سوال 1

الف:

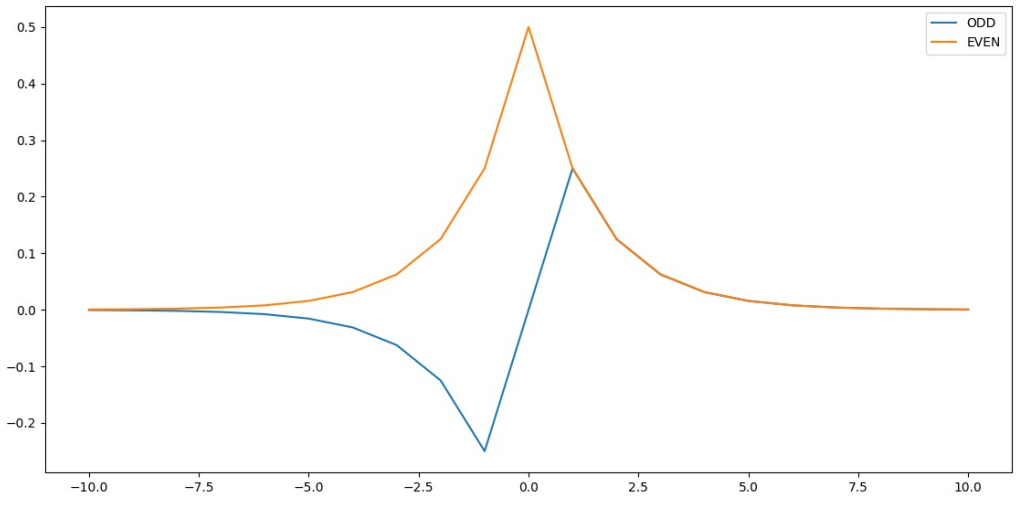
Q1_A.py - D:\DSP\CA1\Q1_A.py (3.8.4)

File Edit Format Run Options Window Help

```
import numpy as np
import matplotlib.pyplot as plt

#defining x[n] = (0.5 ^ n) u[n]
n = range(-10, 11)
x = 0.5 * np.ones(len(n))
x = np.power(x, n).tolist()
u = np.heaviside(n, 0.5)
x = np.multiply(x, u).tolist()

#defining x[-n]
x_reverse = x[::-1]

x_e = (np.add(x, x_reverse) / 2).tolist()
x_o = (np.subtract(x, x_reverse) / 2).tolist()

plt.plot(n, x_o, label = "ODD")
plt.plot(n, x_e, label = "EVEN")
plt.legend()
plt.show()
```

Ln: 1 Col: 0

Figure 1



x=7.69 y=−0.176

ب:

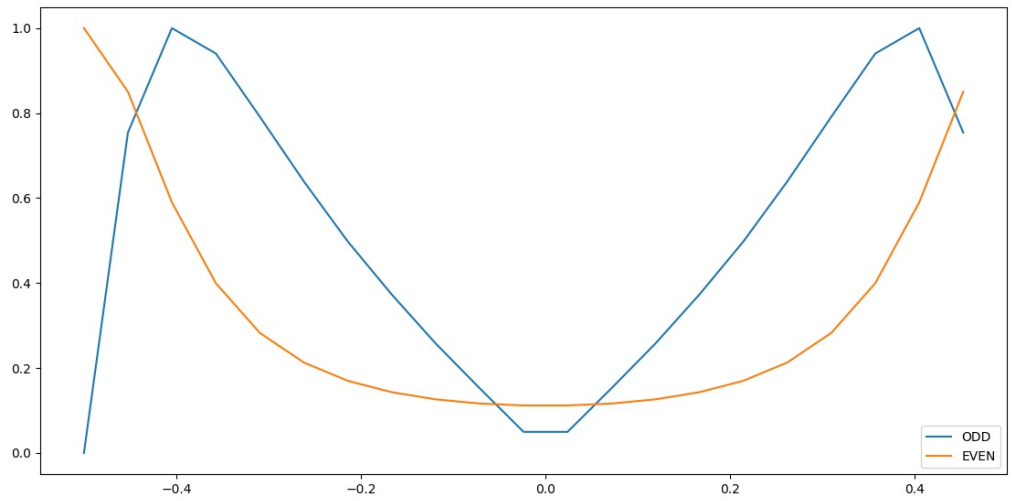
Q1_B.py - D:\DSP\CA1\Q1_B.py (3.8.4)

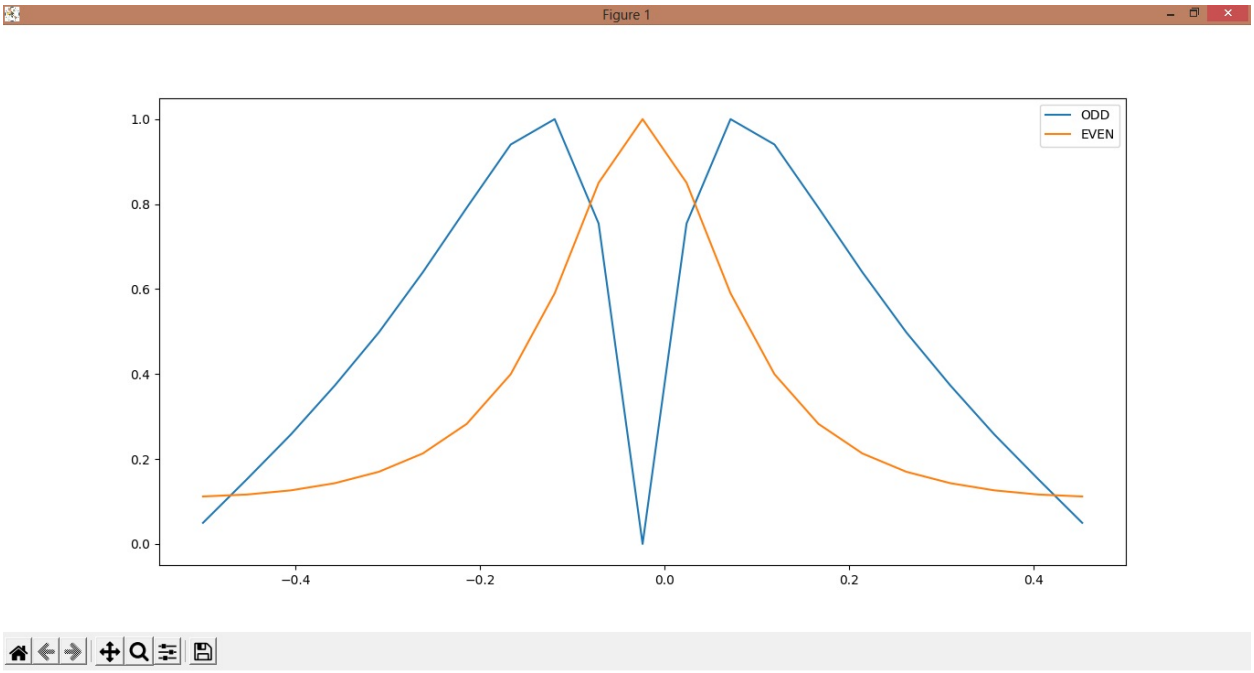
File Edit Format Run Options Window Help

```
import numpy as np
import matplotlib.pyplot as plt

#defining x[n] = (0.5 ^ n) u[n]
n = range(-10, 11)
x = 0.5 * np.ones(len(n))
x = np.power(x, n).tolist()
u = np.heaviside(n, 0.5)
x = np.multiply(x, u).tolist()

#defining x[-n]
x_reverse = x[::-1]

#defining x_e and x_o
x_e = (np.add(x, x_reverse) / 2).tolist()
x_o = (np.subtract(x, x_reverse) / 2).tolist()

#defining frequency domain
N = len(x_o)
n = np.arange(-N/2, N/2)
T = N
w = n/T

DTFT_x_e = np.fft.fft(x_e)
DTFT_x_o = np.fft.fft(x_o)

plt.plot(w, abs(DTFT_x_o / max(abs(DTFT_x_o))), label = "ODD")
plt.plot(w, abs(DTFT_x_e / max(abs(DTFT_x_e))), label = "EVEN")
plt.legend()
plt.show()

DTFT_x_e = np.fft.fftshift(DTFT_x_e)
DTFT_x_o = np.fft.fftshift(DTFT_x_o)

plt.plot(w, abs(DTFT_x_o / max(abs(DTFT_x_o))), label = "ODD")
plt.plot(w, abs(DTFT_x_e / max(abs(DTFT_x_e))), label = "EVEN")
plt.legend()
plt.show()
```

Ln: 2 Col: 31

Figure 1





در واقع با استفاده از تابع fftshift نقطه صفر محور فرکانس که قبلا در منتهی الیه سمت چپ بود، به وسط محور فرکانس منتقل شده(نقطه صفر فرکانس، وسط محور افقی در نظر گرفته می شود) و درنتیجه تبدیل فوریه به سمت راست شیفت داده می شود.

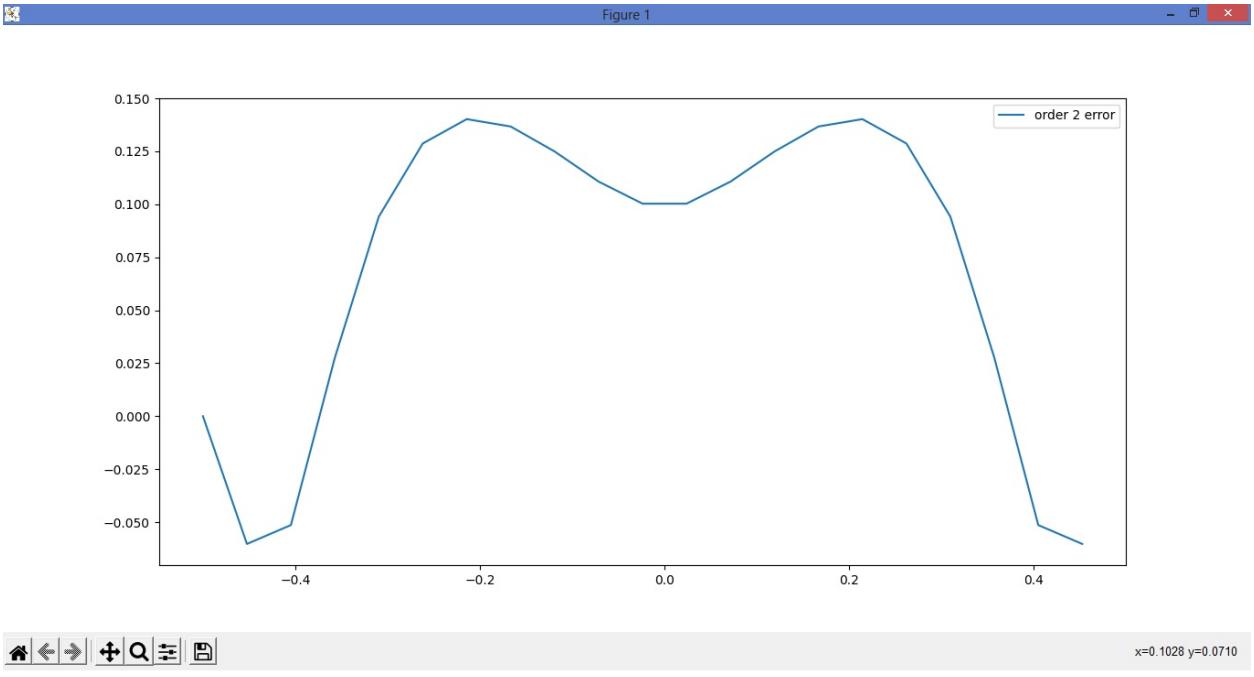
ج:

Q1_C.py - D:\DSP\CA1\Q1_C.py (3.8.4)

File Edit Format Run Options Window Help

```
import numpy as np
import matplotlib.pyplot as plt

#defining x[n] = (0.5 ^ n) u[n]
n = range(-10, 11)
x = 0.5 * np.ones(len(n))
x = np.power(x, n).tolist()
u = np.heaviside(n, 0.5)
x = np.multiply(x, u).tolist()

#defining x[-n]
x_reverse = x[::-1]

#defining x_e and x_o
x_e = (np.add(x, x_reverse) / 2).tolist()
x_o = (np.subtract(x, x_reverse) / 2).tolist()

#defining frequency domain
N = len(x_o)
n = np.arange(-N/2, N/2)
T = N
w = n/T

DTFT_x_e = np.fft.fft(x_e)
DTFT_x_o = np.fft.fft(x_o)
DTFT_x = np.fft.fft(x)

plt.plot(w, abs(DTFT_x / max(abs(DTFT_x))) - abs(DTFT_x_o + DTFT_x_e / max(abs(DTFT_x_o + DTFT_x_e))) ** 2, label = "order 2 error")
plt.legend()
plt.show()
```

Ln: 3 Col: 0



همان طور که دیده می شود خطا مرتبه دوم بسیار کم و حدود صفر است که این نشان می دهد که تبدیل فوریه سیگنال اصلی برابر با مجموع تبدیل فوریه قسمت های زوج و فرد آن است.

:د

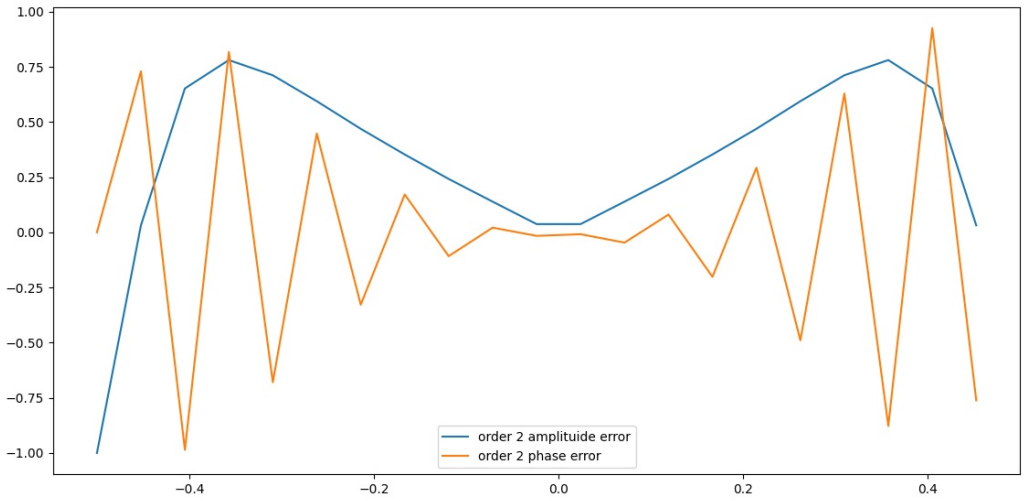
Q1_D.py - D:\DSP\CA1\Q1_D.py (3.8.4)

File Edit Format Run Options Window Help

```
import numpy as np
import matplotlib.pyplot as plt

#defining x[n] = (0.5 ^ n) u[n]
n = range(-10, 11)
x = 0.5 * np.ones(len(n))
x = np.power(x, n).tolist()
u = np.heaviside(n, 0.5)
x = np.multiply(x, u).tolist()

#defining x[-n]
x_reverse = x[::-1]

#defining x_e and x_o
x_e = (np.add(x, x_reverse) / 2).tolist()
x_o = (np.subtract(x, x_reverse) / 2).tolist()

#defining frequency domain
N = len(x_o)
n = np.arange(-N/2, N/2)
T = N
w = n/T

DTFT_x_e = np.fft.fft(x_e)
DTFT_x_o = np.fft.fft(x_o)

plt.plot(w, abs(DTFT_x_o / max(abs(DTFT_x_o))) - abs(DTFT_x_e / max(abs(DTFT_x_e))) ** 2, label = "order 2 amplituide error")
plt.plot(w, np.imag(DTFT_x_o / max(abs(DTFT_x_o))) - np.imag(DTFT_x_e / max(abs(DTFT_x_e))) ** 2, label = "order 2 phase error")
plt.legend()
plt.show()
```

Ln: 10 Col: 0

Figure 1



x=0.1552 y=−0.216

سوال 2

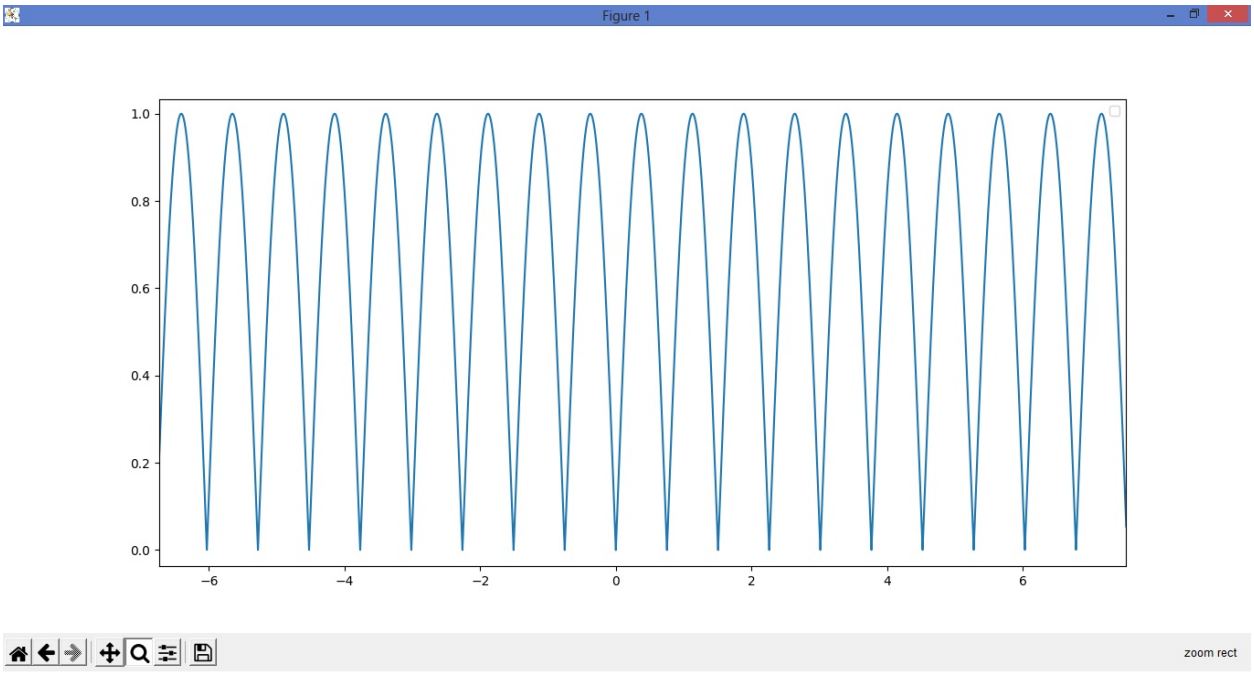
Q2.py - D:\DSP\CA1\Q2.py (3.8.4)

File Edit Format Run Options Window Help

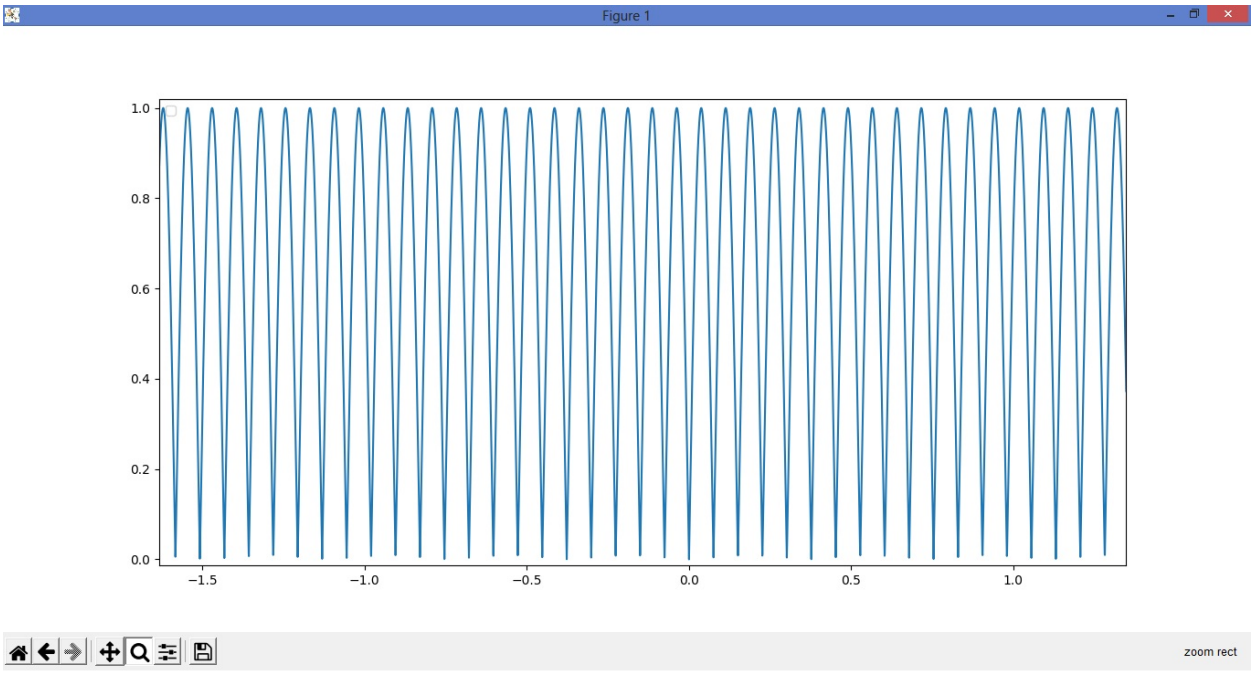
```
import numpy as np
import matplotlib.pyplot as plt

def make_sin_function_with_given_parameters(domain, freqency, time, freqency_offset):
    return domain * np.sin(freqency * time + freqency_offset)

def calculate_normalized_DTFT_of_given_signal_and_plot_it(input_signal, number_of_samples, sampling_rate):
    n = np.arange(-number_of_samples/2, number_of_samples/2)
    T = number_of_samples/sampling_rate
    w = n/(1000 * T)

    DTFT_input_signal = np.fft.fft(input_signal)

    plt.plot(w, abs(input_signal / max(abs(input_signal))))
    plt.legend()
    plt.show()

def aliazing_plotter(signal_frequency, sampling_frequency):
    tstart = -1
    tend = 1
    Ts = 1 / sampling_frequency
    time = np.arange(tstart, tend - Ts, Ts)
    domain = 1
    freqency_offset = 0
    number_of_samples = len(time)

    made_signal = make_sin_function_with_given_parameters(domain, signal_frequency, time, freqency_offset)

    calculate_normalized_DTFT_of_given_signal_and_plot_it(made_signal, number_of_samples, sampling_frequency)

sampling_frequency = 48000
frquencies = [100, 1000, 10000, 108000]

for i in range(len(frquencies)) :
    aliazing_plotter(frquencies[i], sampling_frequency)
```

Ln: 2 Col: 0

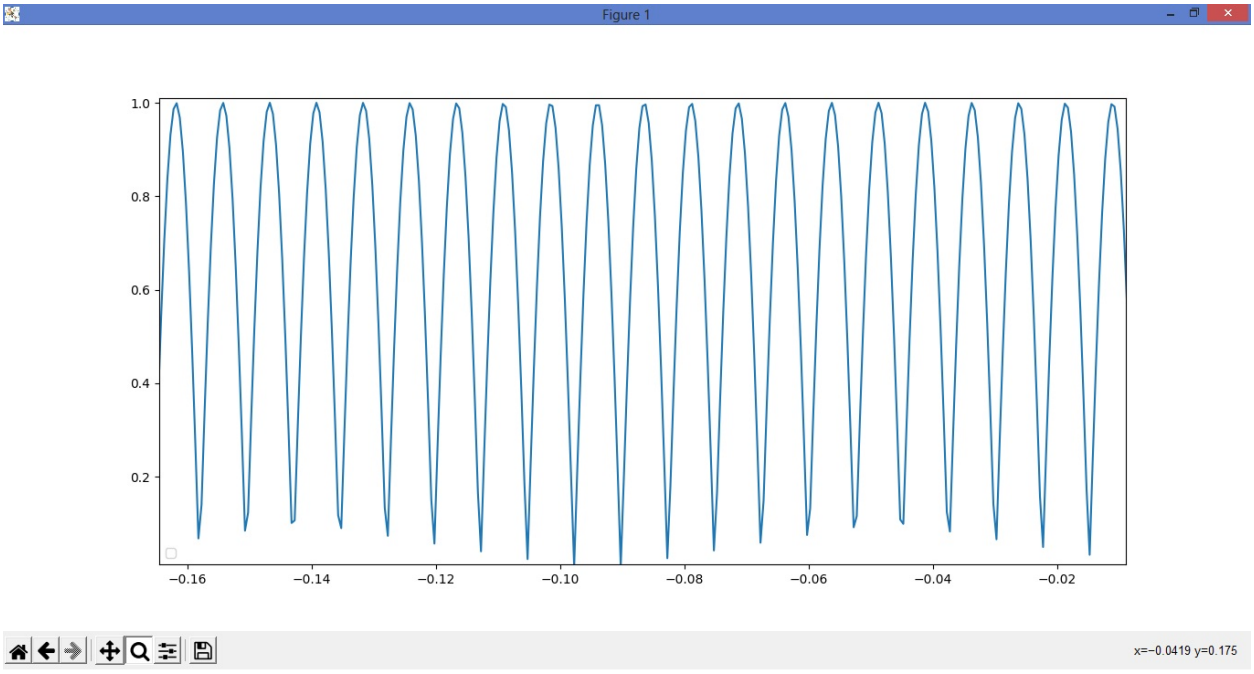
f = 100



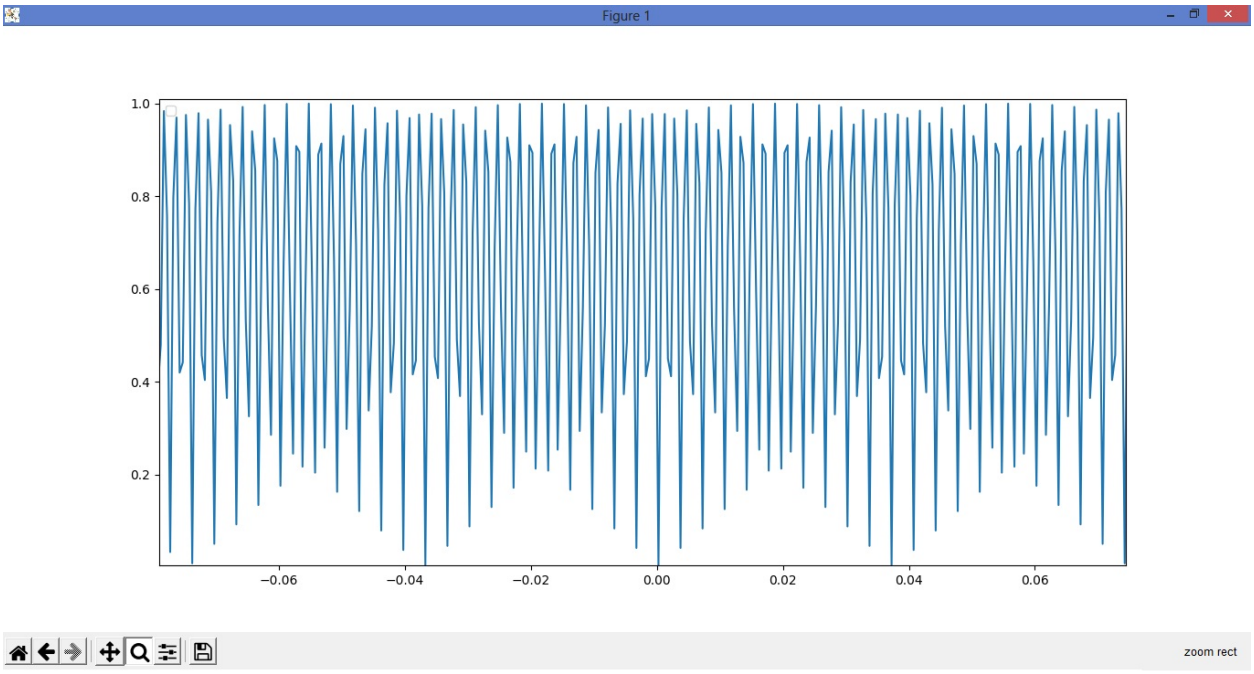
f = 1000



f = 10000



f = 108000



همان طور که دیده می شود در این حالت با توجه به اینکه 108000 بزرگتر از 48000 (فرکانس نمونه برداری) شده است، aliasing رخ داده است.

سوال 3

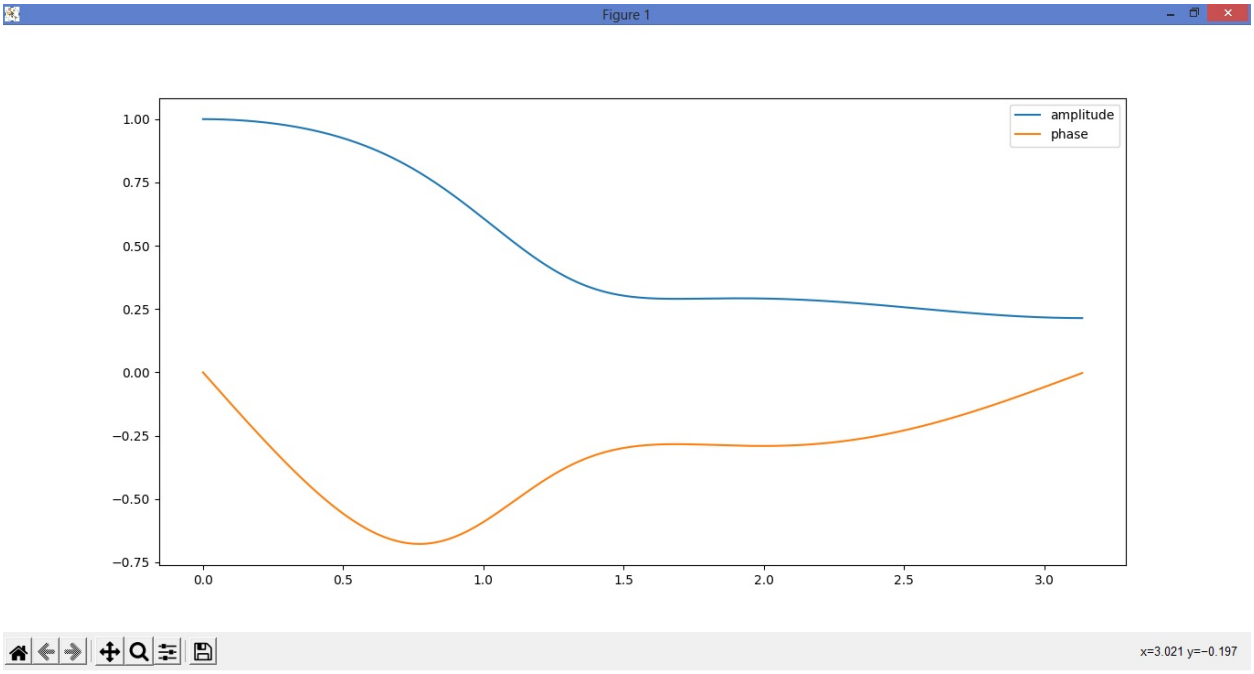
الف:

Q3_A.py - D:\DSP\CA1\Q3_A.py (3.8.4)

File Edit Format Run Options Window Help

```
import numpy as np
from scipy import signal
import matplotlib.pyplot as plt

#defining x(e^(jw)) and y(e^(jw)) coefficeints
y_e_j_w = [1, 2, 0, 1]
x_e_j_w = [1, -0.5, 0.25, 0]

w, h = signal.freqz(y_e_j_w, x_e_j_w)

plt.plot(w, abs(h / max(abs(h))), label = "amplitude")
plt.plot(w, np.imag(h / max(abs(h))), label = "phase")
plt.legend()
plt.show()
```

Ln: 1 Col: 18



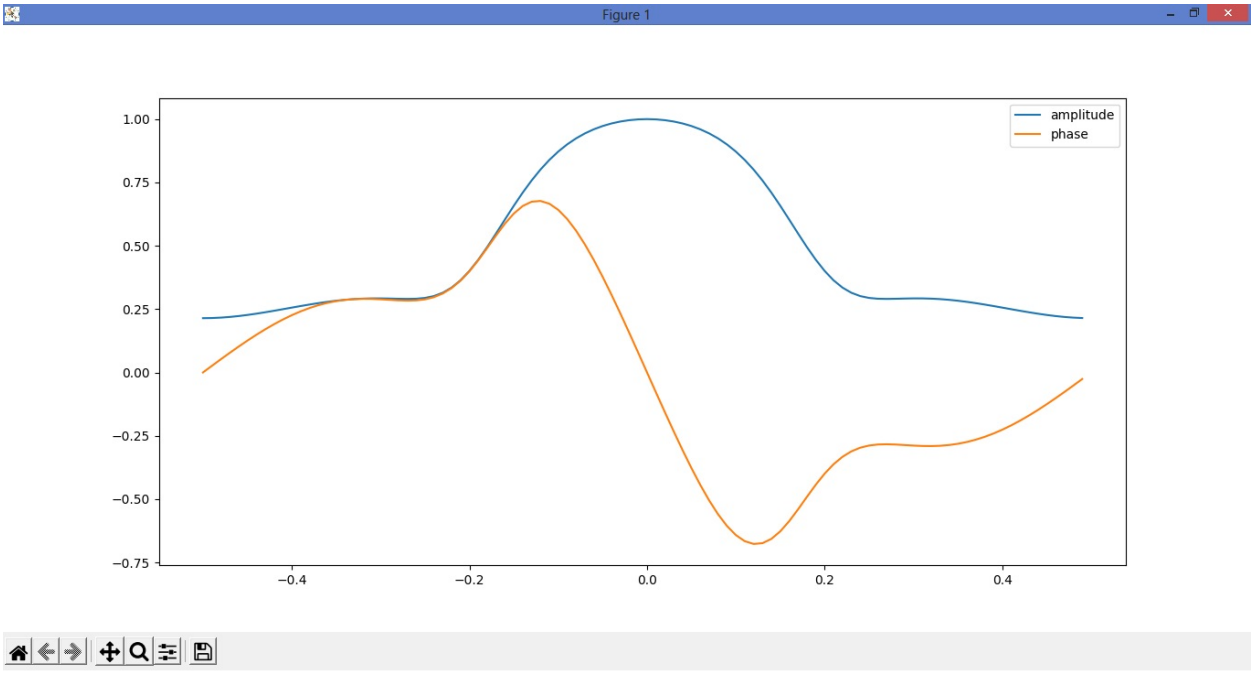
ب:

Q3_B.py - D:\DSP\CA1\Q3_B.py (3.8.4)

File Edit Format Run Options Window Help

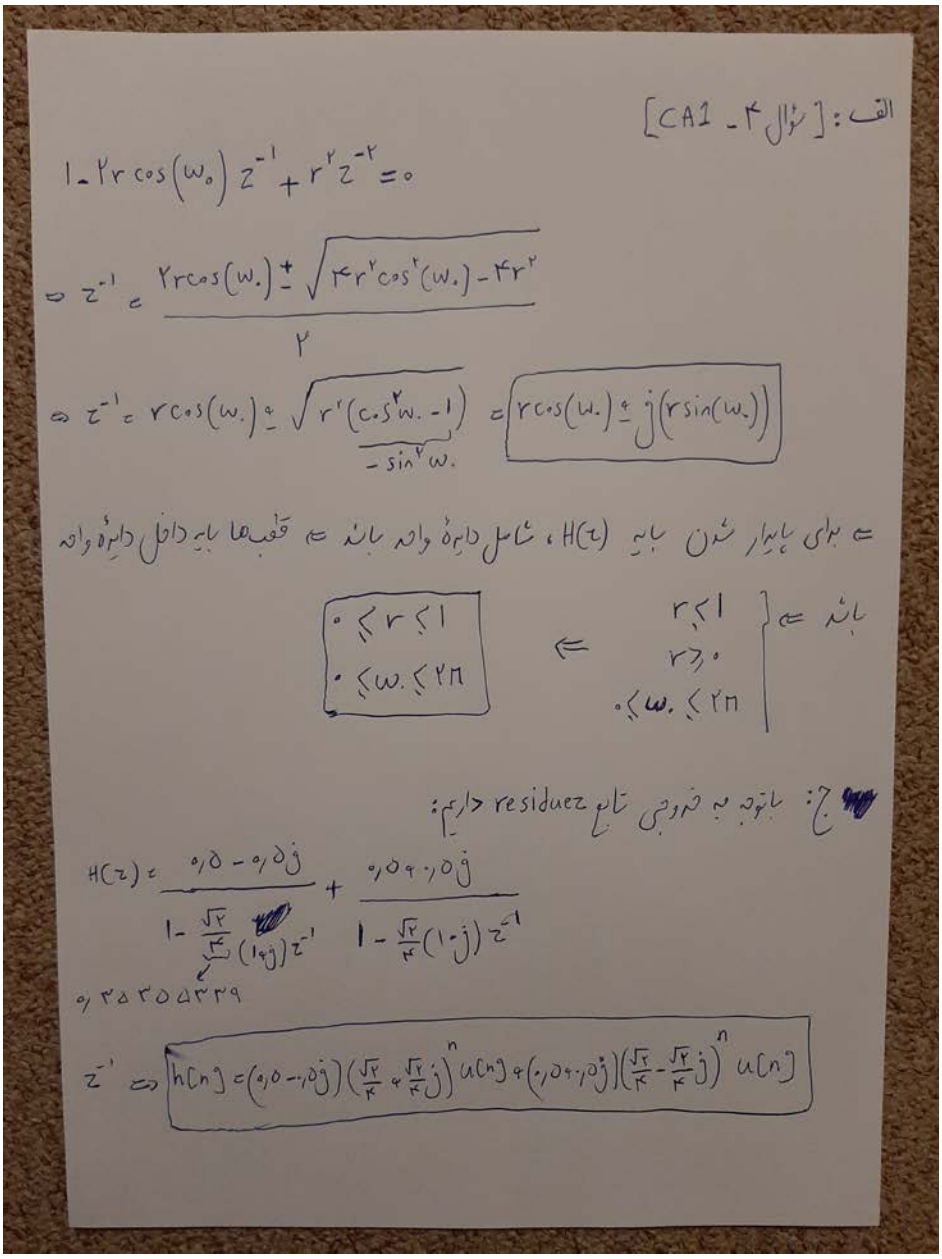
```
import numpy as np
import matplotlib.pyplot as plt

delta = np.zeros(100)
delta[0] = 1
h = np.zeros(100)

#defining h[n]
h[0] = delta[0]
h[1] = 2 * delta[0] + 0.5 * h[0]
h[2] = 0.5 * h[1] - 0.25 * h[0]
for i in range(3, 100):
    h[i] = (delta[i])+ (2 * delta[i - 1]) + (delta[i - 3]) + (0.5 * h[i - 1]) - (0.25 * h[i - 2])

#defining frequency domain
N = len(h)
n = np.arange(-N/2, N/2)
T = N
w = n/T

#defining h(e^(jw))
h_e_j_w = np.fft.fftshift(np.fft.fft(h))

plt.plot(w, abs(h_e_j_w / max(abs(h_e_j_w))), label = "amplitude")
plt.plot(w, np.imag(h_e_j_w / max(abs(h_e_j_w))), label = "phase")
plt.legend()
plt.show()
```

Ln: 1 Col: 1



همان طور که دیده می شود، شکل قسمت الف همان نیمه راست شکل قسمت ب بوده و در واقع جواب دوقسمت الف و ب مانند هم هست.

سوال 4



الف: [سؤال ۴ - CA1]

$$1 - 2r\cos(\omega_0)z^{-1} + r^2 z^{-2} = 0$$

$$\Rightarrow z^{-1} = \frac{2r\cos(\omega_0) \pm \sqrt{4r^2\cos^2(\omega_0) - 4r^2}}{2}$$

$$\Rightarrow z^{-1} = r\cos(\omega_0) \pm \sqrt{r^2(\underbrace{\cos^2\omega_0 - 1}_{-\sin^2\omega_0})} = \boxed{r\cos(\omega_0) \pm j\left(r\sin(\omega_0)\right)}$$

= برای پایدار شدن سیستم $H(z)$، شامل دایرهٔ واحد باشد = قطب‌ها باید داخل دایرهٔ واحد باشند =

$$\left.\begin{array}{l} r < 1 \\ r > 0 \\ 0 < \omega_0 < 2\pi \end{array}\right\} \Leftarrow \boxed{\begin{array}{l} 0 < r < 1 \\ 0 < \omega_0 < 2\pi \end{array}}$$

ج: با توجه به فرمول تابع residuez داریم:

$$H(z) = \frac{0.5 - 0.5j}{1 - \frac{\sqrt{2}}{4}(1+j)z^{-1}} + \frac{0.5 + 0.5j}{1 - \frac{\sqrt{2}}{4}(1-j)z^{-1}}$$

0.35355339

$$z^{-1} \Rightarrow \boxed{h[n] = (0.5 - 0.5j)\left(\frac{\sqrt{2}}{4} + \frac{\sqrt{2}}{4}j\right)^n u[n] + (0.5 + 0.5j)\left(\frac{\sqrt{2}}{4} - \frac{\sqrt{2}}{4}j\right)^n u[n]}$$

ب و ج:

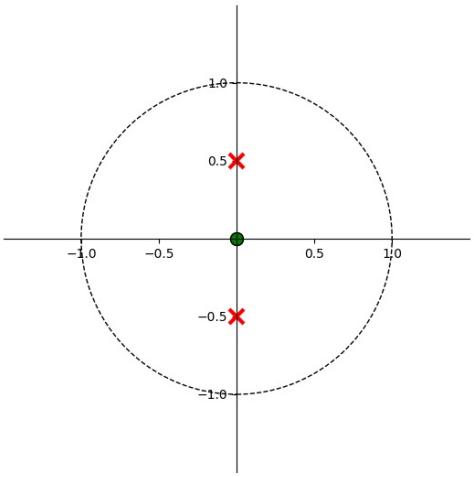
Q4.py - D:\DSP\CA1\Q4.py (3.8.4)

File Edit Format Run Options Window Help

```
import numpy as np
import zplane as zp
from scipy import signal
import matplotlib.pyplot as plt

#defining r, w0
r = 0.5
w_0 = np.pi / 2

#defining b and a coefficeints of H(Z)
b = [1, 0 , 0]
a = [1, -2 * r * np.cos(w_0), r ** 2]

zp.zplane(b, a)

#defining x(e^(jw)) and y(e^(jw)) coefficeints of H(Z), Z = e^(jw)
y_e_j_w = [1, 0, 0]
x_e_j_w = [1, -2 * r * np.cos(w_0), r ** 2]

w, h = signal.freqz(y_e_j_w, x_e_j_w)
plt.plot(w, abs(h / max(abs(h))), label = "amplitude")
plt.plot(w, np.imag(h / max(abs(h))), label = "phase")
plt.legend()
plt.show()

w_0 = np.pi / 4
a = [1, -2 * r * np.cos(w_0), r ** 2]
r, p, k = signal.residuez(b, a)
```

Ln: 2 Col: 0

Figure 1



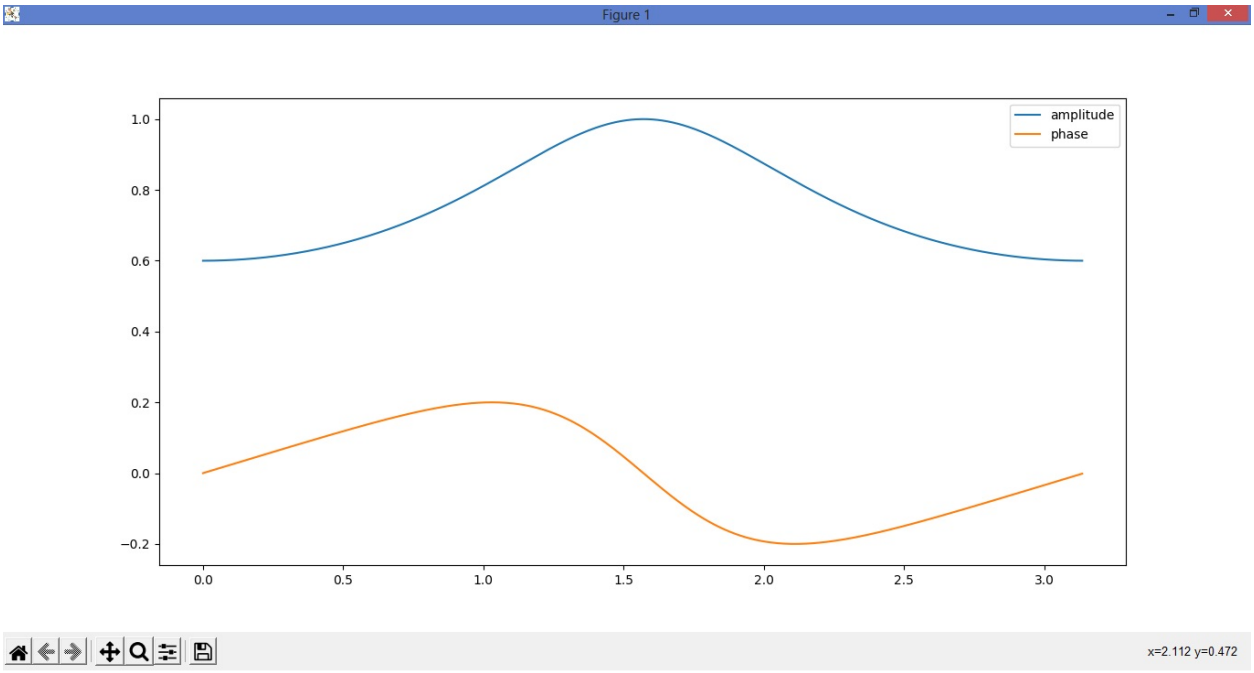


Figure 1
amplitude
phase
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
x=2.112 y=0.472


Python 3.8.4 Shell
File Edit Shell Debug Options Window Help
Python 3.8.4 (tags/v3.8.4:dfa645a, Jul 13 2020, 16:46:45) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:\DSP\CA1\Q4.py ==========
>>> r
array([0.5-0.5j, 0.5+0.5j])
>>> p
array([0.35355339+0.35355339j, 0.35355339-0.35355339j])
>>>
Ln: 9 Col: 4

سوال 5

Q5.py - D:\DSP\CA1\Q5.py (3.8.4)

File Edit Format Run Options Window Help

```
import numpy as np
from scipy.io import wavfile
import matplotlib.pyplot as plt

def M_down_sample(M, input_signal):
    output_signal = np.zeros(int(len(input_signal) / M))
    for i in range(len(output_signal)):
        output_signal[i] = input_signal[M * i]
    return output_signal

def L_up_sample(L, input_signal):
    output_signal = np.zeros(int(len(input_signal) * L))
    for i in range(len(input_signal)):
        output_signal[L * i] = input_signal[i]
    return output_signal

#part a
file_name = "test.wav"
sample_rate, data = wavfile.read(file_name, mmap = False)
print(sample_rate)

#part b
wavfile.write("test result.wav", sample_rate, data[60 * sample_rate : 71 * sample_rate])

#part p
wavfile.write("test result(fs).wav", sample_rate, data[60 * sample_rate : 71 * sample_rate])
wavfile.write("test result(2fs).wav", 2 * sample_rate, data[60 * sample_rate : 71 * sample_rate])
wavfile.write("test result(0.5fs).wav", int(0.5 * sample_rate), data[60 * sample_rate : 71 * sample_rate])

#part t
voice = data[60 * sample_rate : 71 * sample_rate]
voice = voice[:,0]

#defining frequency domain
N = len(voice)
n = np.arange(-N/2, N/2)
T = N/sample_rate
w = n/T

DTFT_voice = np.fft.fft(voice)
DTFT_voice = np.fft.fftshift(DTFT_voice)
```

Ln: 3 Col: 0

Q5.py - D:\DSP\CA1\Q5.py (3.8.4)

File Edit Format Run Options Window Help

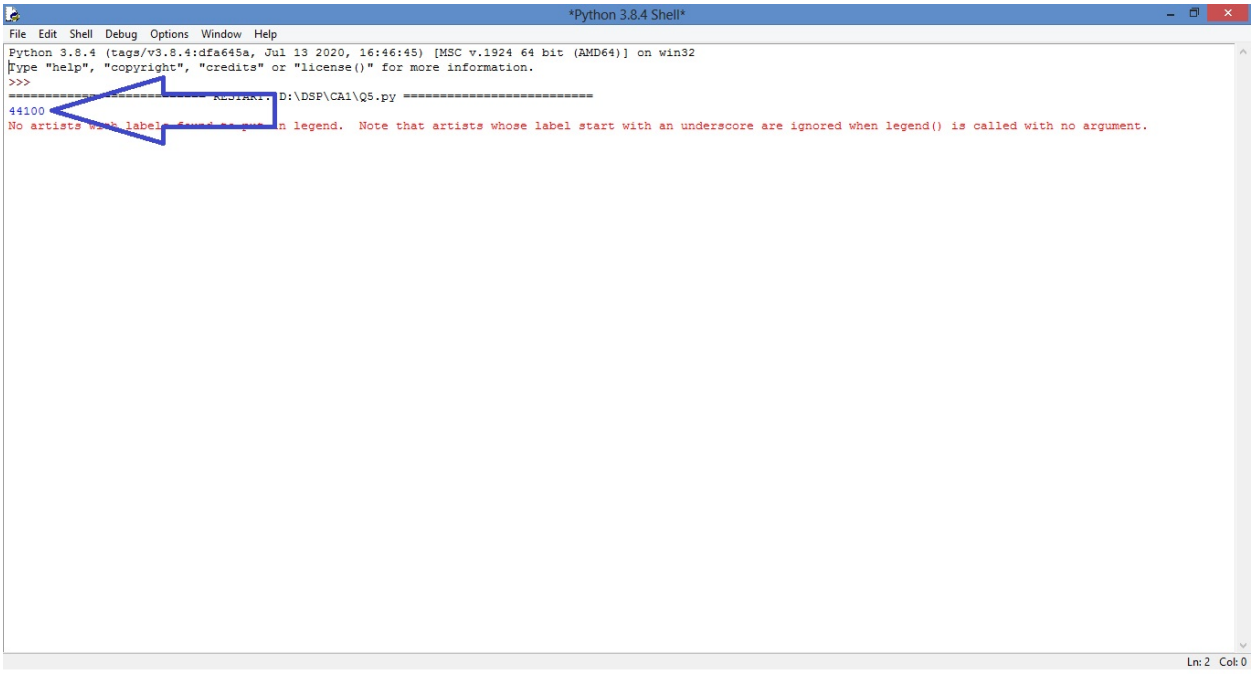
```
#part t
voice = data[60 * sample_rate : 71 * sample_rate]
voice = voice[:,0]

#defining frequency domain
N = len(voice)
n = np.arange(-N/2, N/2)
T = N/sample_rate
w = n/T

DTFT_voice = np.fft.fft(voice)
DTFT_voice = np.fft.fftshift(DTFT_voice)

plt.plot(w, abs(DTFT_voice / max(abs(DTFT_voice))))
plt.legend()
plt.show()

#part g M = 2
DTFT_voice_2_down_sample = np.fft.fft(M_down_sample(2, voice))
DTFT_voice_2_down_sample = np.fft.fftshift(DTFT_voice_2_down_sample)

N2 = N / 2
n = np.arange(-N2/2, N2/2)
T = N2/sample_rate
w = n/T

plt.plot(w, abs(DTFT_voice_2_down_sample / max(abs(DTFT_voice_2_down_sample))))
plt.legend()
plt.show()

#part g M = 5
DTFT_voice_5_down_sample = np.fft.fft(M_down_sample(5, voice))
DTFT_voice_5_down_sample = np.fft.fftshift(DTFT_voice_5_down_sample)

N3 = N / 5
n = np.arange(-N3/2, N3/2)
T = N3/sample_rate
w = n/T

plt.plot(w, abs(DTFT_voice_5_down_sample / max(abs(DTFT_voice_5_down_sample))))
plt.legend()
```

Ln: 45 Col: 10

پ:

با بزرگ تر شدن فرکانس نمونه برداری، صدا پخش شده، زیر تر می شود. و با کوچک تر شدن فرکانس نمونه برداری، صدا پخش شده، بم تر می شود.

الف:


Python 3.8.4 Shell
File Edit Shell Debug Options Window Help
Python 3.8.4 (tags/v3.8.4:dfa645a, Jul 13 2020, 16:46:45) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
D:\DSP\CA1\Q5.py
44100
No artists with labels found to put in legend. Note that artists whose label start with an underscore are ignored when legend() is called with no argument.
Ln: 2 Col: 0

ت:

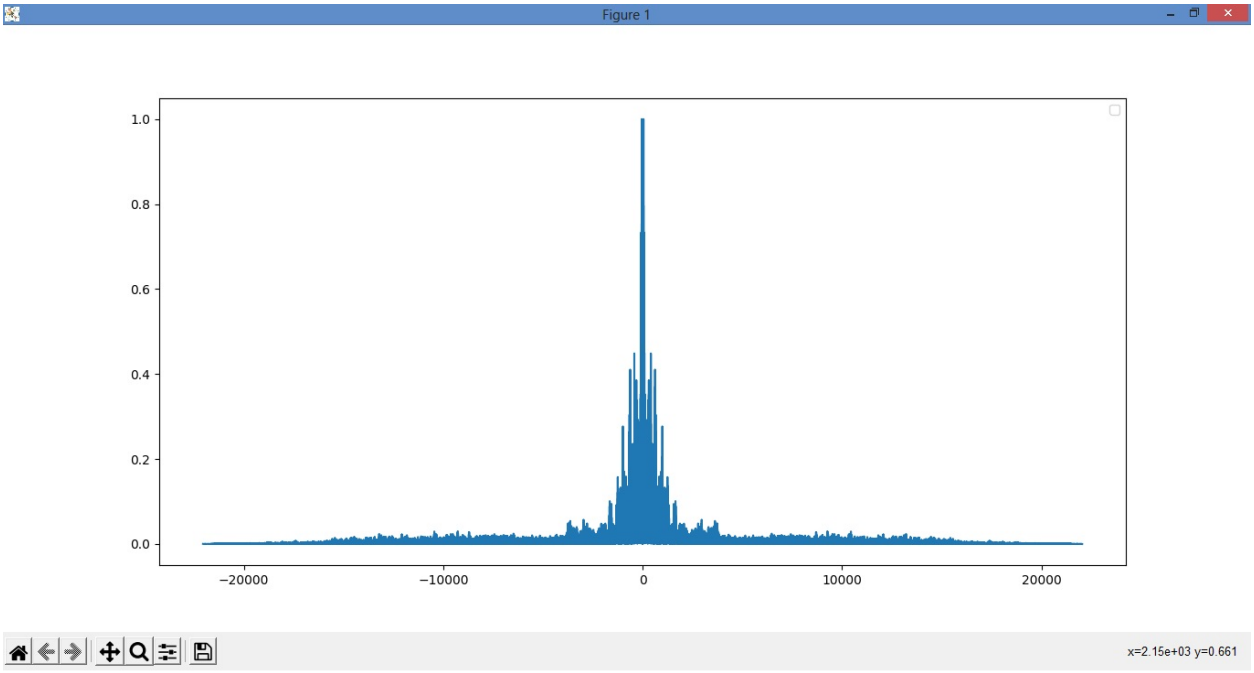
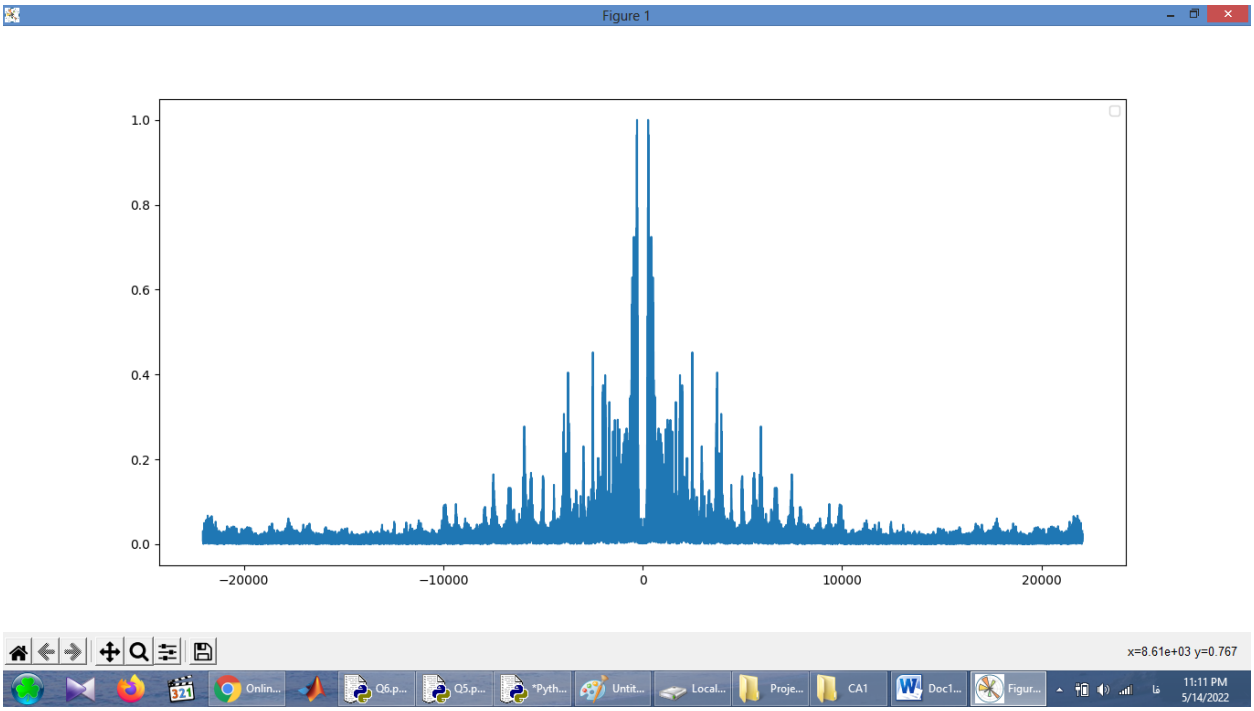


Figure 1
1.0
0.8
0.6
0.4
0.2
0.0
−20000
−10000
0
10000
20000
x=2.15e+03 y=0.661

ج:

Figure 1
1.0
0.8
0.6
0.4
0.2
0.0
−20000
−10000
0
10000
20000
x=1.416e+04 y=0.737
11:03 PM
5/14/2022

Figure 1
1.0
0.8
0.6
0.4
0.2
0.0
−20000
−10000
0
10000
20000
11:04 PM
5/14/2022



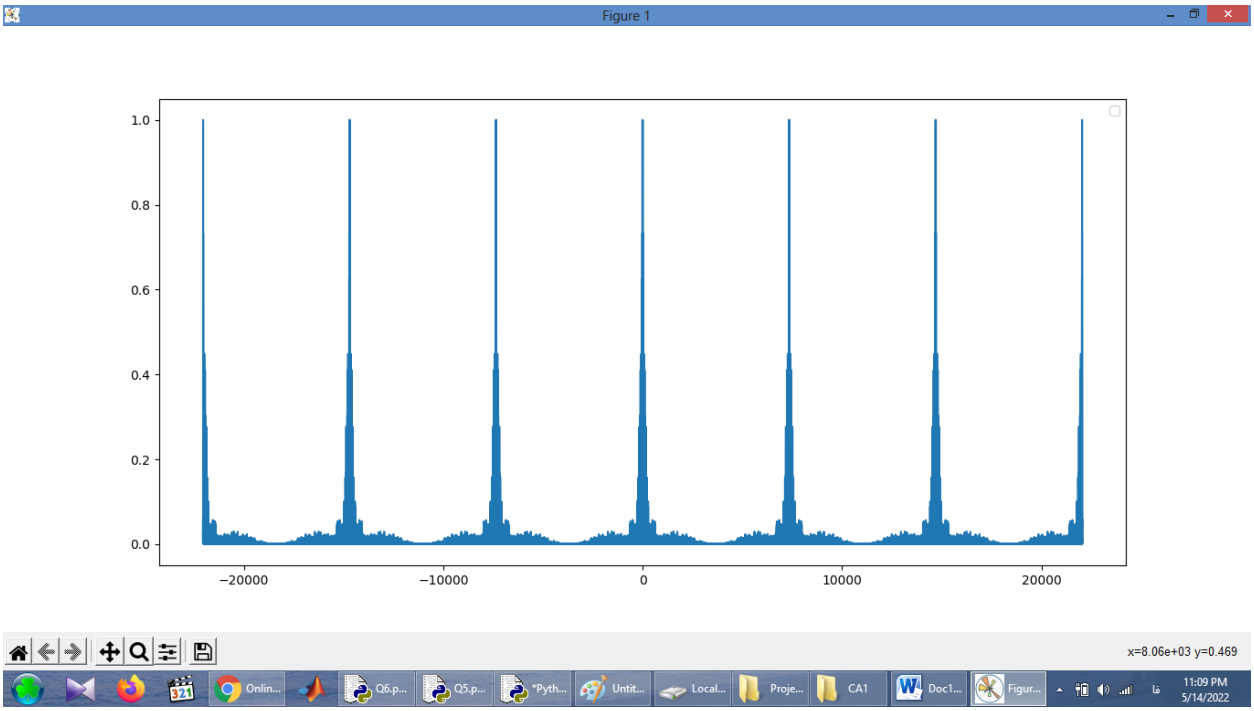
همان طور که مشاهده می شود:

هر چقدرM بزرگتر می شود، سیگنال در حوزه فرکانس، جمع تر و فشرده تر می شود.

:2

Figure 1
1.0
0.8
0.6
0.4
0.2
0.0
−20000
−10000
0
10000
20000
x=6.09e+03 y=0.706
11:09 PM
5/14/2022

Figure 1
1.0
0.8
0.6
0.4
0.2
0.0
−20000
−10000
0
10000
20000
x=4.07e+03 y=0.698
11:09 PM
5/14/2022



همان طور که مشاهده می شود:

هر چقدر L بزرگتر می شود، سیگنال در حوزه فرکانس، باز تر و منبسط تر می شود و درنتیجه آن درهم رفتگی فرکانسی aliasing رخ می دهد.

سوال 6

Q6.py - D:/DSP/CA1/Q6.py (3.8.4)

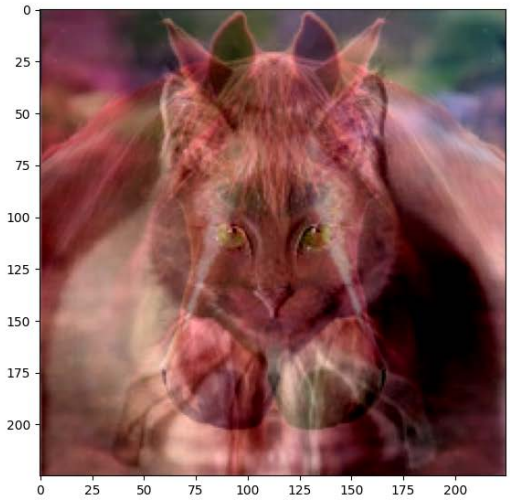
File Edit Format Run Options Window Help

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

#reading images
image_file_1 = "D:/DSP/CA1/cat.jpg"
image_1 = Image.open(image_file_1)
iq = plt.imread(image_file_1)
image_file_2 = "D:/DSP/CA1/horse.jpg"
image_2 = Image.open(image_file_2)
image_2 = image_2.resize(image_1.size)


#defining 2d fft of images and their real and imaginary parts
DTFT_image_1 = np.fft.fft2(image_1)

image_1 = np.fft.ifft2(DTFT_image_1)
image_1 = np.array(image_1.real, dtype = int)


amplitude_of_DTFT_image_1 = np.real(DTFT_image_1)
phase_of_DTFT_image_1 = np.imag(DTFT_image_1)

DTFT_image_2 = np.fft.fft2(image_2)
amplitude_of_DTFT_image_2 = np.real(DTFT_image_2)
phase_of_DTFT_image_2 = np.imag(DTFT_image_2)

DTFT_image_1 = amplitude_of_DTFT_image_2 + 1j * phase_of_DTFT_image_1
DTFT_image_2 = amplitude_of_DTFT_image_1 + 1j * phase_of_DTFT_image_2

image_1 = np.fft.ifft2(DTFT_image_1)
image_1 = np.array(image_1.real, dtype = int)
image_2 = np.fft.ifft2(DTFT_image_2)
image_2 = np.array(image_2.real, dtype = int)

plt.imshow(image_1)
plt.show()

plt.imshow(image_2)
plt.show()
```

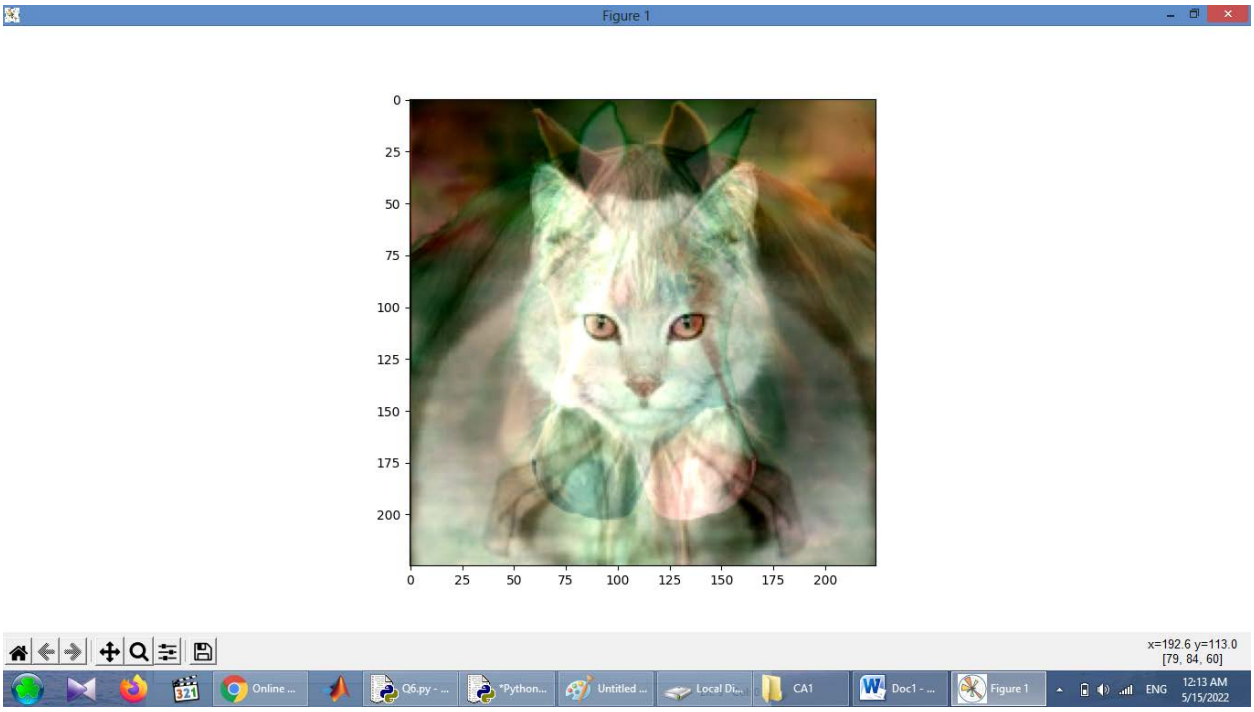
Ln: 39 Col: 18

Online ... Q6.py - ... *Python... Untitled ... Local Di... CA1 Doc1 - ... Figure 1 ENG 12:12 AM 5/15/2022

Figure 1



Online ... Q6.py - ... *Python... Untitled ... Local Di... CA1 Doc1 - ... Figure 1 ENG 12:13 AM 5/15/2022



همان طور که دیده می شود، بر اثر جابجا کردن اندازه دو تبدیل فوریه، اثراتی از عکس اول در عکس دوم و اثراتی از عکس دوم در عکس اول دیده می شود. که این کمی باعث به هم ریختگی عکس نیز شده است. دامنه دارای اطلاعات مهمتری است.