Question 2

Part a

In [ ]:
```
import numpy as np
from numpy import sqrt, exp, pi

def DFT(x):
    n = len(x)

    w = np.exp(2.*np.pi*1j/n)

    # intermediate power array (0, 1, 2, .., n)
    k = np.arange(n)

    # 2d array of powers for DFT
    pow = k.reshape((-1,1)) * k.reshape((1,-1))

    # Fourier matrix of dimension n
    F = w**pow

    return 1/sqrt(n) * F @ x

u = np.array([.75, .25, -.25, .25])
v = np.array([1., 0., -.5, 0., 1., 0., -.5, 0.])

y1 = DFT(u)
y2 = DFT(v)

with np.printoptions(precision=2, suppress=True):
    print(y1)
    print(y2)
```

```
[0.5+0.j 0.5-0.j 0. +0.j 0.5-0.j]
[ 0.35+0.j  0.  +0.j  1.06-0.j -0.  +0.j  0.35+0.j  0.  +0.j  1.06-0.j
 -0.  +0.j]
```

Part b

In [ ]:
```
# Part i)
print(f"Verify that y1_0 is real: {np.imag(y1[0]) == 0.}")
print(f"Verify that y2_0 is real: {np.imag(y2[0]) == 0.}")
print("")

# Part ii)
n = len(u)

print("Verify that y1_(n-k) = conj(y1_k):")
for k in range(1, n):
    print(f"k = {k}: {np.isclose(y1[n-k], np.conj(y1[k]))}")

print("")

n = len(v)
print("Verify that y2_(n-k) = conj(y2_k):")
for k in range(1, n):
    print(f"k = {k}: {np.isclose(y2[n-k], np.conj(y2[k]))}")
```

```
Verify that y1_0 is real: True
Verify that y2_0 is real: True

Verify that y1_(n-k) = conj(y1_k):
k = 1: True
k = 2: True
k = 3: True

Verify that y2_(n-k) = conj(y2_k):
k = 1: True
k = 2: True
k = 3: True
k = 4: True
k = 5: True
k = 6: True
k = 7: True
```

Part c

In [ ]:

```python
def inverse_DFT(x,y):
    n = len(x)

    w = np.exp(2.*np.pi*1j/n)

    # intermediate power array (0, 1, 2, .., n)
    k = np.arange(n)

    # 2d array of powers for DFT
    pow = k.reshape((-1,1)) * k.reshape((1,-1))

    # Fourier matrix of dimension n
    F = w**pow

    return 1/sqrt(n) * np.conjugate(F) @ y

idft_u = inverse_DFT(u,y1)

with np.printoptions(precision=2, suppress=True):
    print(f"Check that inverse_DFT(DFT(u)) = u: {np.allclose(idft_u,u)}")
```

```
Check that inverse_DFT(DFT(u)) = u: True
```