Names of team members:

1- Elsheshtawy, Mohamed

2- Mehta, Saanvi

3- Parkansky, Brandon

In [6]:
```
!pip install -q numpy matplotlib
```

In [7]:
```
import numpy as np
import matplotlib.pyplot as plt
import math
import warnings
from scipy.io.wavfile import write

plt.rcParams["figure.dpi"] = 100
```

0 Useful objects and functions

We suggest implementing the following functions that you can use throughout the lab.

In [9]:
```
def cexp(k, N):
    """
    A function that creates complex exponential with frequency k and length N.

    Args:
        k: discrete frequency
        N: length

    Returns:
        a numpy array of length N.
    """
    ##############################
    # YOUR CODE HERE
    output = np.empty(N, dtype=complex)
    for i in range(N):
        output[i]= np.exp(1j*2*np.pi*k*(i/N))
    return output
    ##############################
def inner_product(x, y):
    """
    Computes the inner product between two numpy arrays x and y.

    Args:
        x: numpy array.
        y: numpy array.
    """
    ##############################
    # YOUR CODE HERE
```

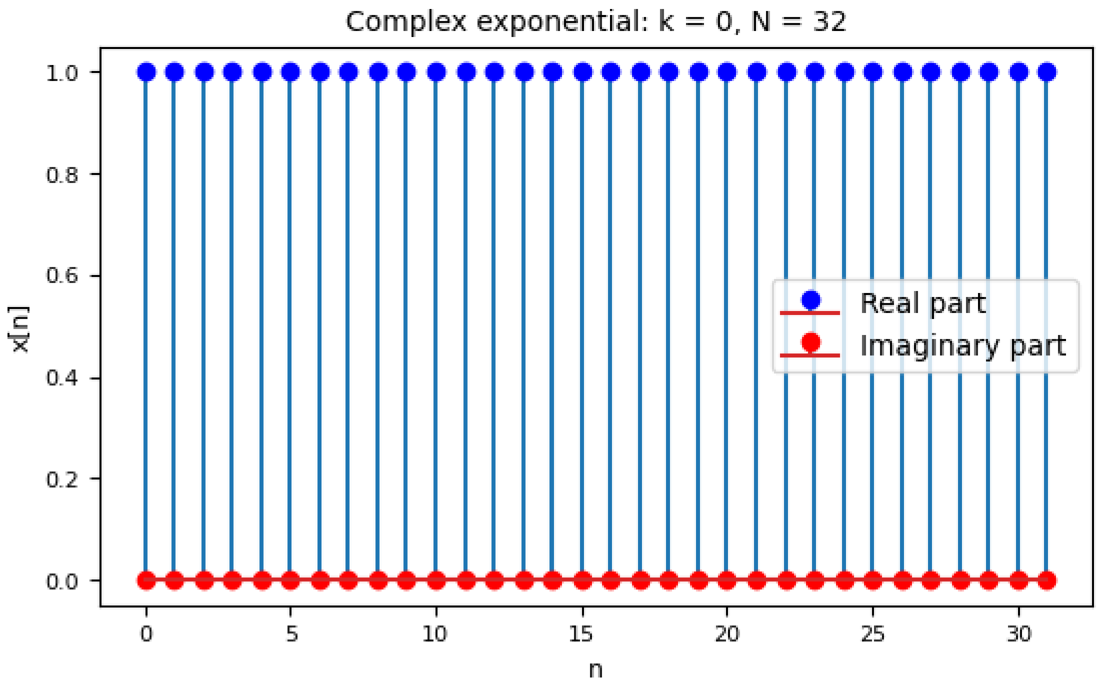
```
    return np.sum(np.conjugate(x) * y)
    ##############################
```

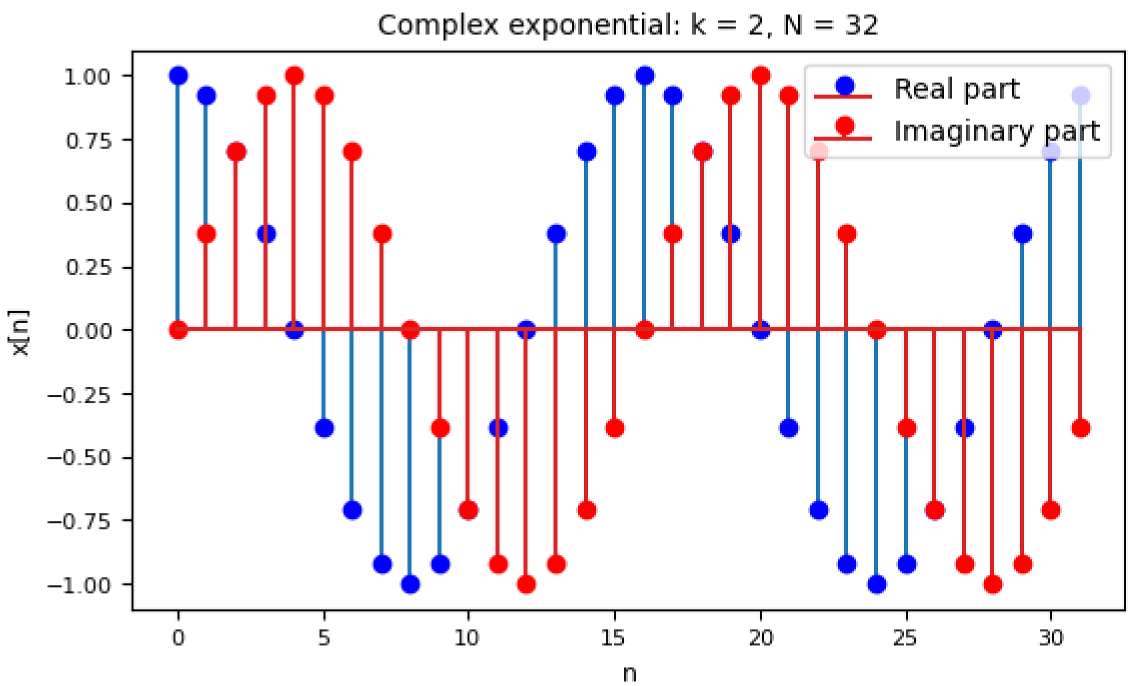
1 Signal Generation

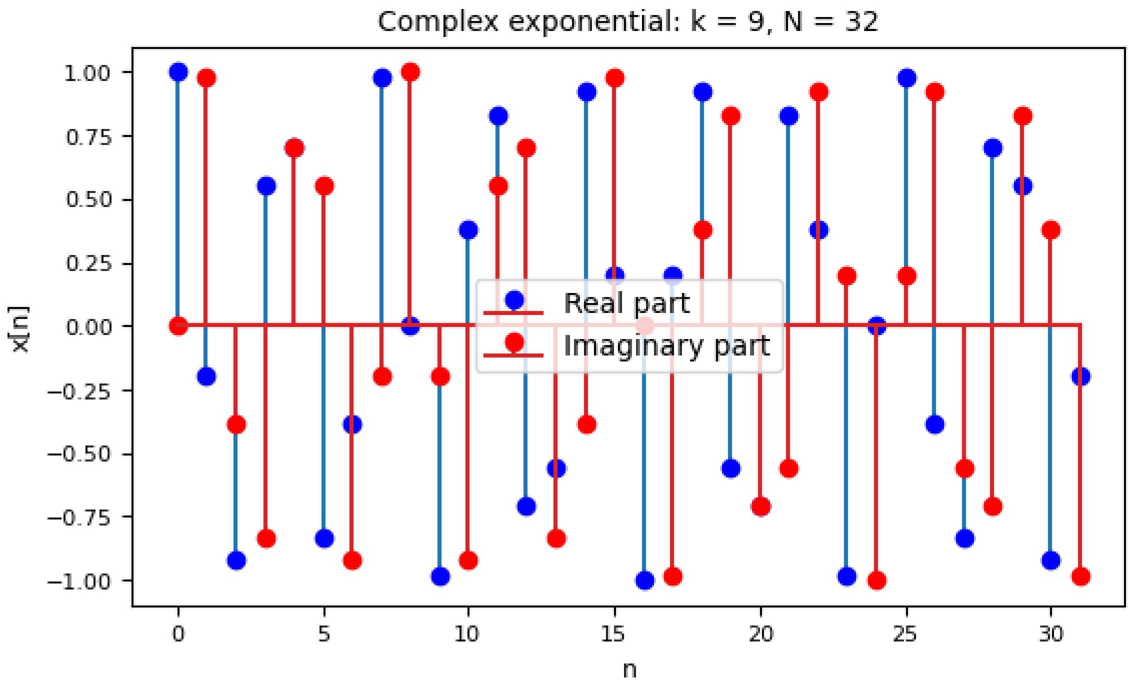
1.1 Generating Complex Exponentials

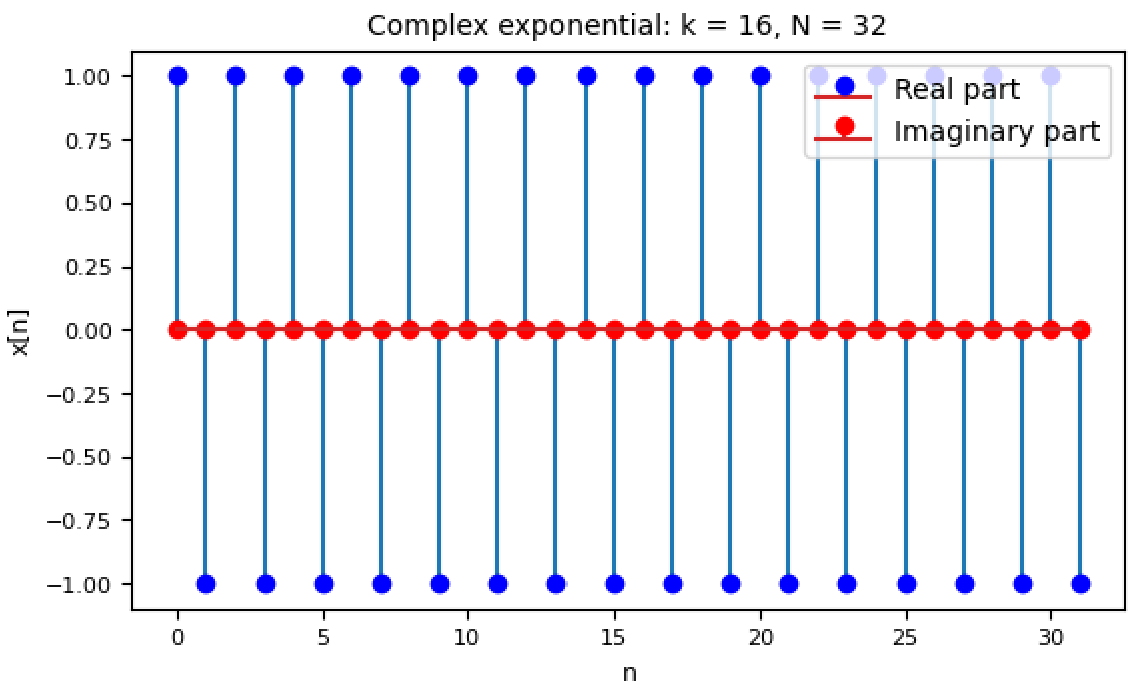
In [12]:
```
def q_11(N, k_list):
    """
    Plot complex exponentials of length N for each frequency k in k_list.
    Args:
        k_list: List of discrete frequencies
        N: length
    """

    for k in k_list:
        # TO DO: Create a complex exponential signal of length N and frequency k
        ##############################
        # YOUR CODE HERE
        data = cexp(k, N)
        ##############################

        # TO DO: Obtain the real and imaginary parts
        ##############################
        # YOUR CODE HERE
        cpx_cos = np.real(data)
        cpx_sin = np.imag(data)
        ##############################

        # Plots real and imaginary parts
        cpx_plt = plt.figure()
        ax = cpx_plt.add_subplot(111)
        plt.stem(np.arange(N), cpx_cos, 'tab:blue', markerfmt='bo', label='Real
        plt.stem(np.arange(N), cpx_sin, 'tab:red', markerfmt='ro', label='Imagin
        plt.title('Complex exponential: k = ' + str(k) + ', N = ' + str(N), font
        plt.xlabel('n', fontsize=9)
        plt.ylabel('x[n]', fontsize=9)
        plt.xticks(fontsize=8)
        plt.yticks(fontsize=8)
        plt.legend()

        # Aspect ratio credit: https://jdhao.github.io/2017/06/03/change-aspect-
        ratio = 1/(16/9)
        xleft, xright = ax.get_xlim()
        ybottom, ytop = ax.get_ylim()
        ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

        # Saves plot
        # fig_path_name = 'q11_cpxexp_k' + str(k) + '.png'
        # plt.savefig(fig_path_name, dpi=300)
        plt.show()
```

In [13]:
```
q_11(32, [0,2,9,16])
```


Complex exponential: k = 0, N = 32
x[n]
n
Real part
Imaginary part


Complex exponential: k = 2, N = 32
x[n]
n
Real part
Imaginary part

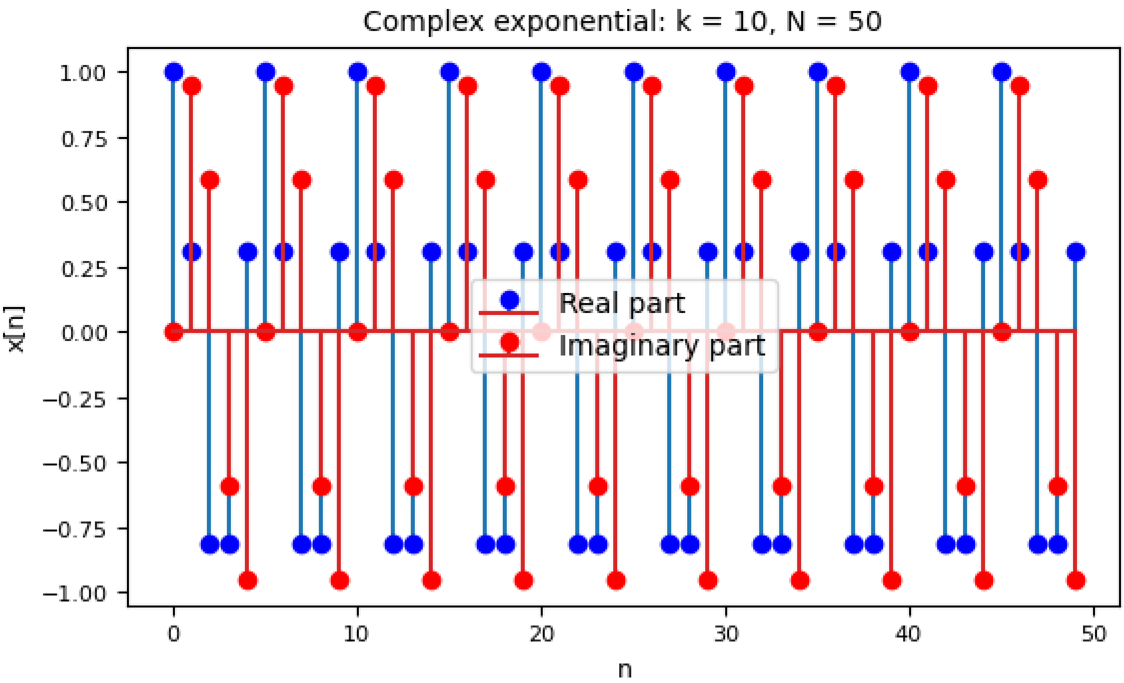


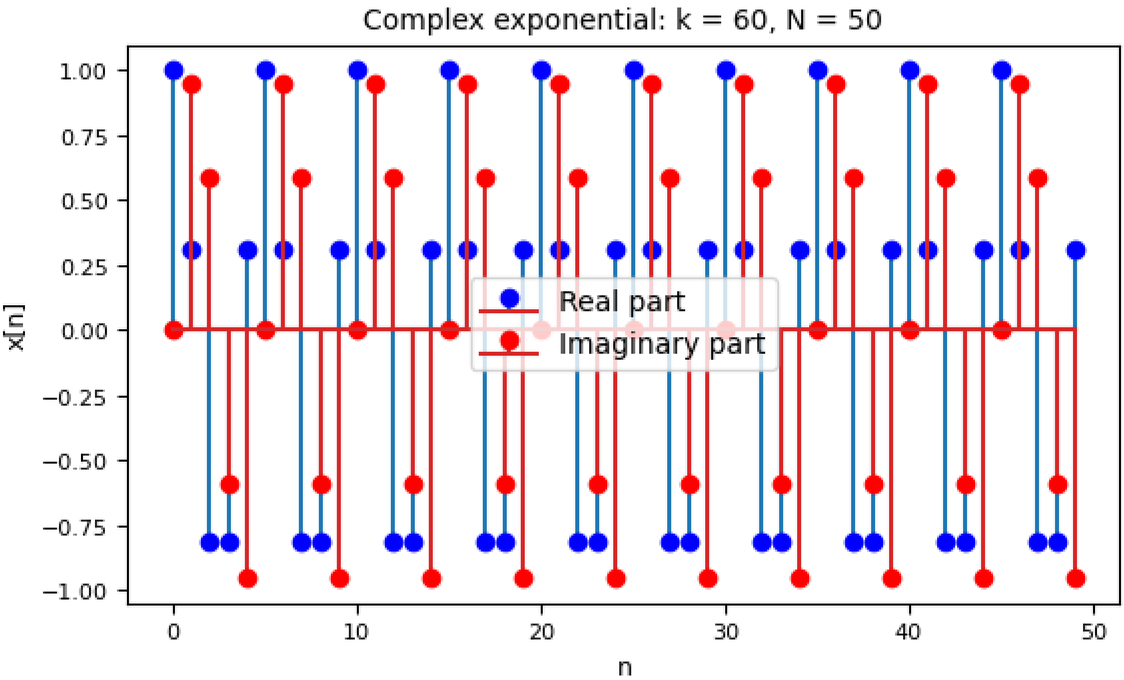


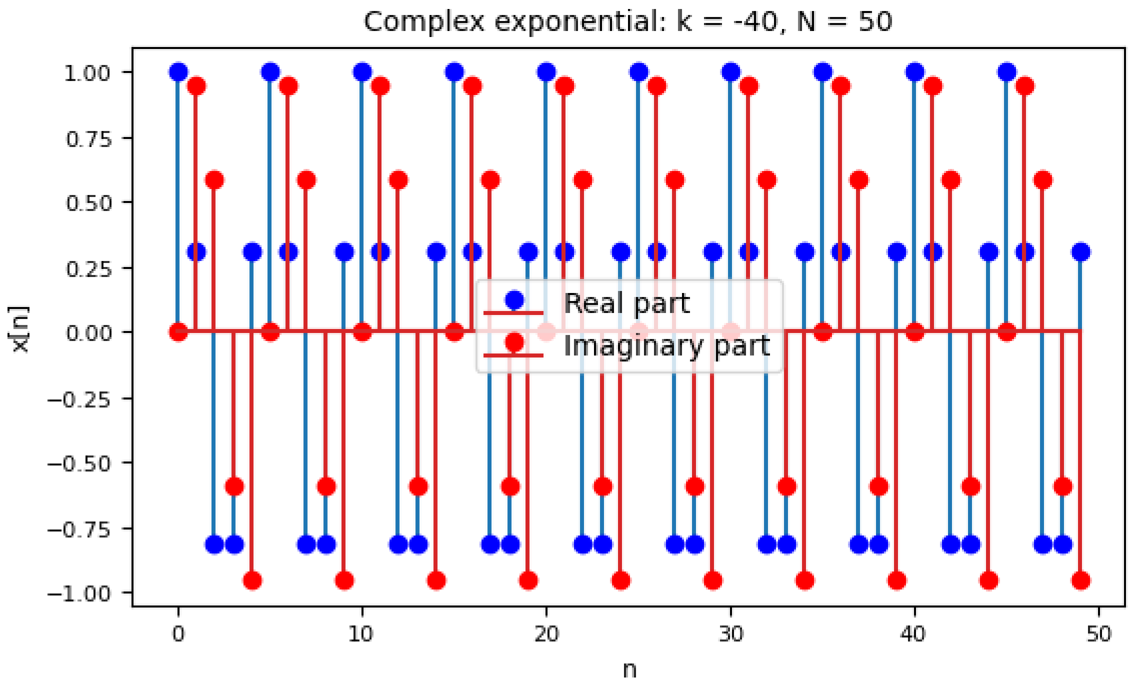
1.2 Equivalent Complex Exponentials

Generate complex exponentials of the same duration and frequencies, k and l, that are N apart. E.g., make $N = 32$ and plot signals for frequencies $k = 3$ and $l = 3 + 32 = 35$ and $l = 3 - 32 = -29$. You should observe that these signals are identical.

In [15]:
```
q_11(50,[10,60,-40])
```


Complex exponential: k = 10, N = 50
x[n]
n
Real part
Imaginary part

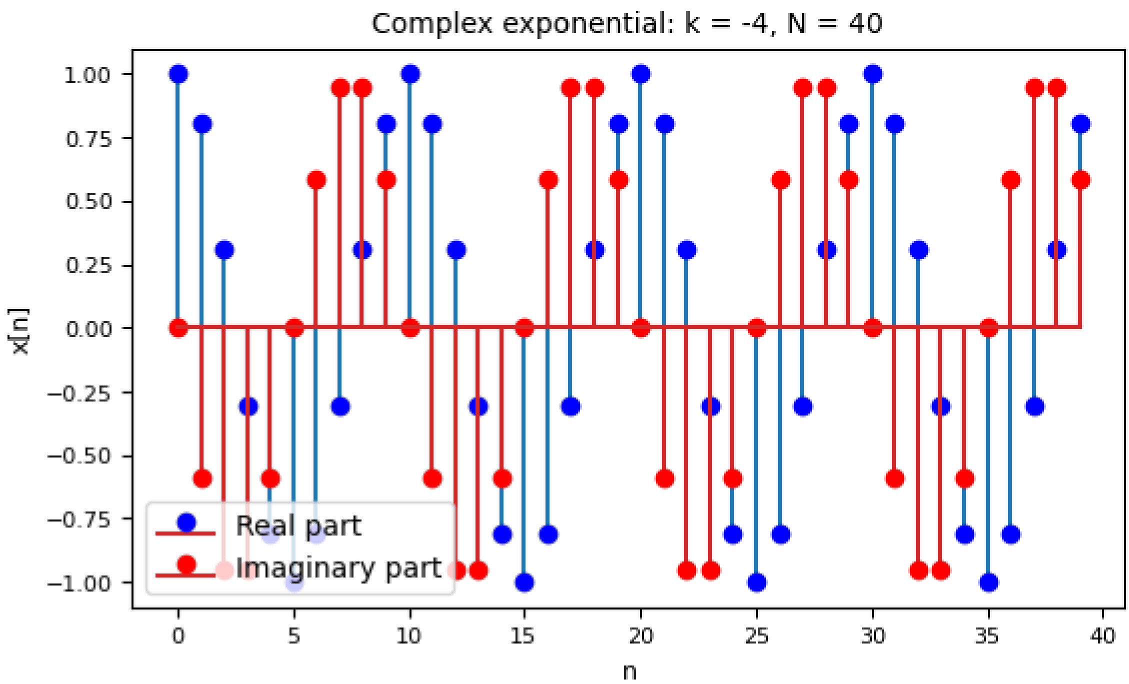

Complex exponential: k = 60, N = 50
x[n]
n
Real part
Imaginary part

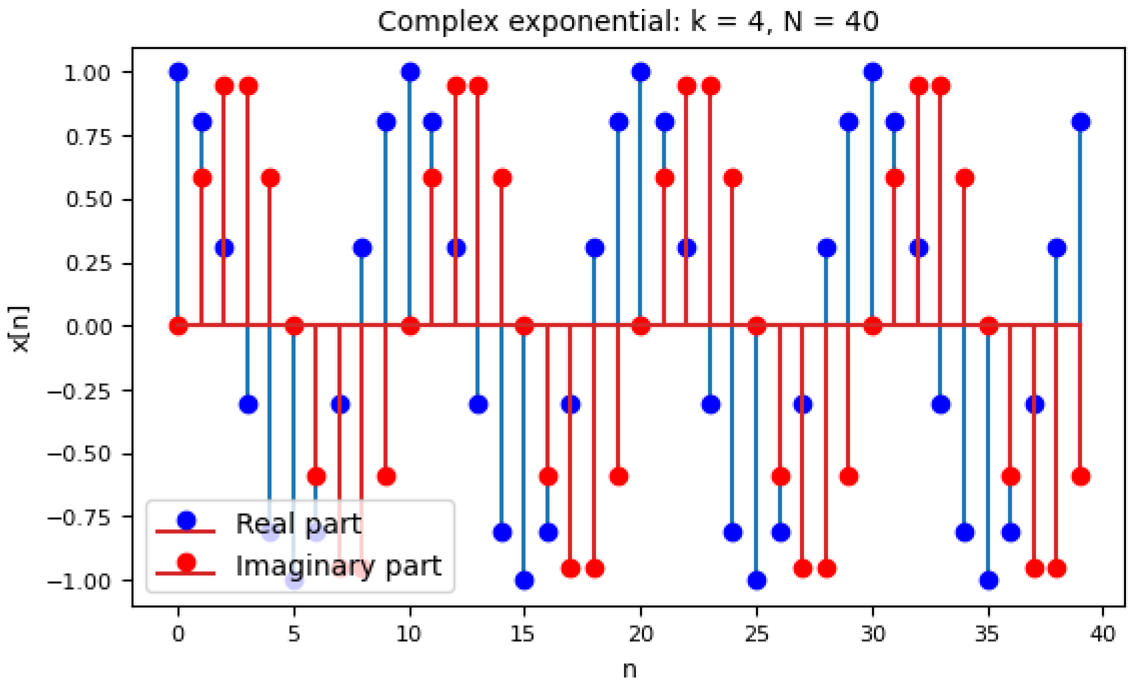


1.3 Conjugate complex exponentials

Generate complex exponentials of the same duration and opposite frequencies k and $-k$. E.g., make $N = 32$ and plot signals for frequencies $k = 3$ and $k = -3$. You should observe that these signals have the same real part and opposite imaginary parts. We say that the signals are conjugates of each other.

In [17]:
```
q_11(40,[-4,4])
```

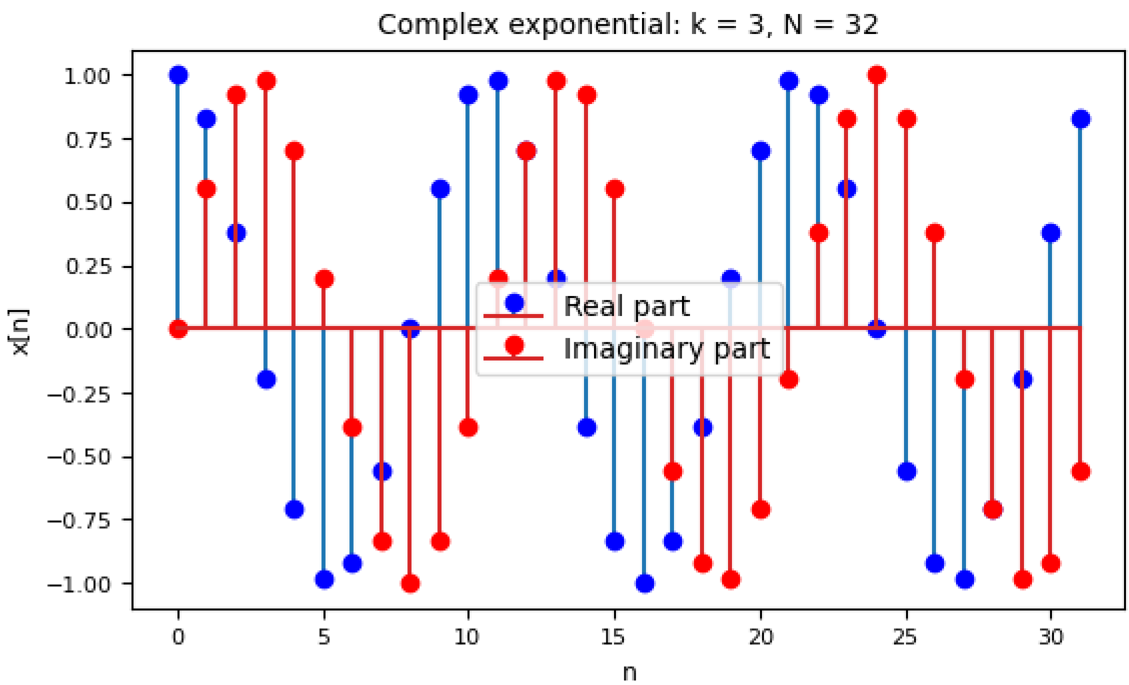


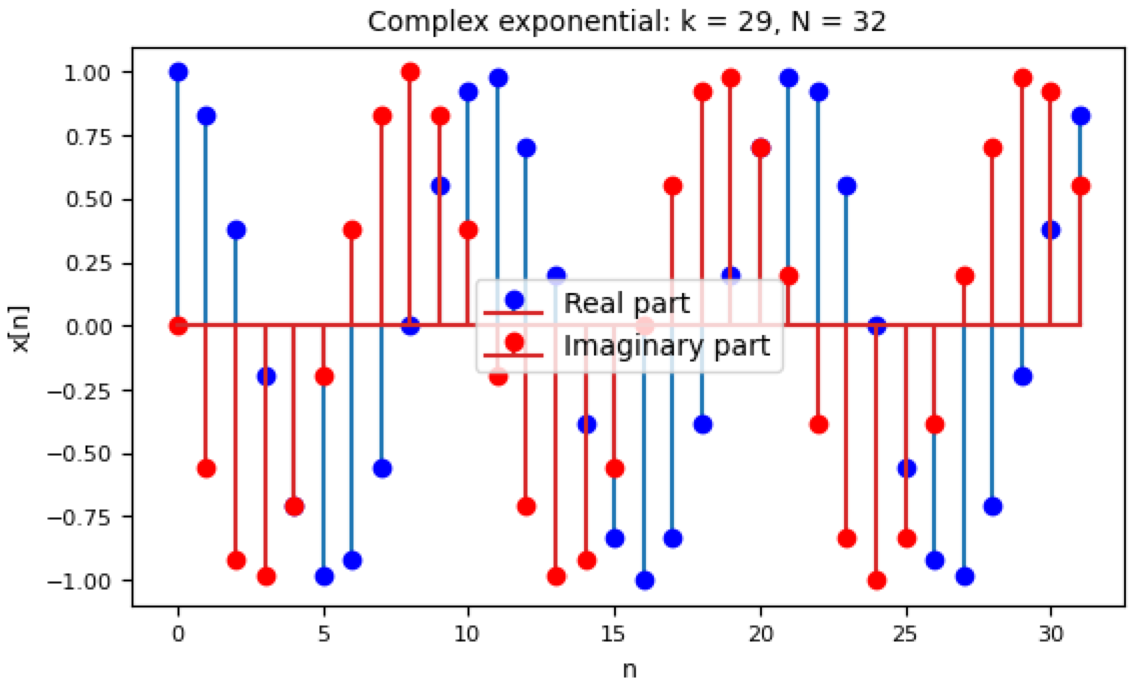


1.4 More conjugate complex exponentials

Consider now frequencies k and l in the interval $[0, N-1]$ such that their sum is $k+l=N$. E.g., verify this for $k=3$ and $l=32-3=29$.

In [19]:
```
q_11(32,[3,29])
```





1.5 Orthonormality

Write a function to compute the inner product $\langle e_{kN}, e_{lN} \rangle$ between all pairs of discrete complex exponentials of length N and frequencies $k, l = 0, 1, \ldots, N-1$. Run and report your result for $N = 16$. You should observe that the complex exponentials have unit energy and are orthogonal to each other. When this happens, we say that the signals form an orthonormal set.

In [37]:

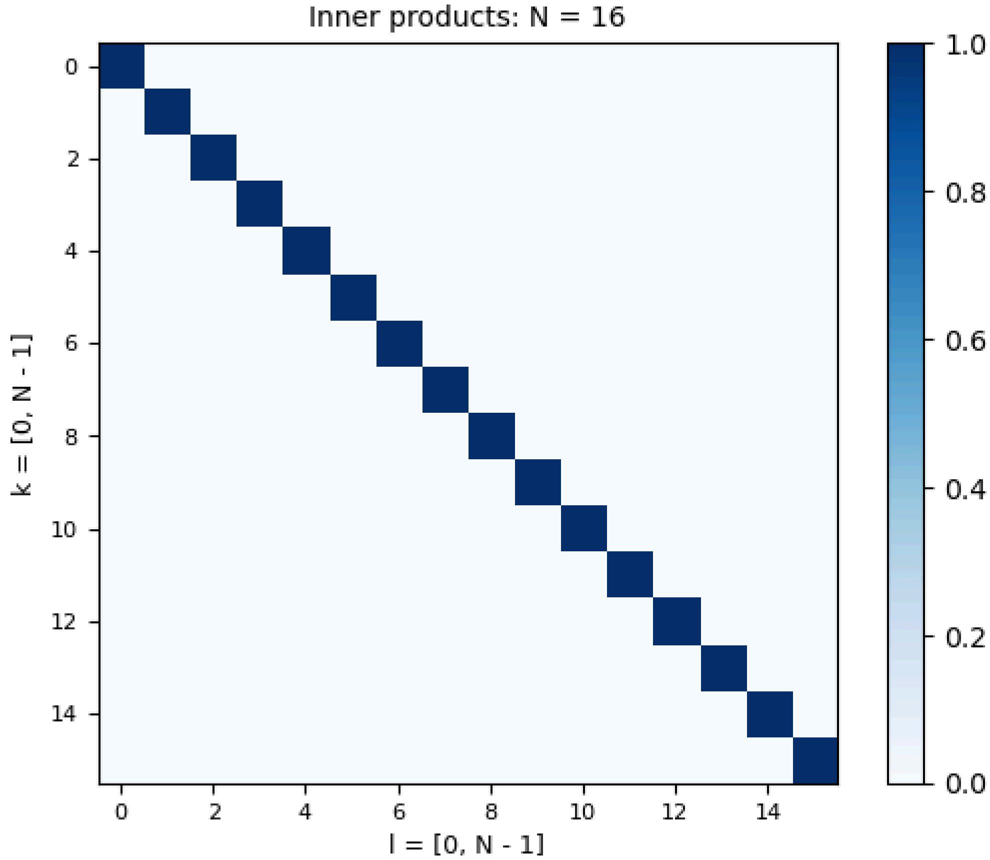
```
def q_15(k_list, N):
    """
    Print and visualize the inner product between pairs of discrete complex expo
    Args:
        k_list: List of discrete frequencies
        N: length
    """

    cpx_exps = np.zeros((N,N), dtype=complex)
    # TO DO: Build a matrix whose jth column stores the signal with k_list[j] fr
    ##############################
    # YOUR CODE HERE
    for k in k_list:
        cpx_exps[k]= cexp(k,N)*(1/math.sqrt(N))
    ##############################

    cpx_exps_conj = np.conjugate(cpx_exps)
    res = np.zeros((N,N), dtype=complex) # store inner products in this matrix
    # TO DO: Loop over the pairs and store the inner products in a matrix called
    ##############################
    # YOUR CODE HERE
    for i in range(N):
        for j in range(N):
            res[i][j] = inner_product(cpx_exps[i],cpx_exps[j])
    ##############################

    res = res.real
```

```python
    print (f"Matrix of inner products: \n{res}")

    # Let us also visualize the inner products
    fig, ax = plt.subplots()
    im = ax.imshow(res, cmap = 'Blues')
    plt.title('Inner products: N = ' + str(N), fontsize=10)
    plt.xlabel('l = [0, N - 1]', fontsize=9)
    plt.ylabel('k = [0, N - 1]', fontsize=9)
    plt.xticks(fontsize=8)
    plt.yticks(fontsize=8)
    fig.colorbar(im, ax=ax)
    # Saves plot
    # fig_path_name = 'q15_colormap.png'
    # plt.savefig(fig_path_name, dpi=300)
    plt.show()

    return res
```

In [39]:
```python
q_15(list(range(16)),16)
```

```
Matrix of inner products:
[[ 1.00000000e+00 -5.65838741e-17 -1.43029718e-18 -8.00485582e-17
  -3.33066907e-16 -1.58019280e-16 -2.52448183e-16  1.87920302e-16
   0.00000000e+00 -1.31239529e-17 -2.79681676e-17  2.16497322e-16
  -5.82867088e-16 -2.74153919e-16  5.65123593e-16  1.77856989e-15]
 [-5.65838741e-17  1.00000000e+00 -4.96449802e-17  5.36343784e-17
  -1.07804134e-16  1.11022302e-16 -8.16914474e-17 -7.34647961e-17
  -9.31049014e-17  0.00000000e+00 -1.65779619e-16  1.50838720e-16
   1.74863958e-16 -1.36002321e-15  1.80343632e-16  1.09814466e-15]
 [-1.43029718e-18 -4.96449802e-17  1.00000000e+00 -1.07272289e-18
   5.50859672e-18  1.83629410e-16  4.71844785e-16  5.98320155e-16
   1.01435406e-16  4.03961325e-18  0.00000000e+00  4.23871983e-17
   5.06326663e-16  3.72622435e-16 -2.49800181e-16 -6.73140318e-16]
 [-8.00485582e-17  5.36343784e-17 -1.07272289e-18  1.00000000e+00
  -9.47477906e-17 -4.64015383e-16 -8.00485582e-17 -1.11022302e-16
  -1.99652644e-16 -2.47258095e-17  1.32409150e-16  0.00000000e+00
  -3.74100816e-17  1.15016038e-16 -1.15046478e-15  8.04911693e-16]
 [-3.33066907e-16 -1.07804134e-16  5.50859672e-18 -9.47477906e-17
   1.00000000e+00 -5.31144272e-17  1.02653111e-16  4.83209790e-17
  -3.33066907e-16 -5.95169596e-16 -4.42813658e-17 -1.33076215e-17
   0.00000000e+00  3.09534614e-16  1.94076437e-16  1.74863958e-16]
 [-1.58019280e-16  1.11022302e-16  1.83629410e-16 -4.64015383e-16
  -5.31144272e-17  1.00000000e+00  5.09689814e-17  1.09145530e-16
  -4.13115466e-16  5.55111512e-17 -1.68427621e-16 -1.20599961e-16
  -2.02465154e-17  0.00000000e+00 -3.04557497e-16 -9.06338813e-16]
 [-2.52448183e-16 -8.16914474e-17  4.71844785e-16 -8.00485582e-17
   1.02653111e-16  5.09689814e-17  1.00000000e+00 -2.53588515e-17
  -9.16359179e-17 -2.80068539e-17 -3.33066907e-16 -9.55692352e-17
  -2.31631501e-16 -3.60252317e-16  0.00000000e+00 -4.12110352e-16]
 [ 1.87920302e-16 -7.34647961e-17  5.98320155e-16 -1.11022302e-16
   4.83209790e-17  1.09145530e-16 -2.53588515e-17  1.00000000e+00
  -1.50258942e-16  9.51010126e-17  3.64040652e-16 -3.88578059e-16
  -7.54764156e-16 -1.47227325e-16  1.77511961e-16  0.00000000e+00]
 [ 0.00000000e+00 -9.31049014e-17  1.01435406e-16 -1.99652644e-16
  -3.33066907e-16 -4.13115466e-16 -9.16359179e-17 -1.50258942e-16
   1.00000000e+00  2.14032988e-16 -7.99403096e-16 -1.59845838e-16
   1.30451205e-15 -1.82305409e-16 -2.59387077e-16  5.38334444e-16]
 [-1.31239529e-17  0.00000000e+00  4.03961325e-18 -2.47258095e-17
  -5.95169596e-16  5.55111512e-17 -2.80068539e-17  9.51010126e-17
   2.14032988e-16  1.00000000e+00 -3.27200736e-16 -5.76308048e-16
  -1.11273581e-16  1.16573418e-15  1.80568769e-15  1.05507420e-15]
 [-2.79681676e-17 -1.65779619e-16  0.00000000e+00  1.32409150e-16
  -4.42813658e-17 -1.68427621e-16 -3.33066907e-16  3.64040652e-16
  -7.99403096e-16 -3.27200736e-16  1.00000000e+00 -1.02455957e-15
   1.02552600e-15  4.83209790e-17  8.32667268e-17 -1.23025083e-17]
 [ 2.16497322e-16  1.50838720e-16  4.23871983e-17  0.00000000e+00
  -1.33076215e-17 -1.20599961e-16 -9.55692352e-17 -3.88578059e-16
  -1.59845838e-16 -5.76308048e-16 -1.02455957e-15  1.00000000e+00
   1.13343643e-15  1.63457398e-15  2.46079455e-16 -3.88578059e-16]
 [-5.82867088e-16  1.74863958e-16  5.06326663e-16 -3.74100816e-17
   0.00000000e+00 -2.02465154e-17 -2.31631501e-16 -7.54764156e-16
   1.30451205e-15 -1.11273581e-16  1.02552600e-15  1.13343643e-15
   1.00000000e+00  1.24445873e-15 -7.92464202e-16  9.05274376e-16]
 [-2.74153919e-16 -1.36002321e-15  3.72622435e-16  1.15016038e-16
   3.09534614e-16  0.00000000e+00 -3.60252317e-16 -1.47227325e-16
  -1.82305409e-16  1.16573418e-15  4.83209790e-17  1.63457398e-15
   1.24445873e-15  1.00000000e+00 -1.32293201e-15 -1.98844004e-15]
 [ 5.65123593e-16  1.80343632e-16 -2.49800181e-16 -1.15046478e-15
   1.94076437e-16 -3.04557497e-16  0.00000000e+00  1.77511961e-16
  -2.59387077e-16  1.80568769e-15  8.32667268e-17  2.46079455e-16
```

```
 -7.92464202e-16 -1.32293201e-15  1.00000000e+00 -1.29864588e-15]
[ 1.77856989e-15  1.09814466e-15 -6.73140318e-16  8.04911693e-16
  1.74863958e-16 -9.06338813e-16 -4.12110352e-16  0.00000000e+00
  5.38334444e-16  1.05507420e-15 -1.23025083e-17 -3.88578059e-16
  9.05274376e-16 -1.98844004e-15 -1.29864588e-15  1.00000000e+00]]
```



Out[39]:
```
array([[ 1.00000000e+00, -5.65838741e-17, -1.43029718e-18,
        -8.00485582e-17, -3.33066907e-16, -1.58019280e-16,
        -2.52448183e-16,  1.87920302e-16,  0.00000000e+00,
        -1.31239529e-17, -2.79681676e-17,  2.16497322e-16,
        -5.82867088e-16, -2.74153919e-16,  5.65123593e-16,
         1.77856989e-15],
       [-5.65838741e-17,  1.00000000e+00, -4.96449802e-17,
         5.36343784e-17, -1.07804134e-16,  1.11022302e-16,
        -8.16914474e-17, -7.34647961e-17, -9.31049014e-17,
         0.00000000e+00, -1.65779619e-16,  1.50838720e-16,
         1.74863958e-16, -1.36002321e-15,  1.80343632e-16,
         1.09814466e-15],
       [-1.43029718e-18, -4.96449802e-17,  1.00000000e+00,
        -1.07272289e-18,  5.50859672e-18,  1.83629410e-16,
         4.71844785e-16,  5.98320155e-16,  1.01435406e-16,
         4.03961325e-18,  0.00000000e+00,  4.23871983e-17,
         5.06326663e-16,  3.72622435e-16, -2.49800181e-16,
        -6.73140318e-16],
       [-8.00485582e-17,  5.36343784e-17, -1.07272289e-18,
         1.00000000e+00, -9.47477906e-17, -4.64015383e-16,
        -8.00485582e-17, -1.11022302e-16, -1.99652644e-16,
        -2.47258095e-17,  1.32409150e-16,  0.00000000e+00,
        -3.74100816e-17,  1.15016038e-16, -1.15046478e-15,
         8.04911693e-16],
       [-3.33066907e-16, -1.07804134e-16,  5.50859672e-18,
        -9.47477906e-17,  1.00000000e+00, -5.31144272e-17,
         1.02653111e-16,  4.83209790e-17, -3.33066907e-16,
        -5.95169596e-16, -4.42813658e-17, -1.33076215e-17,
         0.00000000e+00,  3.09534614e-16,  1.94076437e-16,
         1.74863958e-16],
       [-1.58019280e-16,  1.11022302e-16,  1.83629410e-16,
        -4.64015383e-16, -5.31144272e-17,  1.00000000e+00,
         5.09689814e-17,  1.09145530e-16, -4.13115466e-16,
         5.55111512e-17, -1.68427621e-16, -1.20599961e-16,
        -2.02465154e-17,  0.00000000e+00, -3.04557497e-16,
        -9.06338813e-16],
       [-2.52448183e-16, -8.16914474e-17,  4.71844785e-16,
        -8.00485582e-17,  1.02653111e-16,  5.09689814e-17,
         1.00000000e+00, -2.53588515e-17, -9.16359179e-17,
        -2.80068539e-17, -3.33066907e-16, -9.55692352e-17,
        -2.31631501e-16, -3.60252317e-16,  0.00000000e+00,
        -4.12110352e-16],
       [ 1.87920302e-16, -7.34647961e-17,  5.98320155e-16,
        -1.11022302e-16,  4.83209790e-17,  1.09145530e-16,
        -2.53588515e-17,  1.00000000e+00, -1.50258942e-16,
         9.51010126e-17,  3.64040652e-16, -3.88578059e-16,
        -7.54764156e-16, -1.47227325e-16,  1.77511961e-16,
         0.00000000e+00],
       [ 0.00000000e+00, -9.31049014e-17,  1.01435406e-16,
        -1.99652644e-16, -3.33066907e-16, -4.13115466e-16,
        -9.16359179e-17, -1.50258942e-16,  1.00000000e+00,
         2.14032988e-16, -7.99403096e-16, -1.59845838e-16,
         1.30451205e-15, -1.82305409e-16, -2.59387077e-16,
         5.38334444e-16],
       [-1.31239529e-17,  0.00000000e+00,  4.03961325e-18,
        -2.47258095e-17, -5.95169596e-16,  5.55111512e-17,
        -2.80068539e-17,  9.51010126e-17,  2.14032988e-16,
         1.00000000e+00, -3.27200736e-16, -5.76308048e-16,
        -1.11273581e-16,  1.16573418e-15,  1.80568769e-15,
         1.05507420e-15],
```

```
[-2.79681676e-17, -1.65779619e-16,  0.00000000e+00,
  1.32409150e-16, -4.42813658e-17, -1.68427621e-16,
 -3.33066907e-16,  3.64040652e-16, -7.99403096e-16,
 -3.27200736e-16,  1.00000000e+00, -1.02455957e-15,
  1.02552600e-15,  4.83209790e-17,  8.32667268e-17,
 -1.23025083e-17],
[ 2.16497322e-16,  1.50838720e-16,  4.23871983e-17,
  0.00000000e+00, -1.33076215e-17, -1.20599961e-16,
 -9.55692352e-17, -3.88578059e-16, -1.59845838e-16,
 -5.76308048e-16, -1.02455957e-15,  1.00000000e+00,
  1.13343643e-15,  1.63457398e-15,  2.46079455e-16,
 -3.88578059e-16],
[-5.82867088e-16,  1.74863958e-16,  5.06326663e-16,
 -3.74100816e-17,  0.00000000e+00, -2.02465154e-17,
 -2.31631501e-16, -7.54764156e-16,  1.30451205e-15,
 -1.11273581e-16,  1.02552600e-15,  1.13343643e-15,
  1.00000000e+00,  1.24445873e-15, -7.92464202e-16,
  9.05274376e-16],
[-2.74153919e-16, -1.36002321e-15,  3.72622435e-16,
  1.15016038e-16,  3.09534614e-16,  0.00000000e+00,
 -3.60252317e-16, -1.47227325e-16, -1.82305409e-16,
  1.16573418e-15,  4.83209790e-17,  1.63457398e-15,
  1.24445873e-15,  1.00000000e+00, -1.32293201e-15,
 -1.98844004e-15],
[ 5.65123593e-16,  1.80343632e-16, -2.49800181e-16,
 -1.15046478e-15,  1.94076437e-16, -3.04557497e-16,
  0.00000000e+00,  1.77511961e-16, -2.59387077e-16,
  1.80568769e-15,  8.32667268e-17,  2.46079455e-16,
 -7.92464202e-16, -1.32293201e-15,  1.00000000e+00,
 -1.29864588e-15],
[ 1.77856989e-15,  1.09814466e-15, -6.73140318e-16,
  8.04911693e-16,  1.74863958e-16, -9.06338813e-16,
 -4.12110352e-16,  0.00000000e+00,  5.38334444e-16,
  1.05507420e-15, -1.23025083e-17, -3.88578059e-16,
  9.05274376e-16, -1.98844004e-15, -1.29864588e-15,
  1.00000000e+00]])
```

2 Analysis

I will add a pdf to the end of this pdf

3 Generating and Playing Musical Tones

3.1 & 3.2 Discrete Cosine Generation & Generating an A Note

Write down a function that takes as input the sampling frequency f_s, the time duration T, and the frequency f_0 and returns the associated discrete cosine $x(n)$. Your function has to also return the number of samples N. When T is not a multiple of $T_s = 1/f_s$ you can reduce T to the largest multiple of T_s smaller than T.

The musical A note corresponds to an oscillation at frequency $f_0 = 440$ Hertz.Use the code you have just written to generate an A note of duration $T = 2$ seconds sample data frequency $f_s = 44,100$ Hertz. Play the note in your computer's speakers.

In [23]:
```
def cexpt(f, T, fs):
    """
    This function generates a (sampled) continuous-time complex exponential.
    Arguments:
        f: frequency of the complex exponential
        T: duration
        fs: sampling frequency
    Returns
        x: vector of samples of a complex exponential of frequency f and duratio
        N: number of samples
    """
    assert T > 0, "Duration of the signal cannot be negative."
    assert fs != 0, "Sampling frequency cannot be zero"

    if fs < 0:
        warnings.warn("Sampling frequency is negative. Using absolute value inst
        fs = - fs

    if f < 0:
        warnings.warn("Complex exponential frequency is negative. Using absolute
        f = -f

    # TO DO: Convert the frequency f to a discrete frequency and generate a
    # discrete complex exponential.
    #############################
    # YOUR CODE HERE
    Ts = 1/fs
    N = math.ceil(T/Ts)
    k = (f/fs)*N
    x = cexp(k,N)
    #############################

    return x, N


def q_32(f0, T, fs):
    # Retrieves complex exponential
    cpxexp, num_samples = cexpt(f0, T, fs)
    # Cosine is the real part
    Anote = cpxexp.real
    # Playing the note
    write("Anote.wav", fs, Anote.astype(np.float32))
```

In [24]:
```
q_32(440,2,44100)
```

3.3 & 3.4 Generate Musical Notes

In [26]:
```
def q_33(f0, T, fs):
    cpxexp,num_samples = cexpt(0,0.5,fs)
    audio_sequence = cpxexp.real
    for i in range(len(f0)):
        cpxexp, num_samples = cexpt(f0[i],T[i],fs)
        temp = cpxexp.real
```

```
        audio_sequence = np.concatenate((audio_sequence, temp))
    write("output.wav",fs,audio_sequence.astype(np.float32))
q_33([659.25,622.25,659.25,622.25,659.25,493.88,587.33,523.25,444,0,261.63,329.6
     [0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,1.0,0.5,0.5,0.5,0.5,1,0.5,0.5,0.5,0.5,1,0.
     ,44100)
```

$|k-1| = N$

(2.1) $e_{kN}(n)$

$k = 1 + N$
$k = 1 - N$

$|k-1| = N$
$k = 1 + N$
$k = 1 - N$

$$e^{j 2\pi n \frac{k}{N}} = e^{j 2\pi \frac{(N+1)n}{N}} = e^{j 2\pi \left(\frac{Nn}{\cancel{N}} + \frac{1n}{N}\right)}$$

$$= e^{(j 2\pi n + j 2\pi \frac{n}{N})}$$

$$= 1 \cdot e^{j 2\pi \frac{1n}{N}}$$

$$= e^{j 2\pi \frac{1n}{N}}$$

2.2

a) $K-l \in N$

$|k-L| = xN$

$x \in \mathbb{Z}$

$e^{j2\pi \frac{nk}{N}} = e^{j2\pi \frac{(xN+l)n}{N}} =$

$e^{j2\pi\left(\frac{xNn}{N} + \frac{ln}{N}\right)} = e^{j2\pi\left(xn + \frac{ln}{N}\right)}$

$e^{j2\pi xn} \cdot e^{j2\pi \frac{nL}{N}}$

$\downarrow$

$1 \cdot e^{j2\pi \frac{nL}{N}} = e^{j2\pi \frac{nL}{N}}$

b) $\langle e_{kN}, e_{lN} \rangle = \sum_{n=0}^{N-1} e_{kN}(n) e^*_{lN}(n) = \sum_{n=0}^{N-1} e^{j2\pi kn/N} e^{-j2\pi ln/N}$

$\langle e_{kN}, e_{lN} \rangle = \sum_{n=0}^{N-1} e^{j2\pi(k-l)n/N} = \sum_{n=0}^{N-1} \left[e^{j2\pi(k-l)/N}\right]^n$

$\sum_{n=0}^{N-1} a^n = (1-a^N)/(1-a)$

$\langle e_{kN}, e_{lN} \rangle = \frac{1 - \left[e^{j2\pi(k-l)/N}\right]^N}{1 - e^{j2\pi(k-l)/N}} = \frac{1-1 = 0}{1 - e^{j2\pi(k-l)/N}}$

$$\cancel{\left[e^{j2\pi(k-1)/N}\right]^N = e^{j2\pi(k-1)} = \left[e^{j2\pi}\right]^{(k-l)} = 1}$$

$$\langle e_{kN}, e_{lN}\rangle = \sum_{n=0}^{N-1} e_{kN}(n)\, e_{lN}^*(n) =$$

$$\sum_{n=0}^{N-1} \frac{e^{j2\pi kn/N}}{\sqrt{N}} \, \frac{e^{-j2\pi ln/N}}{\sqrt{N}}$$

$$\langle e_{kN}, e_{lN}\rangle = \frac{1}{N}\sum_{n=0}^{N-1} e^{j2\pi(k-l)n/N} =$$

$$\frac{1}{N}\sum_{n=0}^{N-1}\left[e^{j2\pi(k-l)/N}\right]^n$$

~~$|k-l| \leq$~~

$k-l = xN$

$$\frac{1}{N}\sum_{n=0}^{N-1}\left[e^{j2\pi(xN)/N}\right]^n$$

$$\left[e^{j2\pi\frac{(x\cancel{N})}{\cancel{N}}}\right]^n$$

$$\uparrow$$

$$1$$

$$\rightarrow \frac{1}{N}\sum_{n=0}^{N-1}[1]^n = \frac{N}{N}$$

2.5 $e^{\phi}_{kN}(n) = e^{j\left(2\pi \frac{kn}{N} - \phi\right)}$

$$\overset{*}{e}{}^{\phi}_{lN}(n) = e^{-j\left(2\pi \frac{ln}{N} - \phi\right)}$$

$$\left\langle e^{\phi}_{kN}(n), e^{\phi}_{lN}(n) \right\rangle =$$

$$\sum_{n=0}^{N-1} e^{j\left(2\pi \frac{kn}{N} - \phi\right)} e^{-j\left(2\pi \frac{ln}{N} - \phi\right)}$$

$$= \sum_{n=0}^{N-1} e^{j\left(2\pi \frac{kn}{N}\right)} e^{-j\left(2\pi \frac{Ln}{N}\right)}$$

$$= \sum_{n=0}^{N-1} e^{j\left(2\pi \frac{kn}{N}\right)} e^{-j\left(2\pi \frac{Ln}{N}\right)}$$

ϕ cancelled

$$= \left\langle e_{kn}(n), e_{LN}(n) \right\rangle$$

independent on phase

From 2.3, 2.4

if $|K-L| = xN$ $\quad x \in \mathbb{Z}$ then they are equivalent

else they are orthogonal

2.6 From 2.3 and 2.4

if $|K-L| = xN$ such that $x \in \mathbb{Z}$ then they are equivalent

if $|K-L| = c$ such that $c \in \mathbb{Z}$ but c isn't a multiple of N

then they are orthogonal

else they aren't any of them