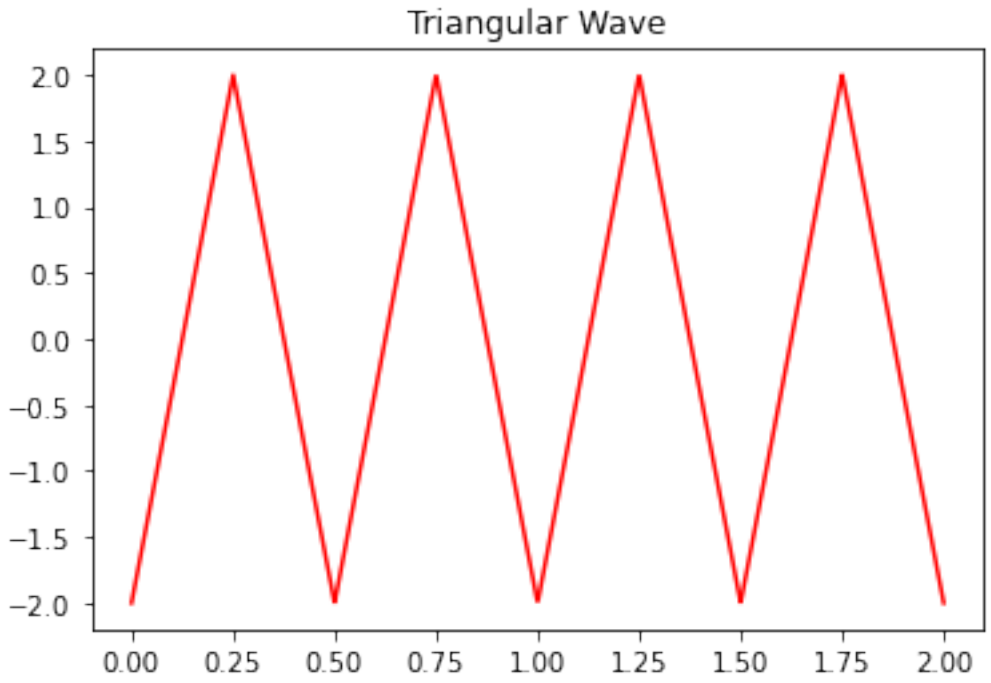
Triangular Wave Quantization

As part of this program, only the following steps need to be done

1. Generate a Triangular Wave Signal
2. Do quantization as you would normally do.

```
#Importing necessary Python libraries

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

#Generation of Triangular Wave

freq=2
amp=2
t=np.linspace(0,2,1000)
signal=amp*sg.sawtooth(2*np.pi*freq*t, width=0.5) #Plotting Triangular
Function
plt.plot(t,signal,'r')
plt.title('Triangular Wave')
```

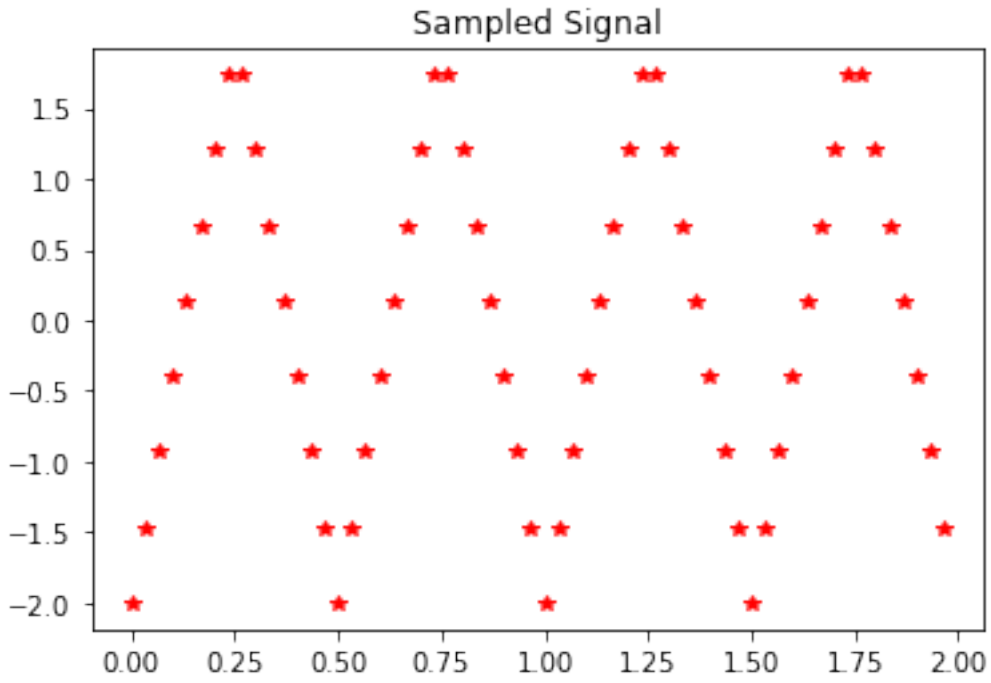
```
Text(0.5, 1.0, 'Triangular Wave')
```




```
#Sampling of the Triangular Wave Signal

fs=15*freq
ts=np.arange(0,4/freq,1/fs)
```

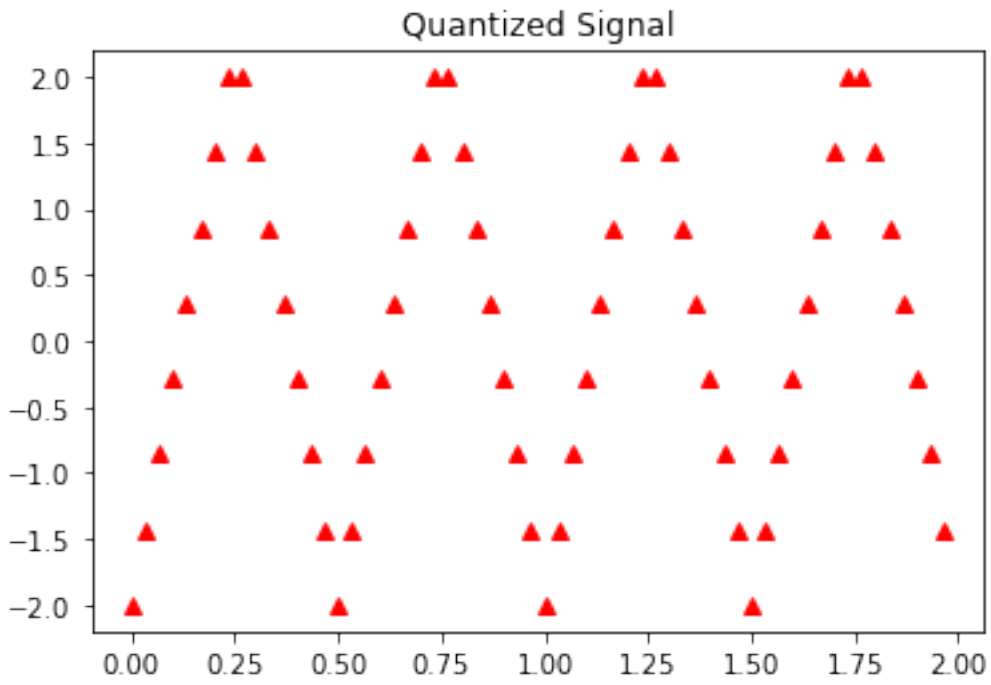
```
sampled_signal=amp*sg.sawtooth(2*np.pi*freq*ts, width=0.5)
plt.plot(ts,sampled_signal,'r*')
plt.title("Sampled Signal")
```

```
Text(0.5, 1.0, 'Sampled Signal')
```




```
#Quantization of Triangular Wave Signal

L=8 #No of Quantization levels
signal_min=min(signal)
signal_max=max(signal)
quant_lvls=np.linspace(signal_min,signal_max,L)
quantized_signal=[]
for i in sampled_signal:
    for j in quant_lvls:
        if(i<=j):
            quantized_signal.append(j)
            break
plt.plot(ts,quantized_signal,'r^')
plt.title('Quantized Signal')
```

```
Text(0.5, 1.0, 'Quantized Signal')
```



BPSK generation from BCD

Here in this program, I'm going to take the last 2 digits from my roll no. "MDL19EC047" , which is '47' in integer and convert it into BCD and hence generate a BPSK signal from it.

```
def int_to_bcd(n):
    return int((str(n)),base=16)


print("The BCD value of the integer 47 is=",int_to_bcd(47))
```

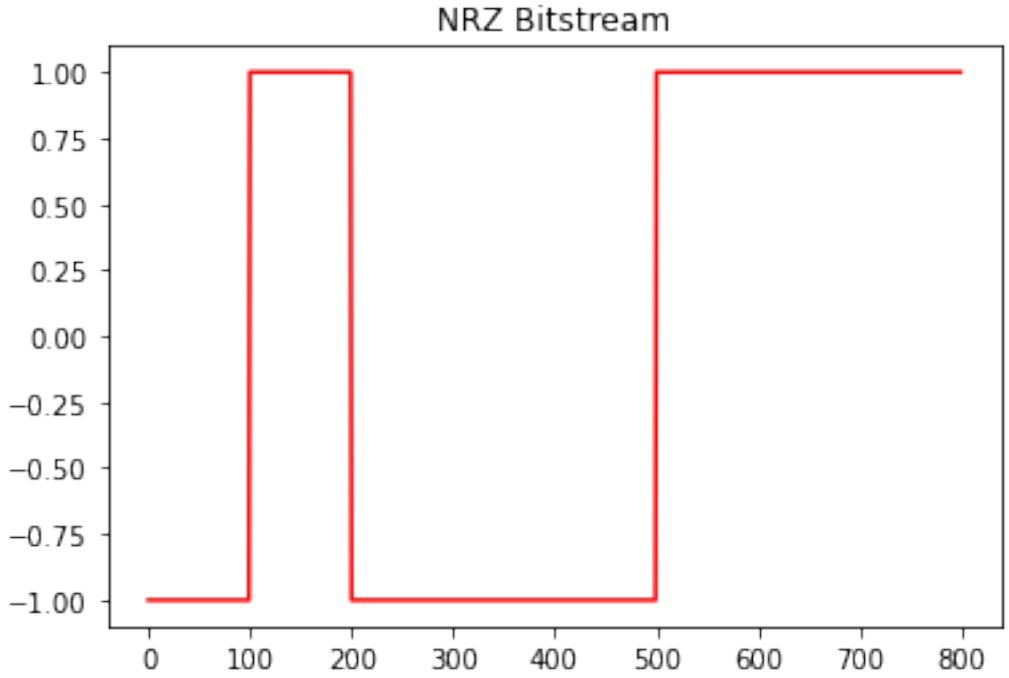
```
The BCD value of the integer 47 is= 71
```

```
def int_to_bin(n):
    return bin(n).replace("0b"," ")

binsig=int_to_bin(71)
print(binsig)
```

```
 1000111
```

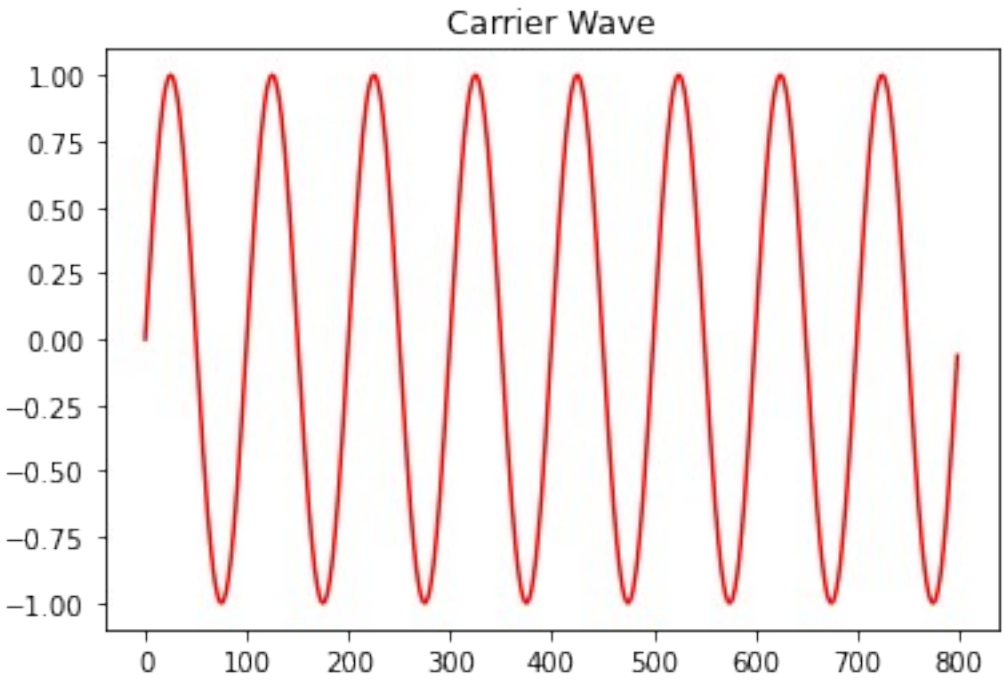
```
#Generating NRZ sequence from BCD

binsig=np.array([0,1,0,0,0,1,1,1])
binsig[binsig==0]=-1
binsig=np.repeat(binsig,100)
plt.plot(binsig,'r')
plt.title('NRZ Bitstream')
```

```
Text(0.5, 1.0, 'NRZ Bitstream')
```

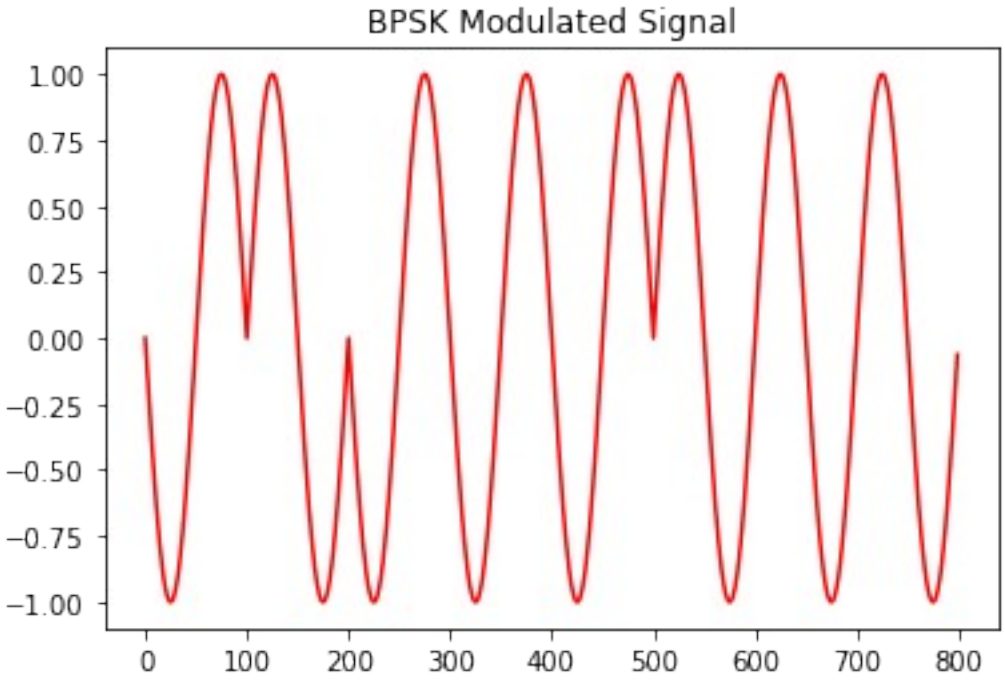



```
#Carrier Wave Coding

f=0.01
t=np.arange(0,8/f)
carwav=np.sin(2*np.pi*f*t)
plt.plot(t,carwav,'r')
plt.title("Carrier Wave")
```

```
Text(0.5, 1.0, 'Carrier Wave')
```




```
#BPSK Modulation of signal

modsig=carwav * binsig
plt.plot(t,modsig,'r')
plt.title('BPSK Modulated Signal')

Text(0.5, 1.0, 'BPSK Modulated Signal')
```



QPSK Generation from XS3 Code

Here in this program, I'm going to take the last digit from my roll no "MDL19EC047", which is 7 in decimal, convert it into BCD, and get the result by adding 3 to it(XS-3)

```
print("The XS-3 code for 7 is =",int_to_bcd(7)+3)
```

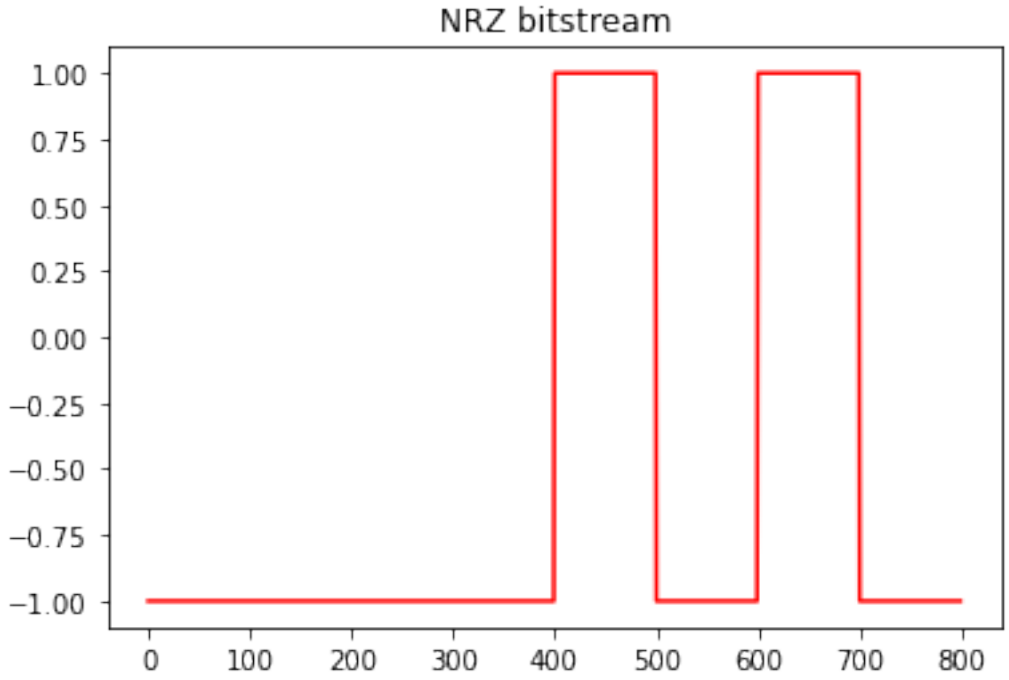
```
The XS-3 code for 7 is = 10
```

```
print("The XS-3 Code in binary is =",int_to_bin(10))
```

```
The XS-3 Code in binary is =  1010
```

```
#Generating NRZ Sequence
binsig=np.array([0,0,0,0,1,0,1,0])
binsig[binsig==0]=-1
plt.plot(np.repeat(binsig,100),'r')
plt.title('NRZ bitstream')
```

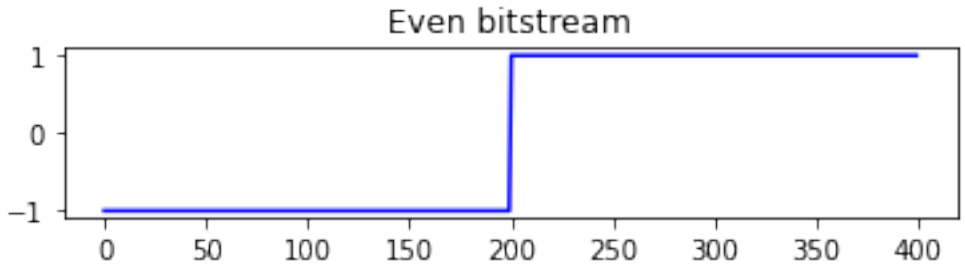
```
Text(0.5, 1.0, 'NRZ bitstream')
```

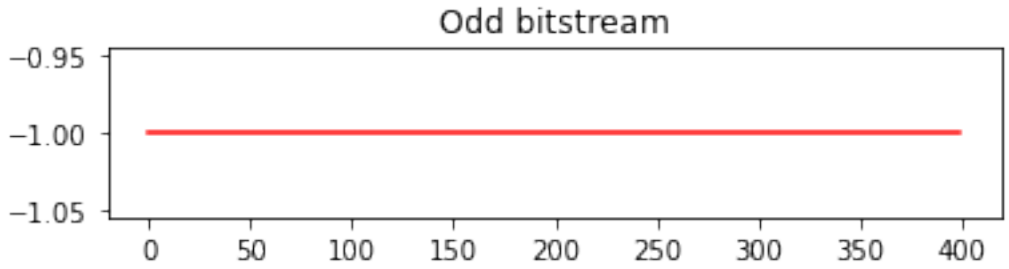



```
evensig=binsig[0::2]
evensig=np.repeat(evensig,100)
oddsig=binsig[1::2]
oddsig=np.repeat(oddsig,100)
plt.subplot(3,1,1)
plt.plot(evensig,'b')
plt.title("Even bitstream")
plt.subplot(3,1,3)
```

```
plt.plot(oddsig,'r')
plt.title("Odd bitstream")
```

```
Text(0.5, 1.0, 'Odd bitstream')
```






```
#Generating Carrier Waves
f=0.01
t=np.arange(0,4/f)
carwaveven=np.sin(2*np.pi*f*t)
carwavodd=np.cos(2*np.pi*f*t)
```

```
---------------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
Input In [87], in <cell line: 5>()
      3 t=np.arange(0,4/f)
      4 carwaveven=np.sin(2*np.pi*f*t)
----> 5 carwavodd

NameError: name 'carwavodd' is not defined
```