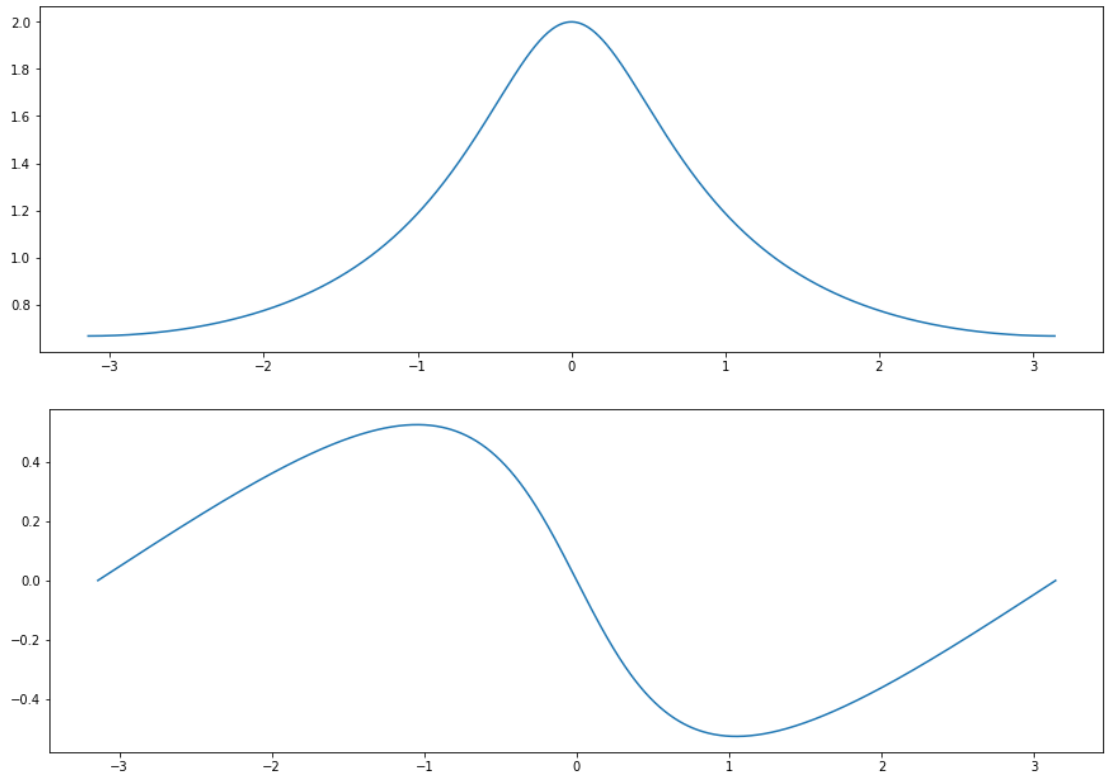
Homework 3

In [1]:
```
import numpy as np
import math
import matplotlib.pyplot as plt
import IPython.display as ipd
from scipy import signal

%matplotlib inline
```

Proakis 5.3a:

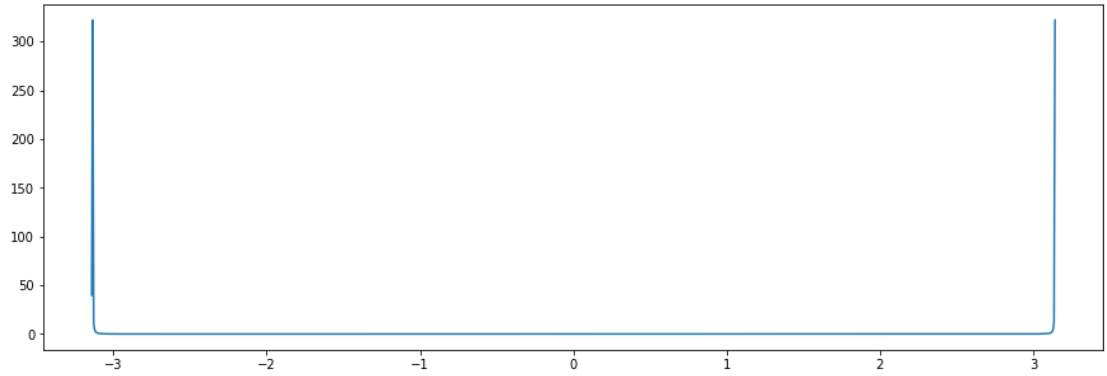
In [2]:
```
numH = [1, 0]
denH = [1, -0.5]
##################

# compute the impulse response
systemH = signal.dlti(numH, denH)
```

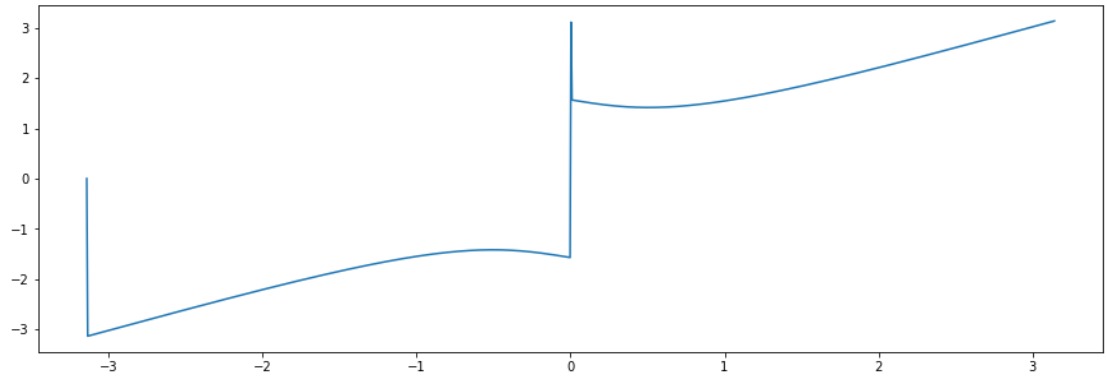
In [3]:
```
w = np.linspace(-np.pi,np.pi,1000)
w, H = signal.dfreqresp(systemH, w)

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(H));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(H));
```



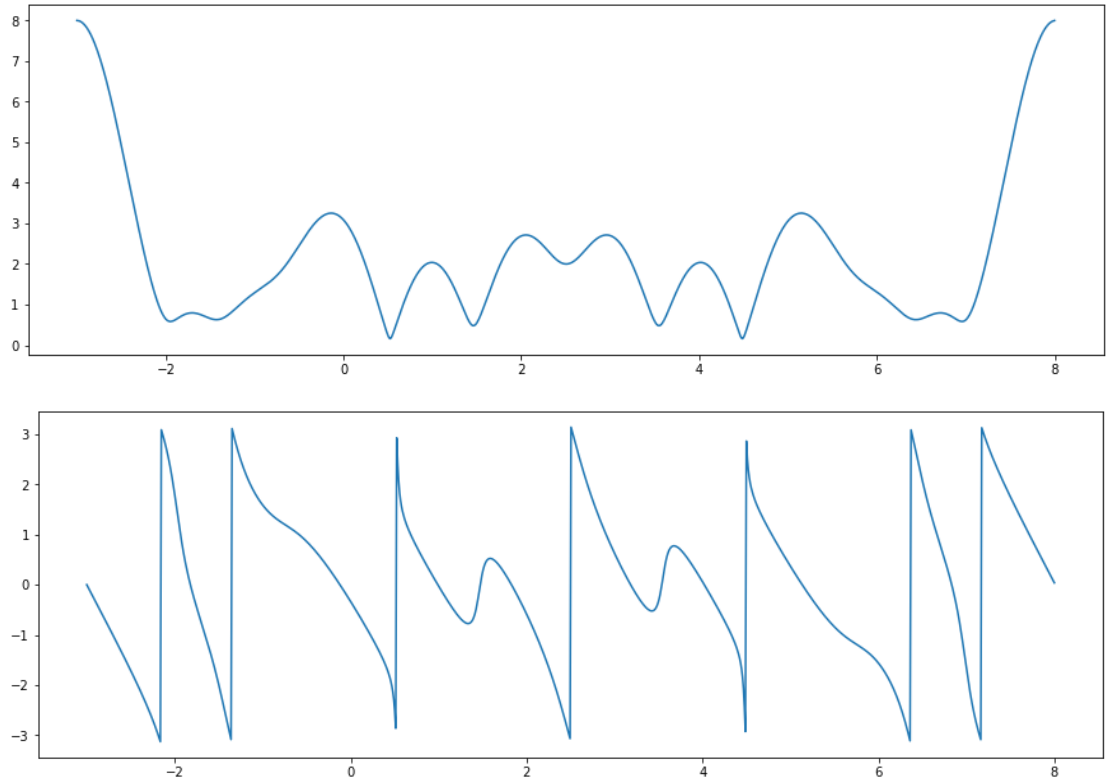
Proakis 5.3b

In [4]:
```
x_n = np.cos(3*np.pi*w/10)

X_k = np.fft.fft(x_n,)

Y_k = X_k*H

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(Y_k));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(Y_k));
```



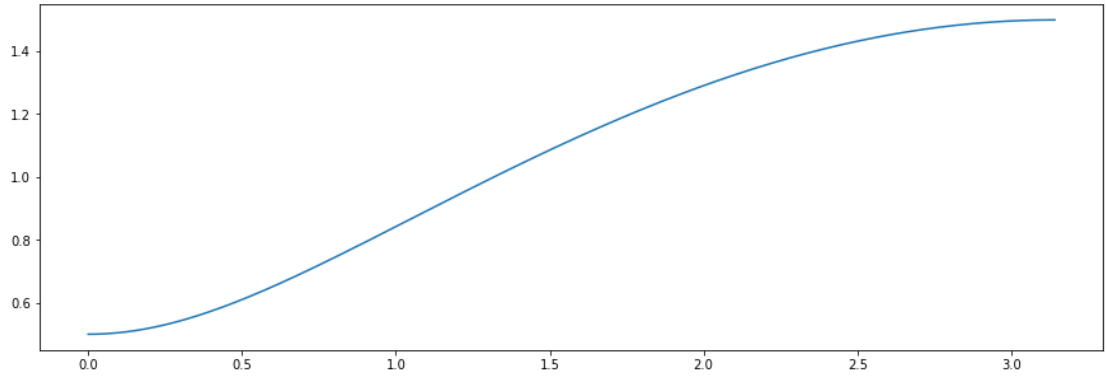


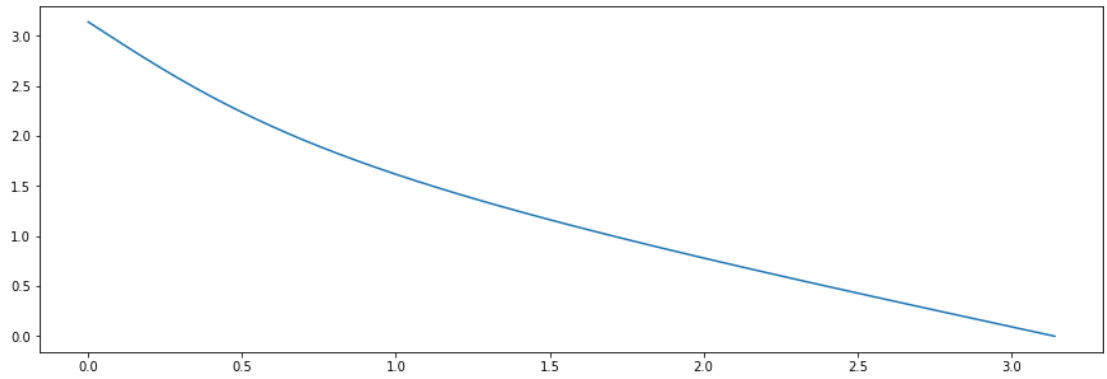
In [5]:
```
x_n = [1,0,0,1,1,1,0,1,1,1,0,1]

X_k = np.fft.fft(x_n, n=1000)

w = np.linspace(-3,8,1000)
w, H = signal.dfreqresp(systemH, w)

Y_k = X_k*H

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(X_k));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(X_k));
```



Proakis 5.4b

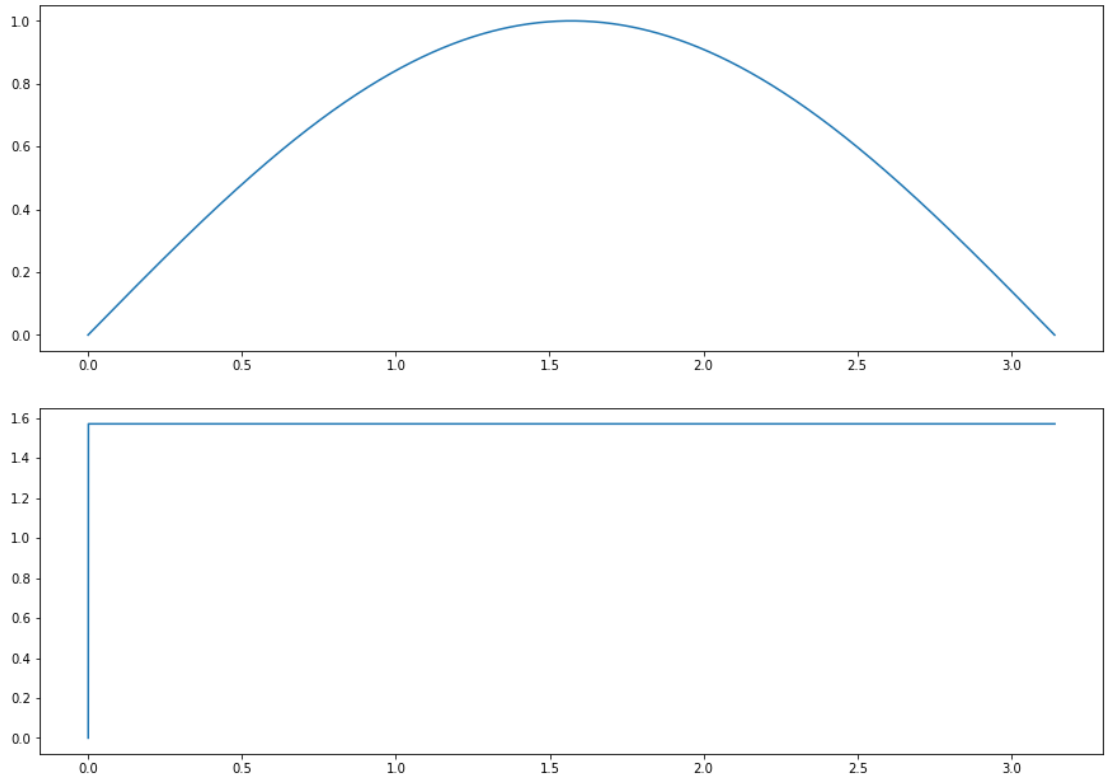
In [6]:
```
numX = [0.5, -1]
denX = [1, 0]
#################

# compute the impulse response
systemX = signal.dlti(numX, denX)

w = np.linspace(0,np.pi,10000)
w, H = signal.dfreqresp(systemX, w)

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(H));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(H));
```





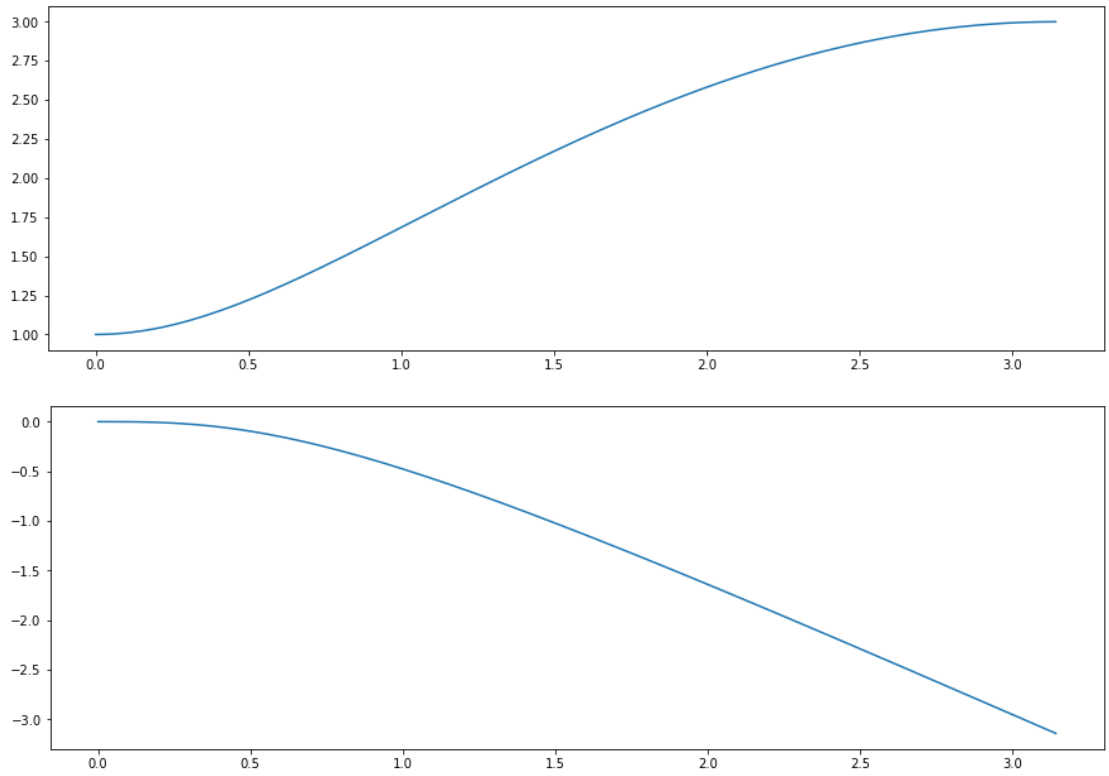
Proakis 5.4c

In [7]:
```
j = np.complex(0,1)

w = np.linspace(0,np.pi,10000)
H = np.sin(w)*np.exp(j*np.pi/2)

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(H));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(H));
```



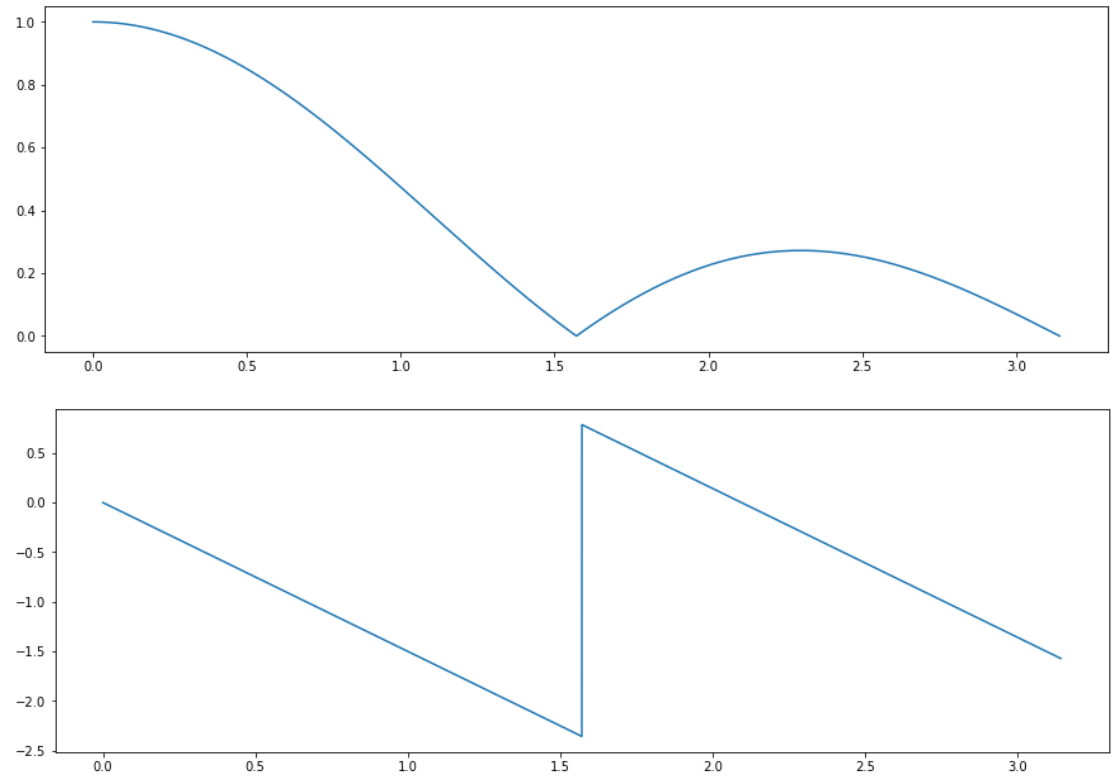
Proakis 5.4i

In [8]:
```
w = np.linspace(0,np.pi,10000)
H = 2*np.exp(-j*w) - np.exp(-j*2*w)

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(H));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(H));
```



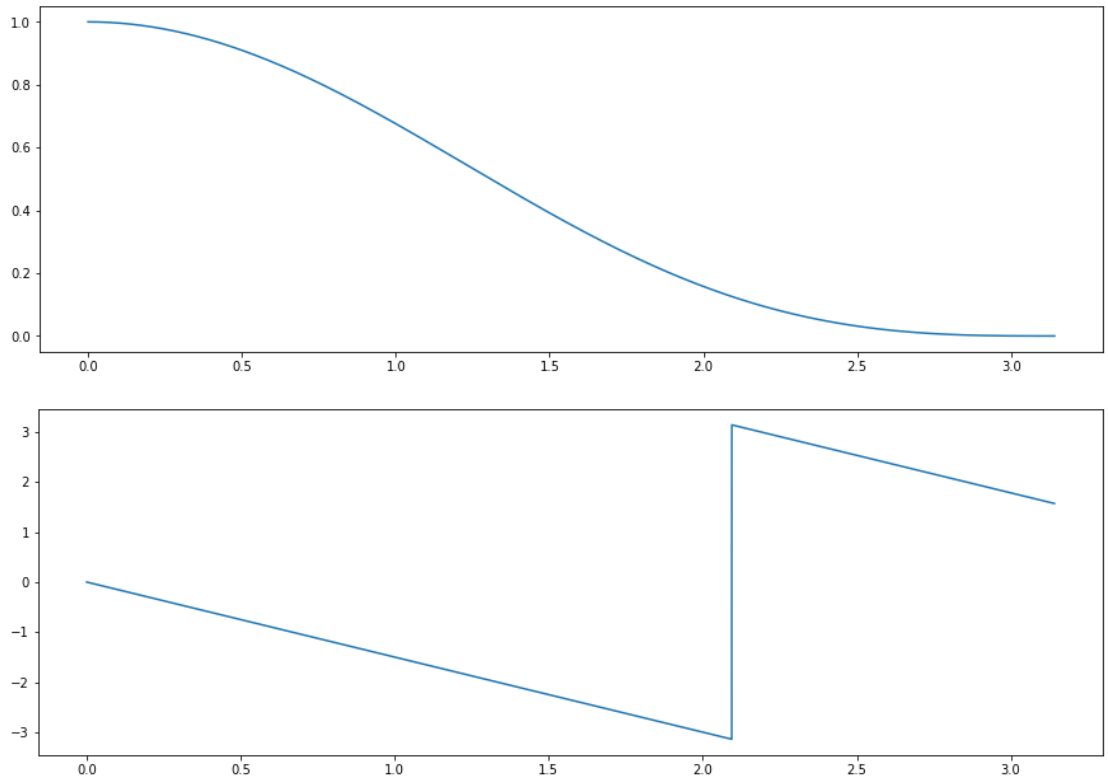
Proakis 5.4j

In [9]:
```
w = np.linspace(0,np.pi,10000)
H = np.cos(w)*np.cos(w/2)*np.exp(-j*3*w/2)

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(H));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(H));
```



Proakis 5.4k

In [10]:
```
w = np.linspace(0,np.pi,10000)
H = np.cos(w/2)*np.cos(w/2)*np.cos(w/2)*np.exp(-j*3*w/2)

plt.figure(figsize = (15,5))
plt.plot(w, np.abs(H));

plt.figure(figsize = (15,5))
plt.plot(w, np.angle(H));
```



Proakis 7.17

In [11]:
```
x_n = [1,1,1,1,1,1,0,0]
N = 8

X_k = np.fft.fft(x_n, n=N)

k = np.linspace(0,N,N)

print(k)

plt.figure(figsize = (15,5))
plt.stem(k, np.abs(X_k));

plt.figure(figsize = (15,5))
plt.stem(k, np.angle(X_k));
```

```
[0.         1.14285714 2.28571429 3.42857143 4.57142857 5.71428571
 6.85714286 8.        ]
```

```
/home/sweet/2-coursework/ecec434/labs/lib/python3.6/site-packages/ipykernel_launcher.py:11: UserWarning: In Matplotlib 3.3 individual lines on a stem plot will be added as a LineCollection instead of individual lines. This significantly improves the performance of a stem plot. To remove this warning and switch to the new behaviour, set the "use_line_collection" keyword argument to True.
  # This is added back by InteractiveShellApp.init_path()
/home/sweet/2-coursework/ecec434/labs/lib/python3.6/site-packages/ipykernel_launcher.py:14: UserWarning: In Matplotlib 3.3 individual lines on a stem plot will be added as a LineCollection instead of individual lines. This significantly improves the performance of a stem plot. To remove this warning and switch to the new behaviour, set the "use_line_collection" keyword argument to True.
```