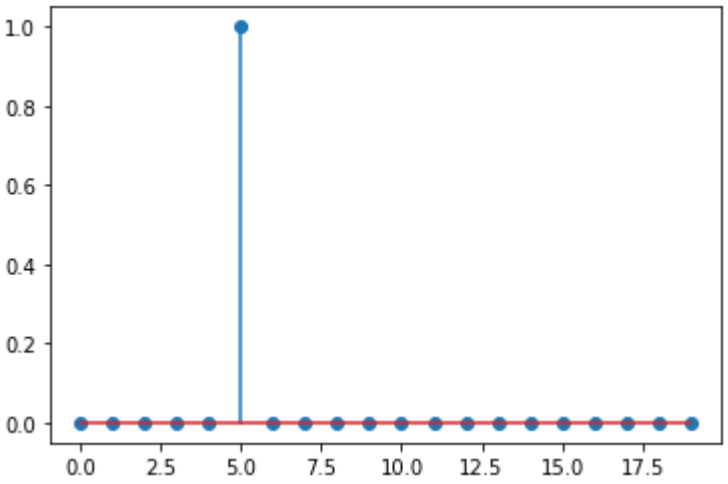
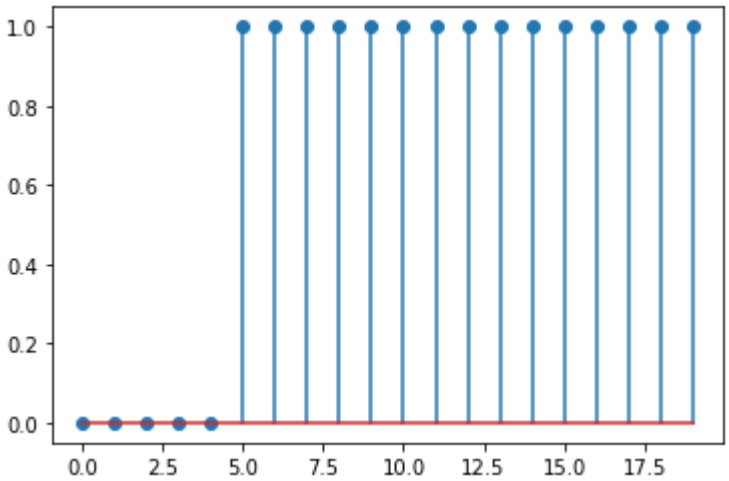
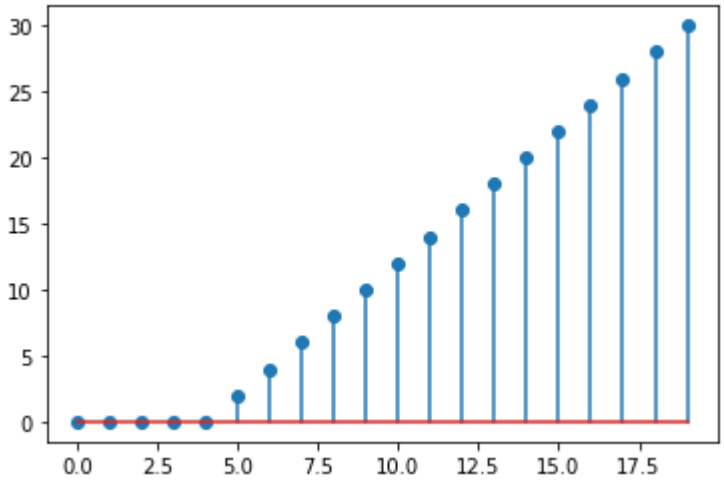


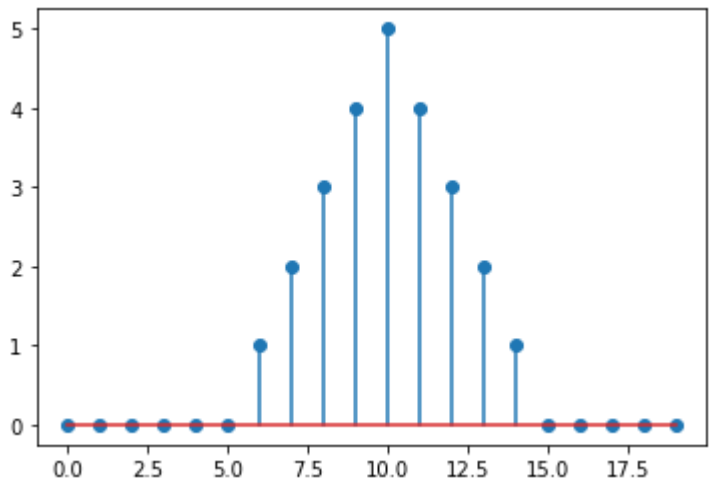

```
#%%
#Question 1
def genSig1(N,k) :
    r = np.zeros(N)
    r[k] = 1
    return r

R = genSig1(20,5)
plt.figure(1)
plt.stem(R)
```



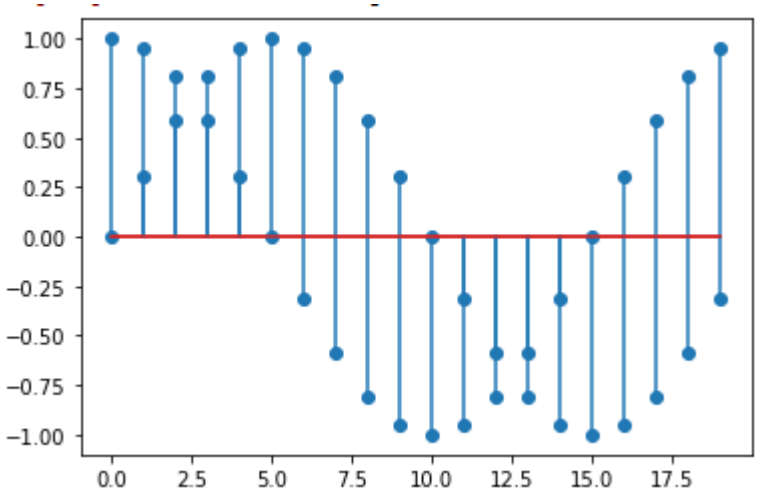

```
#%%
#Question 2
def genSig2(N,k) :
    r = np.zeros(N)
    for i in range(k,N):
        r[i] = 1
    return r

R = genSig2(20,5)
plt.figure(1)
plt.stem(R)
```



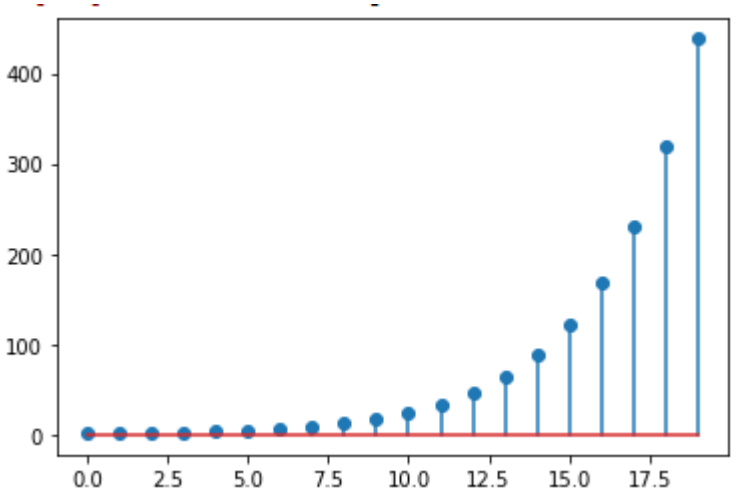

```
#%%
#Question 3
def genSig3(N,k,R) :
    r = np.zeros(N)
    tmp = 0
    for i in range(k,N):
        tmp += R
        r[i] = tmp
    return r

R = genSig3(20,5,2)
plt.figure(1)
plt.stem(R)
```




```
#%%
#Question 4
def genSig4(N,k,R) :
    r = np.zeros(N)
    tmp = 0
    for i in range(k,k+k+1):
        r[i] = tmp
        tmp += R
    for i in range(k+k,N-k):
        tmp -= R
        r[i] = tmp
    return r

R = genSig4(20,5,1)
plt.figure(1)
plt.stem(R)
```




```
#%%
#Question 5
def genSig5Sin(N,f,fs) :
    r = np.zeros(N)
    for i in range(N):
        r[i] = np.sin(2*np.pi*f*(float)(i/fs))
    return r

def genSig5Cos(N,f,fs) :
    r = np.zeros(N)
    t = np.linspace(0,(N-1)/float(fs), N)
    r = np.cos(2*np.pi*f*t)
    return r

R = genSig5Sin(20,1,20)
plt.figure(1)
plt.stem(R)
R = genSig5Cos(20,1,20)
plt.figure(1)
plt.stem(R)
```




```
#%%
#Question 6
def expsig(N,a) :
    ret = np.zeros(N)
    (r,phi) = cmath.polar(a)
    for i in range(N):
        ret[i] = (np.power(r,i)*np.exp(phi*r*i))
    return ret

R = expsig(20,0.95+(np.pi/10)*1j)
plt.figure(1)
plt.stem(R)
```