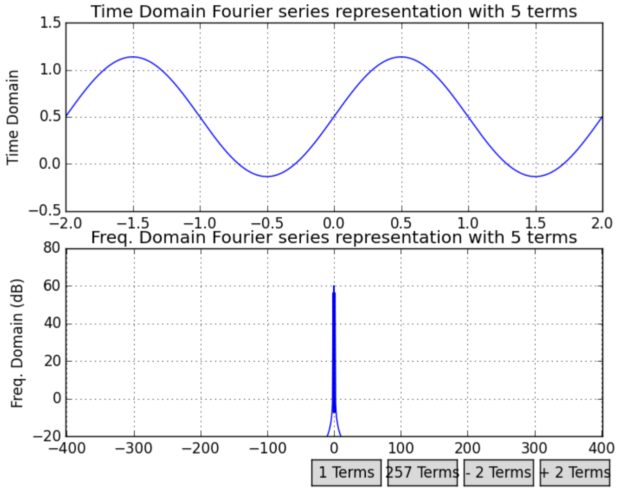
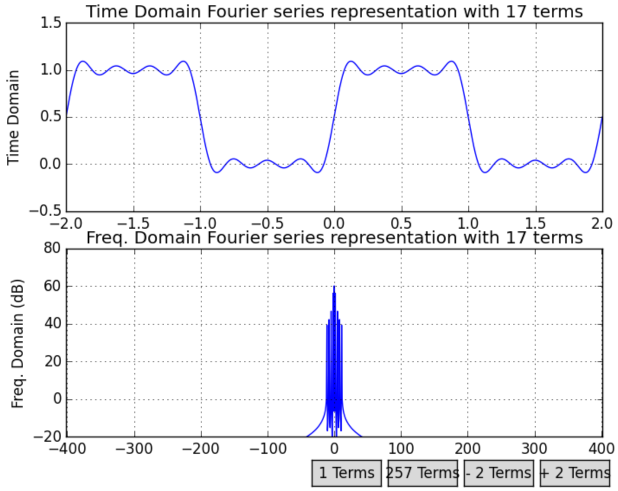
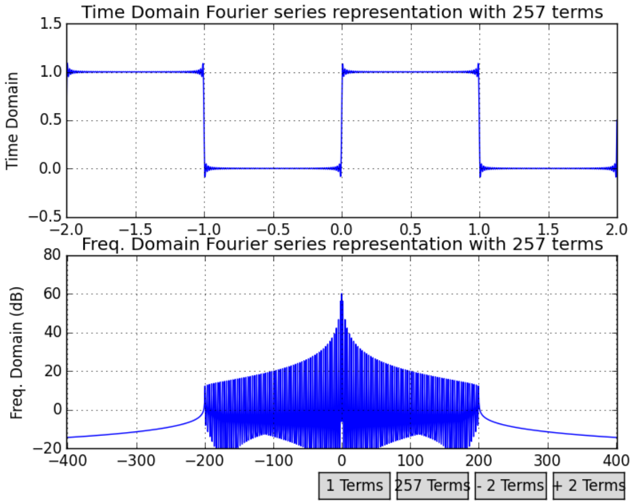
Question 3B:

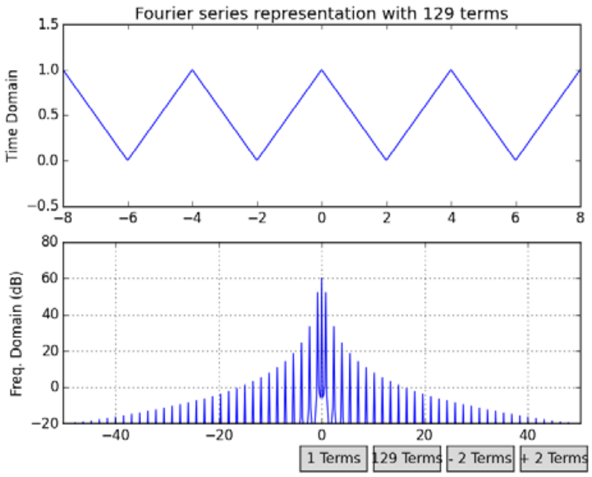

Time Domain Fourier series representation with 5 terms
Time Domain
1.5
1.0
0.5
0.0
−0.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Freq. Domain Fourier series representation with 5 terms
Freq. Domain (dB)
80
60
40
20
0
−20
−400
−300
−200
−100
0
100
200
300
400
1 Terms
257 Terms
- 2 Terms
+ 2 Terms


Time Domain Fourier series representation with 17 terms
Time Domain
1.5
1.0
0.5
0.0
−0.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Freq. Domain Fourier series representation with 17 terms
Freq. Domain (dB)
80
60
40
20
0
−20
−400
−300
−200
−100
0
100
200
300
400
1 Terms
257 Terms
- 2 Terms
+ 2 Terms

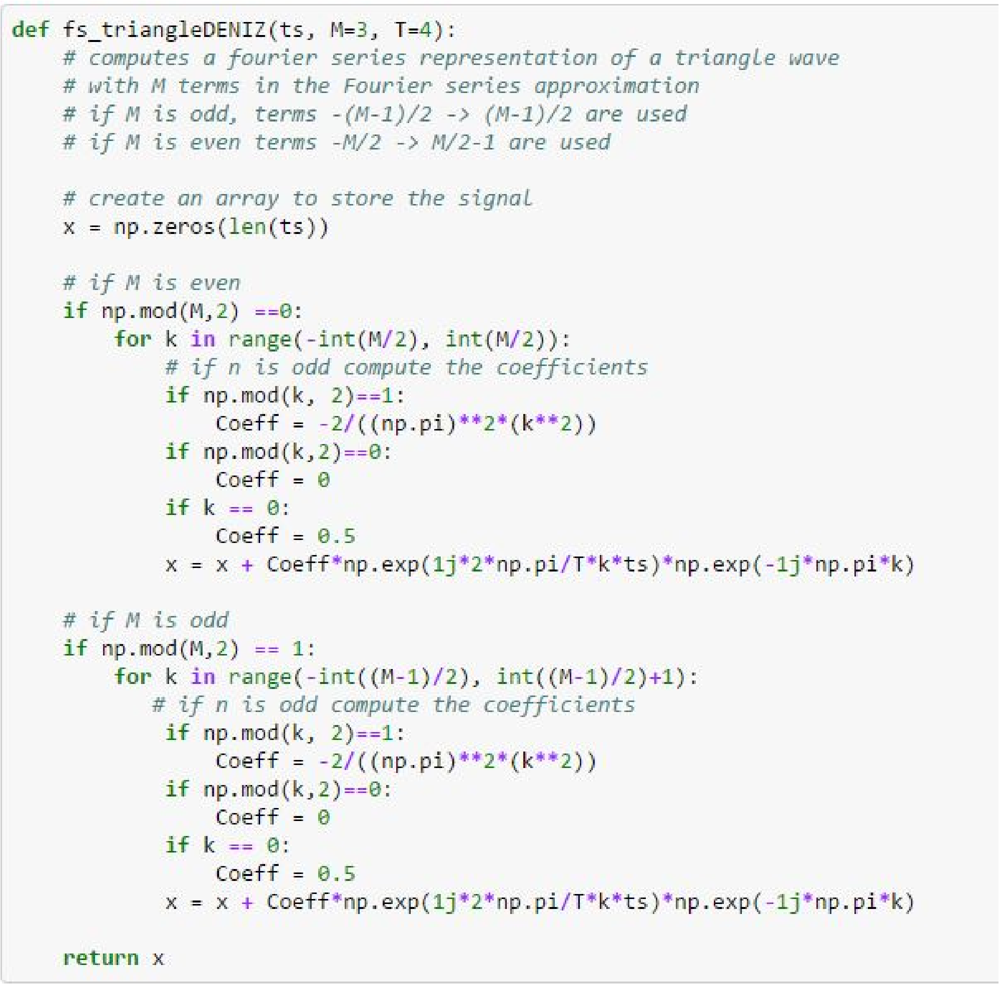

Time Domain Fourier series representation with 257 terms
Time Domain
1.5
1.0
0.5
0.0
−0.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Freq. Domain Fourier series representation with 257 terms
Freq. Domain (dB)
80
60
40
20
0
−20
−400
−300
−200
−100
0
100
200
300
400
1 Terms
257 Terms
- 2 Terms
+ 2 Terms

Question 4B:

Graphs:



Code:



```
def fs_triangleDENIZ(ts, M=3, T=4):
    # computes a fourier series representation of a triangle wave
    # with M terms in the Fourier series approximation
    # if M is odd, terms -(M-1)/2 -> (M-1)/2 are used
    # if M is even terms -M/2 -> M/2-1 are used

    # create an array to store the signal
    x = np.zeros(len(ts))

    # if M is even
    if np.mod(M,2) ==0:
        for k in range(-int(M/2), int(M/2)):
            # if n is odd compute the coefficients
            if np.mod(k, 2)==1:
                Coeff = -2/((np.pi)**2*(k**2))
            if np.mod(k,2)==0:
                Coeff = 0
            if k == 0:
                Coeff = 0.5
            x = x + Coeff*np.exp(1j*2*np.pi/T*k*ts)*np.exp(-1j*np.pi*k)

    # if M is odd
    if np.mod(M,2) == 1:
        for k in range(-int((M-1)/2), int((M-1)/2)+1):
           # if n is odd compute the coefficients
            if np.mod(k, 2)==1:
                Coeff = -2/((np.pi)**2*(k**2))
            if np.mod(k,2)==0:
                Coeff = 0
            if k == 0:
                Coeff = 0.5
            x = x + Coeff*np.exp(1j*2*np.pi/T*k*ts)*np.exp(-1j*np.pi*k)

    return x
```