ÖRNEK 1.ipynb

Gerekli kütüphanelerin çağrılması:

In [ ]:
```
import matplotlib.pyplot as plt  # grafik çizimi için gerekli kütüphanenin aktifleştirilmesi
import numpy as np               # dizi işlemleri için kütüphane aktif hale getirilir.
```

Sürekli zamanlı $x(t) = 5cos(200\pi t)$ işaretinin temel frekansı olan F=100 Hz' dir ve bilirndiği üzere temel periyot olan T ile temel frekans olan F arasında $T = 1/F$ ilişkisi vardır.

In [ ]:
```
F=100  # x işaretinin temel frekansı
T=1/F # x işaretinin temel periyodu
```

$x(t)$ işareti, iki periyot boyunca çizdirilmek istenirse aşağıdaki kodlar çalıştırılır:

In [ ]:
```
t=np.arange(0.,2*T,0.0001)     # x işaretine ait t değişkeninin tanımlanması(2 periyot boyunca)
x=5*np.cos(200*np.pi*t)  # x(t) işaretinin tanımlanması
```

Grafik çizdirme ve gösterme işlemleri:

In [ ]:
```
plt.xlabel("t")                  # grafiğin x ekseninin isimlendirilmesi
plt.title("x(t)=5*cos(200*pi*t)") # Grafik başlığının oluşturulması
plt.plot(t,x)        # işaretin grafiğinin çizdirilmesi
plt.show()      # grafiklerin gösterilmesini sağlar
```

In [ ]:
```

```

ÖRNEK 2.ipynb

Gerekli kütüphanelerin çağrılması:

In [ ]:
```
import matplotlib.pyplot as plt  # grafik çizimi için gerekli kütüphanenin aktifleştirilmesi
import numpy as np               # dizi işlemleri için kütüphane aktif hale getirilir.
```

Soruda verilen örnekleme frekansı F_s için ayrık zamanlı $x[n]$ işaretinin 2 periyot boyunca oluşturulması

In [ ]:
```
Fs=2200     # örnekleme frekansının tanımlanması
Ts=1/Fs   # örnekleme periyodunun tanımlanması
N=22      # örnek sayısının tanımlanması
n=np.arange(0.,2*N)  # örnekleme indisinin 0'dan iki periyot olacak şekilde array olarak tanımlanması
xn=5*np.cos(200*np.pi*n*Ts)  #örneklenmiş x[n] işaretinin tanımlanması
```

Grafik çizdirme ve gösterme işlemleri:

In [ ]:
```
plt.stem(n,xn)      # x[n] işaretinin grafiğinin çizdirilmesi
plt.title("x[n]")   # x[n] işaretinin grafiğinin isimlendirilmesi
plt.xlabel("örnek") # grafiğin x ekseninin isimlendirilmesi
plt.ylabel("x[n]")  # grafiğin y ekseninin isimlendirilmesi
plt.show()      # grafiğin gösterilmesi
```

In [ ]:
```

```

ÖRNEK 3.ipynb

Gerekli kütüphanelerin çağrılması:

In [ ]:
```
import matplotlib.pyplot as plt  # grafik çizimi için gerekli kütüphanenin aktifleştirilmesi
import numpy as np               # dizi işlemleri için kütüphane aktif hale getirilir.
```

Soruda verilen üç sürekli zamanlı x(t) işaretinin tanımlanması:

In [ ]:
```
t=np.arange(0.,2/60,1/4000)      # x işaretine ait t değişkeninin tanımlanması
xt_1=np.cos(2*np.pi*60*t+np.pi/3)  # x(t) işaretinin tanımlanması
xt_2=np.cos(2*np.pi*340*t-np.pi/3)  # x(t) işaretinin tanımlanması
xt_3=np.cos(2*np.pi*460*t+np.pi/3)  # x(t) işaretinin tanımlanması
```

Soruda verilen örnekleme frekansı F_s için ayrık zamanlı $x[n]$ işaretlerinin oluşturulması:

NOT: Dokümanda da belirtildiği üzere indis alt eksen uyuşmazlığı nedeniyle $x(t)$ ve $x[n]$ işaretleri aynı grafiklerde çizdirilemez. Çünkü biri sürekli zamanlı işarettir ve alt eksen t zaman indisine bağlıdır; diğeri ise ayrık zamanlı işarettir ve alt eksen n indis değerleri göstermektedir. Bu nedenle bu örnek için $x(t)$ ve $x(nT_s)$ grafikleri aynı figure de gösterilmiştir.

In [ ]:
```
Fs=400 # örnekleme frekansının tanımlanması
Ts=1/Fs # örnekleme periyodunun tanımlanması

nTs=np.arange(0.,2/60,Ts)      # x işaretine ait t değişkeninin tanımlanması( 2 periyot boyunca)

xnTs_1=np.cos(2*np.pi*60*nTs+np.pi/3)  # x1(nTs) işaretinin tanımlanması
xnTs_2=np.cos(2*np.pi*340*nTs-np.pi/3)  # x2(nTs) işaretinin tanımlanması
xnTs_3=np.cos(2*np.pi*460*nTs+np.pi/3)  # x3(nTs) işaretinin tanımlanması
```

Grafik çizdirme ve gösterme işlemleri:

In [ ]:
```
# x1(t) için:
plt.subplot(3,1,1)  # 3 grafiği alt alta gösterebilmek için kullanılır ve ilk grafiği belirtir.
plt.xlabel("t")                  # grafiğin x ekseninin isimlendirilmesi
plt.ylabel("x_1(t) için")                  # grafiğin y ekseninin isimlendirilmesi
plt.title("x(t) ve x(nTs) grafikleri")  # grafik başlığının oluşturulması
plt.plot(t,xt_1,color='orange')      # x(t) işaretinin grafiğinin zaman domaininde çizdirilmesi
plt.stem(nTs,xnTs_1)  # x(nTs) işaretinin grafiğinin çizdirilmesi
plt.hold()    # iki işaretin aynı grafik üzerinde gösterilmesi

# x2(t) için:
plt.subplot(3,1,2)  # 3 grafiği alt alta gösterebilmek için kullanılır ve ikinci grafiği belirtir.
plt.xlabel("t")                  # grafiğin x ekseninin isimlendirilmesi
plt.ylabel("x_2(t) için")                  # grafiğin y ekseninin isimlendirilmesi
plt.plot(t,xt_2,color='orange')      # x(t) işaretinin grafiğinin zaman domaininde çizdirilmesi
plt.stem(nTs,xnTs_2)  # x(nTs) işaretinin grafiğinin çizdirilmesi
plt.hold()    # iki işaretin aynı grafik üzerinde gösterilmesi

# x3(t) için:
plt.subplot(3,1,3)  # 3 grafiği alt alta gösterebilmek için kullanılır ve üçüncü grafiği belirtir.
plt.xlabel("t")                  # grafiğin x ekseninin isimlendirilmesi
plt.ylabel("x_3(t) için")                  # grafiğin y ekseninin isimlendirilmesi
plt.plot(t,xt_3,color='orange')      # x(t) işaretinin grafiğinin zaman domaininde çizdirilmesi
plt.stem(nTs,xnTs_3)  # x(nTs) işaretinin grafiğinin çizdirilmesi
plt.hold()    # iki işaretin aynı grafik üzerinde gösterilmesi
plt.show()    # grafiklerin gösterilmesi
```

In [ ]:
```
```

ÖRNEK 4.ipynb

In [ ]:
```
import matplotlib.pyplot as plt  # grafik çizimi için gerekli kütüphanenin aktifleştirilmesi
import numpy as np               # dizi işlemleri için kütüphane aktif hale getirilir.
```

In [ ]:
```
def signall(Fs,N,k):    # Fs: Örnekleme frekansı
                        # N : çizdirilmek istenen toplam örnek sayısı
                        # k = x(t) işaretinin kaç periyot çizdirileceği
    F=100
    T=1/F
    t = np.linspace(-k*T/2,k*T/2,1000)
    xt = np.sinc(2*F*t)**2   # sürekli zamanlı işaret

    Ts=1/Fs
    n = np.arange(-N/2, N/2) # x[n] işaretinin indis ekseninin tanımlanması
    xn = np.sinc(2*F*n*Ts)**2  # x[n] işaretinin tanımlanması

    # x[n] işaretinin fourier transformu
    w = np.arange(-np.pi, np.pi, 2*np.pi/N)  # omega ekseninin -pi ile +pi arasında tanımlanması
    xw = np.fft.fftshift(np.fft.fft(xn,N)/N)  # ayrık zamanlı işaretin Fourier transformu

    plt.figure()
    plt.subplot(1,3,1)
    plt.plot(t,xt)   # sürekli zamanlı x(t) işaretinin grafiğinin çizdirilmesi
    plt.xlabel("t (sn)")  # grafiğin x ekseninin isimlendirilmesi
    plt.title("$x(t)$")     # grafiğin y ekseninin isimlendirilmesi

    plt.subplot(1,3,2)
    plt.stem(n,xn)      # x[n] işaretinin grafiğinin çizdirilmesi
    plt.xlabel("n (örnek)")  # grafiğin x ekseninin isimlendirilmesi
    plt.title("$x[n]$")     # grafiğin y ekseninin isimlendirilmesi

    plt.subplot(1,3,3)
    plt.stem(w/np.pi,abs(xw))      # X(w) işaretinin grafiğinin çizdirilmesi
    plt.xlabel("$\omega$ / $\pi$")  # grafiğin x ekseninin isimlendirilmesi
    plt.title("$X(\omega)$ (Genlik)")     # grafiğin y ekseninin isimlendirilmesi
    plt.show()
```

In [ ]:
```
signall(1000,30,3)  # Fs= 1000 Hz ,  N = 30 örnek , x(t) toplam 3 periyot için çizdirildi
signall(600,30,5)   # Fs= 600 Hz ,  N = 30 örnek , x(t) toplam 5 periyot içi çizdirildi
signall(300,30,10)  # Fs= 300 Hz ,  N = 30 örnek , x(t) toplam 10 periyotta çizdirildi
signall(100,30,30)  # Fs= 100 Hz ,  N = 30 örnek , x(t) toplam 3 periyotta çizdirildi
```

In [ ]:
```

```

ÖRNEK 5.ipynb

Gerekli kütüphanelerin çağrılması:

In [ ]:
```
import matplotlib.pyplot as plt  # grafik çizimi için gerekli kütüphanenin aktifleştirilmesi
import numpy as np               # dizi işlemleri için kütüphane aktif hale getirilir.
```

Soruda verilen örnekleme frekansı F_s için ayrık zamanlı $x[n]$ işaretinin 2 periyot boyunca oluşturulması

In [ ]:
```
Fs=2200     # örnekleme frekansının tanımlanması
Ts=1/Fs   # örnekleme periyodunun tanımlanması
N=22      # örnek sayısının tanımlanması
n=np.arange(0.,2*N)  # örnekleme indisinin 0'dan iki periyot olacak şekilde array olarak tanımlanması
xn=5*np.cos(200*np.pi*n*Ts)  #örneklenmiş x[n] işaretinin tanımlanması
```

Grafik çizdirme ve gösterme işlemleri:

In [ ]:
```
plt.stem(n,xn)      # x[n] işaretinin grafiğinin çizdirilmesi
plt.title("x[n]")   # x[n] işaretinin grafiğinin isimlendirilmesi
plt.xlabel("örnek") # grafiğin x ekseninin isimlendirilmesi
plt.ylabel("x[n]")  # grafiğin y ekseninin isimlendirilmesi
plt.show()      # grafiklerin gösterilmesini sağlar
```

Şimdi bu $x[n]$ işaretinin Fourier transformunu alalım:

In [ ]:
```
w=np.linspace(-np.pi,np.pi-(2*np.pi/N),N)   # -pi,pi aralığında N örnek olacak şekilde x ekseni değerleri etik
x_w=np.fft.fft(xn,N)/N  # x[n] işaretinin fourier transformunun alınması
x_w=np.fft.fftshift(x_w)  # x(w) işaretinin fftshift ile -pi,+pi aralığında gösterilmesi
```

Grafik çizdirme ve gösterme işlemleri:

In [ ]:
```
plt.figure() #yeni bir figure penceresi açar
plt.stem(w,np.abs(x_w))    # x(w) işaretinin genlik değerlerinin bulunması ve grafiğinin çizdirilmesi
plt.title("X(w)'nın Genlik Grafiği")     # grafiğin isimlendirilmesi
plt.xlabel("w")       # grafiğin x ekseninin isimlendirilmesi
plt.show()      # grafiklerin gösterilmesini sağlar
plt.figure() #yeni bir figure penceresi açar
plt.stem(w,np.angle(x_w))    # x_(w) işaretinin faz değerlerinin bulunması ve grafiğinin çizdirilmesi
plt.title("X(w)'nın Faz Grafiği")          # x_(w) işaretinin fazının grafiğinin isimlendirilmesi
plt.xlabel("w")             # grafiğin x ekseninin isimlendirilmesi
plt.show()         # grafiklerin gösterilmesini sağlar
```

In [ ]:
```

```

ÖRNEK 7.ipynb

Gerekli kütüphanelerin çağrılması:

In [ ]:
```
import matplotlib.pyplot as plt  # grafik çizimi için gerekli kütüphanenin aktifleştirilmesi
import numpy as np               # dizi işlemleri için kütüphane aktif hale getirilir.
```

Sürekli zamanlı $x(t) = 5cos(200\pi t)$ işaretinin temel frekansı olan F=100 Hz' dir ve bilirndiği üzere temel periyot olan T ile temel frekans olan F arasında $T = 1/F$ ilişkisi vardır.

In [ ]:
```
F=100  # x işaretinin temel frekansı
T=1/F # x işaretinin temel periyodu
```

$x(t)$ işaretini k periyot boyunca elde edelim ve bu k değerini değişken olarak tanımlayalım:

In [ ]:
```
k=4  # Bu örnek için işareti 4 periyot boyunca çizdirelim
t=np.arange(0.,k*T,0.002/22)      # x işaretine ait t değişkeninin tanımlanması
x=5*np.cos(200*np.pi*t)  # x(t) işaretinin tanımlanması
```

Grafik çizdirme ve gösterme işlemleri:

In [ ]:
```
plt.figure() # yeni bir figure penceresi açar
plt.xlabel("t")                  # grafiğin x ekseninin isimlendirilmesi
plt.title("x(t)=5*cos(200*pi*t)") # x[n] işaretinin grafiğinin isimlendirilmesi
plt.plot(t,x)      # işaretin grafiğinin çizdirilmesi
plt.show()      # grafiklerin gösterilmesini sağlar
```

$x[n]$ işaretlerinin k periyot boyunca oluşturulması ve grafiğinin çizdirilmesi:

In [ ]:
```
Fs=2200     # örnekleme frekansının tanımlanması
Ts=1/Fs   # örnekleme periyodunun tanımlanması
N=22      # örnek sayısının tanımlanması
n=np.arange(0,k*N)  # örnekleme indisinin 0'dan iki periyot olacak şekilde array olarak tanımlanması
xn=5*np.cos(200*np.pi*n*Ts)  #örneklenmiş x[n] işaretinin tanımlanması

plt.figure() # yeni bir figure penceresi açar
plt.stem(n,xn)      # x[n] işaretinin grafiğinin çizdirilmesi
plt.title("x[n]")   # x[n] işaretinin grafiğinin isimlendirilmesi
plt.xlabel("örnek") # grafiğin x ekseninin isimlendirilmesi
plt.ylabel("x[n]")  # grafiğin y ekseninin isimlendirilmesi
plt.show()      # grafiğin gösterilmesi
```

Yeniden elde etme (Reconstruction) işlemi için zaman domaininde sinc(.) ile konvolüsyon işlemi ve konvolüsyon işlemindeki her bileşenin ayrı şekilde grafikte gösterilmesi::

In [ ]:
```
x_sinc = np.zeros([k*N,np.size(x,axis=0)])  # İşlem sonucunu yazabilmek için oluşturulan yeni dizi

plt.figure() #yeni bir figure penceresi açar
for ni in n:
    x_sinc[ni,:]=xn[ni]*np.sinc((t-(ni)*Ts)/Ts)
    plt.plot(x_sinc[ni,:])

plt.title("işaretin zaman domaininde sinc(.) ile konvolüsyon işlemi ") # Grafik başlığının oluşturulması
plt.xlabel("örnek") # grafiğin x ekseninin isimlendirilmesi
plt.show() # grafiğin gösterilmesi
```

Konvolüsyon bileşenlerinin toplanarak konvolüsyon toplamının sonucunun elde edilmesi ve elde edilen $x_r(t)$ işaretinin grafiğinin çizdirilmesi:

In [ ]:
```
xr_t = np.sum(x_sinc, axis=0)
plt.figure() # yeni bir figure penceresi açar
plt.xlabel("t")                  # grafiğin x ekseninin isimlendirilmesi
plt.title("x_r(t)") # x[n] işaretinin grafiğinin isimlendirilmesi
plt.plot(t,xr_t)                      # x[n] işaretinin grafiğinin zaman domaininde çizdirilmesi
plt.show()      # grafiklerin gösterilmesini sağlar
```

In [ ]:
```

```

ÖRNEK 8.ipynb

Gerekli kütüphanelerin çağrılması:

In [ ]:
```
import matplotlib.pyplot as plt  # grafik çizimi için gerekli kütüphanenin aktifleştirilmesi
import numpy as np               # dizi işlemleri için kütüphane aktif hale getirilir.
```

Sürekli zamanlı $x(t) = 5cos(200\pi t)$ işaretini k periyot boyunca elde edip çizdirelim:

In [ ]:
```
F=100  # x işaretinin frekansı
T=1/F # x işaretinin periyodu
k=2  # Periyot sayısı
t=np.arange(0.,k*T,0.00001)      # x işaretine ait t değişkeninin tanımlanması
x=5*np.cos(200*np.pi*t)  # x(t) işaretinin tanımlanması

plt.figure() # yeni bir figure penceresi açar
plt.xlabel("t")     # grafiğin x ekseninin isimlendirilmesi
plt.title("x(t)")  # Grafiğin başlığının oluşturulması
plt.plot(t,x)   # işaretinin grafiğinin çizdirilmesi
plt.show()      # grafiğin gösterilmesi
```

$x[n]$ işaretlerinin k periyot boyunca oluşturulması:

In [ ]:
```
Fs=2200     # örnekleme frekansının tanımlanması
Ts=1/Fs   # örnekleme periyodunun tanımlanması
N=22      # bir periyottaki örnek sayısının tanımlanması
n=np.arange(0.,k*N)  # örnekleme indisinin 0'dan k periyot olacak şekilde array olarak tanımlanması
xn=5*np.cos(200*np.pi*n*Ts)  #örneklenmiş x[n] işaretinin tanımlanması
```

Kuantalama fonksiyonunun yazılması

In [ ]:
```
def quantize(x, S):
    X = x.reshape((-1,1))
    S = S.reshape((1,-1))
    dists = abs(X-S)

    nearestIndex = dists.argmin(axis=1)
    quantized = S.flat[nearestIndex]

    return quantized.reshape(x.shape)
```

İşaretin değerlerinin 'k_bit' ile ifade edileceğinin tanımlanması ve buna göre gerekli parametrelerin tanımlanması:

In [ ]:
```
k_bit= 2   # işaret değerlerinin kaç bit ile ifade edileceği
Max_range= np.amax(xn)-np.amin(xn)   # işaretin y ekseninde aldığı max değer aralığı
q=pow(2,k_bit)   # seviye sayısı
delta = Max_range/q  # adım aralığı

S_midrise = -np.amax(xn) + delta/2 + np.arange(q)*delta  # mid-rise yöntemi için formül
S_midtread = -np.amax(xn)  + np.arange(q)*delta   # mid-tread yöntemi için formül
```

Kuantalama için yazılan $quantize(x, S)$ fonksiyonunun uygun parametreler ile çağrılarak kuantalanmış işaretin elde edilmesi:

In [ ]:
```
y_midtrise = quantize(xn, S_midrise) # mid-rise yöntemi ile kuantalanmış değerler
y_midtread = quantize(xn, S_midtread) # mid-tread yöntemi ile kuantalanmış değerler
```

Örnekleme sonucunda elde edilen $x[n]$ ve kuantalanmış $x[n]$ grafiklerinin aynı figurede çizdirilmesi:

In [ ]:
```
plt.figure() #yeni bir figure penceresi açar
fig, ax = plt.subplots()
ax.stem(n,xn)      # x[n] işaretinin grafiğinin çizdirilmesi
ax.bar(n,y_midtrise,color='orange',edgecolor = "darkorange")    # kuantalanmış işaretin grafiğinin çizdirilmes
plt.xlabel("örnek") # grafiğin x ekseninin isimlendirilmesi
plt.title('midrise yöntemiyle kuantalanmış işaret')

fig, ax = plt.subplots()
ax.stem(n,xn)      # x[n] işaretinin grafiğinin çizdirilmesi
ax.bar(n,y_midtread,color='orange',edgecolor = "darkorange")    # kuantalanmış işaretin grafiğinin çizdirilmes
plt.xlabel("örnek") # grafiğin x ekseninin isimlendirilmesi
plt.title('midtread yöntemiyle kuantalanmış işaret')
plt.show()      # grafiklerin gösterilmesi
```

In [ ]:
```
```