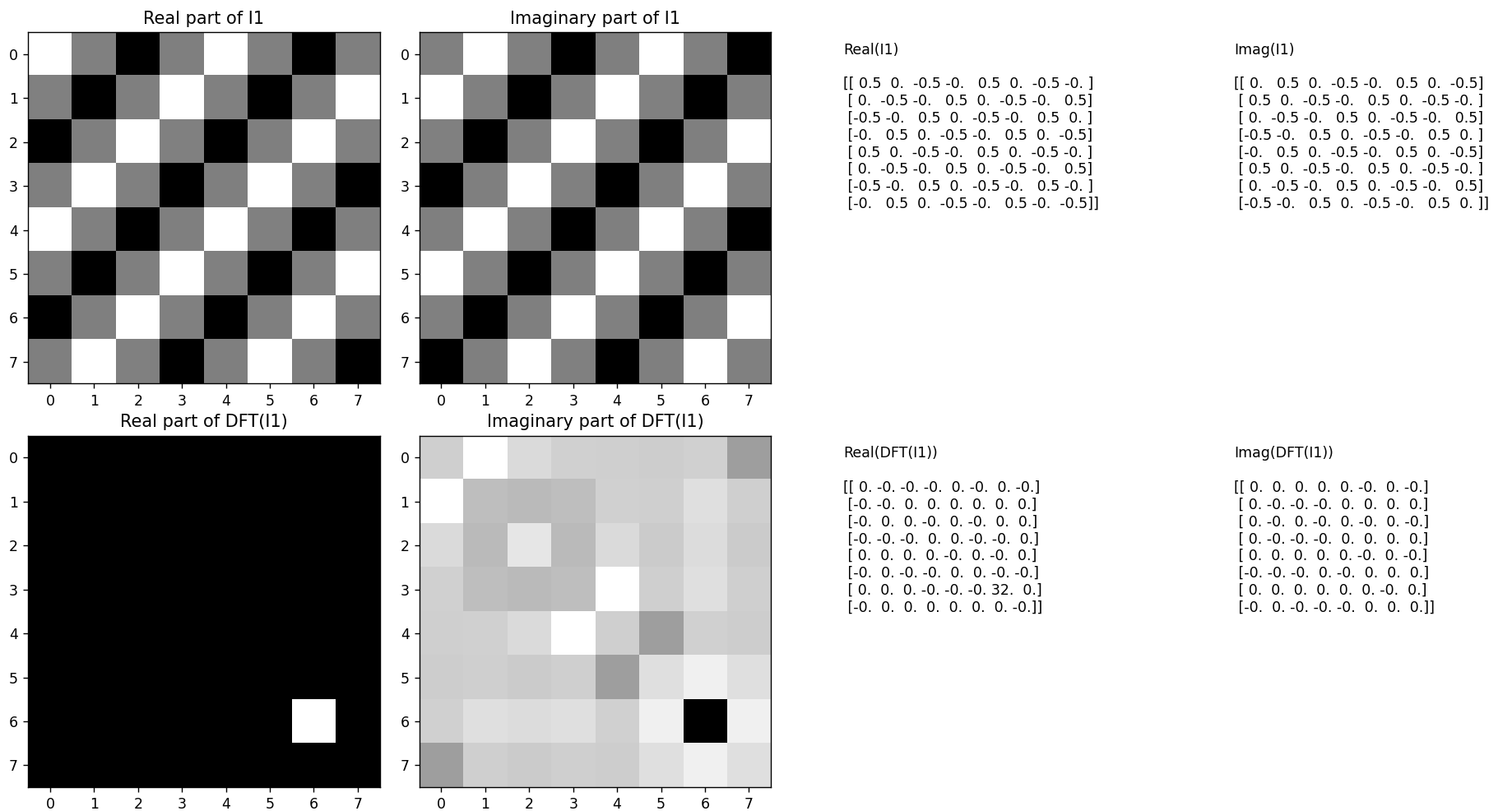
1.

* Code:
* import numpy as np  
  import matplotlib.pyplot as plt  
    
  COLS, ROWS = np.meshgrid(np.arange(8), np.arange(8))  
    
  u0 = 2  
  v0 = 2   
    
  I1 = 0.5 \* np.exp(1j \* 2 \* np.pi \* (u0 \* COLS + v0 \* ROWS) / 8)  
    
  Itilde1 = np.fft.fftshift(np.fft.fft2(I1))  
    
  print("Real part of I1:")  
  print(np.real(I1))  
  print("\nImaginary part of I1:")  
  print(np.imag(I1))  
    
  print("\nReal part of DFT(I1):")  
  np.set\_printoptions(precision=4, suppress=True)  
  print(np.real(Itilde1))  
    
  print("\nImaginary part of DFT(I1):")  
  print(np.imag(Itilde1))  
    
  fig, axes = plt.subplots(2, 4, figsize=(16, 8))  
    
  axes[0, 0].imshow(np.real(I1), cmap='gray')  
  axes[0, 0].set\_title('Real part of I1')  
  axes[0, 1].imshow(np.imag(I1), cmap='gray')  
  axes[0, 1].set\_title('Imaginary part of I1')  
    
  axes[1, 0].imshow(np.real(Itilde1), cmap='gray')  
  axes[1, 0].set\_title('Real part of DFT(I1)')  
  axes[1, 1].imshow(np.imag(Itilde1), cmap='gray')  
  axes[1, 1].set\_title('Imaginary part of DFT(I1)')  
    
  axes[0, 2].axis('off')  
  axes[0, 3].axis('off')  
  axes[1, 2].axis('off')  
  axes[1, 3].axis('off')  
    
  axes[0, 2].text(0.1, 0.5, "Real(I1)\n\n" + str(np.real(I1)), fontsize=10)  
  axes[0, 3].text(0.1, 0.5, "Imag(I1)\n\n" + str(np.imag(I1)), fontsize=10)  
  axes[1, 2].text(0.1, 0.5, "Real(DFT(I1))\n\n" + str(np.real(Itilde1)), fontsize=10)  
  axes[1, 3].text(0.1, 0.5, "Imag(DFT(I1))\n\n" + str(np.imag(Itilde1)), fontsize=10)  
    
  plt.tight\_layout()  
  plt.show()
* Result:

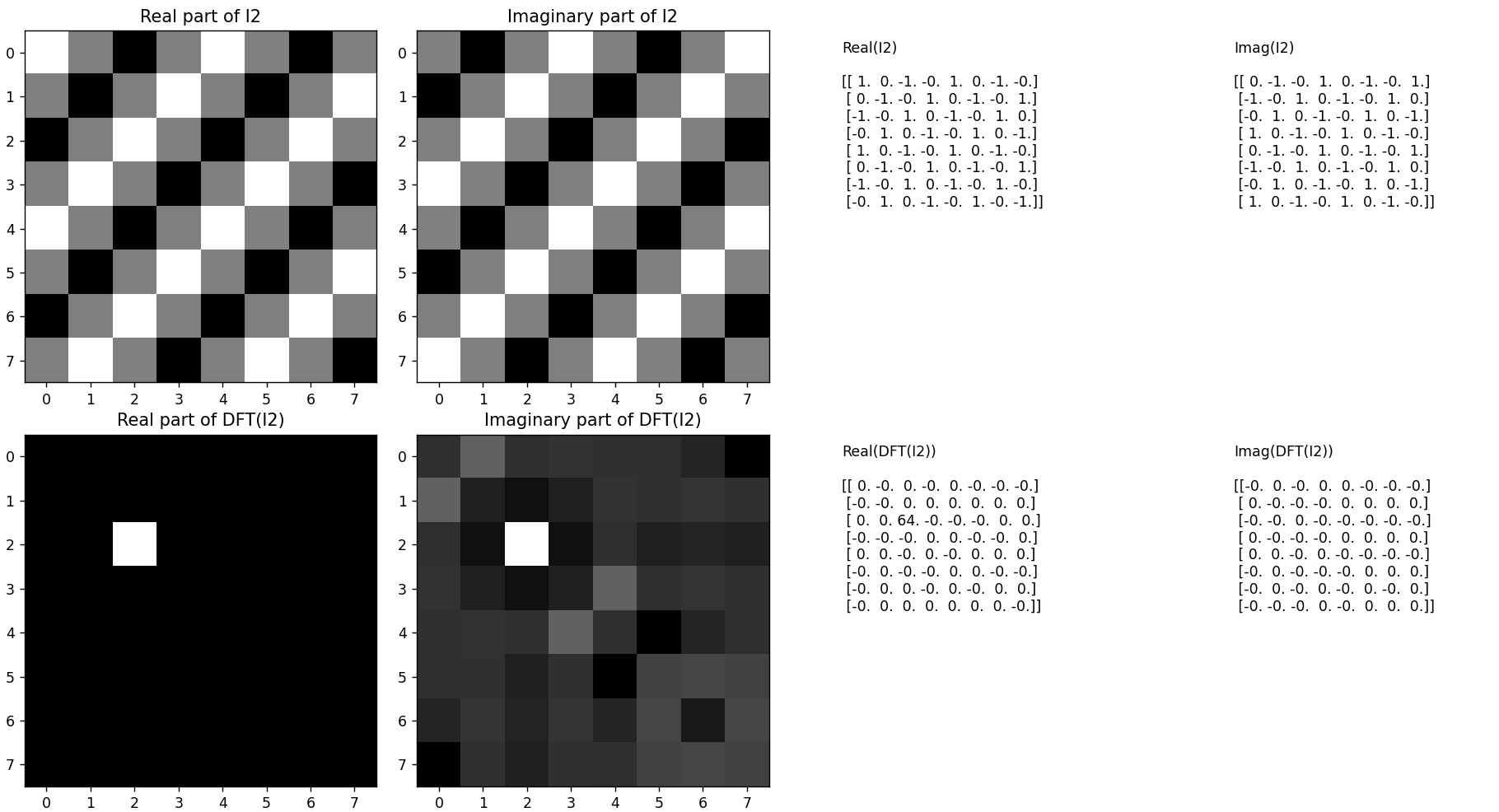


2.

- Code:

import numpy as np  
import matplotlib.pyplot as plt  
  
COLS, ROWS = np.meshgrid(np.arange(8), np.arange(8))  
  
u0 = 2  
v0 = 2  
  
I2 = np.exp(-1j \* 2 \* np.pi \* (u0 \* COLS + v0 \* ROWS) / 8)  
  
Itilde2 = np.fft.fftshift(np.fft.fft2(I2))  
  
print("Real part of I2:")  
print(np.real(I2))  
print("\nImaginary part of I2:")  
print(np.imag(I2))  
  
print("\nReal part of DFT(I2):")  
np.set\_printoptions(precision=4, suppress=True)  
print(np.real(Itilde2))  
  
print("\nImaginary part of DFT(I2):")  
print(np.imag(Itilde2))  
  
fig, axes = plt.subplots(2, 4, figsize=(16, 8))  
  
axes[0, 0].imshow(np.real(I2), cmap='gray')  
axes[0, 0].set\_title('Real part of I2')  
axes[0, 1].imshow(np.imag(I2), cmap='gray')  
axes[0, 1].set\_title('Imaginary part of I2')  
  
axes[1, 0].imshow(np.real(Itilde2), cmap='gray')  
axes[1, 0].set\_title('Real part of DFT(I2)')  
axes[1, 1].imshow(np.imag(Itilde2), cmap='gray')  
axes[1, 1].set\_title('Imaginary part of DFT(I2)')  
  
axes[0, 2].axis('off')  
axes[0, 3].axis('off')  
axes[1, 2].axis('off')  
axes[1, 3].axis('off')  
  
axes[0, 2].text(0.1, 0.5, "Real(I2)\n\n" + str(np.real(I2)), fontsize=10)  
axes[0, 3].text(0.1, 0.5, "Imag(I2)\n\n" + str(np.imag(I2)), fontsize=10)  
axes[1, 2].text(0.1, 0.5, "Real(DFT(I2))\n\n" + str(np.real(Itilde2)), fontsize=10)  
axes[1, 3].text(0.1, 0.5, "Imag(DFT(I2))\n\n" + str(np.imag(Itilde2)), fontsize=10)  
  
plt.tight\_layout()  
plt.show()

- Result:

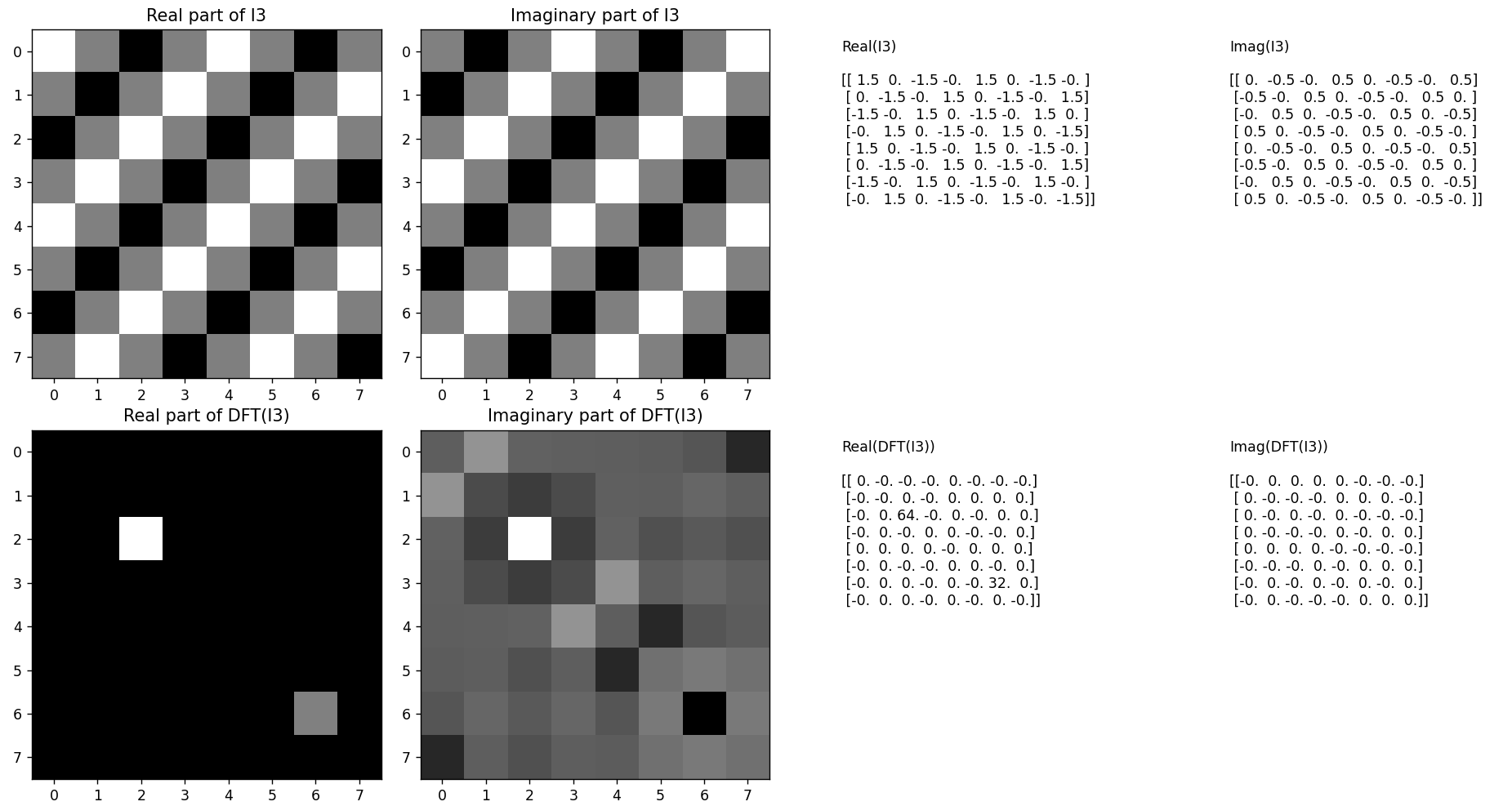


3.

- Code:

import numpy as np  
import matplotlib.pyplot as plt  
  
COLS, ROWS = np.meshgrid(np.arange(8), np.arange(8))  
  
u0 = 2  
v0 = 2  
  
I1 = 0.5 \* np.exp(1j \* 2 \* np.pi \* (u0 \* COLS + v0 \* ROWS) / 8)  
I2 = np.exp(-1j \* 2 \* np.pi \* (u0 \* COLS + v0 \* ROWS) / 8)  
I3 = I1 + I2  
  
Itilde3 = np.fft.fftshift(np.fft.fft2(I3))  
  
print("Real part of I3:")  
print(np.real(I3))  
print("\nImaginary part of I3:")  
print(np.imag(I3))  
  
print("\nReal part of DFT(I3):")  
np.set\_printoptions(precision=4, suppress=True)  
print(np.real(Itilde3))  
  
print("\nImaginary part of DFT(I3):")  
print(np.imag(Itilde3))  
  
fig, axes = plt.subplots(2, 4, figsize=(16, 8))  
  
axes[0, 0].imshow(np.real(I3), cmap='gray')  
axes[0, 0].set\_title('Real part of I3')  
axes[0, 1].imshow(np.imag(I3), cmap='gray')  
axes[0, 1].set\_title('Imaginary part of I3')  
  
axes[1, 0].imshow(np.real(Itilde3), cmap='gray')  
axes[1, 0].set\_title('Real part of DFT(I3)')  
axes[1, 1].imshow(np.imag(Itilde3), cmap='gray')  
axes[1, 1].set\_title('Imaginary part of DFT(I3)')  
  
axes[0, 2].axis('off')  
axes[0, 3].axis('off')  
axes[1, 2].axis('off')  
axes[1, 3].axis('off')  
  
axes[0, 2].text(0.1, 0.5, "Real(I3)\n\n" + str(np.real(I3)), fontsize=10)  
axes[0, 3].text(0.1, 0.5, "Imag(I3)\n\n" + str(np.imag(I3)), fontsize=10)  
axes[1, 2].text(0.1, 0.5, "Real(DFT(I3))\n\n" + str(np.real(Itilde3)), fontsize=10)  
axes[1, 3].text(0.1, 0.5, "Imag(DFT(I3))\n\n" + str(np.imag(Itilde3)), fontsize=10)  
  
plt.tight\_layout()  
plt.show()

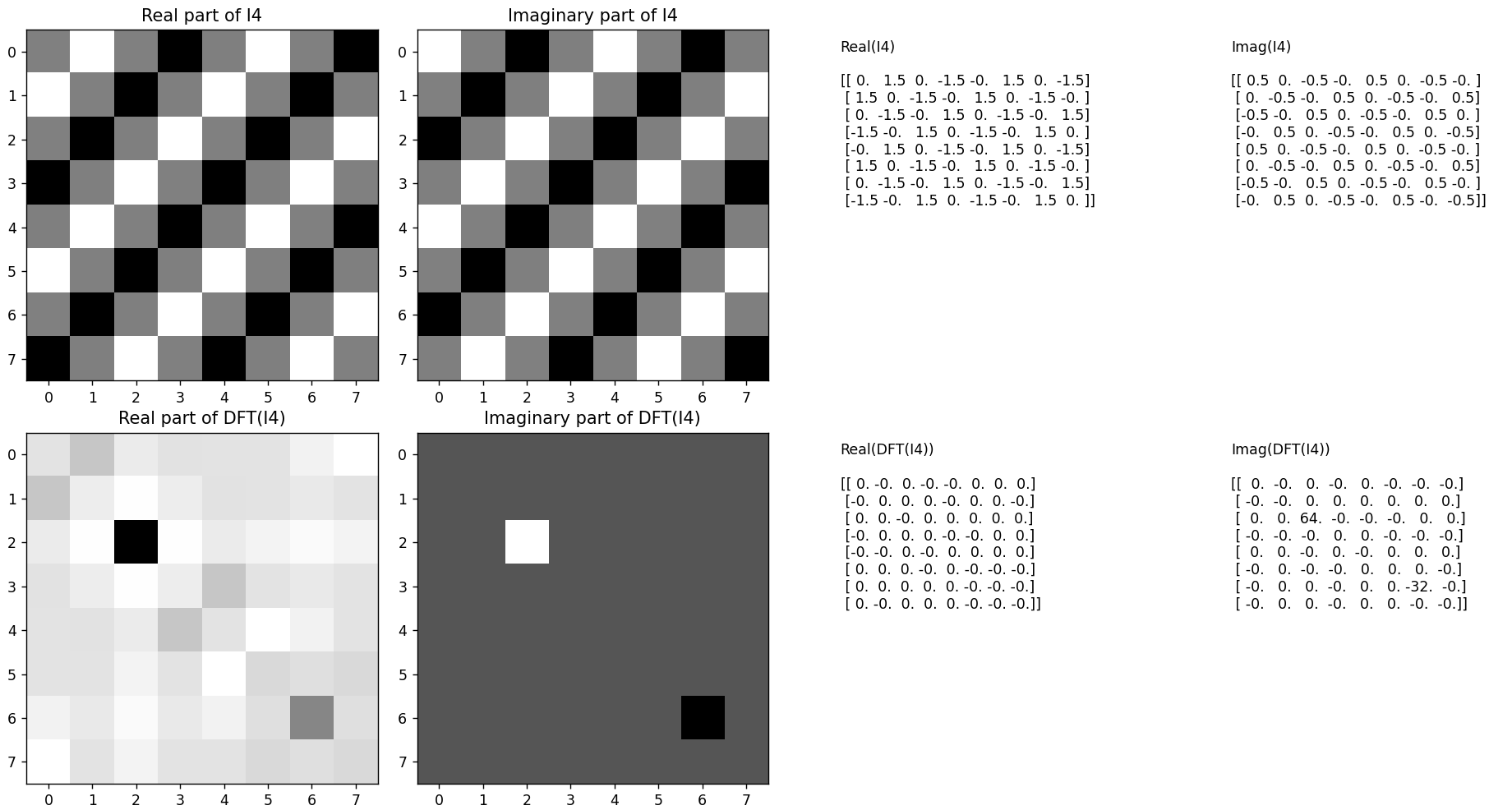
- Result:



4.

- Code: import numpy as np  
import matplotlib.pyplot as plt  
  
COLS, ROWS = np.meshgrid(np.arange(8), np.arange(8))  
  
u0 = 2  
v0 = 2  
  
I1 = 0.5 \* np.exp(1j \* 2 \* np.pi \* (u0 \* COLS + v0 \* ROWS) / 8)  
I2 = np.exp(-1j \* 2 \* np.pi \* (u0 \* COLS + v0 \* ROWS) / 8)  
I4 = -1j \* (I1 - I2)  
  
Itilde4 = np.fft.fftshift(np.fft.fft2(I4))  
  
print("Real part of I4:")  
print(np.real(I4))  
print("\nImaginary part of I4:")  
print(np.imag(I4))  
  
print("\nReal part of DFT(I4):")  
np.set\_printoptions(precision=4, suppress=True)  
print(np.real(Itilde4))  
  
print("\nImaginary part of DFT(I4):")  
print(np.imag(Itilde4))  
  
fig, axes = plt.subplots(2, 4, figsize=(16, 8))  
  
axes[0, 0].imshow(np.real(I4), cmap='gray')  
axes[0, 0].set\_title('Real part of I4')  
axes[0, 1].imshow(np.imag(I4), cmap='gray')  
axes[0, 1].set\_title('Imaginary part of I4')  
  
axes[1, 0].imshow(np.real(Itilde4), cmap='gray')  
axes[1, 0].set\_title('Real part of DFT(I4)')  
axes[1, 1].imshow(np.imag(Itilde4), cmap='gray')  
axes[1, 1].set\_title('Imaginary part of DFT(I4)')  
  
axes[0, 2].axis('off')  
axes[0, 3].axis('off')  
axes[1, 2].axis('off')  
axes[1, 3].axis('off')  
  
axes[0, 2].text(0.1, 0.5, "Real(I4)\n\n" + str(np.real(I4)), fontsize=10)  
axes[0, 3].text(0.1, 0.5, "Imag(I4)\n\n" + str(np.imag(I4)), fontsize=10)  
axes[1, 2].text(0.1, 0.5, "Real(DFT(I4))\n\n" + str(np.real(Itilde4)), fontsize=10)  
axes[1, 3].text(0.1, 0.5, "Imag(DFT(I4))\n\n" + str(np.imag(Itilde4)), fontsize=10)  
  
plt.tight\_layout()  
plt.show()

- Result:

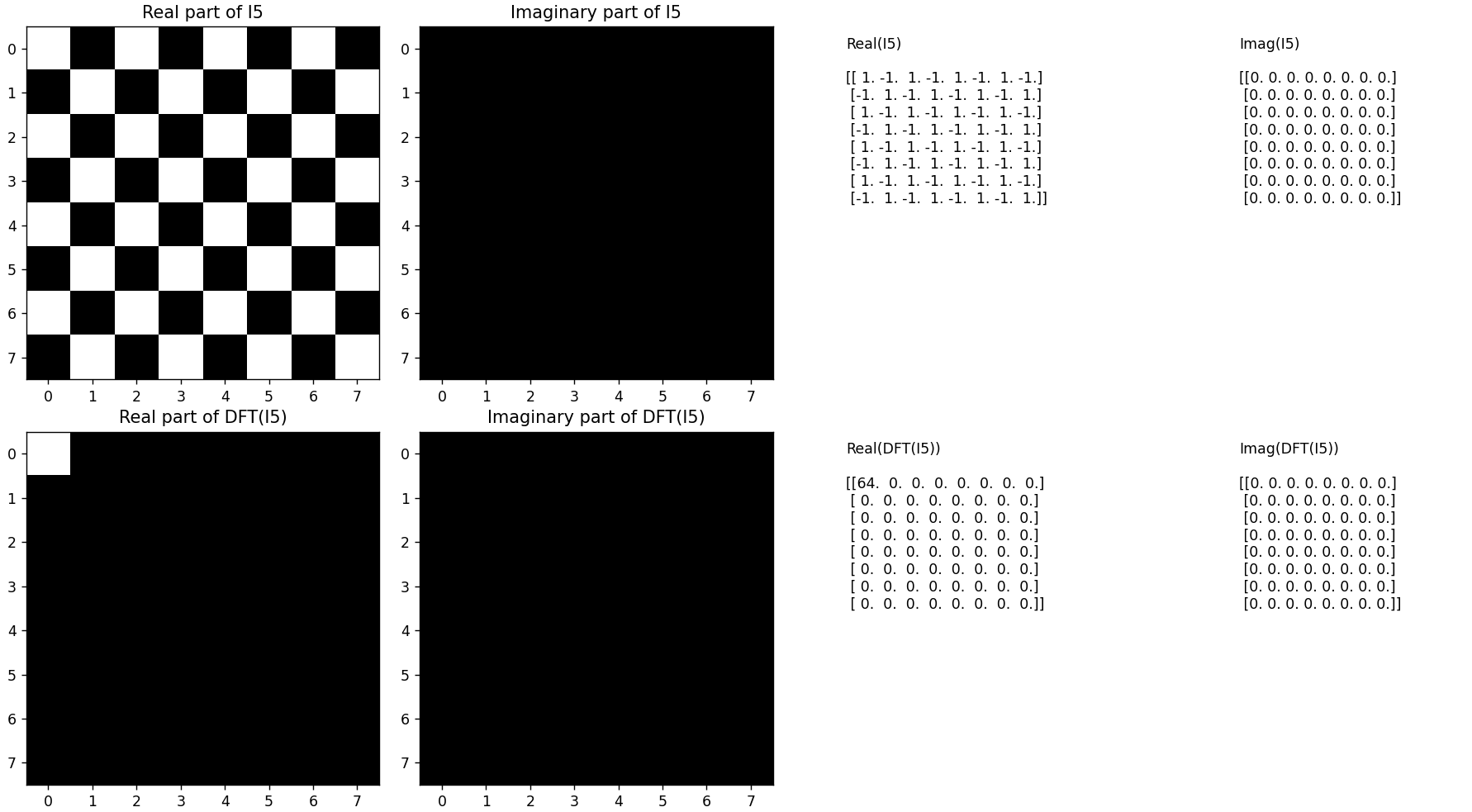


5.

- Code:

import numpy as np  
import matplotlib.pyplot as plt  
  
COLS, ROWS = np.meshgrid(np.arange(8), np.arange(8))  
u1, v1 = 1.5, 1.5  
  
I5 = np.cos(2 \* np.pi \* (u1 \* COLS + v1 \* ROWS))  
  
Itilde5 = np.fft.fftshift(np.fft.fft2(I5))  
  
print("Real part of I5:")  
print(np.real(I5))  
print("\nImaginary part of I5:")  
print(np.imag(I5))  
  
print("\nReal part of DFT(I5):")  
np.set\_printoptions(precision=4, suppress=True)  
print(np.real(Itilde5))  
  
print("\nImaginary part of DFT(I5):")  
print(np.imag(Itilde5))  
  
fig, axes = plt.subplots(2, 4, figsize=(16, 8))  
  
axes[0, 0].imshow(np.real(I5), cmap='gray')  
axes[0, 0].set\_title('Real part of I5')  
axes[0, 1].imshow(np.imag(I5), cmap='gray')  
axes[0, 1].set\_title('Imaginary part of I5')  
  
axes[1, 0].imshow(np.real(Itilde5), cmap='gray')  
axes[1, 0].set\_title('Real part of DFT(I5)')  
axes[1, 1].imshow(np.imag(Itilde5), cmap='gray')  
axes[1, 1].set\_title('Imaginary part of DFT(I5)')  
  
axes[0, 2].axis('off')  
axes[0, 3].axis('off')  
axes[1, 2].axis('off')  
axes[1, 3].axis('off')  
  
axes[0, 2].text(0.1, 0.5, "Real(I5)\n\n" + str(np.real(I5)), fontsize=10)  
axes[0, 3].text(0.1, 0.5, "Imag(I5)\n\n" + str(np.imag(I5)), fontsize=10)  
axes[1, 2].text(0.1, 0.5, "Real(DFT(I5))\n\n" + str(np.real(Itilde5)), fontsize=10)  
axes[1, 3].text(0.1, 0.5, "Imag(DFT(I5))\n\n" + str(np.imag(Itilde5)), fontsize=10)  
  
plt.tight\_layout()  
plt.show()

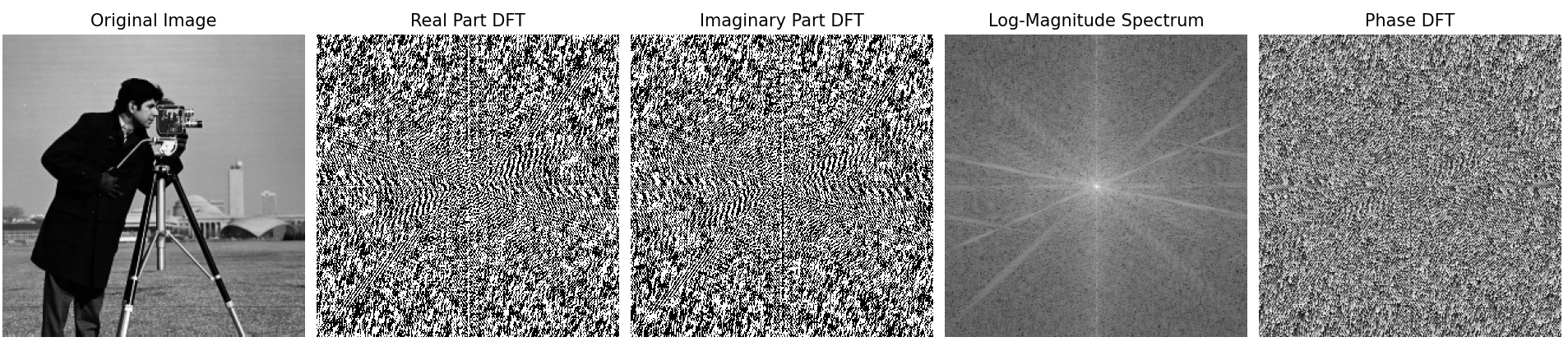
* Result:

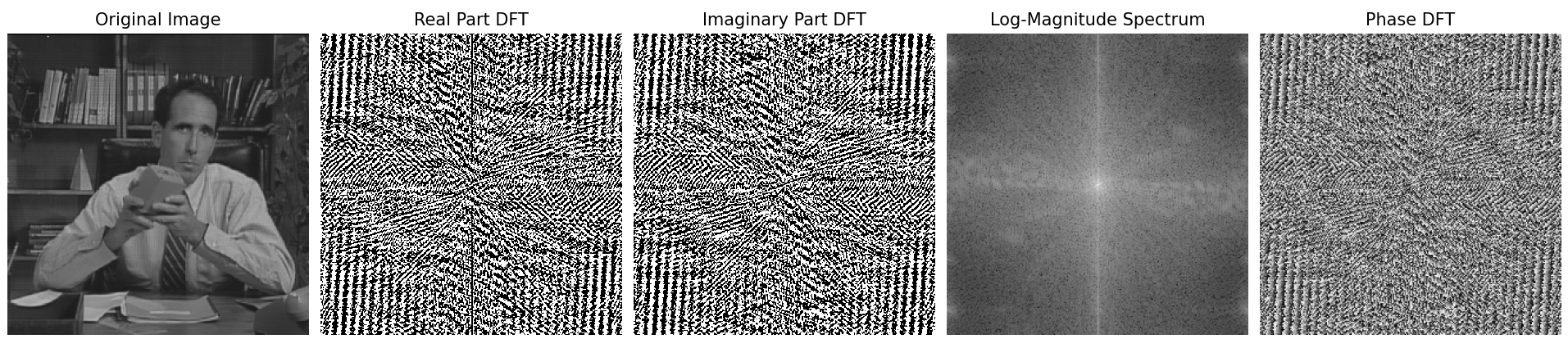


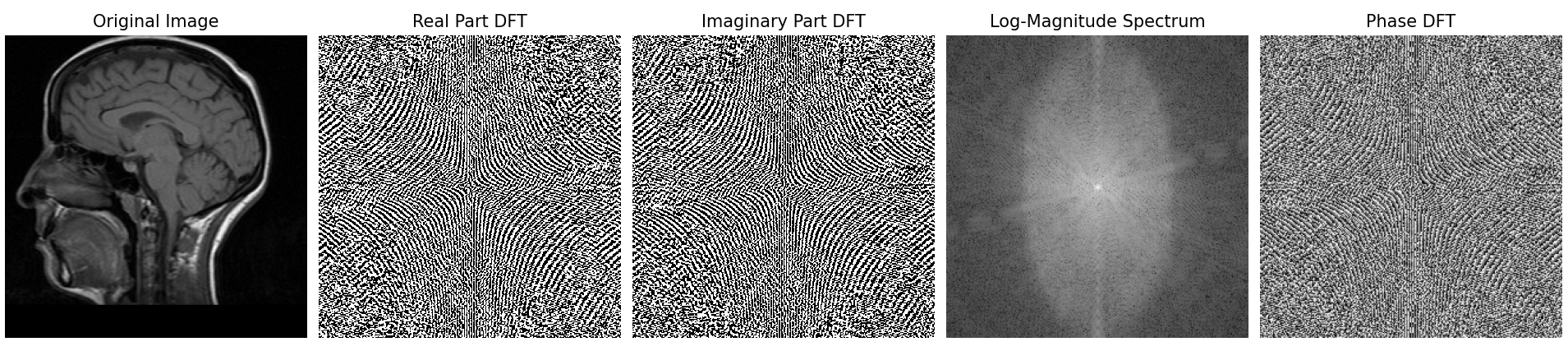
6.

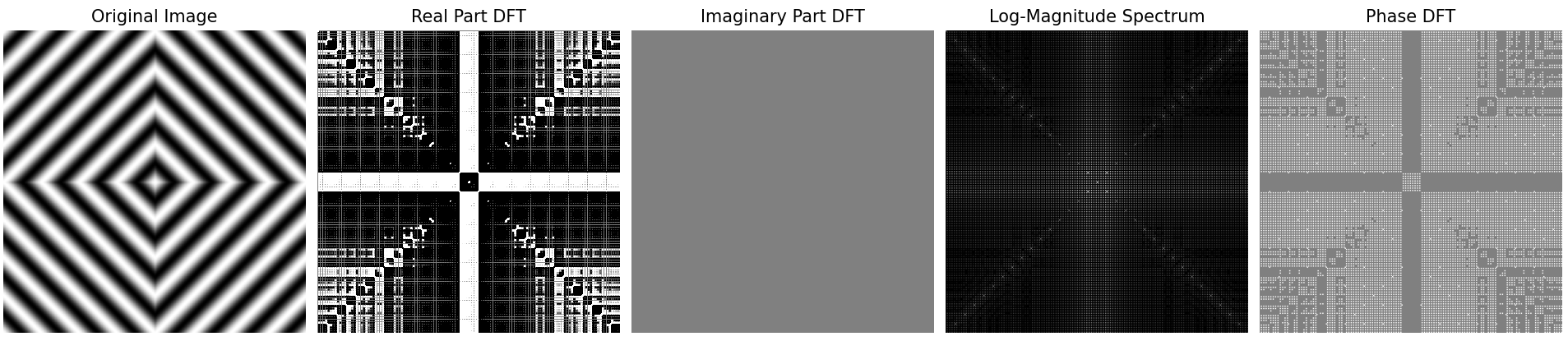
- Code:

* import numpy as np  
  import matplotlib.pyplot as plt  
    
  image\_filenames = ['camera.bin', 'salesman.bin', 'head.bin', 'eyeR.bin']  
    
  def process\_and\_display\_image(image\_filename):  
    
   image\_data = np.fromfile(image\_filename, dtype=np.uint8).reshape(256, 256)  
    
   image\_dft = np.fft.fftshift(np.fft.fft2(image\_data))  
    
   real\_part = np.real(image\_dft)  
   imaginary\_part = np.imag(image\_dft)  
    
   magnitude\_spectrum = np.log(np.abs(image\_dft) + 1)  
    
   phase = np.angle(image\_dft)  
    
   fig, axes = plt.subplots(1, 5, figsize=(20, 4))  
    
   axes[0].set\_title('Original Image')  
   axes[0].imshow(image\_data, cmap='gray', vmin=0, vmax=255)  
   axes[0].axis('off')  
    
   axes[1].set\_title('Real Part DFT')  
   axes[1].imshow(real\_part, cmap='gray', vmin=-1, vmax=1)  
   axes[1].axis('off')  
    
   axes[2].set\_title('Imaginary Part DFT')  
   axes[2].imshow(imaginary\_part, cmap='gray', vmin=-1, vmax=1)  
   axes[2].axis('off')  
    
   axes[3].set\_title('Log-Magnitude Spectrum')  
   axes[3].imshow(magnitude\_spectrum, cmap='gray')  
   axes[3].axis('off')  
    
   axes[4].set\_title('Phase DFT')  
   axes[4].imshow(phase, cmap='gray', vmin=-np.pi, vmax=np.pi)  
   axes[4].axis('off')  
    
   plt.tight\_layout()  
   plt.show()  
    
  for image\_filename in image\_filenames:  
   process\_and\_display\_image(image\_filename)
* Result:









7.

- Code:

import numpy as np  
import matplotlib.pyplot as plt  
  
I6 = np.fromfile('camera.bin', dtype=np.uint8).reshape(256, 256)  
  
J1 = np.abs(I6)  
J2 = np.angle(I6)  
  
fig, axes = plt.subplots(1, 2, figsize=(12, 6))  
  
axes[0].set\_title('J2: DFT Phase Contribution')  
axes[0].imshow(J2, cmap='gray', vmin=-np.pi, vmax=np.pi)  
axes[0].axis('off')  
  
JJ1 = np.log(J1 + 1e-10)  
  
axes[1].set\_title('JJ1: Log(DFT Magnitude Contribution)')  
axes[1].imshow(JJ1, cmap='gray')  
axes[1].axis('off')  
  
plt.show()

- Result:

