ 1.+0.j  0.-1.j -1.+0.j  0.+1.j]
 [ 1.+0.j -1.+0.j  1.+0.j -1.+0.j]
 [ 1.+0.j  0.+1.j -1.+0.j  0.-1.j]]
```

+ Code + Text

2. Calculate the discrete Fourier transform of the vector (0, 1, 2).

```
vector = np.array([0, 1, 2])
DFT_vector = fft(vector)
print("DFT of vector (0, 1, 2):\n", DFT_vector)
```

```
DFT of vector (0, 1, 2):
 [ 3. +0.j        -1.5+0.8660254j -1.5-0.8660254j]
```

3. Find the cyclic convolution of the vectors a = (0, 1, 2) and b = (3, 1, 2) in Python using the fast Fourier transform. Compare with the results calculated by hand in Lab 4.

```
a = np.array([0, 1, 2])
b = np.array([3, 1, 2])

cyclic_conv_ab_fft = ifft(fft(a) * fft(b)).real
print("Cyclic convolution of a and b using FFT:\n", cyclic_conv_ab_fft)
```

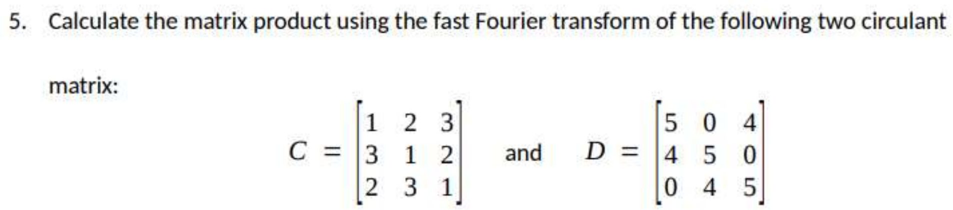
```
Cyclic convolution of a and b using FFT:
 [4. 7. 7.]
```

4. Find the cyclic convolution of the vectors x = (0, 1, 0, 1) and y = (0, 1, 2, 3) in Python using the fast Fourier transform. Compare with the results calculated by hand in Lab 4.

```
x = np.array([0, 1, 0, 1])
y = np.array([0, 1, 2, 3])


cyclic_conv_xy_fft = ifft(fft(x) * fft(y)).real
print("Cyclic convolution of x and y using FFT:\n", cyclic_conv_xy_fft)
```

```
Cyclic convolution of x and y using FFT:
 [4. 2. 4. 2.]
```



5. Calculate the matrix product using the fast Fourier transform of the following two circulant matrix:

$$C = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 2 & 3 & 1 \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} 5 & 0 & 4 \\ 4 & 5 & 0 \\ 0 & 4 & 5 \end{bmatrix}$$

```
C = np.array([[1, 2, 3],
              [3, 1, 2],
              [2, 3, 1]])

D = np.array([[5, 0, 4],
              [4, 5, 0],
              [0, 4, 5]])

C_fft = fft(C, axis=0)
D_fft = fft(D, axis=0)
product_CD_fft = ifft(C_fft * D_fft, axis=0).real
product_CD_fft = np.roll(product_CD_fft, shift=-1, axis=0)
print("Product of circulant matrices C and D using FFT:\n", product_CD_fft)
```

```
Product of circulant matrices C and D using FFT:
 [[19. 22. 13.]
 [22. 13. 19.]
 [13. 19. 22.]]
```