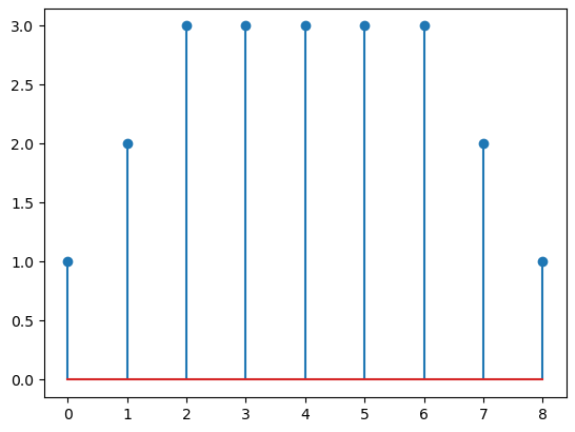
Circular Convolution

Part a.

Write a Python program called `conv` to compute the convolution of two discrete-time signals: $x_1[n]$ and $x_2[n]$. Assume that the signal $x_1[n]$ is zero outside the range $0 \leq n < N_1$ and that the signal $x_2[n]$ is zero outside the range $0 \leq n < N_2$. Represent these signals as Python lists `x_1` and `x_2` where the first element in each list represents the value of the signal at $n = 0$ and the lengths of `x_1` and `x_2` are N_1 and N_2 respectively.

Demonstrate the use of your program by convolving two rectangular pulses: one of length 3 and the other of length 7.

```
def conv(x,y):
    answer = []
    for n in range(len(x)+len(y)-1):
        answer.append(sum([x[m]*y[n-m] for m in range(max(0,n-len(y)+1),min(n+1,len(x)))]))
    return answer
```

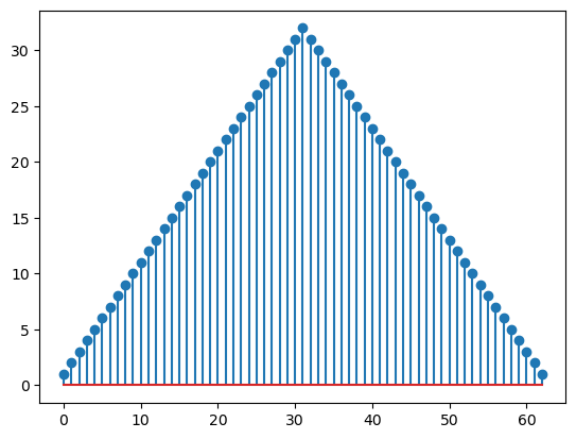


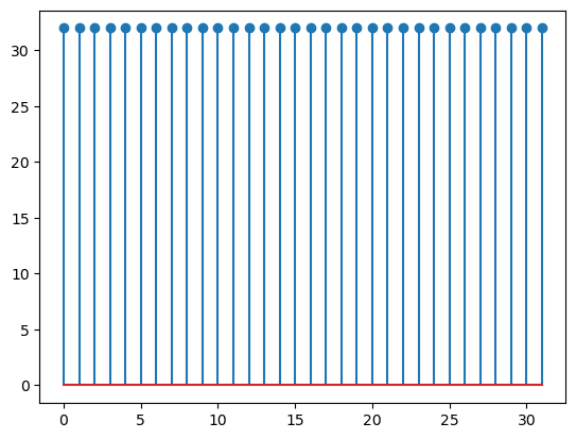
Part b.

Write a Python program called `circconv` to compute the circular convolution of two discrete-time signals: $x_1[n]$ and $x_2[n]$, as described in the previous part. Inputs to `circconv` should include the two input lists as well as the analysis width N of the circular convolution.

Demonstrate the use of your program by circularly convolving two rectangular pulses that are each of length 32 using an analysis width of $N = 32$. Compare the result of circular convolution with that of conventional convolution. Briefly explain the relation of these two results.

```
def circconv(x,y,N):
    answer = N*[0]
    z = conv(x,y)
    for n in range(len(z)):
        answer[n%N] += z[n]
    return answer
```





In this problem we convolved two rectangular signals with non-zero samples in the range $0 \leq n < 32$ to get a result that had non-zero samples in the range $0 \leq n < 64$.

However, the result of circular convolution is confined to the range $0 \leq n < 32$. Samples outside that range are wrapped (i.e., aliased) back into the range, so that

$$(x_2 \circledast x_2)[n] = (x_2 * x_2)[n] + (x_2 * x_2)[n + 32]$$

for $0 \leq n < 32$. Because the inputs were rectangular, the convolution was triangular. Wrapping the triangle combined the portion of the convolution with positive slope with the portion with negative slope. The result was rectangular.

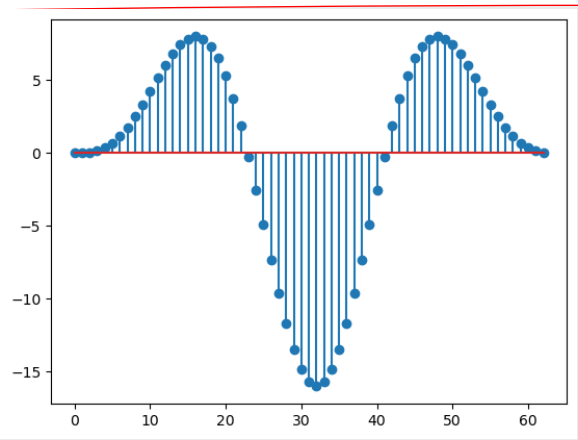
Part c.

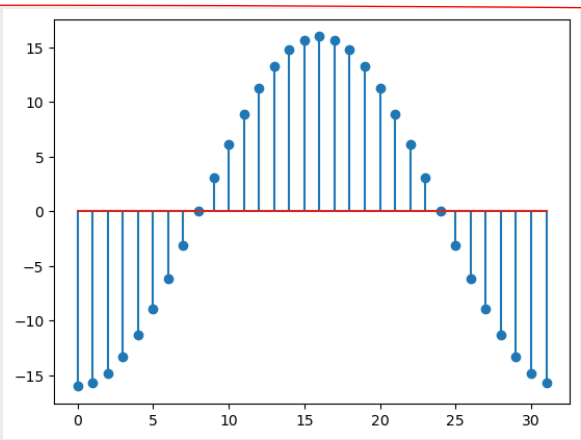
Let $x_3[n]$ represent the following signal:

$$x_3[n] = \begin{cases} \sin(2\pi n/32) & \text{if } 0 \leq n < 32 \\ 0 & \text{otherwise} \end{cases}$$

Compute the conventional convolution of $x_3[\cdot]$ with itself.

Compare the result with an analogous circular convolution, where the analysis window is $N = 32$. Briefly explain how the two results differ.





In this problem, one cycle of a sine wave with length 32 was convolved with itself. The result was greatest when the positive half cycles of the signals overlapped ($n = 16$) and smallest when the positive and negative halves overlapped ($n = 32$).

As in the previous part, circular convolution is confined to the range $0 \leq n < 32$. Samples outside that range are wrapped (i.e., aliased) back into the range, so that

$$(x_2 \circledast x_2)[n] = (x_2 * x_2)[n] + (x_2 * x_2)[n + 32]$$

for $0 \leq n < 32$.

Interestingly, The result is a negative cosine wave. This result makes sense in the frequency domain. The Fourier series representation of the original sine wave is $X_4[k]$:

$$X_4[k] = -\frac{j}{2}\delta[k-1] + \frac{j}{2}\delta[k+1]$$

Convolution in time is equivalent to multiplication in frequency. If

$$x_5[n] = (x_4 * x_4)[n]$$

then

$$X_5[k] = X_4[k] \times X_4[k] = -\frac{1}{2}\delta[k-1] - \frac{1}{2}\delta[k+1]$$

which is a negative cosine signal.