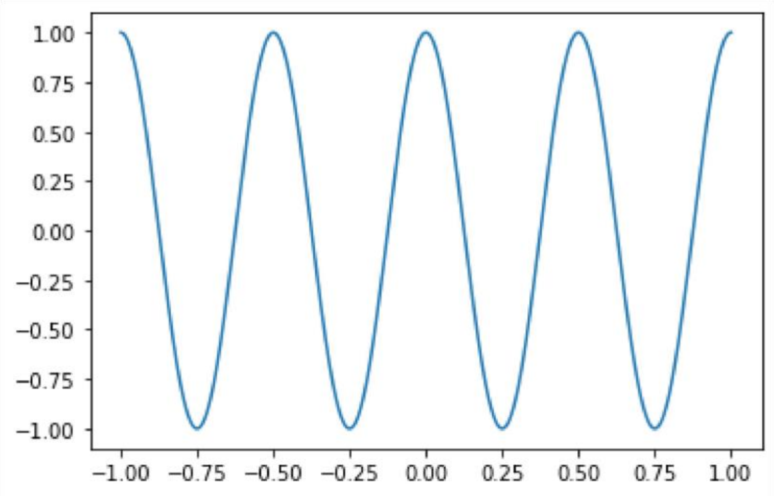
INDEX NO : 200014B GROUP NO : A - (01)

```
import numpy as np
print(np.pi, np.sin(np.pi/2))
```

```
3.141592653589793 1.0
```

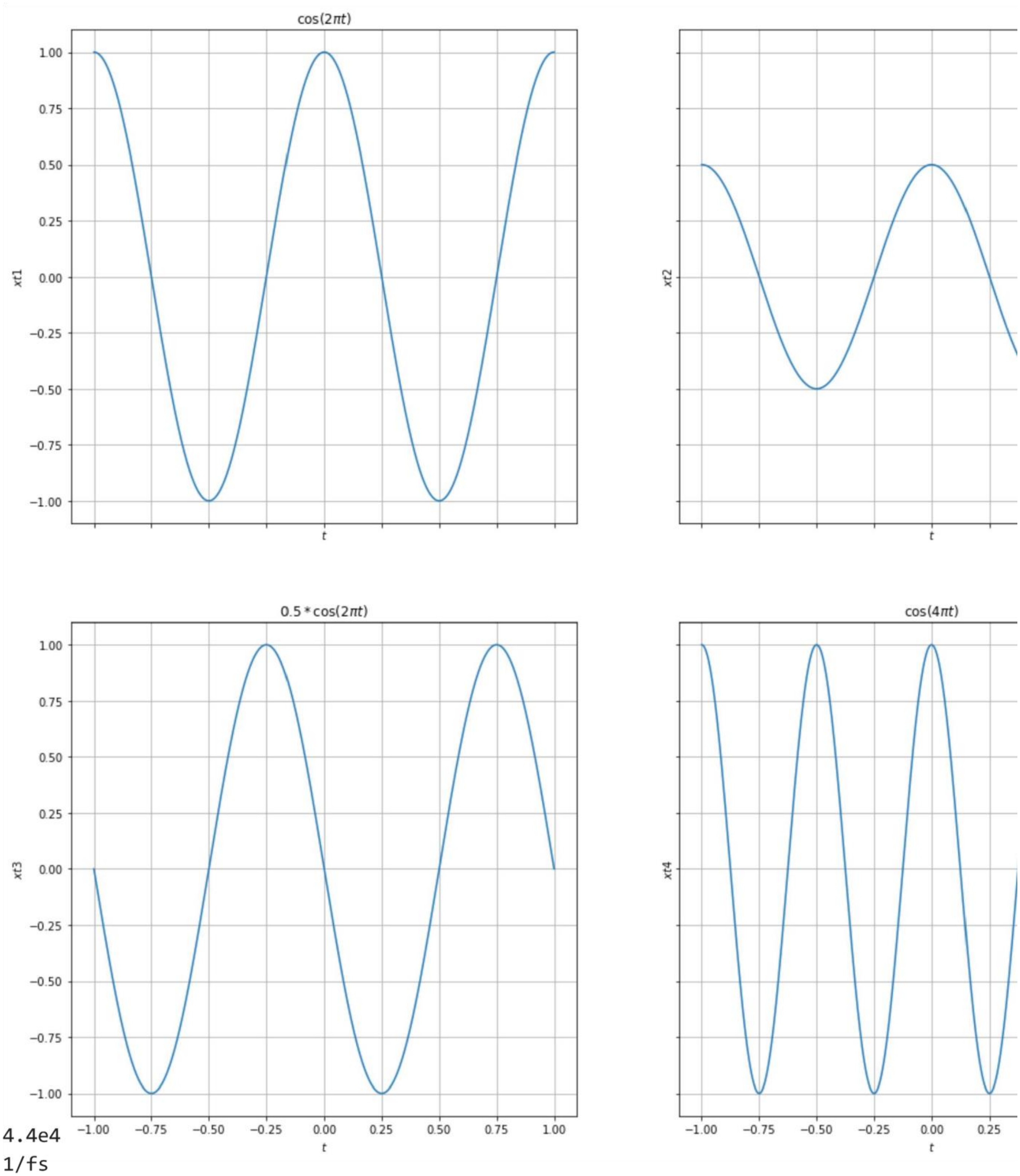
```
import matplotlib.pyplot as plt
plt.plot(np.linspace(0,10,10), np.linspace(0,10,10))
```

```
fs = 44100
ts = 1/fs
t = np.arange(-1., 1., ts)
A = 1.
f = 2
w0 = 2*np.pi*f
phi = 0.
xt = A*np.cos(w0*t + phi)
fig, ax = plt.subplots()
ax.plot(t,xt)
plt.show()
```

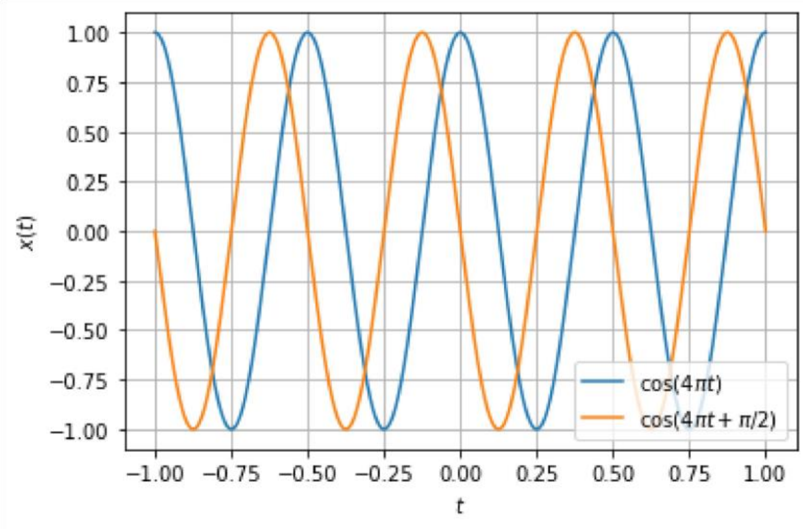


```
fs = 44100
ts = 1/fs
t = np.arange(-1., 1., ts)
f = 2
fig, axes = plt.subplots(2,2, sharex='all', sharey='all', figsize=(18,18))

𝐴 = 1
𝜔0 = 2*np.pi
phi = 0
xt1 = np.cos(2*np.pi*𝑡)
𝐴 = 0.5
```

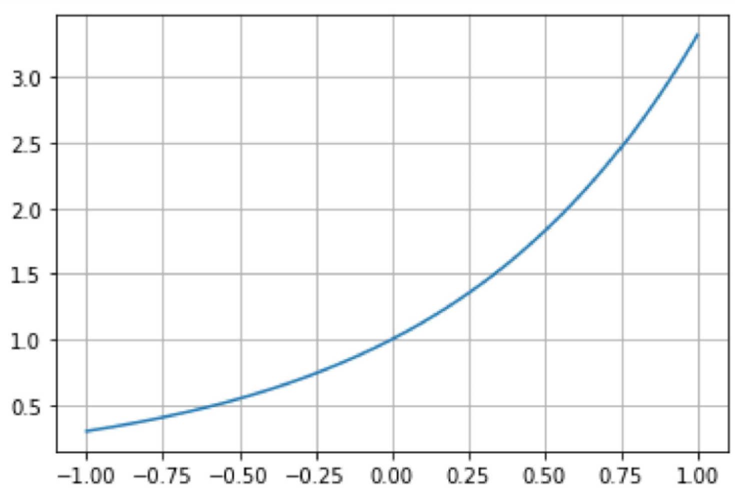
```
ω0 = 2*np.pi
phi = 0
xt2 = 0.5*np.cos(2*np.pi*t)
A = 1
ω0 = 2*np.pi
phi = 0
xt3 = np.sin(-2*np.pi*t)
A = 1
ω0 = 4*np.pi
phi = np.pi/2
xt4 = np.cos(4*np.pi*t)
axes[0, 0].plot(t,xt1)
axes[0, 0].set_xlabel('$t$')
axes[0, 0].set_ylabel('$xt1$')
axes[0, 0].grid(True)
axes[0, 0].title.set_text('$\cos(2\pi t)$')

axes[0, 1].plot(t,xt2)
axes[0, 1].set_xlabel('$t$')
axes[0, 1].set_ylabel('$xt2$')
axes[0, 1].grid(True)
axes[1, 0].title.set_text('$0.5*\cos(2\pi t)$')
axes[1, 0].plot(t,xt3)
axes[1, 0].set_xlabel('$t$')
axes[1, 0].set_ylabel('$xt3$')
axes[1, 0].grid(True)
axes[1, 1].title.set_text('$\cos(2\pi t + \phi)$')
axes[1, 1].plot(t,xt4)
axes[1, 1].set_xlabel('$t$')
axes[1, 1].set_ylabel('$xt4$')
axes[1, 1].grid(True)
axes[1, 1].title.set_text('$\cos(4\pi t)$')
plt.show()
```



```
fs = 4.4e4
ts = 1/fs
t = np.arange(-1., 1., ts)
f = 2
w0 = 2*np.pi*f
phi = 0.
x1t = np.cos(w0*t + phi)
phi = np.pi/2
x2t = np.cos(w0*t + phi)
fig, ax = plt.subplots()
ax.plot(t,x1t, label='$\cos(4\pi t)$')
ax.plot(t,x2t, label='$\cos(4\pi t + \pi/2)$')
ax.set_xlabel('$t$')
ax.set_ylabel('$x(t)$')
plt.legend(loc='lower right')
ax.grid(True)
plt.show()
```



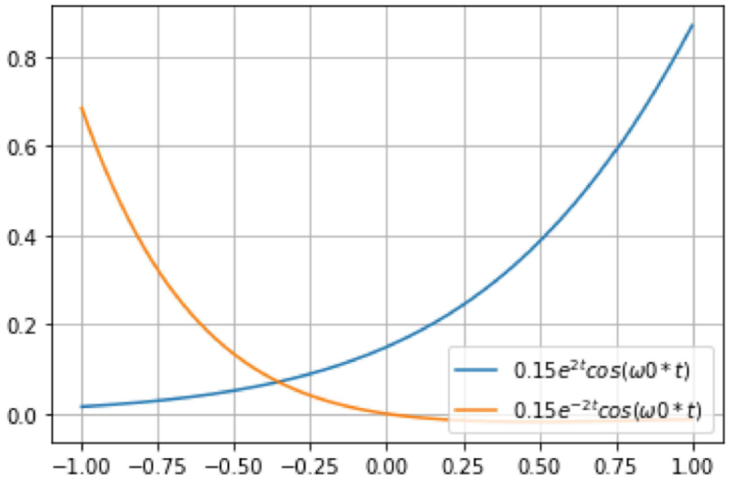
```
import numpy as np
import matplotlib.pyplot as plt

fs = 4.4e4
ts = 1/fs
t = np.arange(-1., 1., ts)

C = 1.
t0 = 0
alpha = 1.2
xt = C*np.exp(alpha*t + t0)
fig, ax = plt.subplots()
ax.plot(t, xt, label='$e^{1.2t}$')
ax.grid(True)
plt.show()
```



Comment on the effect due to the sign of α on the signal.

1. If α is positive exponential growth
2. If α is negative exponentiao decay

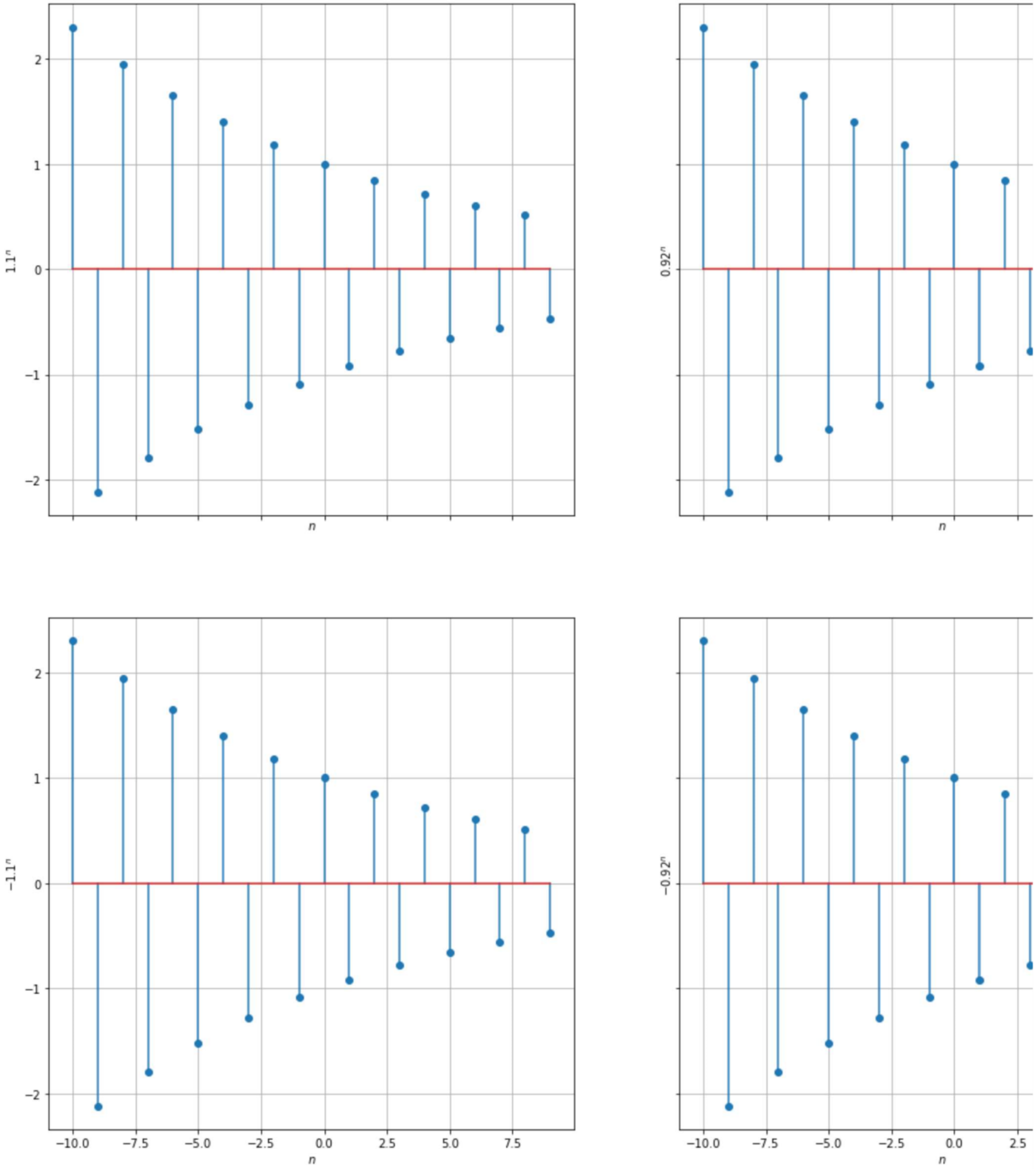
```
fs = 4.4e4
ts = 1/fs
t = np.arange(-1., 1., ts)
f = 5

C = 0.15
r = 2
phi = 0
omega0 = 2*np.pi*f
xt1 = C*np.exp(r*t)*np.cos(w0*t+phi)

C = 0.15
r = -2
phi = np.pi/2
omega0 = 2*np.pi*f
xt2 = C*np.exp(r*t)*np.cos(w0*t+phi)

fig, ax = plt.subplots()
ax.plot(t, xt1, label='$0.15e^{2t}cos(\omega0*t)$')
ax.plot(t, xt2, label='$0.15e^{-2t}cos(\omega0*t)$')
ax.legend(loc='lower right')
ax.grid(True)
plt.show()
```



```
n = np.arange(-10, 10, 1)
fig, axes = plt.subplots(2,2, sharex='all', sharey='all', figsize=(18,18))
C = 1.0
alpha = 1.1
xn1 = C*alpha**(n)
```

```
C = 1.0
alpha = 0.92
xn1 = C*alpha**(n)

C = 1.0
alpha = -1.1
xn1 = C*alpha**(n)

C = 1.0
alpha = -0.92
xn1 = C*alpha**(n)

axes[0, 0].stem(n,xn1)
axes[0, 0].set_xlabel('$n$')
axes[0, 0].set_ylabel('$1.1^{n}$')
axes[0, 0].grid(True)

axes[0, 1].stem(n,xn1)
axes[0, 1].set_xlabel('$n$')
axes[0, 1].set_ylabel('$0.92^{n}$')
axes[0, 1].grid(True)

axes[1, 0].stem(n,xn1)
axes[1, 0].set_xlabel('$n$')
axes[1, 0].set_ylabel('$-1.1^{n}$')
axes[1, 0].grid(True)

axes[1, 1].stem(n,xn1)
axes[1, 1].set_xlabel('$n$')
axes[1, 1].set_ylabel('$-0.92^{n}$')
axes[1, 1].grid(True)

plt.show()
```

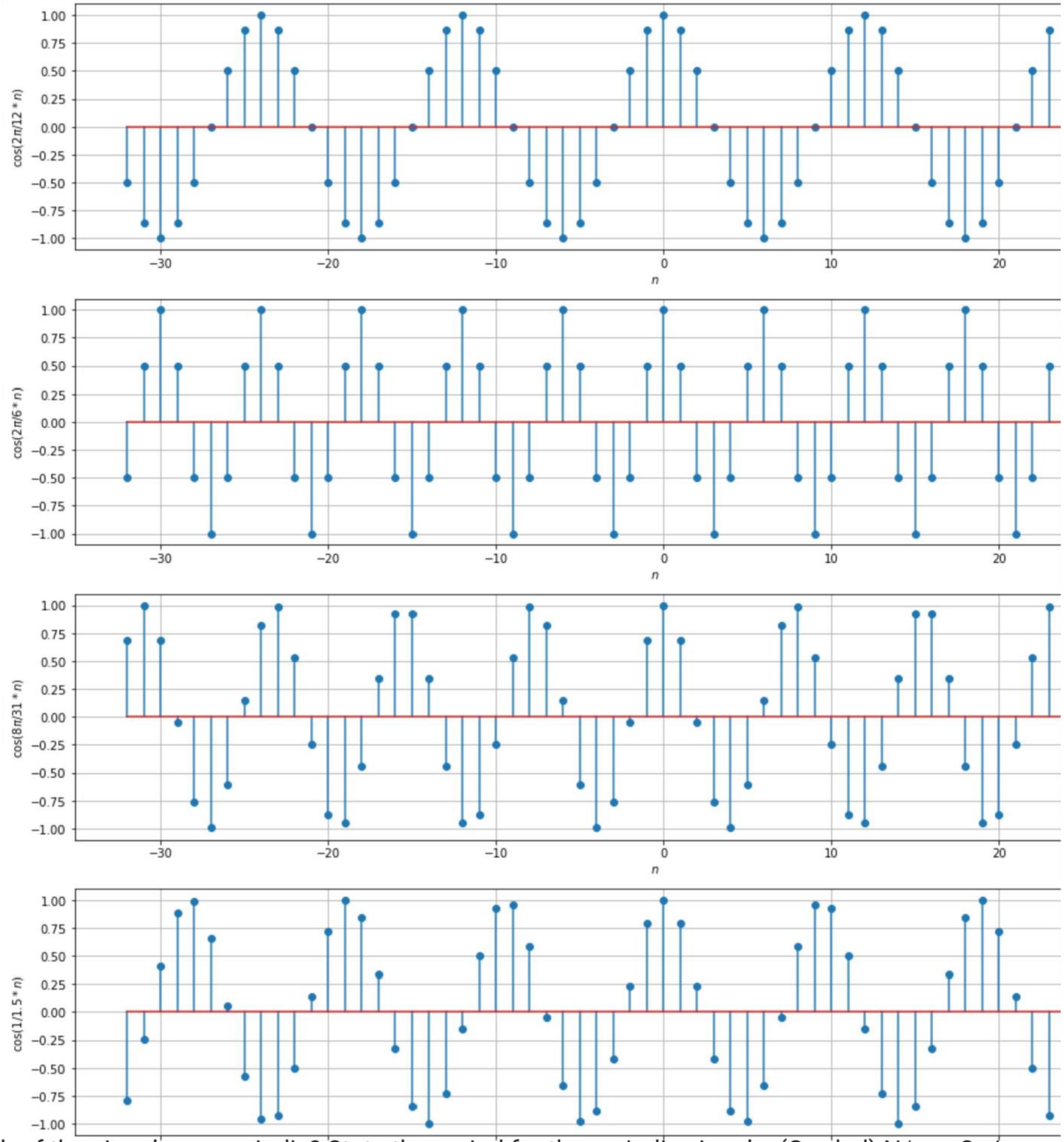
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launc er.py:  : UserWarning: In Matplot
                    .

                                                            h      19
/usr/local/lib/python3 7/dist-packages/ipykernel_launcher.py:24: UserWarning: In Matplot
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:29: UserWarning: In Matplot
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:34: UserWarning: In Matplot
```



```
n = np.arange(-32,32,1)
fig, axes = plt.subplots(4,1, sharey='all', figsize=(18,18))

A = 1
w0 = 2*np.pi/12
xn1 = A*np.cos(w0*n)
```

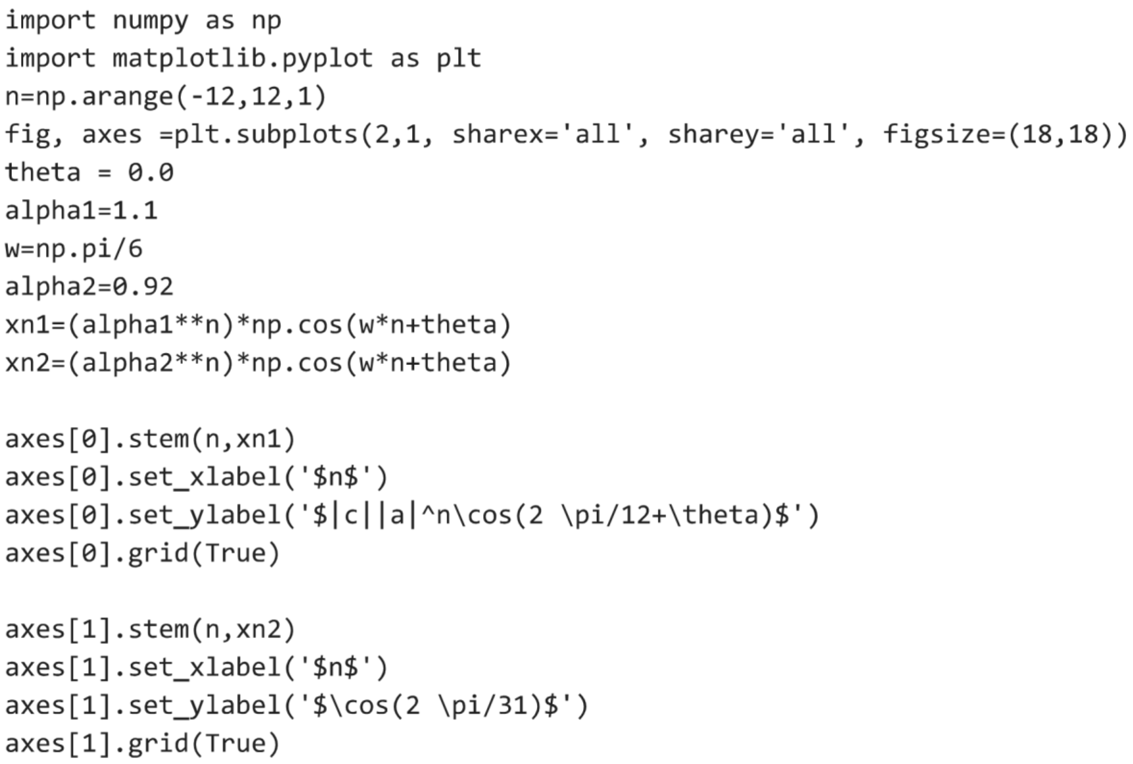
```
A = 1
w0 = 2*np.pi/6
xn2 = A*np.cos(w0*n)

A = 1
w0 = 8*np.pi/31
xn3 = A*np.cos(w0*n)

A = 1
w0 = 1/1.5
xn4 = A*np.cos(w0*n)

axes[0].stem(n,xn1)
axes[0].set_xlabel('$n$')
axes[0].set_ylabel('$\cos(2\pi/12*n)$')
axes[0].grid(True)

axes[1].stem(n,xn2)
axes[1].set_xlabel('$n$')
axes[1].set_ylabel('$\cos(2\pi/6*n)$')
axes[1].grid(True)

axes[2].stem(n,xn3)
axes[2].set_xlabel('$n$')
axes[2].set_ylabel('$\cos(8\pi/31*n)$')
axes[2].grid(True)

axes[3].stem(n,xn4)
axes[3].set_xlabel('$n$')
axes[3].set_ylabel('$\cos(1/1.5*n)$')
axes[3].grid(True)

plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launc er.py:  : UserWarning: In Matplot
                    .

                                                          h      20
/usr/local/lib/python3 7/dist-packages/ipykernel_launcher.py:25: UserWarning: In Matplot
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:30: UserWarning: In Matplot
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:35: UserWarning: In Matplot
```



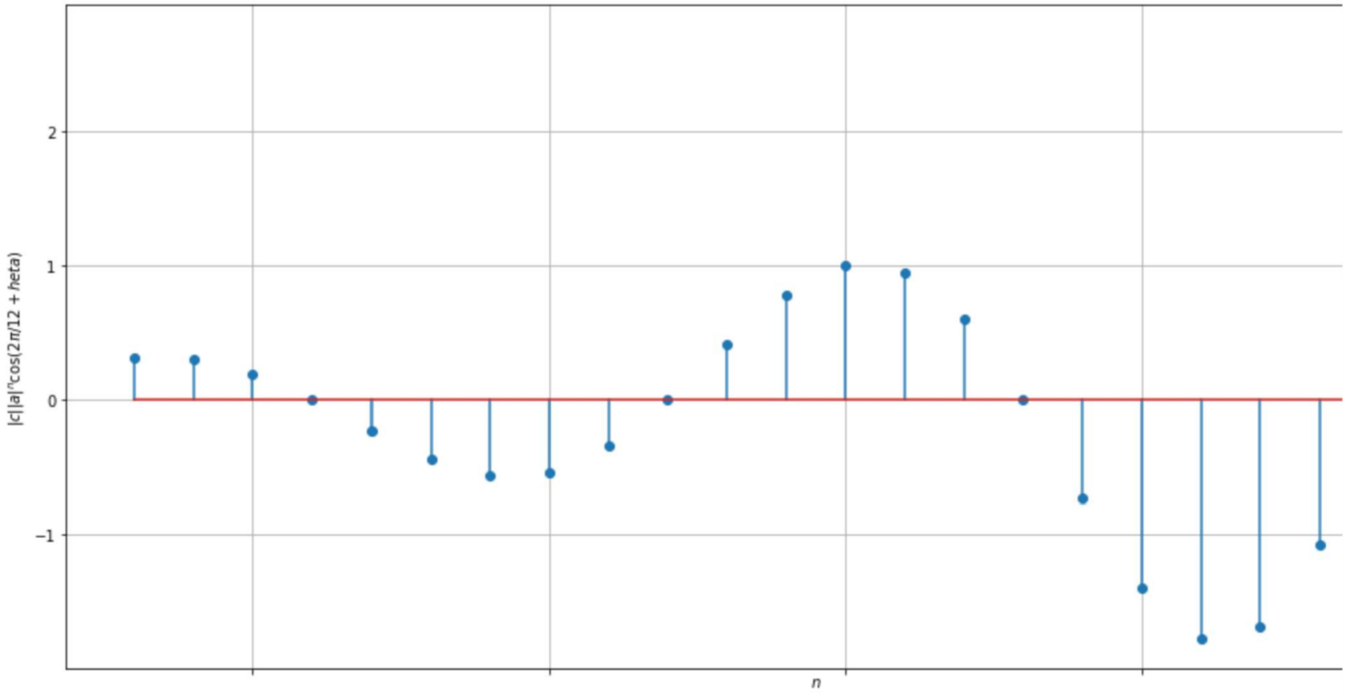
Which of the signals are periodic? State the period for the periodic signals. (Graded) N/m = 2π/w - mustbe an integer

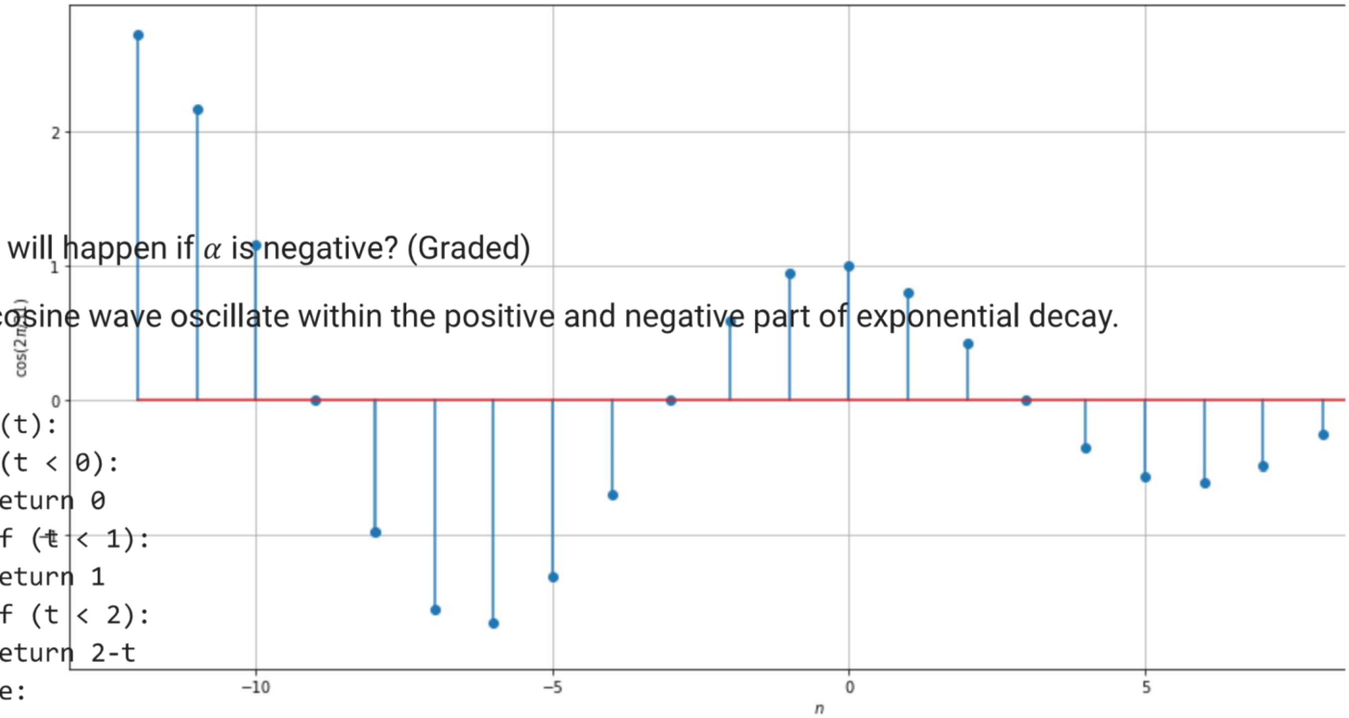
1. $x1[n]$ - Periodic signal
2. $x2[n]$ - periodic signal
3. $x3[n]$ - periidic signal
4. $x4[n]$ - aperiodic signal



```
import numpy as np
import matplotlib.pyplot as plt
n=np.arange(-12,12,1)
fig, axes =plt.subplots(2,1, sharex='all', sharey='all', figsize=(18,18))
theta = 0.0
alpha1=1.1
w=np.pi/6
alpha2=0.92
xn1=(alpha1**n)*np.cos(w*n+theta)
xn2=(alpha2**n)*np.cos(w*n+theta)

axes[0].stem(n,xn1)
axes[0].set_xlabel('$n$')
axes[0].set_ylabel('$|c||a|^n\cos(2 \pi/12+\theta)$')
axes[0].grid(True)

axes[1].stem(n,xn2)
axes[1].set_xlabel('$n$')
axes[1].set_ylabel('$\cos(2 \pi/31)$')
axes[1].grid(True)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launc er.py:  : UserWarning: In Matplot
                      .

                                                 h     12
  if sys.path[0] == '':
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:17: UserWarning: In Matplot
```





What will happen if α is negative? (Graded)

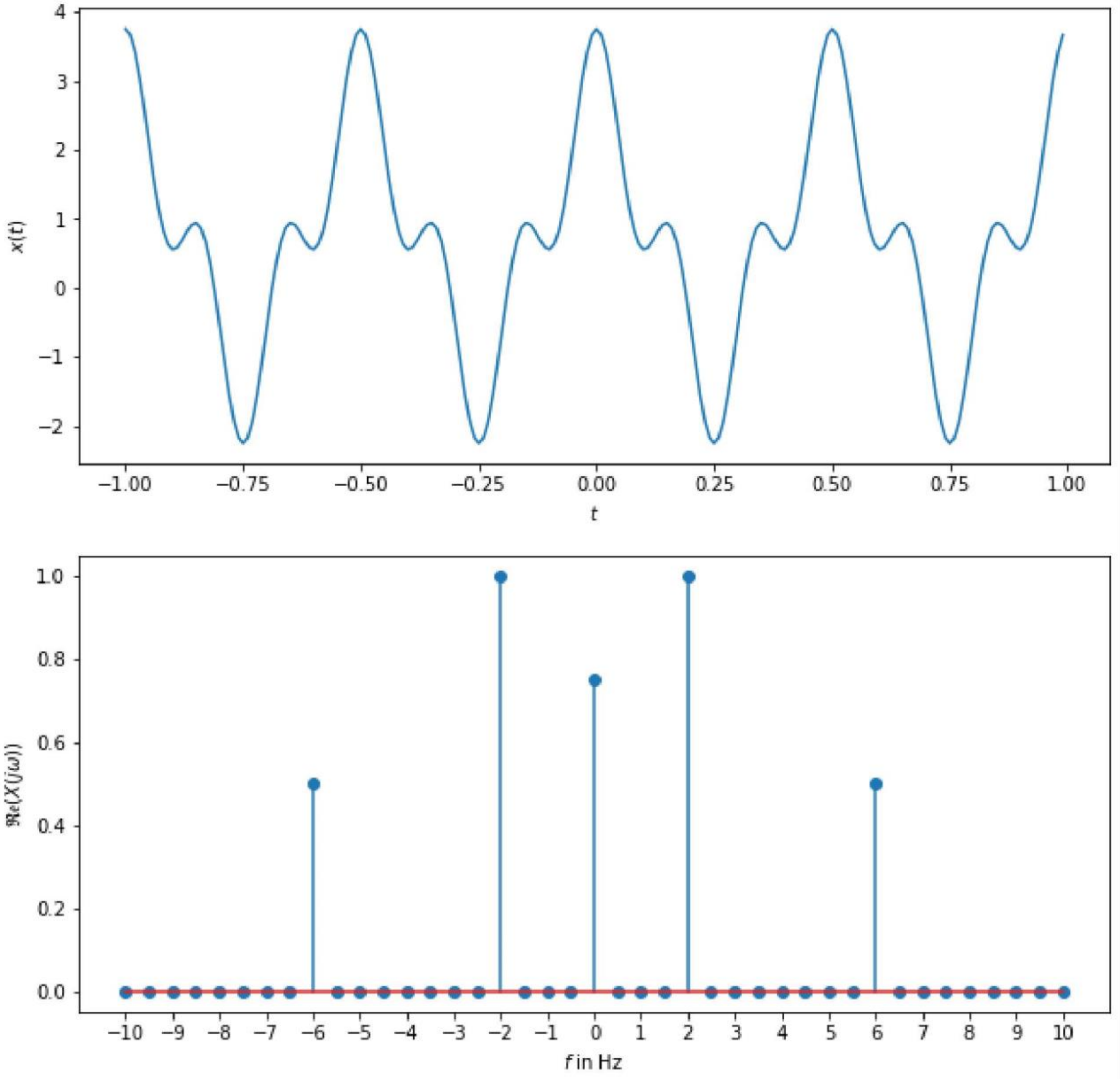
The cosine wave oscillate within the positive and negative part of exponential decay.

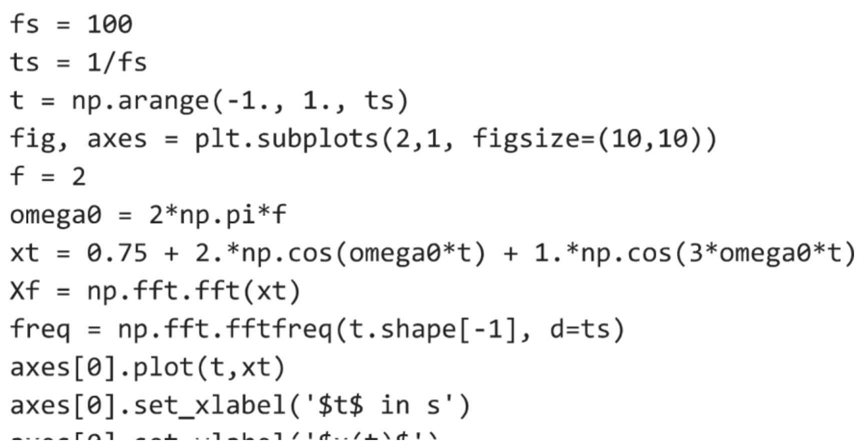
```
def x(t):
  if (t < 0):
    return 0
  elif (t < 1):
    return 1
  elif (t < 2):
    return 2-t
  else:
    return 0
fs = 4.4e4 # 44,000-Hz sampling frequency
ts = 1/fs
t = np.arange(-3.5,3.5, ts) # A linearly-spaced array with step ts
fig, axes = plt.subplots(7,1, sharey='all', figsize=(10,15))
# x(t)
axes[0].plot(t, [x(t_) for t_ in t])
axes[0].set_xlabel('$t$')
```

```
axes[0].set_ylabel('$x(t)$')
axes[0].grid(True)

axes[1].plot(t,[x(t_-1) for t_ in t ])
axes[1].set_xlabel('$t$')
axes[1].set_ylabel('$x(t-1)$')
axes[1].grid(True)

axes[2].plot(t,[x(t_+1) for t_ in t])
axes[2].set_xlabel('$t$')
axes[2].set_ylabel('$x(t+1)$')
axes[2].grid(True)

axes[3].plot(t,[x(-1*t_) for t_ in t ])
axes[3].set_xlabel('$t$')
axes[3].set_ylabel('$x(-t)$')
axes[3].grid(True)

axes[4].plot(t,[x(-1*t1+1) for t1 in t ])
axes[4].set_xlabel('$t$')
axes[4].set_ylabel('$x(-t+1)$')
axes[4].grid(True)

axes[5].plot(t,[x(1.5*t1) for t1 in t ])
axes[5].set_xlabel('$t$')
axes[5].set_ylabel('$x(3t/2)$')
axes[5].grid(True)

axes[6].plot(t,[x(1.5*t1+1) for t1 in t ])
axes[6].set_xlabel('$t$')
axes[6].set_ylabel('$x(3t/2+1)$')
axes[6].grid(True)
plt.show()

fs=100
ts=1/fs
t=np.arange(-1,1,ts)
fig,axes=plt.subplots(2,1,figsize=(10,10))
f=2
w0=2*np.pi*f
xt=0.75+ 2*np.cos(w0*t) + np.cos(3*w0*t)
Xf=np.fft.fft(xt)
freq=np.fft.fftfreq(t.shape[-1],d=ts)
axes[0].plot(t,xt)
axes[0].set_xlabel('$t$')
axes[0].set_ylabel('$x(t)$')
valsubrange=np.concatenate((np.arange(0,21,1),np.arange(-1,-21,-1)))
freqsubrange=np.concatenate((np.arange(0,21,1),np.arange(-1,-21,-1)))
axes[1].stem(freq[freqsubrange],Xf.real[valsubrange]/len(t))
axes[1].set_xlabel('$f$ in Hz')
axes[1].set_ylabel('$\mathfrak{Re}(X(j\omega))$')
```

```
plt.xticks(np.arange(-10,11))
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:15: UserWarning: In Matplot
  from ipykernel import kernelapp as app
```

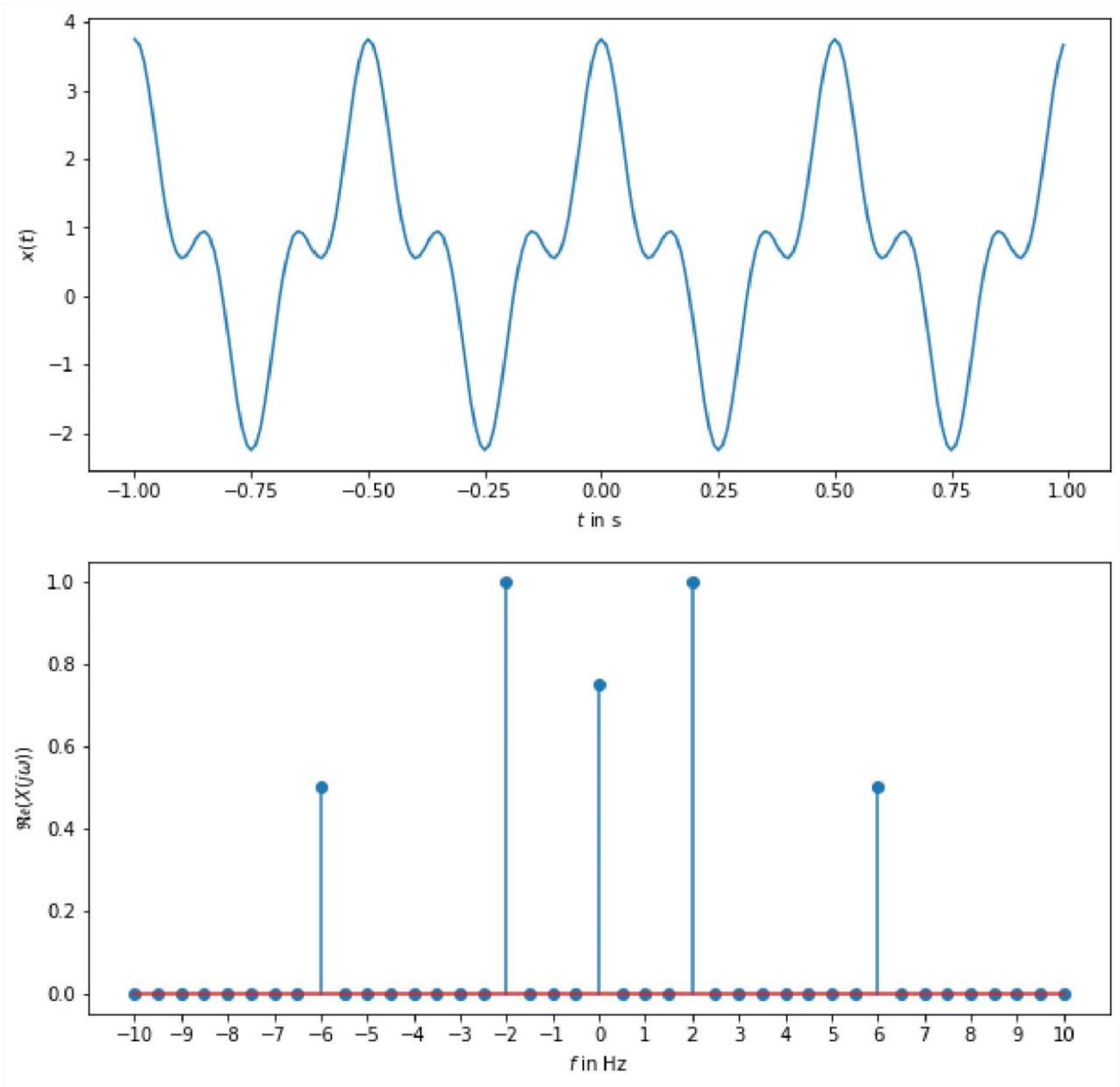




```
fs = 100
ts = 1/fs
t = np.arange(-1., 1., ts)
fig, axes = plt.subplots(2,1, figsize=(10,10))
f = 2
omega0 = 2*np.pi*f
xt = 0.75 + 2.*np.cos(omega0*t) + 1.*np.cos(3*omega0*t)
Xf = np.fft.fft(xt)
freq = np.fft.fftfreq(t.shape[-1], d=ts)
axes[0].plot(t,xt)
axes[0].set_xlabel('$t$ in s')
axes[0].set_ylabel('$x(t)$')
```

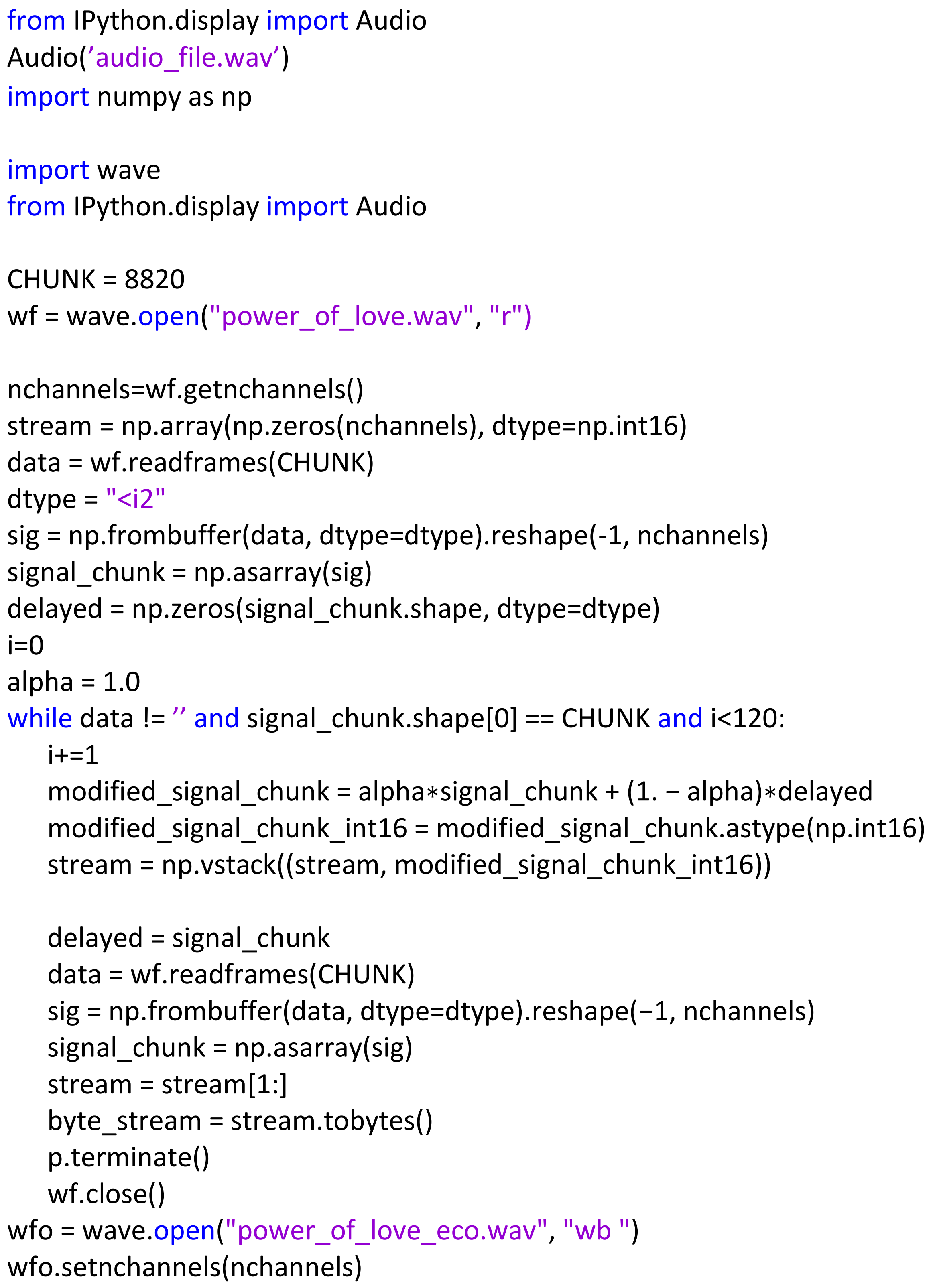
```
axes[0].set_ylabel('$x(t)$')
valsubrange = np.concatenate((np.arange(0,21,1), np.arange(-1,-21,-1)))
freqsubrage = np.concatenate((np.arange(0,21,1), np.arange(-1,-21,-1)))
axes[1].stem(freq[freqsubrage], Xf.real[valsubrange]/len(t))

axes[1].set_xlabel('$f$ in Hz')
axes[1].set_ylabel('$\mathfrak{Re}(X(j\omega))$')
plt.xticks(np.arange(-10,11))
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:15: UserWarning: In Matplot
  from ipykernel import kernelapp as app
```



Interpret the frequency domain representation. [Graded]

In above diagram we can find frequency domain representation



```
from IPython.display import Audio
Audio(’audio_file.wav’)
import numpy as np

import wave
from IPython.display import Audio

CHUNK = 8820
wf = wave.open("power_of_love.wav", "r")

nchannels=wf.getnchannels()
stream = np.array(np.zeros(nchannels), dtype=np.int16)
data = wf.readframes(CHUNK)
dtype = "<i2"
sig = np.frombuffer(data, dtype=dtype).reshape(-1, nchannels)
signal_chunk = np.asarray(sig)
delayed = np.zeros(signal_chunk.shape, dtype=dtype)
i=0
alpha = 1.0
while data != ’’ and signal_chunk.shape[0] == CHUNK and i<120:
    i+=1
    modified_signal_chunk = alpha∗signal_chunk + (1. − alpha)∗delayed
    modified_signal_chunk_int16 = modified_signal_chunk.astype(np.int16)
    stream = np.vstack((stream, modified_signal_chunk_int16))

    delayed = signal_chunk
    data = wf.readframes(CHUNK)
    sig = np.frombuffer(data, dtype=dtype).reshape(−1, nchannels)
    signal_chunk = np.asarray(sig)
    stream = stream[1:]
    byte_stream = stream.tobytes()
    p.terminate()
    wf.close()
wfo = wave.open("power_of_love_eco.wav", "wb ")
wfo.setnchannels(nchannels)
wfo.setsampwidth(wf.getsampwidth())
wfo.setframerate(wf.getframerate())
wfo.writeframes(byte_stream)
wfo.close()
Audio(’power_of_love_eco.wav’)
```

```
          0:00 / 0:24
```