Example with Sound:

We have **8000 Hz sampling rate**, and want to build a **band pass** filter. Our low stop band is between 0 and 0.05, our **pass band** between 0.1 and 0.2, and high stop band between 0.3 and 0.5. Since here, 1 corresponds to the sampling frequency, our **pass band** will be between $0.1 * 8000 = 800Hz$ and $0.2 * 8000 = 1600Hz$. Hence our vector bands is: [0.0, 0.05, 0.1, 0.2, 0.3, 0.5]

The vector desired contains the desired output per band. Hence here for our bandpass filter it is: [0.0, 1.0, 0.0]

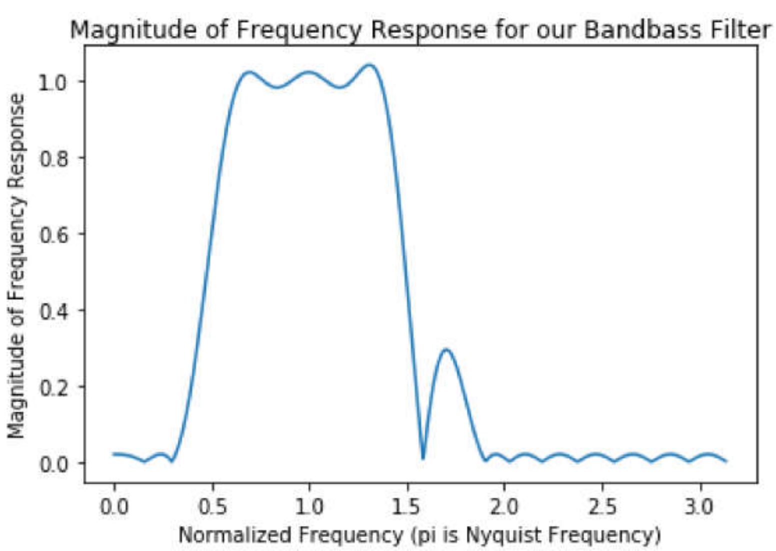
We choose our weights all to 1: weight=[1.0, 1.0, 1.0]

and our numtaps to be 32. Hence our design function in Python is:

In [1]:
```
%matplotlib inline
import numpy as np
import scipy.signal
import matplotlib.pyplot as plt

N=32
bpass=scipy.signal.remez(N, [0.0, 0.05, 0.1, 0.2, 0.3, 0.5] , [0.0, 1.0, 0.0],
 weight=[1.0, 1.0, 1.0])

#Plot the magnitude of the frequencyresponse:
fig = plt.figure()
[freq, response] = scipy.signal.freqz(bpass)
plt.plot(freq, np.abs(response))
plt.xlabel('Normalized Frequency (pi is Nyquist Frequency)')
plt.ylabel("Magnitude of Frequency Response")
plt.title("Magnitude of Frequency Response for our Bandbass Filter")
plt.show()
```

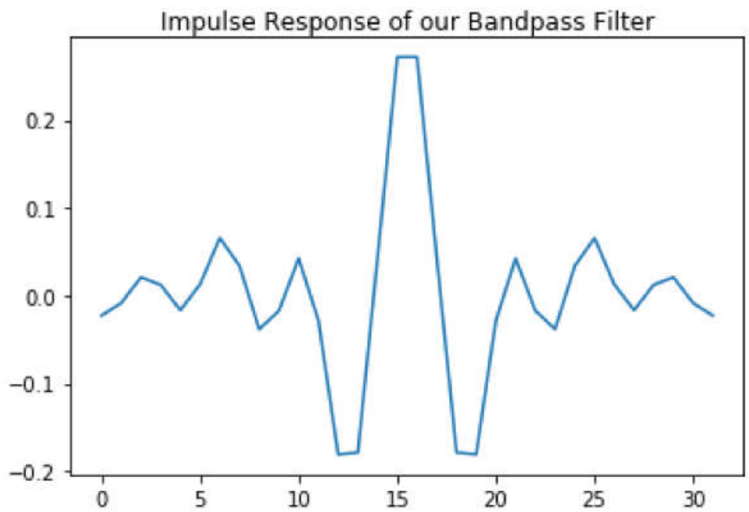


Observe:

The equi-ripple behaviour inside each band is clearly visible, and we see our pass band a little left of the center. The side lobe to its right is from the transition band there.

Next we plot its **impulse response**,

In [2]:
```
fig2=plt.figure()
plt.plot(bpass)
plt.title('Impulse Response of our Bandpass Filter')
plt.show()
```



Observe:

The impulse response is symmetric around the center, because it is a linear phase flter, and it stll has similarity with a sinc functon.