Time domain processing of LTI systems

For the LTI systems described by the following difference equations, generate its impulse response, and unit step response. Comment on the properties of the system (Stable, Causal)

1. $y[n] = y[n-1] + x[n]$
2. $y[n] = x[n] - x[n-1]$
3. $y[n] = 0.9y[n-1] + x[n]$
4. $y[n] = \frac{1}{4}\sum_{k=0}^{3} x[n-k]$

Also plot the input and output of these systems if you pass the following inputs

1. $\sin[0.05\pi n](u[n] - u[n-100])$
2. $(-1)^n(u[n] - u[n-100])$
3. $[(n\%10) - 5](u[n] - u[n-100])$ "%" denotes the reminder of division
4. $(0.9)^n(u[n] - u[n-100])$

(use function *scipy.signal.impulse* to determine the impulse response and *scipy.signal.lfilter* to find the output)

Details about *scipy.signal.impulse* can be found here, https://docs.scipy.org/doc/scipy-0.19.0/reference/generated/scipy.signal.impulse.html#scipy.signal.impulse

Details about *scipy.signal.lfilter* can be found here, https://docs.scipy.org/doc/scipy-0.19.0/reference/generated/scipy.signal.lfilter.html#scipy.signal.lfilter

Submitted By:

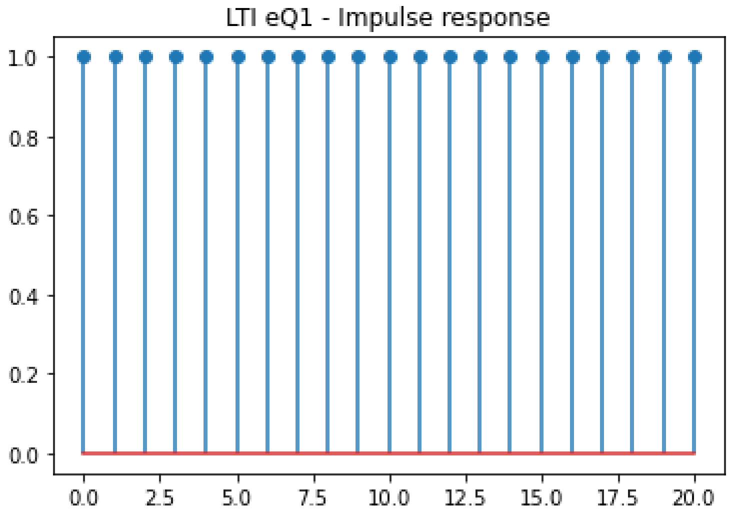
Utkarsh Mahajan 201EC164

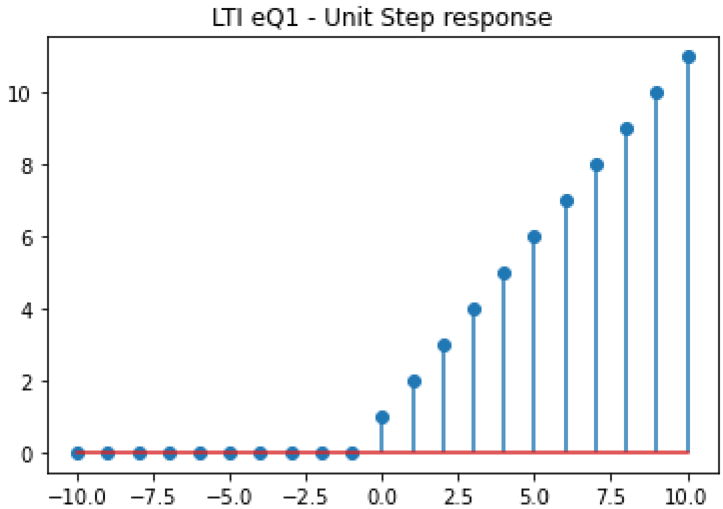
Arnav Raj 201EC109

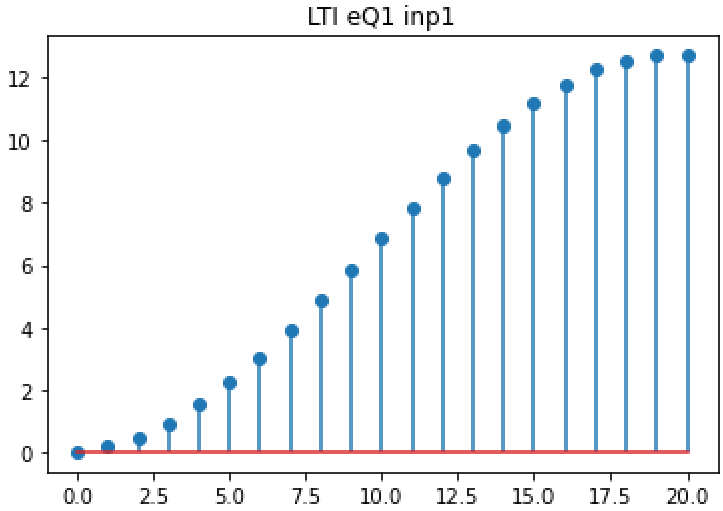
In [62]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sig

def plot(q, w, title):
    plt.stem(q,w)
    plt.title(title)
    plt.show()

n1= range(0,21)
n2 = range(-10,11)
n= np.array(n1)
d = sig.unit_impulse(21,0)
u = [*[0]*10,*(np.heaviside(n,1) for n in range(0,11))]

inp1 = [np.sin(np.pi*n*0.05)*(np.heaviside(n,1)-np.heaviside(n-100,1))]
inp2 = np.power(-1,n)*(np.heaviside(n,1)-np.heaviside(n-100,1))
inp3 = (((n%10)-5)*(np.heaviside(n,1)-np.heaviside(n-100,1)))
inp4 = np.power(0.9, n)*(np.heaviside(n,1)-np.heaviside(n-100,1))

#LTI eQ1
```

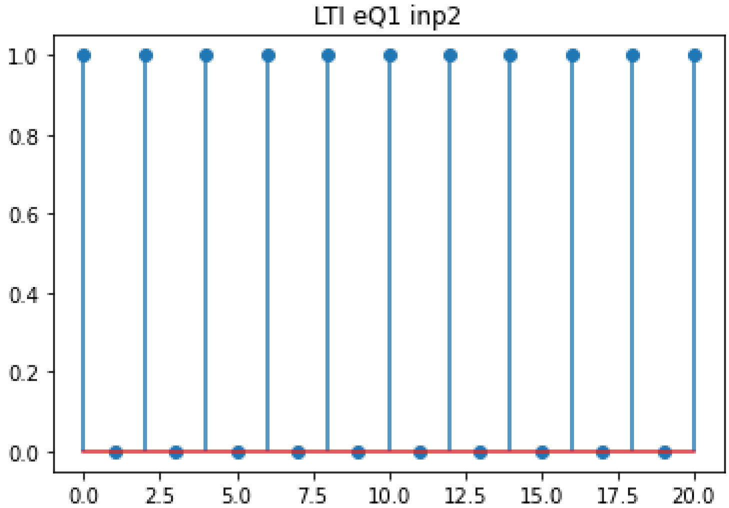
```
b = np.array([1.0, 0])
a = np.array([1.0, -1.0])
# Impulse Response
ir1 = sig.lfilter(b, a, d);
plot(n1,ir1,  "LTI eQ1 - Impulse response")
#unit step Response
usr1 = sig.lfilter(b, a, u);
plot(n2,usr1,"LTI eQ1 - Unit Step response")
# eQ1 Input1
y11 = sig.lfilter(b, a, inp1)[0]
plot(n1,y11,"LTI eQ1 inp1")
# eQ1 Input2
y12 = sig.lfilter(b, a, inp2);
plot(n1,y12,"LTI eQ1 inp2")
# eQ1 Input3
y13 = sig.lfilter(b, a, inp3);
plot(n1,y13,"LTI eQ1 inp3")
# eQ1 Input4
y14 = sig.lfilter(b, a, inp4);
plot(n1,y14,"LTI eQ1 inp4")

#LTI eQ2
b = np.array([1.0, -1.0])
a = np.array([1.0])
# Impulse Response
ir1 = sig.lfilter(b, a, d);
plot(n1,ir1,  "LTI eQ2 - Impulse response")
#unit step Response
usr1 = sig.lfilter(b, a, u);
plot(n2,usr1,"LTI eQ2 - Unit Step response")
# eQ1 Input1
y11 = sig.lfilter(b, a, inp1)[0]
plot(n1,y11,"LTI eQ2 inp1")
# eQ1 Input2
y12 = sig.lfilter(b, a, inp2);
plot(n1,y12,"LTI eQ2 inp2")
# eQ1 Input3
y13 = sig.lfilter(b, a, inp3);
plot(n1,y13,"LTI eQ2 inp3")
# eQ1 Input4
y14 = sig.lfilter(b, a, inp4);
plot(n1,y14,"LTI eQ2 inp4")

#LTI eQ3
b = np.array([1.0])
a = np.array([1,0.9])
# Impulse Response
ir1 = sig.lfilter(b, a, d);
plot(n1,ir1,  "LTI eQ3 - Impulse response")
#unit step Response
usr1 = sig.lfilter(b, a, u);
plot(n2,usr1,"LTI eQ3 - Unit Step response")
# eQ1 Input1
y11 = sig.lfilter(b, a, inp1)[0]
plot(n1,y11,"LTI eQ3 inp1")
# eQ1 Input2
y12 = sig.lfilter(b, a, inp2);
plot(n1,y12,"LTI eQ3 inp2")
# eQ1 Input3
y13 = sig.lfilter(b, a, inp3);
plot(n1,y13,"LTI eQ3 inp3")
# eQ1 Input4
y14 = sig.lfilter(b, a, inp4);
plot(n1,y14,"LTI eQ3 inp4")
```

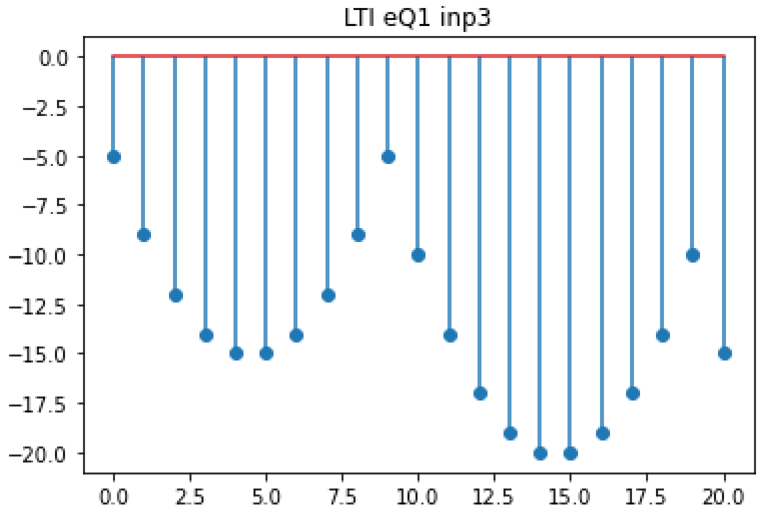
```
#LTI eQ4
b = np.array([0.25, 0.25, 0.25, 0.25])
a = np.array([1.0])
# Impulse Response
ir1 = sig.lfilter(b, a, d);
plot(n1,ir1,  "LTI eQ4 - Impulse response")
#unit step Response
usr1 = sig.lfilter(b, a, u);
plot(n2,usr1,"LTI eQ4 - Unit Step response")
# eQ1 Input1
y11 = sig.lfilter(b, a, inp1)[0]
plot(n1,y11,"LTI eQ4 inp1")
# eQ1 Input2
y12 = sig.lfilter(b, a, inp2);
plot(n1,y12,"LTI eQ4 inp2")
# eQ1 Input3
y13 = sig.lfilter(b, a, inp3);
plot(n1,y13,"LTI eQ4 inp3")
# eQ1 Input4
y14 = sig.lfilter(b, a, inp4);
plot(n1,y14,"LTI eQ4 inp4")
```

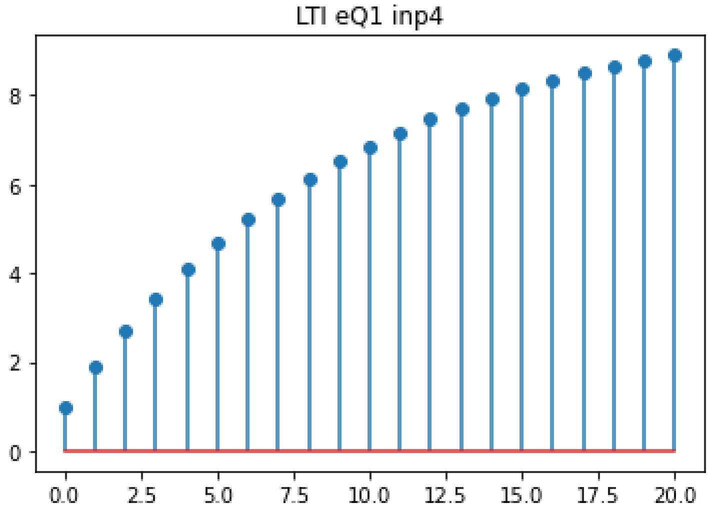


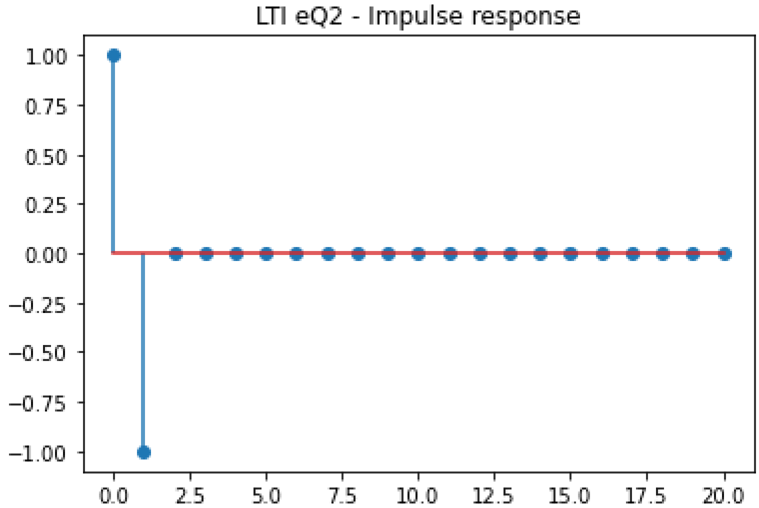


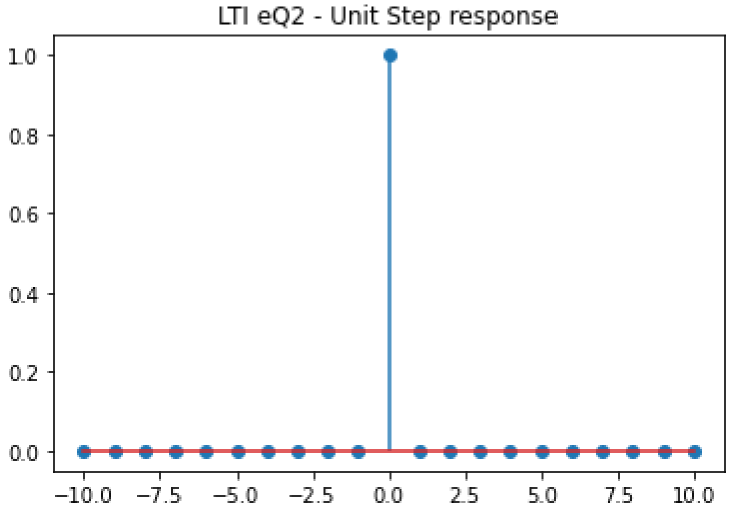

LTI eQ1 inp1
12
10
8
6
4
2
0
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0

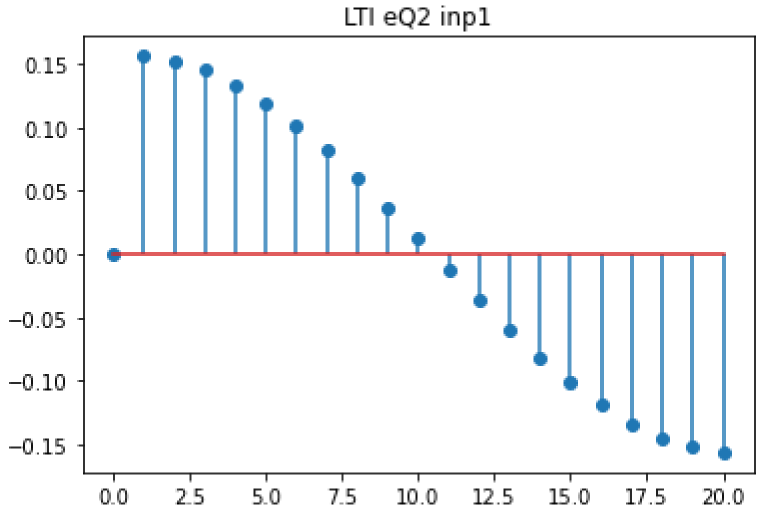

LTI eQ1 inp2
1.0
0.8
0.6
0.4
0.2
0.0
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0

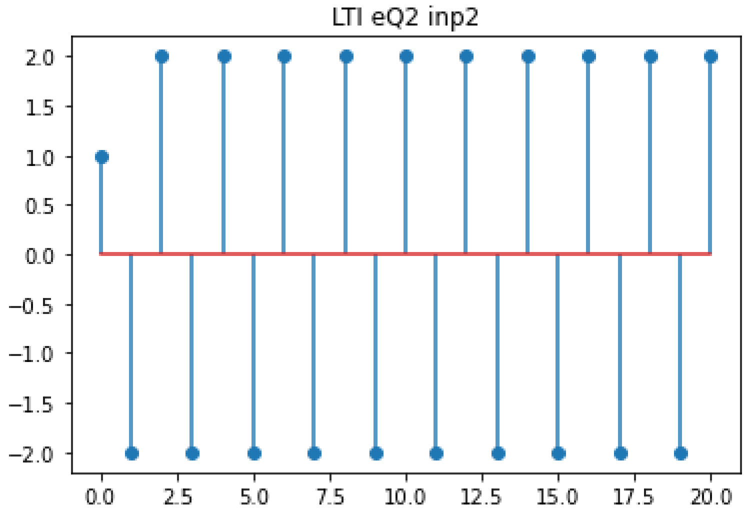

LTI eQ1 inp3
0.0
−2.5
−5.0
−7.5
−10.0
−12.5
−15.0
−17.5
−20.0
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0

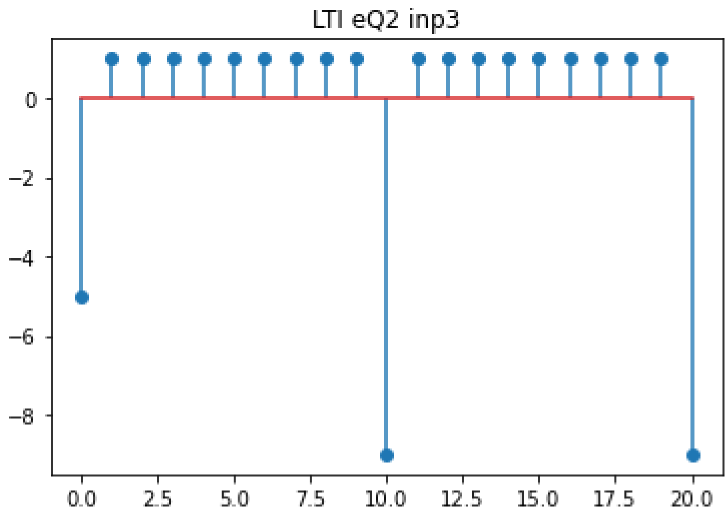

LTI eQ1 inp4

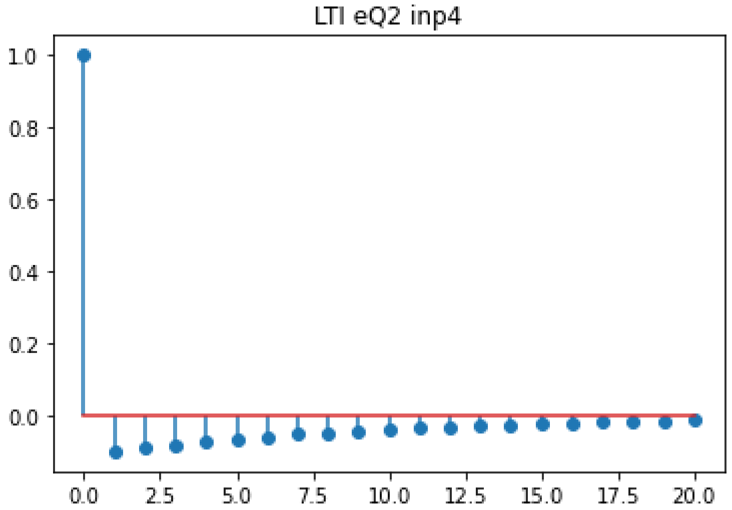

LTI eQ2 - Impulse response

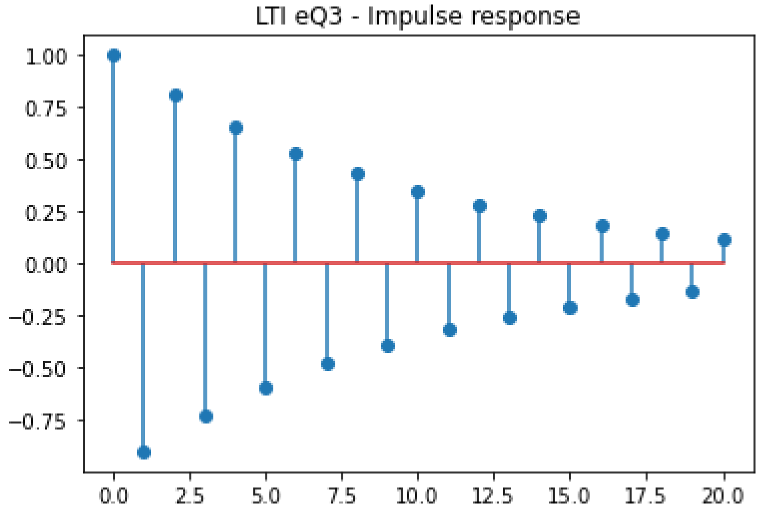

LTI eQ2 - Unit Step response

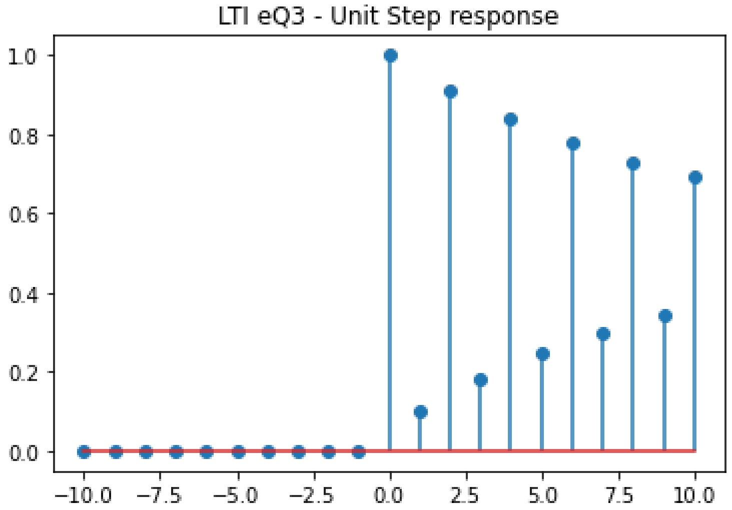

LTI eQ2 inp1
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

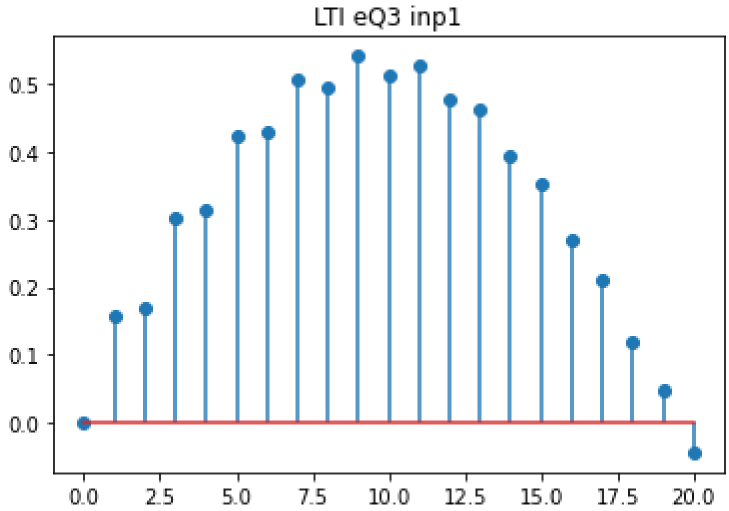

LTI eQ2 inp2
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

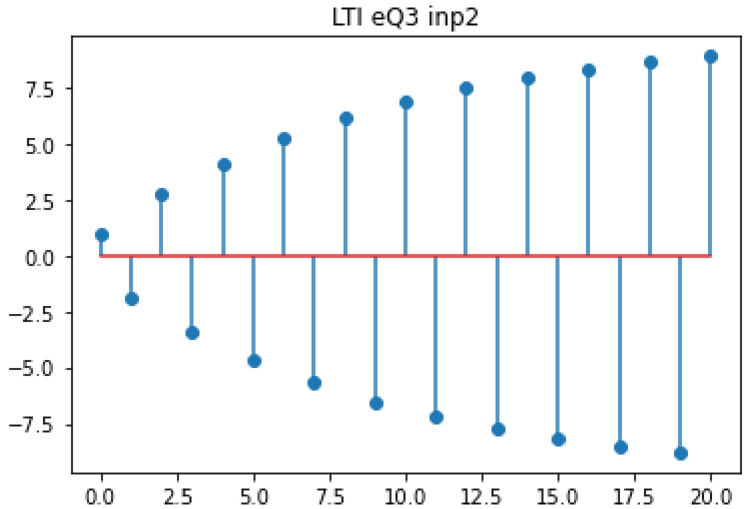

LTI eQ2 inp3
0
−2
−4
−6
−8
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

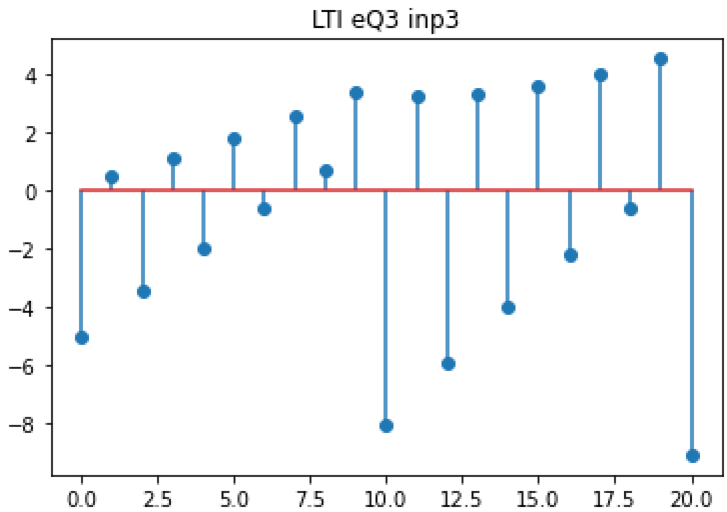

LTI eQ2 inp4
1.0
0.8
0.6
0.4
0.2
0.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

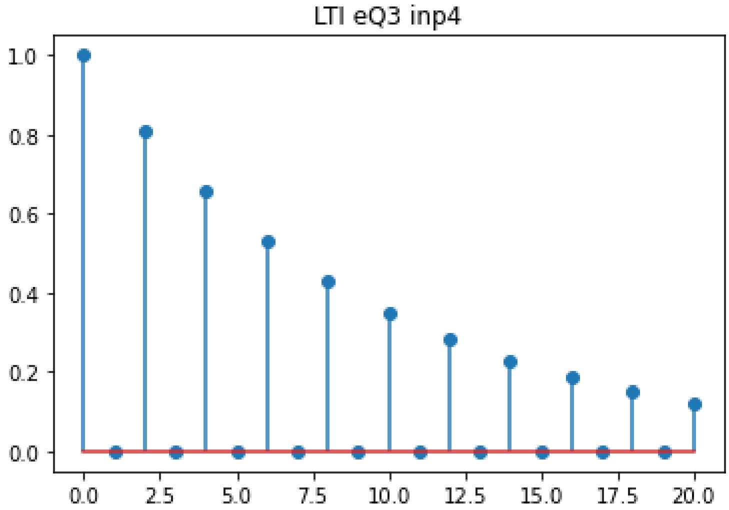

LTI eQ3 - Impulse response
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

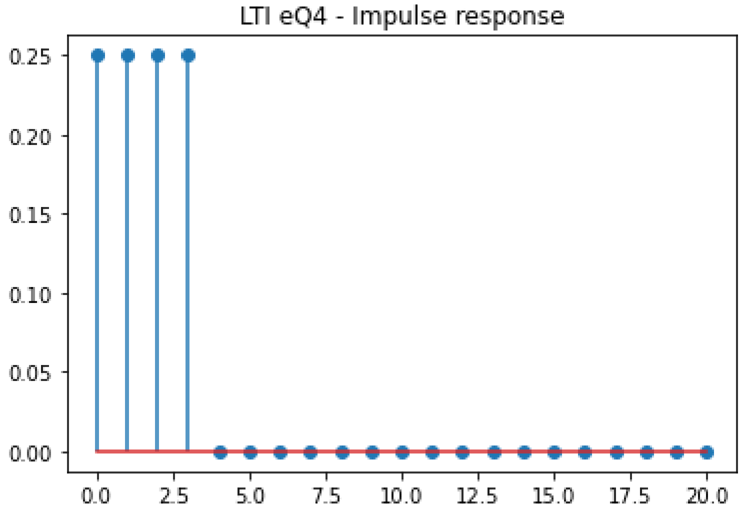

LTI eQ3 - Unit Step response
1.0
0.8
0.6
0.4
0.2
0.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

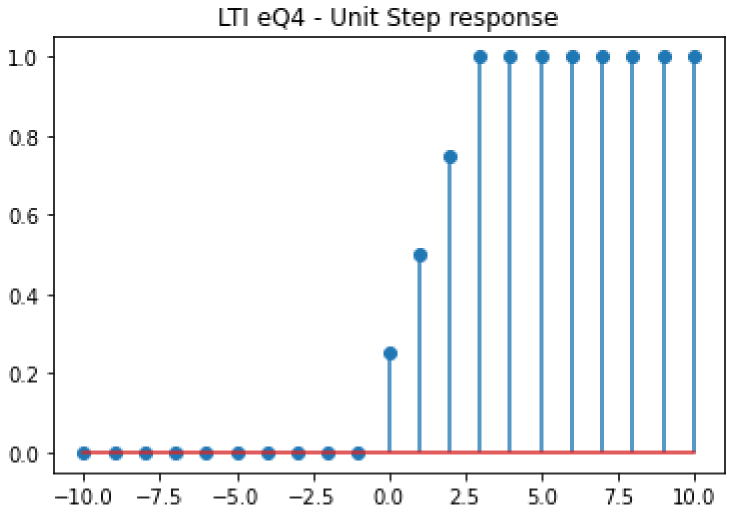

LTI eQ3 inp1
0.5
0.4
0.3
0.2
0.1
0.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

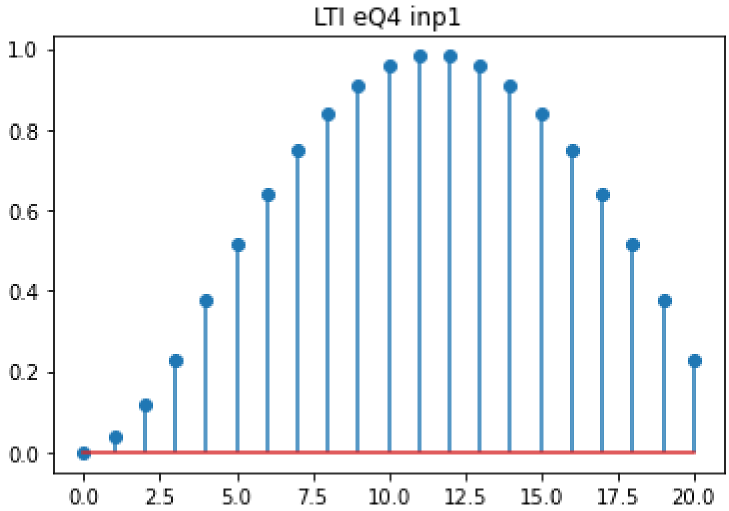

LTI eQ3 inp2
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

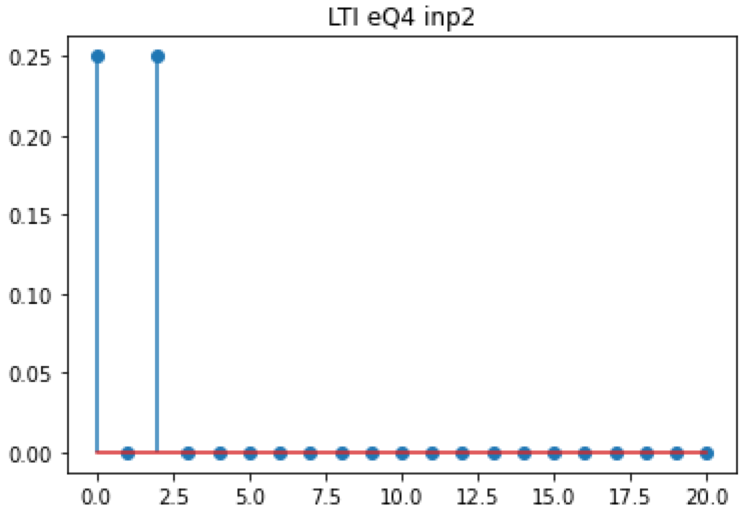

LTI eQ3 inp3
4
2
0
−2
−4
−6
−8
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

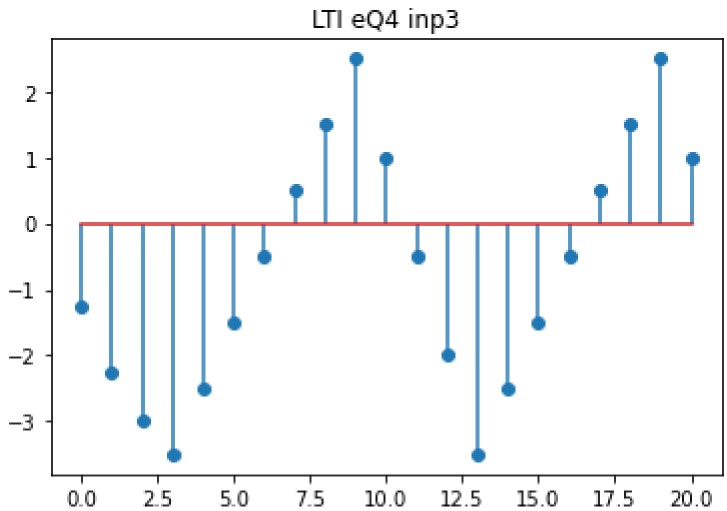

LTI eQ3 inp4
1.0
0.8
0.6
0.4
0.2
0.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

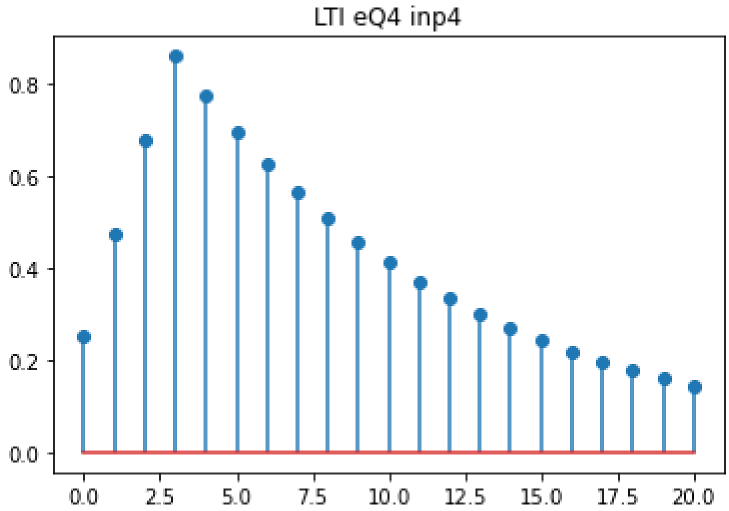

LTI eQ4 - Impulse response
0.25
0.20
0.15
0.10
0.05
0.00
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


LTI eQ4 - Unit Step response
1.0
0.8
0.6
0.4
0.2
0.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


LTI eQ4 inp1
1.0
0.8
0.6
0.4
0.2
0.0
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0


LTI eQ4 inp2
0.25
0.20
0.15
0.10
0.05
0.00
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0


LTI eQ4 inp3
2
1
0
−1
−2
−3
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0



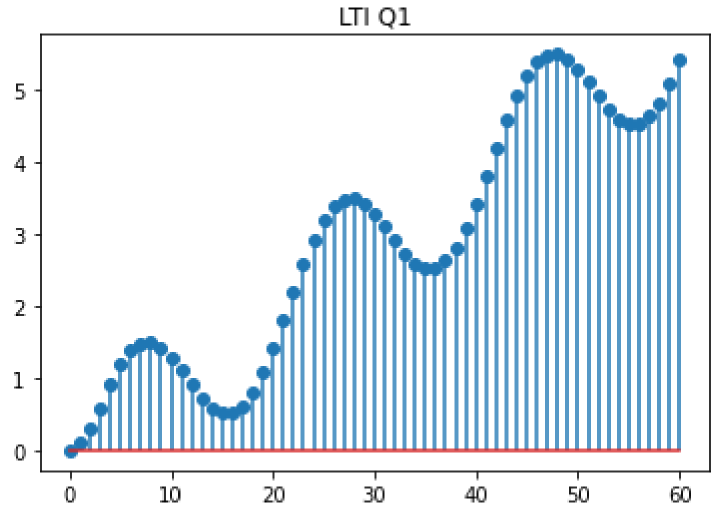
Plot the response of the following filters if the input is

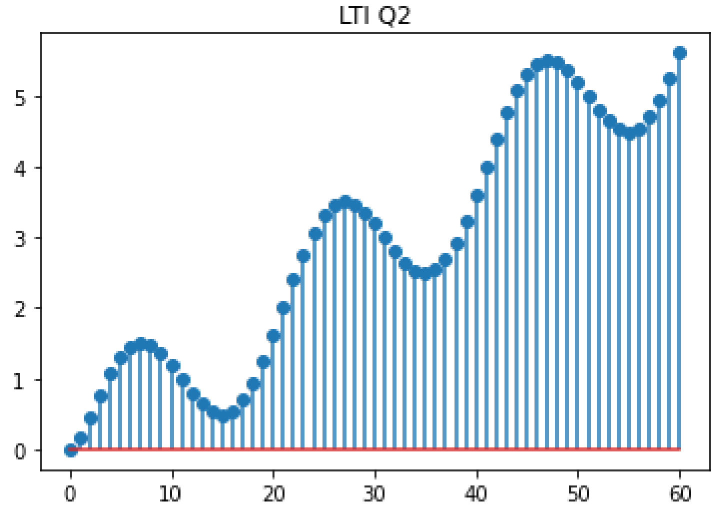
- $x[n] = 0.1n + sin(0.1n\pi); 0 \leq n \leq 60$

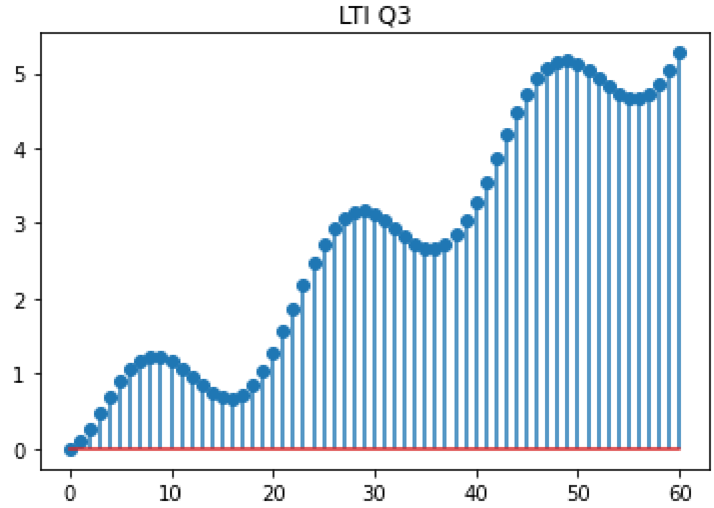
1. $y[n] = \frac{1}{4}\sum_{k=0}^{3} x[n-k]$
2. $y[n] = \frac{2}{N(N+1)}\sum_{k=0}^{N-1}(N-k)x[n-k]; N = 4$
3. $y[n] - \alpha y[n-1] = (1-\alpha)x[n]; \alpha = 3/4$

In [71]:

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

def plot(q, w, title):
    plt.stem(q,w)
    plt.title(title)
    plt.show()

n=np.arange(0,61)
inp =  0.1*n+ np.sin(0.1*np.pi*n)


#LTI q1
b = np.array([0.25, 0.25, 0.25, 0.25])
a = np.array([1.0])
response = sig.lfilter(b, a, inp)
plot(n,response,"LTI Q1")

#LTI q2
b = np.array([0.4, 0.3, 0.2, 0.1])
a = np.array([1.0])
response2 = sig.lfilter(b, a, inp)
plot(n,response2,"LTI Q2")

#LTI q3
b = np.array([0.25])
a = np.array([1.0, -0.75])
response3 = sig.lfilter(b, a, inp)
plot(n,response3,"LTI Q3")
```







Consider a round theatre where an orchestra is in the middle of two concentric circles and the walls on one half side are at a radial distances of $17.15m$ (inner circle) and $34.3m$ (outer circle) on the other side from the orchestra. The speed of sound is $343m/s$. Assume that the recorded signal is the sum of the original signal and the attenuated echoes from the two walls and is given by $r[n] = y[n] + 0.8y[n - N1] + 0.6y[n - N2]$ where, $N1$ is the delay caused by the closest wall and N2 is the delay caused by the farther wall. The recorder is in the centre of the theatre. Take any audio signal available and generate $r[n]$ and listen to both the original and the echoed signal.

In [4]:
```
import numpy as np
```

```
import matplotlib.pyplot as plt
import scipy.io.wavfile as wav

N1 = 17.15*2/343
N2 = 34.3*2/343

rate, data = wav.read('input.wav')
# echo due to wall 1
echo1 = [0.8*data[round(n-N1)] for n in range(0, np.shape(data)[0])]
# echo due to wall 2
echo2 = [0.6*data[round(n-N2)] for n in range(0, np.shape(data)[0])]
echoed_signal = [data[n] + echo1[n] + echo2[n] for n in range(0, np.shape(d
#to write the output file
wav.write("out.wav", rate, np.asarray(echoed_signal))
```

In [ ]: