Homework 2

(A) Given a grayscale image I

(1) Problem statement

Step 1: Use the dithering matrix D_2 to generate an array D of image size by repeating D_2

$$D_2 = \begin{bmatrix} 0 & 128 & 32 & 160 \\ 192 & 64 & 224 & 96 \\ 48 & 176 & 16 & 144 \\ 240 & 112 & 208 & 80 \end{bmatrix}$$

D

D_2	D_2	D_2	D_2
D_2	D_2	D_2	D_2
D_2	D_2	D_2	D_2
D_2	D_2	D_2	D_2

Step 2: Threshold image I by

$$I'(i,j) = \begin{cases} 255 & \text{if } I(i,j) > D(i,j) \\ 0 & \text{if } I(i,j) \le D(i,j) \end{cases}$$

Step 3: Show images I and I'

(2.1) Experimental results

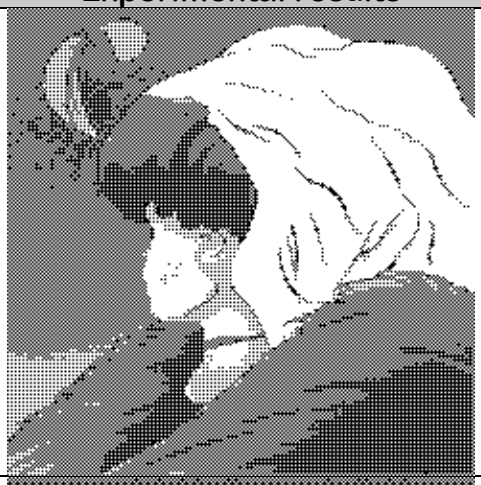
Original Image I

Image I'

(2.2) Source code

```python
# HW2-A (Implement Dithering)
from PIL import Image
import numpy as np

# Input a grayscale image I
I = Image.open("grayscale.png").convert('L')
width, height = I.size

# Step1 : Define dithering matrix(D2) and generate dithering array(D)
# Define Dithering Matrix (4X4)
D2 = np.array([[0, 128, 32, 160],
               [192, 64, 224, 96],
                [48, 176, 16, 144],
                [240, 112, 208, 80]])

# Generate Dithering Array D of image size by repeating D2
D = np.tile(D2, (int(width/4), int(height/4)))

# Step2 : Threshold image I
for y in range(I.size[1]):      # y = height
    for x in range(I.size[0]):  # x = width

        ori_pixel = I.getpixel((x, y))

        # threshold and update pixel val.
        if ori_pixel > D[y][x]:
            I.putpixel((x, y), 255)
        else:
            I.putpixel((x, y), 0)

# Save new image I'
I.save("I'.png")

# Step3 : Show images I and I'(dithering_img)
I = Image.open("grayscale.png")
I.show()
dithering_img = Image.open("I'.png")
dithering_img.show()
```

(2.3) Comments

Original Grayscale Image I on the right	
Dithering matrix	**Experimental results**
$D_1 = \begin{bmatrix} 0 & 128 \\ 192 & 64 \end{bmatrix}$	
$D_2 = \begin{bmatrix} 0 & 128 & 32 & 160 \\ 192 & 64 & 224 & 96 \\ 48 & 176 & 16 & 144 \\ 240 & 112 & 208 & 80 \end{bmatrix}$	

The larger the dithering matrix, the more random in dot scattering.

(B) Extend to n = 4 gray values

(1) Problem statement

1. $255 / 3 = 85$

2. $Q(i,j) = [I(i,j)/85]$

3. $D_1 = \begin{bmatrix} 0 & 56 \\ 84 & 28 \end{bmatrix} \underset{\text{extend}}{\Rightarrow} D$

4. $I'(i,j) = Q(i,j) + \begin{cases} 1 & \text{if } I(i,j) - 85Q(x,y) > D(i,j) \\ 0 & \text{if } I(i,j) - 85Q(x,y) \le D(i,j) \end{cases}$

5. Scale values of I' so that its values are in $[0, 255]$ for displaying

(2.1) Experimental results

Original Image I

Image I'

(2.2) Source code

```python
# HW2-B (Implement Dithering extend to n=4 gray values)
from PIL import Image
import numpy as np

# Input a grayscale image I
I = Image.open("grayscale.png").convert('L')
width, height = I.size

# Step1 : n output and m range Declaration
n = 4
m = int(255/(n-1))

# Step2 : Define and Calculate Q
column = height
row = width
# Define Q (2D Array)
Q = [[0 for _ in range(row)] for _ in range(column)]
# Calculate Q(i,j) value
for j in range(I.size[1]):      # j = column
    for i in range(I.size[0]):  # i = row
        Q[j][i] = int(I.getpixel((i, j))/m)  # Q(i,j) = [I(i,j)/85]

# Step3 :Define dithering matrix(D1) and generate dithering array(D)
# Define Dithering Matrix (2X2)
D1 = np.array([[0, 56],
               [84, 28]])

# Generate Dithering Array D of image size by repeating D1
D = np.tile(D1, (int(width/2), int(height/2)))

# Step4 : Threshold image I
for j in range(I.size[1]):      # j = column
    for i in range(I.size[0]):  # i = row

        ori_pixel = I.getpixel((i, j))

        if ori_pixel - m * Q[j][i] > D[j][i]:
            I.putpixel((i, j), m * (Q[j][i] + 1))
        else:
            I.putpixel((i, j), m * (Q[j][i] + 0))


```

```
# Save new image I'
I.save("I'.png")

# Step5 : Display images I and I'(dithering_img)
I = Image.open("grayscale.png")
I.show()
dithering_img = Image.open("I'.png")
dithering_img.show()
```

(2.3) Comments

Extend to more than two gray values :

I

I_4

I_2

I_8