EXPT NO 4: Discrete Fourier Transform

Aim: The aim of this experiment is to study Fourier transform the DT signal.

Input Specifications:

1. **Length of first Signal N**

Objective

1. Develop a function to perform DFT of N point signal
2. Calculate DFT of a DT signal
3. Calculate the number of real multi. And real additions required to find DFT.

Problem Definition

– Take any four-point sequence x[n].

– Find DFT X[k].

– Compute number of real and complex multiplications and additions required to find X[k].

– Append the input signal by four zeros.

– Find DFT and plot magnitude Spectrum

– Give your conclusion

Experimentation and Result Analysis

Sample example

1. To find DFT of 4 point sequence

- Input x[n] = {1, 2, 3, 4} Length L= 4
- Output X[k] = {10, -2+2j, -2, -2-2j}
- Total no of Complex Multiplications = N^2 = 16
- Total no of Complex additions = N(N-1) = 12
- Total no of Real Multiplications = 16
- Total no of Real Additions = 12

2. To find DFT off zero padded signal

- Input x[n] = {1, 2, 3, 4, 0, 0, 0, 0} Length L=8
- Output X[k] = {10, -0.41-7.24j, -2+2j, 2.41-1.24j,

  -2, 2.41+1.24j, -2-2j, -0.41+7.24j}

3. To find DFT off zero padded signal

- Input x[n] = {1, 2, 3, 4, 0, 0, 0, 0, 0, 0, 0, 0}
- Length L=12
- Output X[k] = {10, 4.23-7.6j, -3.5-4.33j, -2+2j, 2.5+0.87j, 0.77-2.4j,

  -2, 0.77+2.4j, 2.5-0.87j, -2-2j, -3.5+4.33j, 4.23+7.6j}

Conclusion:

1. DFT coefficients are defined in frequency domain.

2. DFT gives discrete and periodic spectrum in frequency domain.

3. As the length of signal is increased by zero padding, frequency spacing decreases, resolution of the spectrum increases and so the quality of spectrum increases.

Postlab:

1. **Compute 4 point DFT of x(n)={ 1,2,3,4} Using DFT Equation and Twiddle Matrix Method**
2. **Compute 8 point DFT of { 1,0,2,0,3,0,4,0} Using DFT Equation and Twiddle Matrix Method**
3. **List any 5 DFT properties**

Name: Flavion D'sa   Roll No.: 7371

Program: Discrete Fourier Transform

Code:

```
import cmath
import math

pi2 = math.pi * 2.0

def DFT(fnList):
    N = len(fnList)
    FmList = []
    for m in range(N):
        Fm = 0.0
        for n in range(N):
            Fm  += fnList[n] * cmath.exp(-1j*pi2*m*n/N)
        FmList.append(Fm)
    print("x(k) : ")
    return FmList

def InvDFT(FmList):
    N = len(FmList)
    fnList = []
    for n in range(N):
        fn = 0.0
        for m in range(N):
            fn += FmList[m] * cmath.exp(1j*pi2*m*n/N)
        fnList.append(fn/N)
    print("x(n) : ")
    return fnList

if __name__ == '__main__':
    print("===== DISCRETE FOURIER TRANSFORM =====")
    print("Enter samples :")
    x = [complex(temp) for temp in input().split()]
    print("\n1.Fourier Transform.\n2.Inverse Fourier Transform")
    choice = lambda ch : DFT(x) if ch == 1 else InvDFT(x)
    print("======================================")
    res = choice(int(input()))
    print('['+', '.join('{:.1f}'.format(f) for f in list(res))+']')
```

1. Input: x[n] = {1, 2, 3, 4}

```
===== DISCRETE FOURIER TRANSFORM =====
Enter samples :
1 2 3 4

1.Fourier Transform.
2.Inverse Fourier Transform
======================================
1
x(k) :
[10.0+0.0j, -2.0+2.0j, -2.0-0.0j, -2.0-2.0j]
```

2. Input: x[n] = {1, 2, 3, 4, 0, 0, 0, 0}

```
===== DISCRETE FOURIER TRANSFORM =====
Enter samples :
1 2 3 4 0 0 0 0

1.Fourier Transform.
2.Inverse Fourier Transform
======================================
1
x(k) :
[10.0+0.0j, -0.4-7.2j, -2.0+2.0j, 2.4-1.2j, -2.0-0.0j, 2.4+1.2j, -2.0-2.0j, -
0.4+7.2j]
```

3. Input: x[n] = {1, 2, 3, 4, 0, 0, 0, 0, 0, 0, 0, 0}

```
===== DISCRETE FOURIER TRANSFORM =====
Enter samples :
1 2 3 4 0 0 0 0 0 0 0 0

1.Fourier Transform.
2.Inverse Fourier Transform
======================================
1
x(k) :
[10.0+0.0j, 4.2-7.6j, -3.5-4.3j, -2.0+2.0j, 2.5+0.9j, 0.8-2.4j, -2.0-0.0j,
0.8+2.4j, 2.5-0.9j, -2.0-2.0j, -3.5+4.3j, 4.2+7.6j]
```