EXPERIMENT NO 4

NAME: Vishal Shashikant Salvi **UID: 2019230069**
CLASS: TE COMPS **BATCH: C**

Aim: To perform auto and cross-correlation between two signals.

Theory:

Correlation:

Correlation is a measure of similarity between two signals. The general formula for correlation is:

$$\int_{-\infty}^{\infty} x_1(t)x_2(t-\tau)dt$$

In general, correlation describes the mutual relationship which exists between two or more things. The same definition holds good even in the case of signals. That is, correlation between signals indicates the measure up to which the given signal resembles another signal. In other words, if we want to know how much similarity exists between the signals 1 and 2, then we need to find out the correlation of Signal 1 with respect to Signal 2 or vice versa.

Types of Correlation:

Depending on whether the signals considered for correlation are same or different, we have two kinds of correlation:

1. **Autocorrelation**
2. **Cross-correlation.**

Autocorrelation:

This is a type of correlation in which the given signal is correlated with itself, usually the time-shifted version of itself. The autocorrelation of the discrete time signal x[n] is expressed as:

$$R_{xx}[m] = \sum_{n=-\infty}^{\infty} x[n]\, x^{\star}[n-m]$$

The autocorrelation of any given signal can also be computed by resorting to graphical technique. The procedure involves sliding the time-shifted version of the given signal upon itself while computing the samples at every interval. That is, if the given signal is digital, then we shift the given signal by one sample every time and overlap it with the original signal. While doing so, for every shift and overlap, we perform multiply and add.

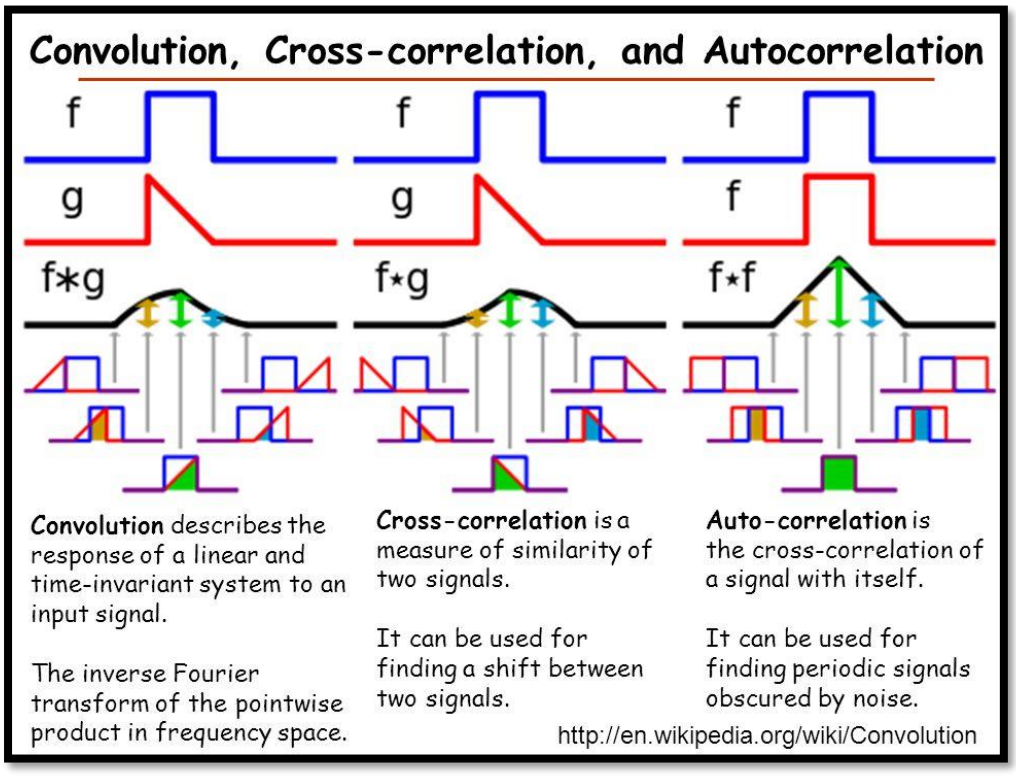
Cross correlation:

This is a kind of correlation, in which the signal in-hand is correlated with another signal so as to know how much resemblance exists between them.

The cross-correlation of the discrete time signals x [n] and y [n] is expressed as

$$R_{xy}[m] = \sum_{n=-\infty}^{\infty} x[n]\, y^{\star}[n-m]$$

Cross-correlation of any two given signals can be found via graphical techniques. Here, one signal is slid upon the other while computing the samples at every interval. That is, in the case of digital signals, one signal is shifted by one sample to the right each time, at which point the sum of the product of the overlapping samples is computed.



Code:

Cross Correlation:

```
import numpy as np
from tabulate import tabulate
import matplotlib.pyplot as plt


x = list ( map ( int , input ( "Enter values in x(n): " ).split()))
xstart = int ( input ( "Enter x start:" ))
y = list ( map ( int , input ( "Enter values in y(n):" ).split()))
ystart = int ( input ( "Enter y start:" ))

temp_y = y[:]
y.reverse()
ystart = len (y) - ystart - 1
table = [[ 'x(n) \\ y(-n)' ,]+y,]

Rxh = [ 0 ]*( len (x) + len (y) - 1 )
Rxh_start = max (xstart,ystart)

for i,x_i in enumerate (x):
   row,s = [x_i,], 0
   for j,y_i in enumerate (y):
      Rxh[i+j] += x_i*y_i
      row.append(x_i*y_i)
   table.append(row)

print ( "y(-n) :" ,y)
print ( "y(-n) origin:" , ystart)
print ( " \n Table:" )
print (tabulate(table, headers = "firstrow" , tablefmt = "pretty" ))
print ( " \n Rxy(n) :" , Rxh)
print ( "Rxy(n) origin:" , Rxh_start)

# plotting xn
xn1 = np.array(x)
xn2 = np.array( range (xstart, len (x)+xstart))
plt.figure( 1 )
plt.stem(xn2,xn1, use_line_collection = True )
plt.title( 'x(n)' )
plt.grid( True )
plt.show()

#plotting y(-n)
hn1 = np.array(y)
hn2 = np.array( range (-ystart,( len (y)-ystart)))
plt.figure( 2 )
plt.stem(hn2,hn1, use_line_collection = True )
plt.title( 'y(-n)' )
plt.grid( True )
plt.show()

# plotting RXH
xaxis = np.array(Rxh)
yaxis = np.array( range (-Rxh_start,( len (Rxh)-Rxh_start)))
plt.figure( 3 )
plt.stem(yaxis,xaxis, use_line_collection = True )
plt.title( 'RXH' )
plt.grid( True )
```

```
plt.show()
print ( "********************************************** \n " )
print ( "y(n) :" ,y)
print ( "y(n) origin:" , ystart)
print ( "x(n) :" ,x)
print ( "x(n) origin:" , xstart)

x.reverse()
xstart = len (x) - xstart - 1
y = temp_y
ystart = len (y) - ystart - 1
table = [[ 'y(n) \\ x(-n)' ,]+x,]

Rhx = [ 0 ]*( len (x) + len (y) - 1 )
Rhx_start = max (xstart,ystart)

for j,y_i in enumerate (y):
    row,s = [y_i,], 0
    for i,x_i in enumerate (x):
        Rhx[i+j] += x_i*y_i
        row.append(x_i*y_i)
    table.append(row)
print ( "x(-n) :" ,x)
print ( "x(-n) origin:" , xstart)
print ( " \n Table:" )
print (tabulate(table, headers = "firstrow" , tablefmt = "pretty" ))
print ( " \n Ryx(n) :" , Rhx)
print ( "Ryx(n) origin:" , Rhx_start)
print ( "********************************************** \n " )

# plotting yn
xn1 = np.array(y)
xn2 = np.array( range (ystart, len (y)+ystart))
plt.figure( 4 )
plt.stem(xn2,xn1, use_line_collection = True )
plt.title( 'y(n)' )
plt.grid( True )
plt.show()

#plotting x(-n)
hn1 = np.array(x)
hn2 = np.array( range (-xstart,( len (x)-xstart)))
plt.figure( 5 )
plt.stem(hn2,hn1, use_line_collection = True )
plt.title( 'x(-n)' )
plt.grid( True )
plt.show()

# plotting Rhx
xaxis = np.array(Rhx)
yaxis = np.array( range (-Rhx_start,( len (Rhx)-Rhx_start)))
plt.figure( 6 )
plt.stem(yaxis,xaxis, use_line_collection = True )
plt.title( 'RHX' )
plt.grid( True )
plt.show()

y = temp_y
ystart = len (y) - ystart - 1
x.reverse()
```

```
xstart = len (x) - xstart - 1
table = [[ 'x(n) \\ x(-n)' ,]+x,]

Rxx = [ 0 ]*( len (x) + len (y) - 1 )
Rxx_start = max (xstart,ystart)

for i,x_i in enumerate (x):
   row,s = [x_i,], 0
   for j,y_i in enumerate (y):
      Rxx[i+j] += x_i*y_i
      row.append(x_i*y_i)
   table.append(row)

print ( "y(-n) :" ,y)
print ( "y(-n) origin:" , ystart)
print ( " \n Table:" )
print (tabulate(table, headers = "firstrow" , tablefmt = "pretty" ))
print ( " \n Rxx(n) :" , Rxh)
print ( "Rxx(n) origin:" , Rxh_start)

# plotting xn
xn1 = np.array(x)
xn2 = np.array( range (xstart, len (x)+xstart))
plt.figure( 7 )
plt.stem(xn2,xn1, use_line_collection = True )
plt.title( 'x(n)' )
plt.grid( True )
plt.show()

#plotting x(-n)
hn1 = np.array(x)
hn2 = np.array( range (-xstart,( len (x)-xstart)))
plt.figure( 8 )
plt.stem(hn2,hn1, use_line_collection = True )
plt.title( 'x(-n)' )
plt.grid( True )
plt.show()

# plotting RXX
xaxis = np.array(Rxx)
yaxis = np.array( range (-Rxx_start,( len (Rxx)-Rxx_start)))
plt.figure( 9 )
plt.stem(yaxis,xaxis, use_line_collection = True )
plt.title( 'RXX' )
plt.grid( True )
plt.show()
print ( "************************************************** \n " )
print ( "y(n) :" ,y)
print ( "y(n) origin:" , ystart)
print ( "x(n) :" ,x)
print ( "x(n) origin:" , xstart)

temp_y = y[:]
y.reverse()
ystart = len (y) - ystart - 1
table = [[ 'y(n) \\ y(-n)' ,]+y,]

Rhh = [ 0 ]*( len (x) + len (y) - 1 )
Rhh_start = max (xstart,ystart)
```

```
for j,y_i in enumerate (y):
    row,s = [y_i,], 0
    for i,x_i in enumerate (x):
        Rhh[i+j] += x_i*y_i
        row.append(x_i*y_i)
    table.append(row)
print ( "x(-n) :" ,x)
print ( "x(-n) origin:" , xstart)
print ( " \n Table:" )
print (tabulate(table, headers = "firstrow" , tablefmt = "pretty" ))
print ( " \n Rhh(n) :" , Rhh)
print ( "Rhh(n) origin:" , Rhh_start)
print ( "************************************************** \n " )

# plotting yn
xn1 = np.array(y)
xn2 = np.array( range (ystart, len (y)+ystart))
plt.figure( 10 )
plt.stem(xn2,xn1, use_line_collection = True )
plt.title( 'y(n)' )
plt.grid( True )
plt.show()
#plotting y(-n)
hn1 = np.array(y)
hn2 = np.array( range (-ystart,( len (y)-ystart)))
plt.figure( 11 )
plt.stem(hn2,hn1, use_line_collection = True )
plt.title( 'y(-n)' )
plt.grid( True )
plt.show()


# plotting Rhh
xaxis = np.array(Rhh)
yaxis = np.array( range (-Rhh_start,( len (Rhh)-Rhh_start)))
plt.figure( 12 )
plt.stem(yaxis,xaxis, use_line_collection = True )
plt.title( 'RHH' )
plt.grid( True )
plt.show()
```

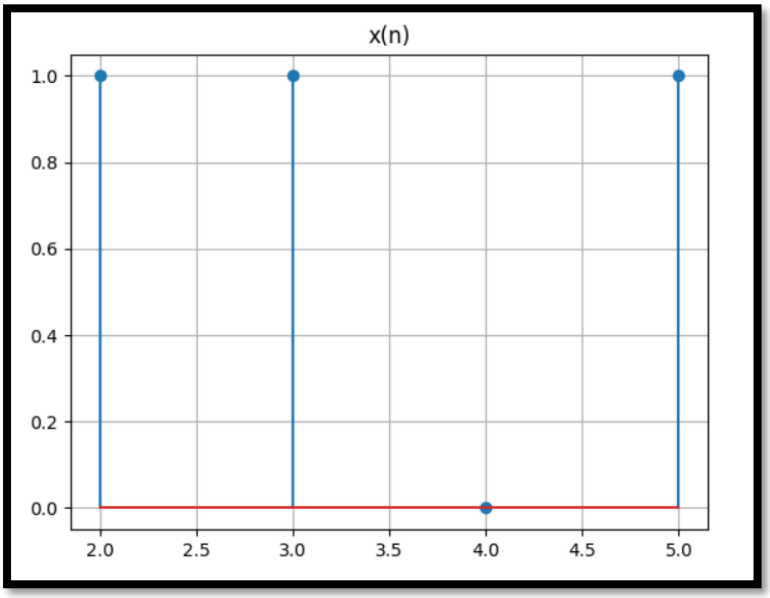
Output:

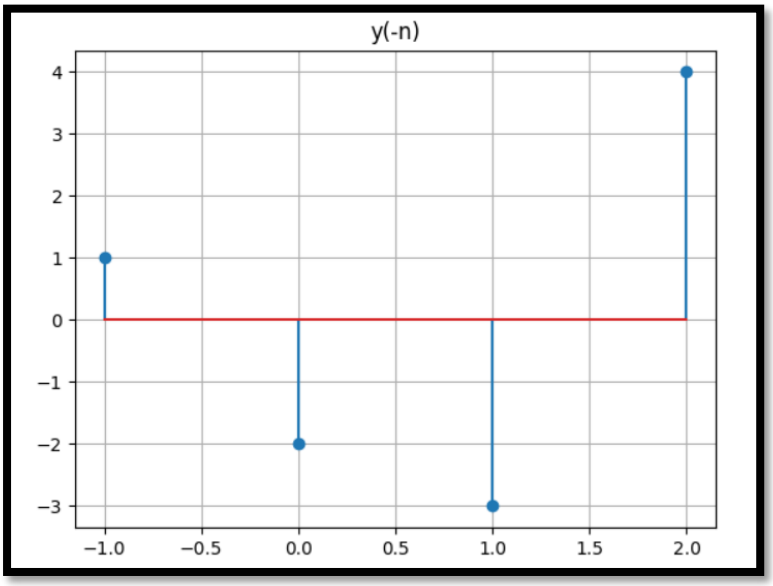
```
C:\Users\Vishal\AppData\Local\Programs\Python\Python38\python.exe C:/Python/Cross_Co.py
Enter values in x(n): 1 1 0 1
Enter x start:2
Enter values in y(n):4 -3 -2 1
Enter y start:2
y(-n) : [1, -2, -3, 4]
y(-n) origin: 1

 Table:
+--------------+---+----+----+---+
| x(n) \ y(-n) | 1 | -2 | -3 | 4 |
+--------------+---+----+----+---+
|      1       | 1 | -2 | -3 | 4 |
|      1       | 1 | -2 | -3 | 4 |
|      0       | 0 | 0  | 0  | 0 |
|      1       | 1 | -2 | -3 | 4 |
+--------------+---+----+----+---+

 Rxy(n) : [1, -1, -5, 2, 2, -3, 4]
Rxy(n) origin: 2
**********************************************
```

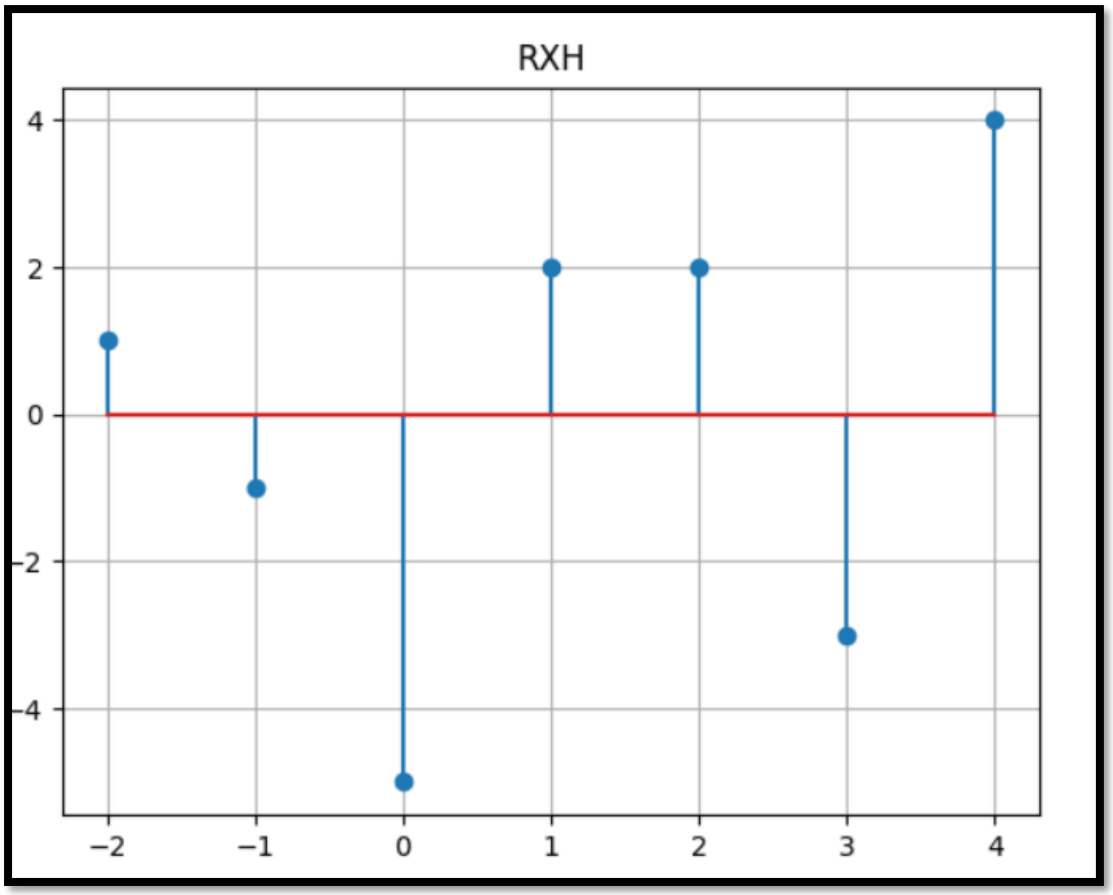
```
y(n) : [1, -2, -3, 4]
y(n) origin: 1
x(n) : [1, 1, 0, 1]
x(n) origin: 2
x(-n) : [1, 0, 1, 1]
x(-n) origin: 1

 Table:
+--------------+----+---+----+----+
| y(n) \ x(-n) | 1  | 0 | 1  | 1  |
+--------------+----+---+----+----+
|      4       | 4  | 0 | 4  | 4  |
|      -3      | -3 | 0 | -3 | -3 |
|      -2      | -2 | 0 | -2 | -2 |
|      1       | 1  | 0 | 1  | 1  |
+--------------+----+---+----+----+

 Ryx(n) : [4, -3, 2, 2, -5, -1, 1]
Ryx(n) origin: 2
**********************************************
```

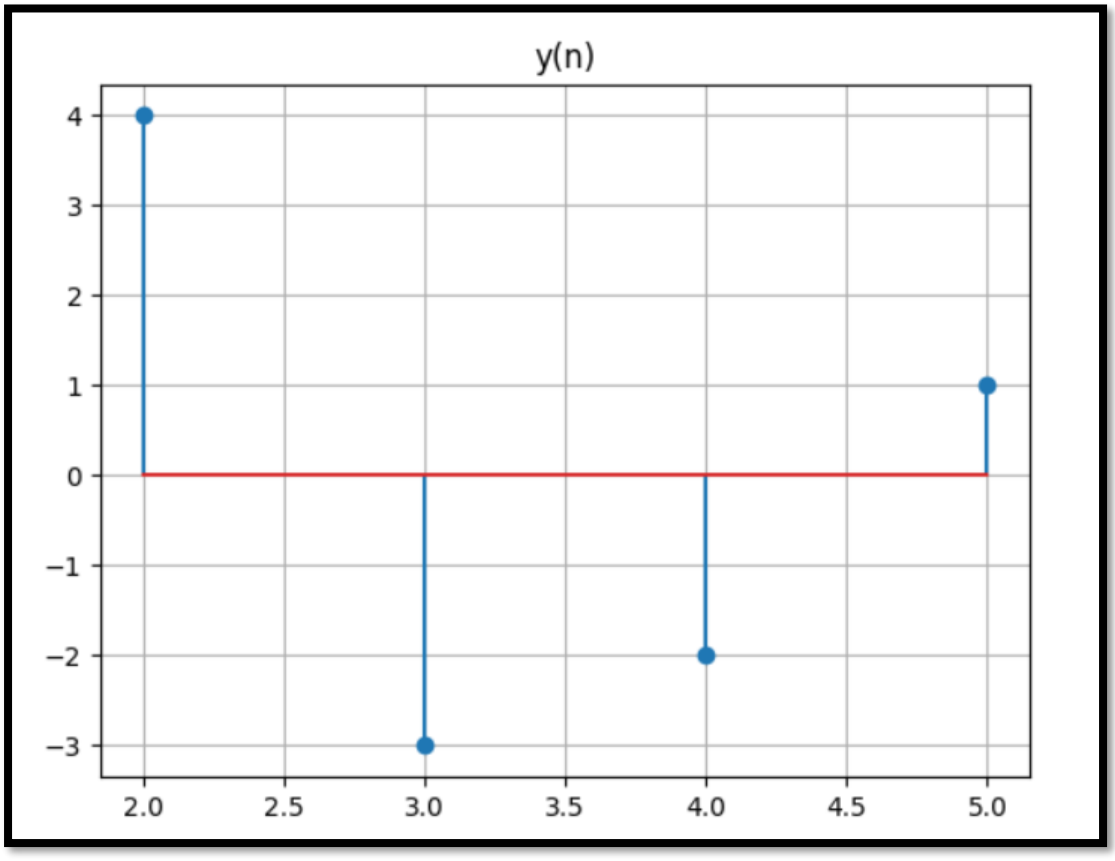
```
y(-n) : [4, -3, -2, 1]
y(-n) origin: 1

 Table:
+--------------+---+----+----+---+
| x(n) \ x(-n) | 1 | 1  | 0  | 1 |
+--------------+---+----+----+---+
|      1       | 4 | -3 | -2 | 1 |
|      1       | 4 | -3 | -2 | 1 |
|      0       | 0 | 0  | 0  | 0 |
|      1       | 4 | -3 | -2 | 1 |
+--------------+---+----+----+---+

 Rxx(n) : [1, -1, -5, 2, 2, -3, 4]
Rxx(n) origin: 2
************************************************
```

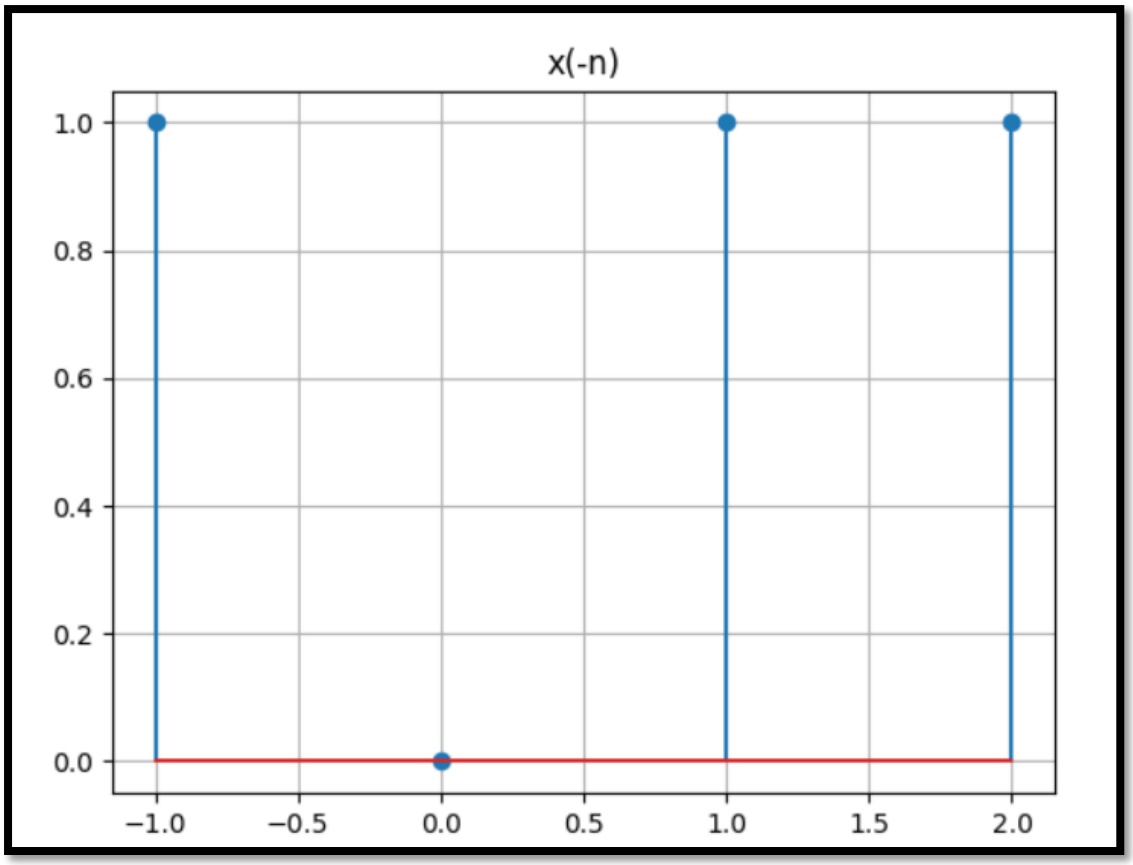
```
y(n) : [4, -3, -2, 1]
y(n) origin: 1
x(n) : [1, 1, 0, 1]
x(n) origin: 2
x(-n) : [1, 1, 0, 1]
x(-n) origin: 2

 Table:
+--------------+----+----+----+----+
| y(n) \ y(-n) | 1  | -2 | -3 | 4  |
+--------------+----+----+----+----+
|      1       | 1  | 1  | 0  | 1  |
|      -2      | -2 | -2 | 0  | -2 |
|      -3      | -3 | -3 | 0  | -3 |
|      4       | 4  | 4  | 0  | 4  |
+--------------+----+----+----+----+

 Rhh(n) : [1, -1, -5, 2, 2, -3, 4]
Rhh(n) origin: 2
```

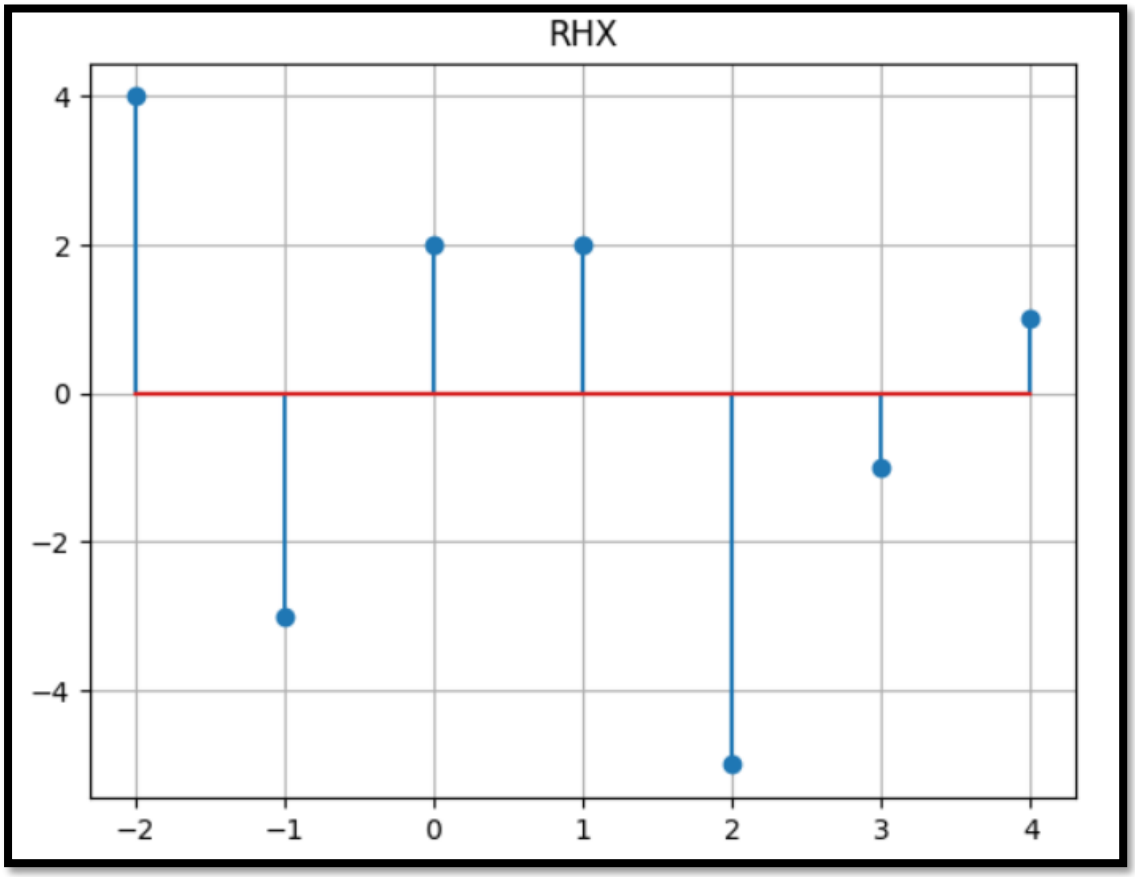


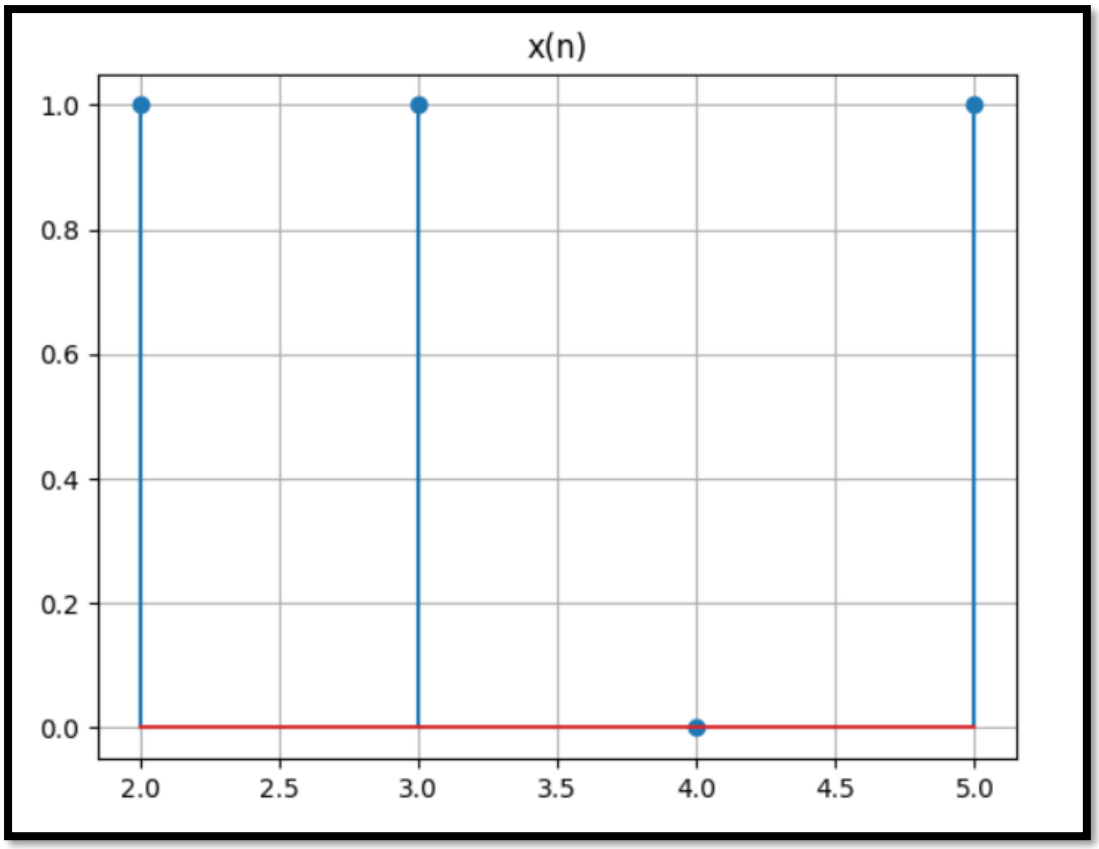


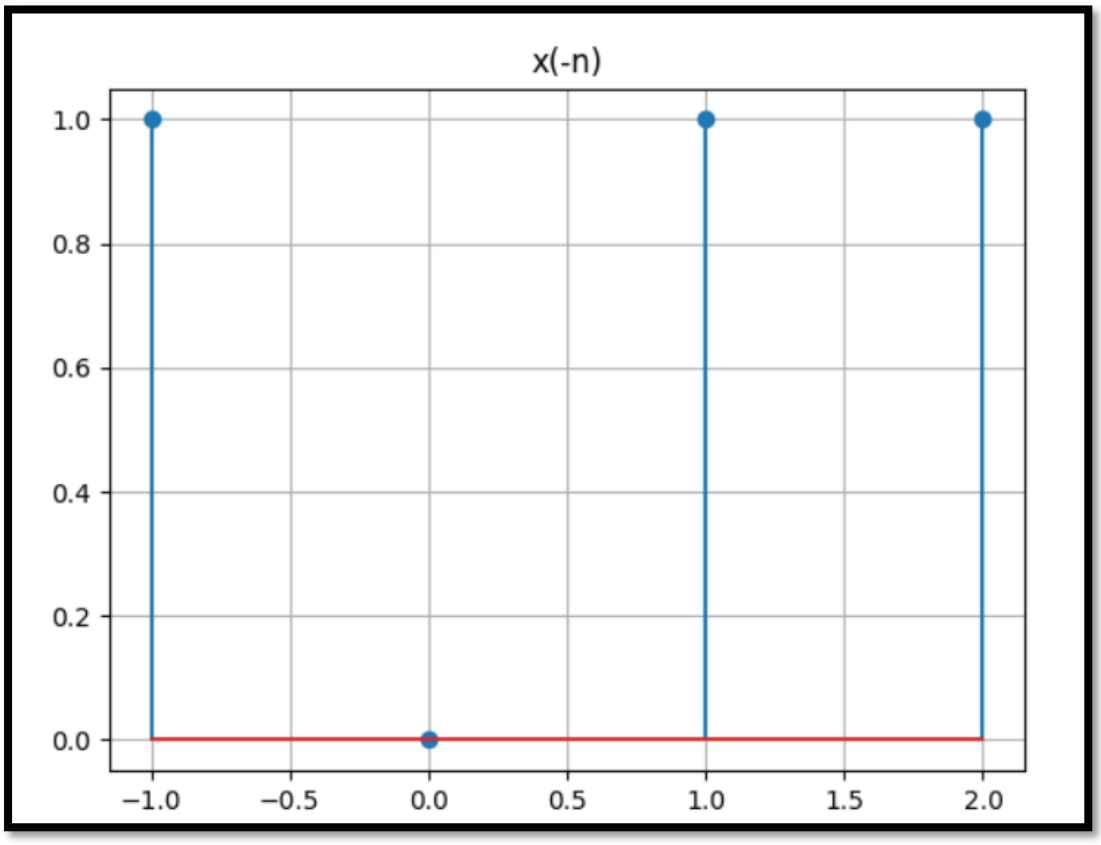

RXH
4
2
0
−2
−4
−2
−1
0
1
2
3
4

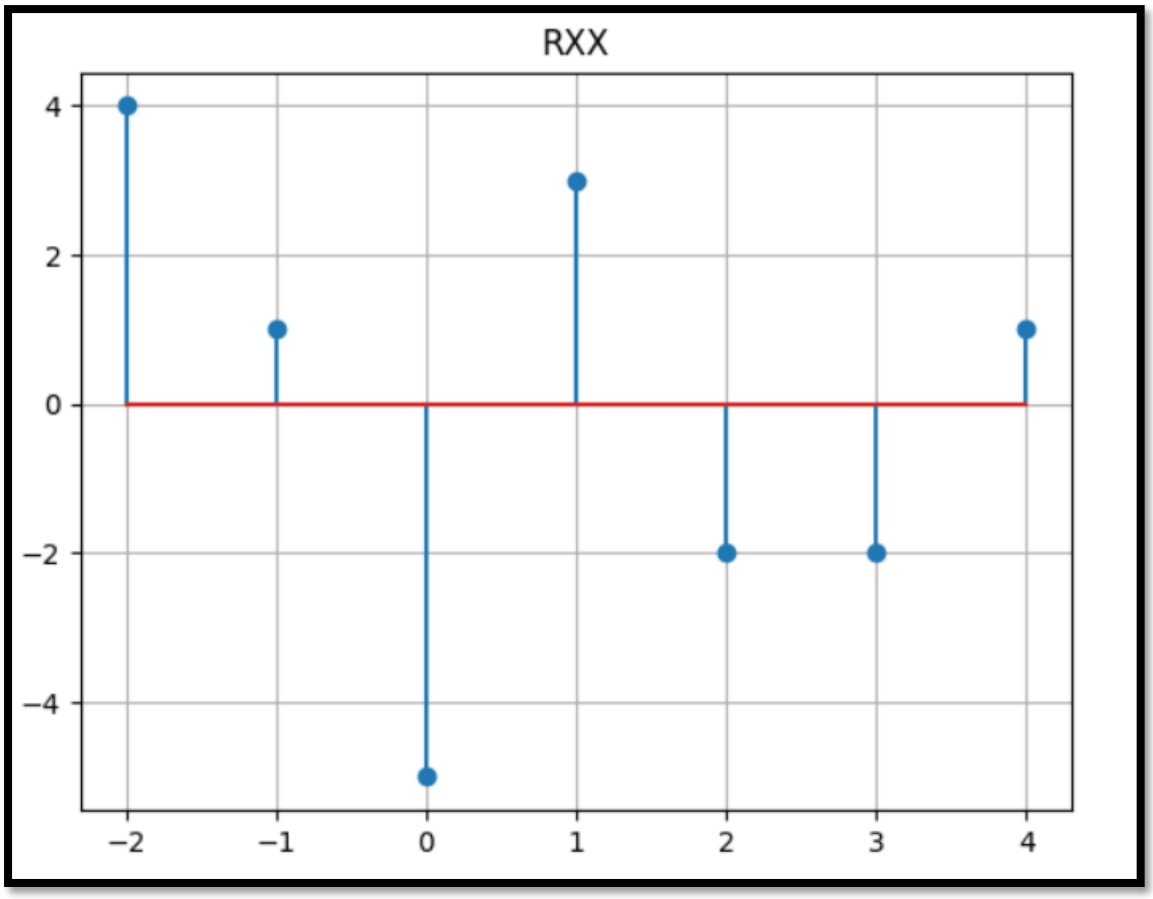

y(n)
4
3
2
1
0
−1
−2
−3
2.0
2.5
3.0
3.5
4.0
4.5
5.0

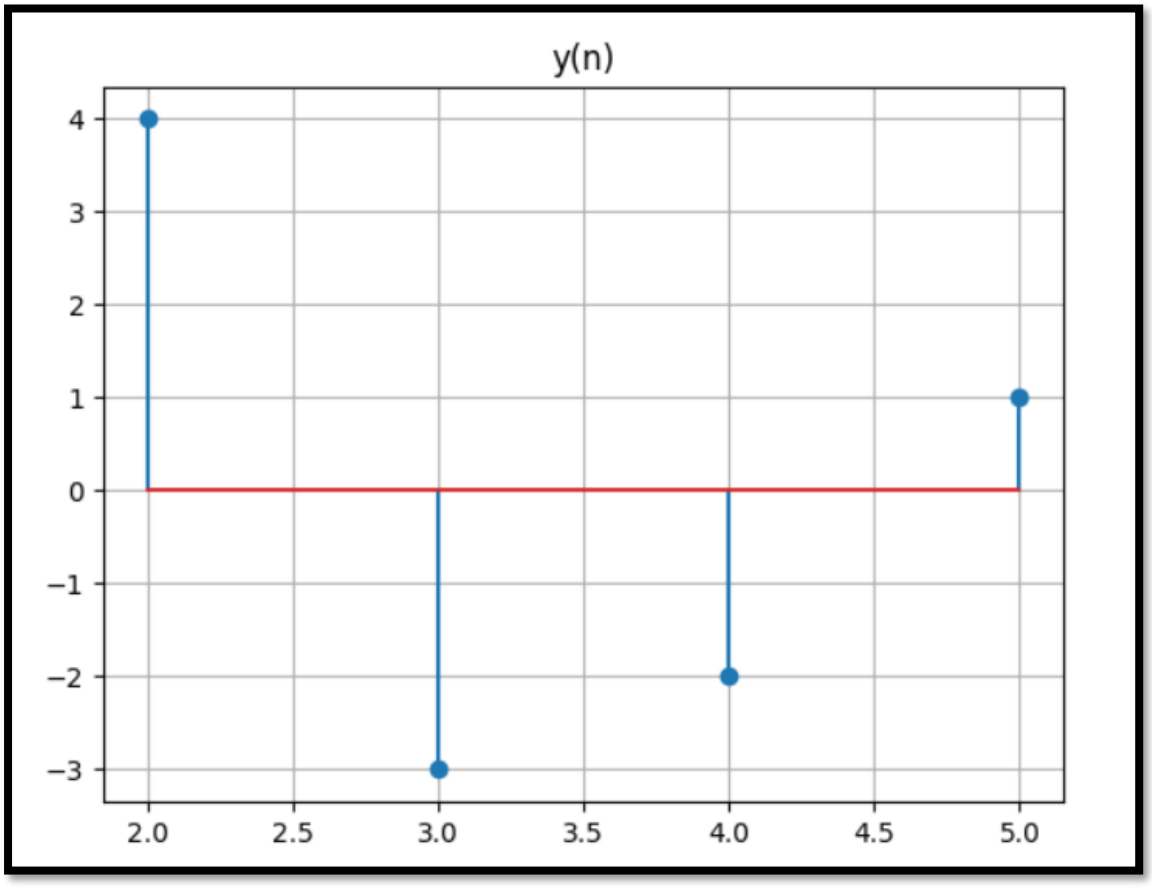

x(-n)
1.0
0.8
0.6
0.4
0.2
0.0
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

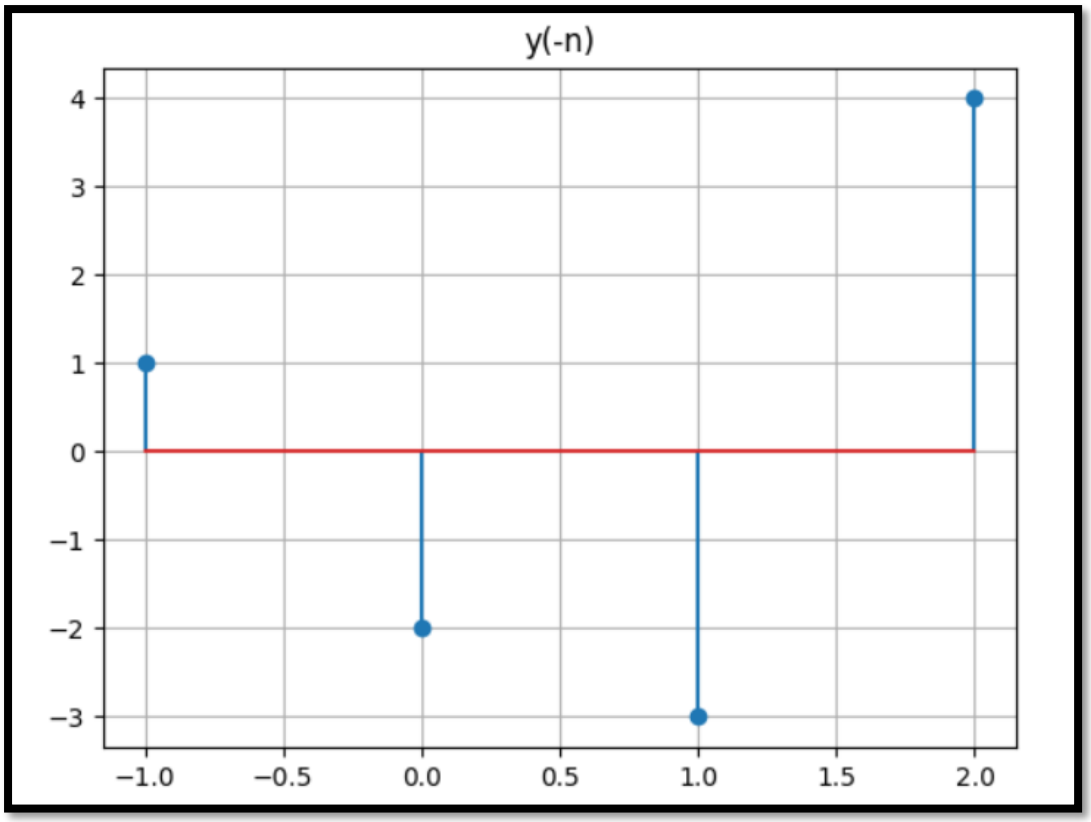

RHX
4
2
0
−2
−4
−2
−1
0
1
2
3
4

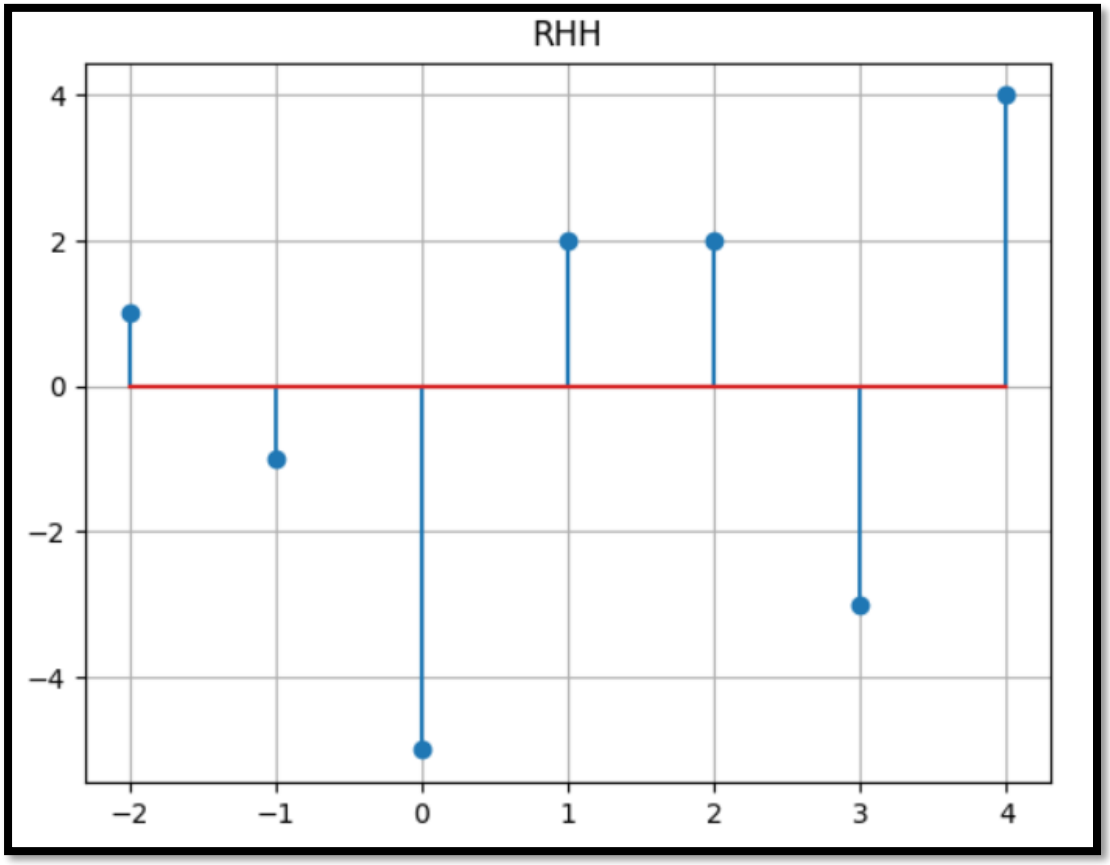

x(n)
1.0
0.8
0.6
0.4
0.2
0.0
2.0
2.5
3.0
3.5
4.0
4.5
5.0


x(-n)
1.0
0.8
0.6
0.4
0.2
0.0
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0


RXX
4
2
0
−2
−4
−2
−1
0
1
2
3
4


y(n)
4
3
2
1
0
−1
−2
−3
2.0
2.5
3.0
3.5
4.0
4.5
5.0


y(-n)
4
3
2
1
0
−1
−2
−3
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

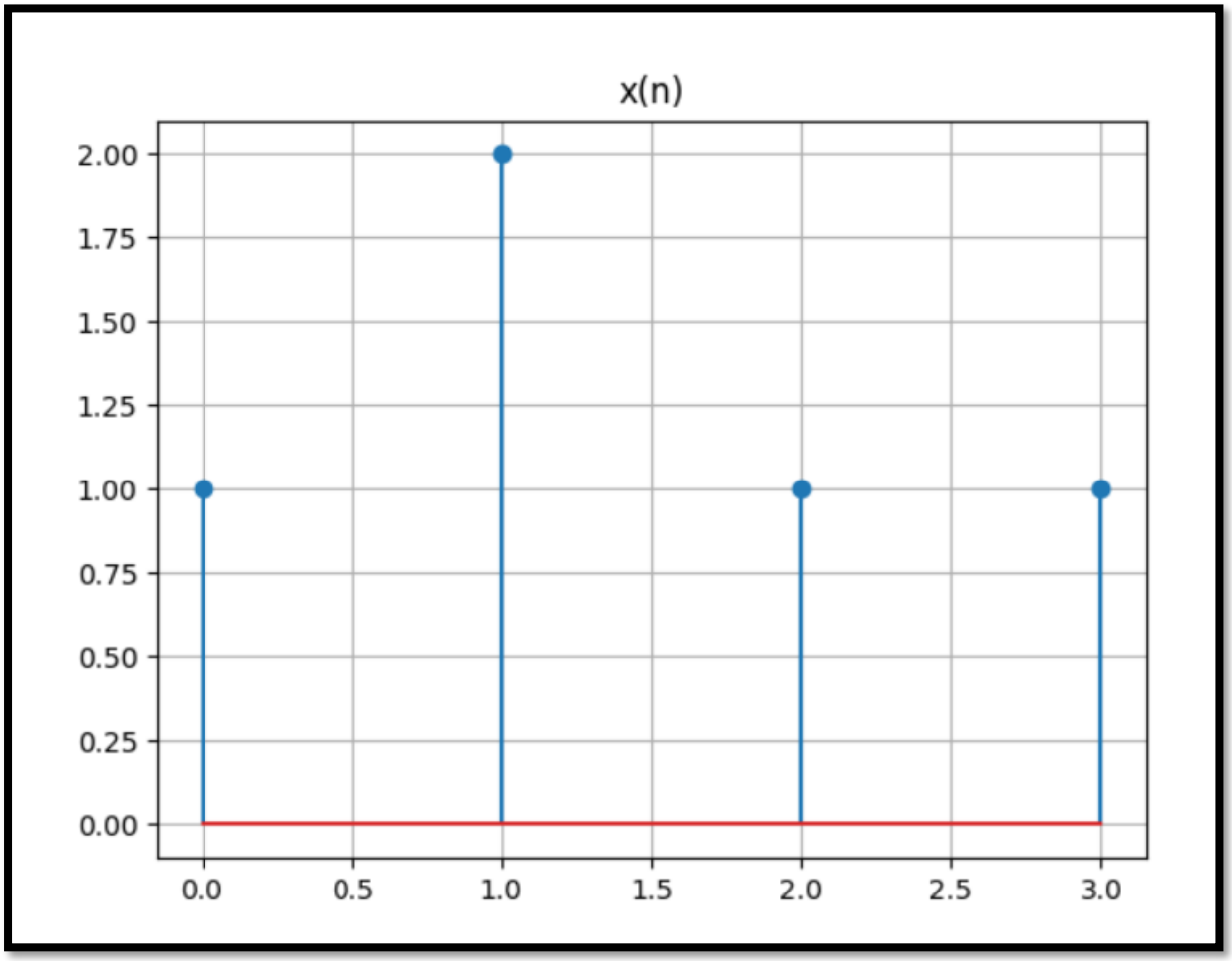

RHH
4
2
0
−2
−4
−2
−1
0
1
2
3
4

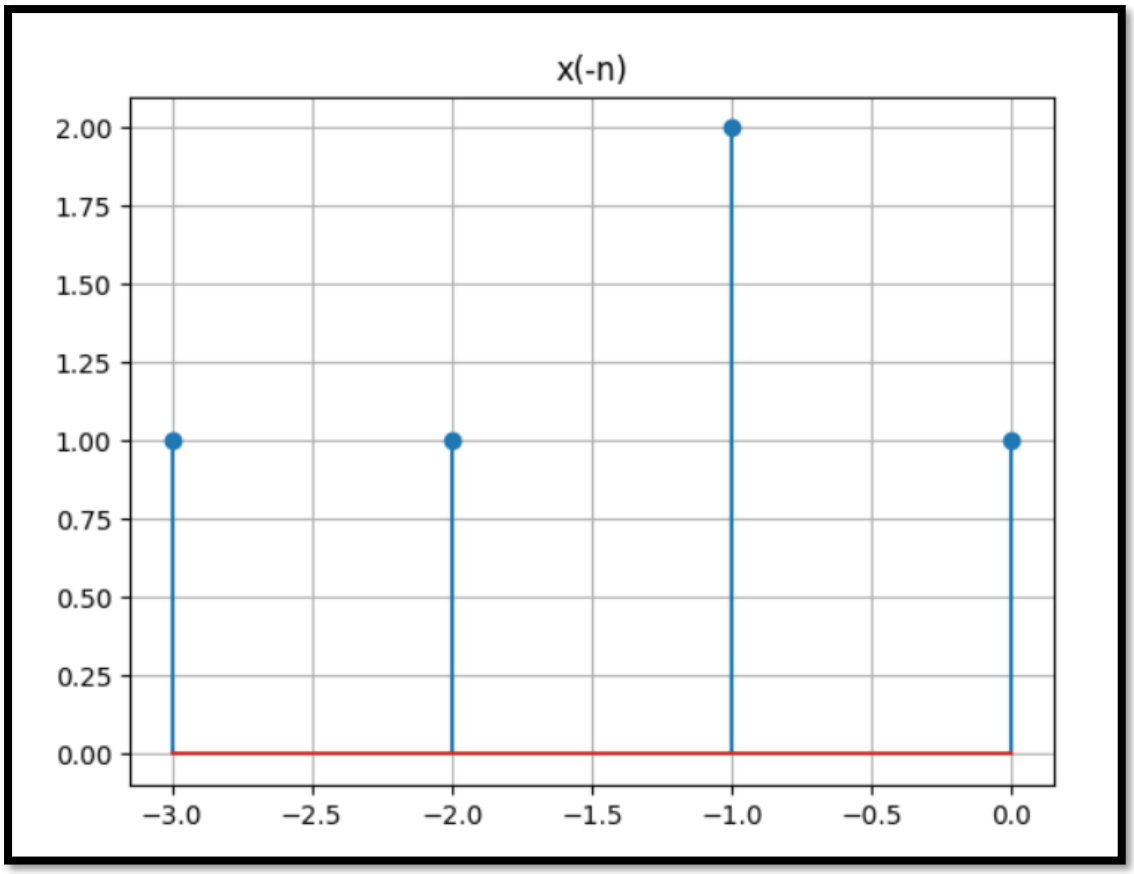
Auto Correlation:

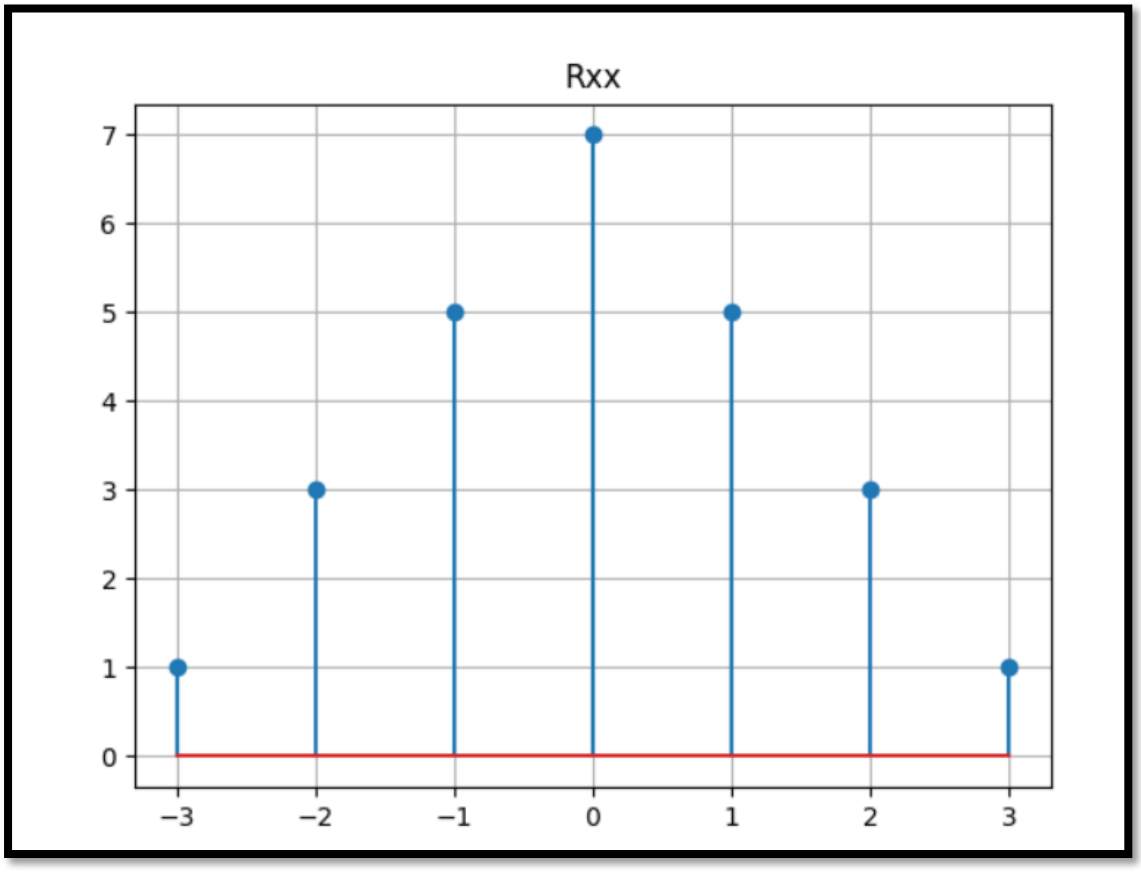
```
import numpy as np
from tabulate import tabulate
import matplotlib.pyplot as plt

# taking inputs
x = list ( map ( int , input ( "Enter values in x(n): " ).split()))
xstart = int ( input ( "Enter values in x start:" ))

temp_x = list ( reversed (x))
temp_x_start = len (x) - xstart - 1

print ( "x(n) :" ,x)
print ( "x(n) origin:" , xstart)
print ( "x(-n) :" ,temp_x)
print ( "x(-n) origin:" , temp_x_start)
table = [[ 'x(n) \\ x(-n)' ,]+temp_x,]
R_xx = [ 0 ]*( len (x) + len (temp_x) - 1 )
r_xxstart = max (xstart,temp_x_start)
for i,x_i in enumerate (x):
    row,s = [x_i,], 0
    for j,y_i in enumerate (temp_x):
        R_xx[i+j] += x_i*y_i
        row.append(x_i*y_i)
    table.append(row)
print ( " \n Table:" )
print (tabulate(table, headers = "firstrow" , tablefmt = "pretty" ))
print ( " \n Rxx(n) :" , R_xx)
print ( "Rxx(n) origin:" , r_xxstart)

# plotting xn
xn1 = np.array(x)
xn2 = np.array( range (xstart, len (x)+xstart))
plt.figure( 1 )
plt.stem(xn2,xn1, use_line_collection = True )
plt.title( 'x(n)' )
plt.grid( True )
plt.show()

#plotting x(-n)
hn1 = np.array(temp_x)
hn2 = np.array( range (-temp_x_start,( len (temp_x)-temp_x_start)))
plt.figure( 2 )
plt.stem(hn2,hn1, use_line_collection = True )
plt.title( 'x(-n)' )
plt.grid( True )
plt.show()

# plotting Rxx
xaxis = np.array(R_xx)
yaxis = np.array( range (-r_xxstart,( len (R_xx)-r_xxstart)))
plt.figure( 3 )
plt.stem(yaxis,xaxis, use_line_collection = True )
plt.title( 'Rxx' )
plt.grid( True )
plt.show()
print ( "********************************************** \n " )
```

Output:

```
C:\Users\Vishal\AppData\Local\Programs\Python\Python38\python.exe C:/Python/Auto_co.py
Enter values in x(n): 1 2 1 1
Enter values in x start:0
x(n) : [1, 2, 1, 1]
x(n) origin: 0
x(-n) : [1, 1, 2, 1]
x(-n) origin: 3

 Table:
+--------------+---+---+---+---+
| x(n) \ x(-n) | 1 | 1 | 2 | 1 |
+--------------+---+---+---+---+
|      1       | 1 | 1 | 2 | 1 |
|      2       | 2 | 2 | 4 | 2 |
|      1       | 1 | 1 | 2 | 1 |
|      1       | 1 | 1 | 2 | 1 |
+--------------+---+---+---+---+

 Rxx(n) : [1, 3, 5, 7, 5, 3, 1]
Rxx(n) origin: 3
```

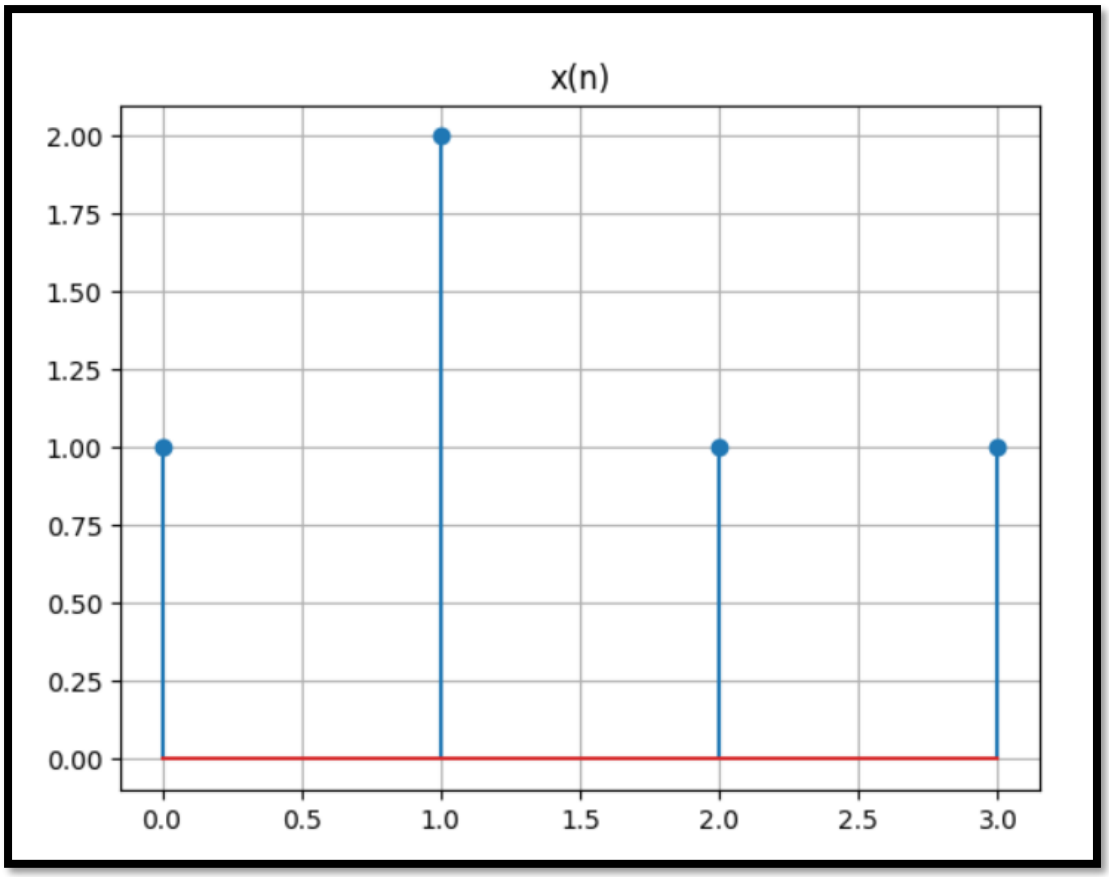


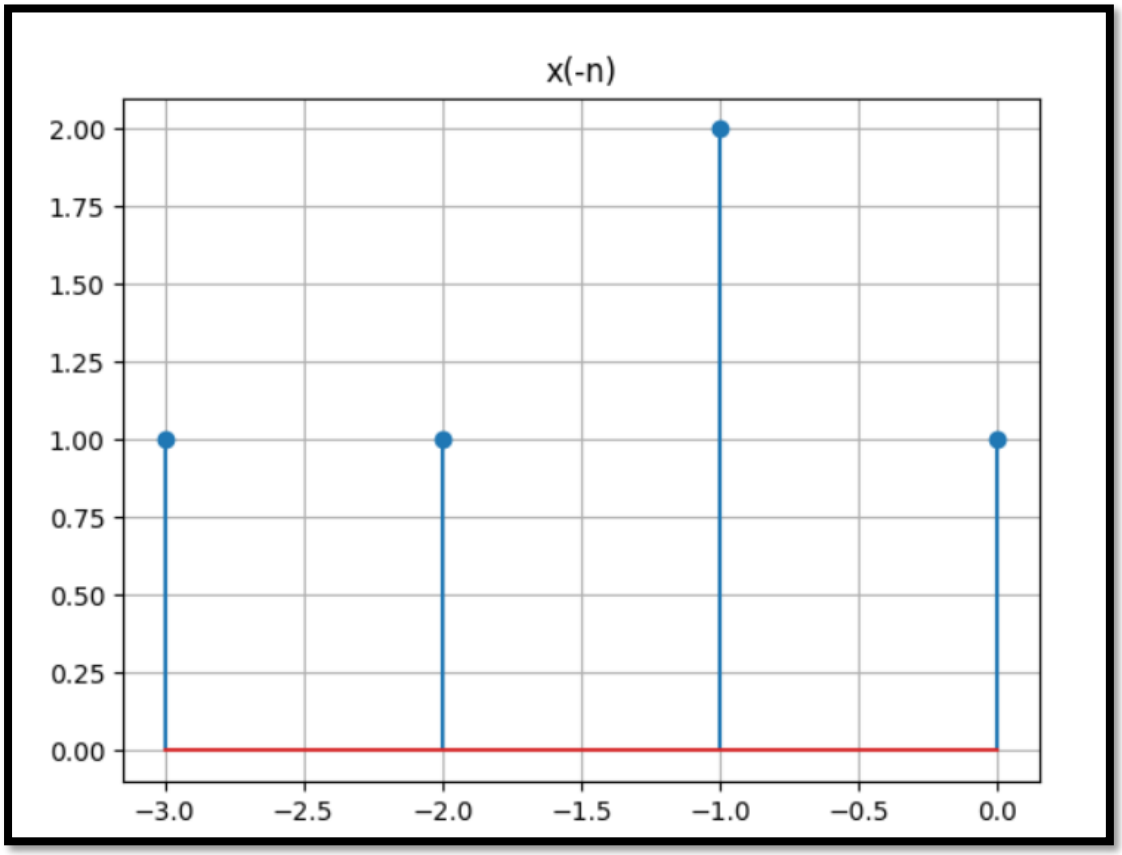

x(-n)
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−3.0
−2.5
−2.0
−1.5
−1.0
−0.5
0.0

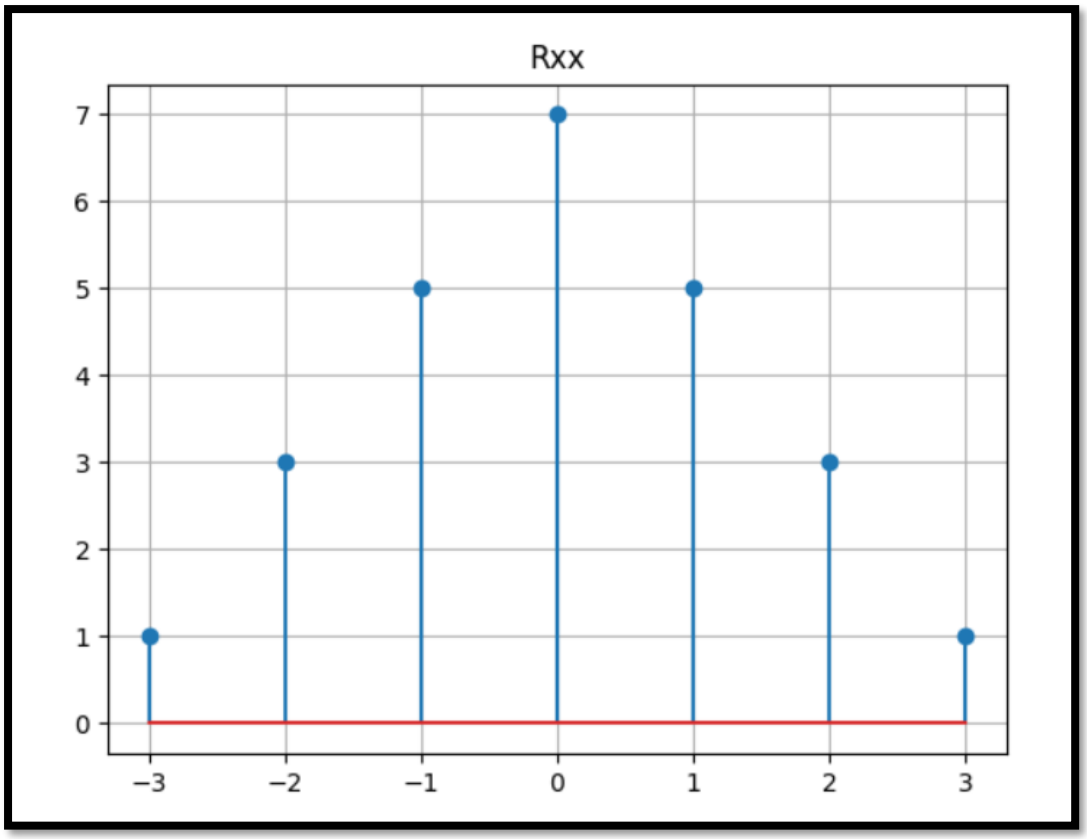

Rxx
7
6
5
4
3
2
1
0
−3
−2
−1
0
1
2
3

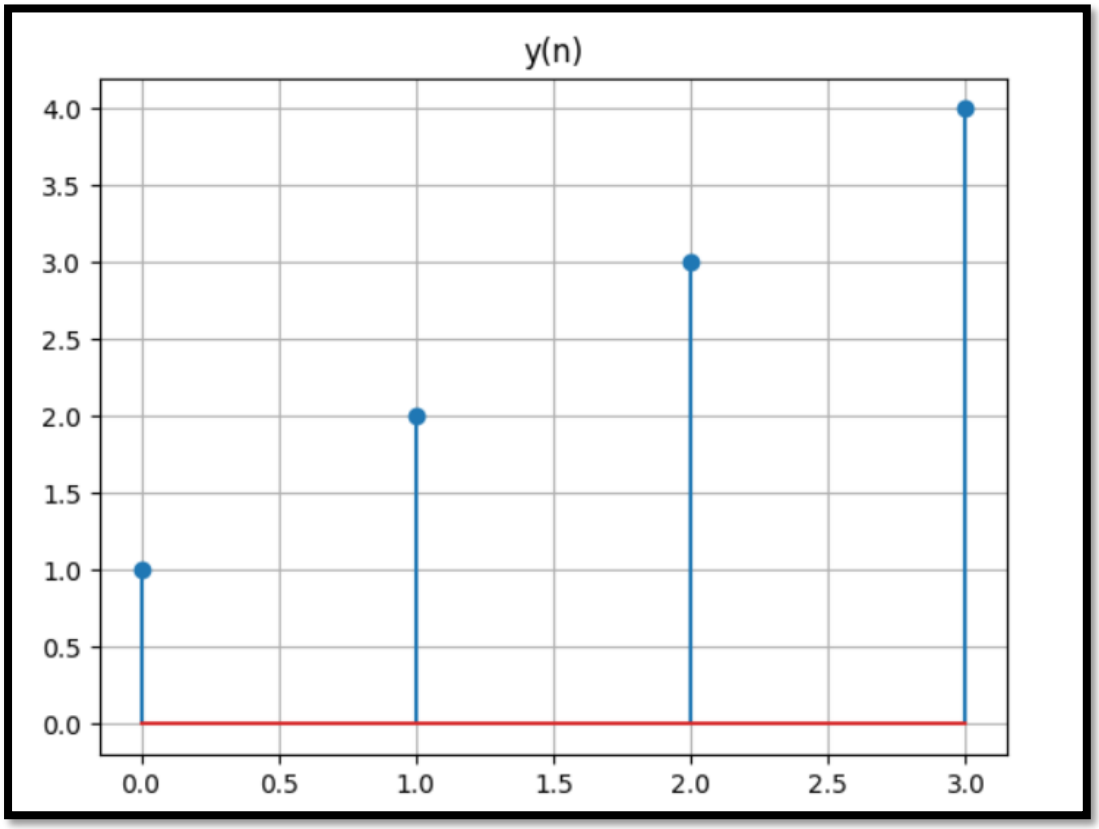
```
C:\Users\Vishal\AppData\Local\Programs\Python\Python38\python.exe C:/Python/Auto_co.py
Enter values in x(n): 1 2 1 1
Enter value at x start:0
Enter values in y(n):1 2 3 4
Enter value at y start:0
x(n) : [1, 2, 1, 1]
x(n) origin: 0
x(-n) : [1, 1, 2, 1]
x(-n) origin: 3

 Table:
+--------------+---+---+---+---+
| x(n) \ x(-n) | 1 | 1 | 2 | 1 |
+--------------+---+---+---+---+
|      1       | 1 | 1 | 2 | 1 |
|      2       | 2 | 2 | 4 | 2 |
|      1       | 1 | 1 | 2 | 1 |
|      1       | 1 | 1 | 2 | 1 |
+--------------+---+---+---+---+

 Rxx(n) : [1, 3, 5, 7, 5, 3, 1]
Rxx(n) origin: 3
************************************************
```

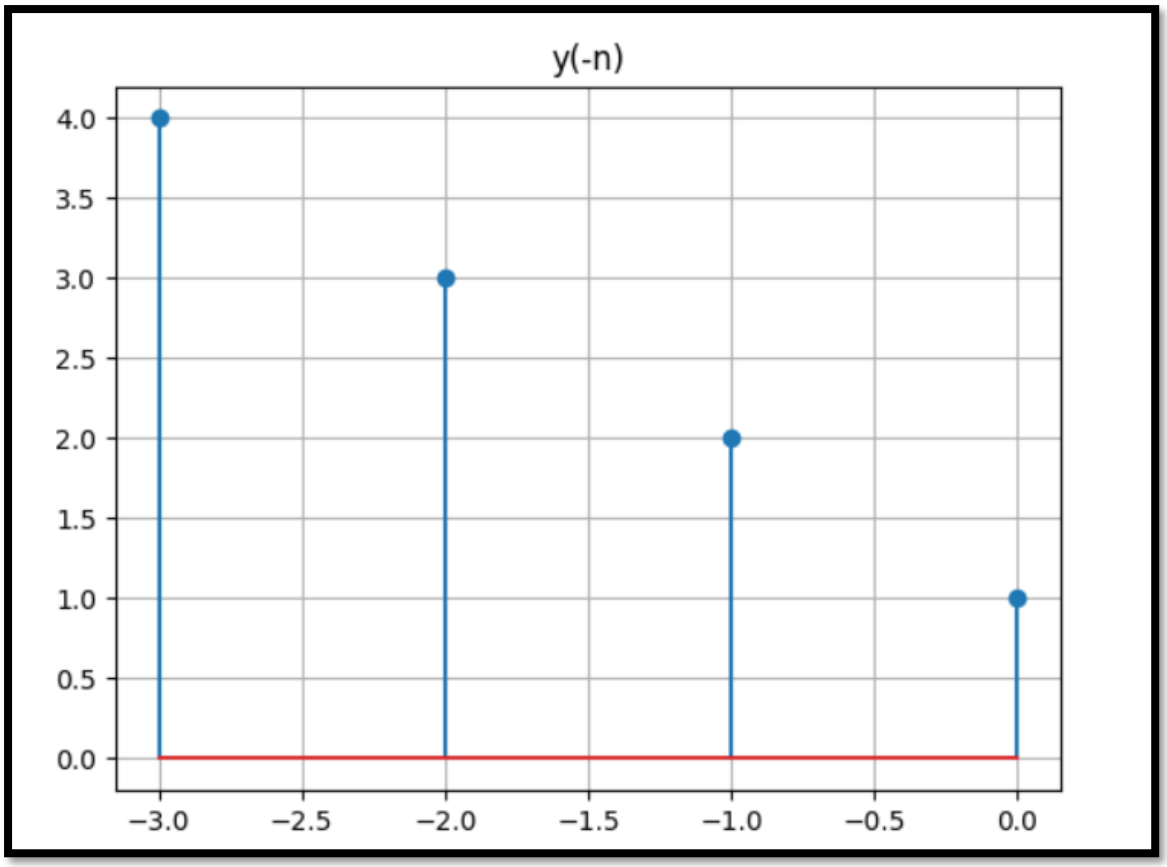
```
y(n) : [1, 2, 3, 4]
y(n) origin: 0
y(-n) : [4, 3, 2, 1]
y(-n) origin: 3

 Table:
+--------------+----+----+---+---+
| y(n) \ y(-n) | 4  | 3  | 2 | 1 |
+--------------+----+----+---+---+
|      1       | 4  | 3  | 2 | 1 |
|      2       | 8  | 6  | 4 | 2 |
|      3       | 12 | 9  | 6 | 3 |
|      4       | 16 | 12 | 8 | 4 |
+--------------+----+----+---+---+

 Ryy(n) : [4, 11, 20, 30, 20, 11, 4]
Ryy(n) origin: 3
************************************************


Process finished with exit code 0
```

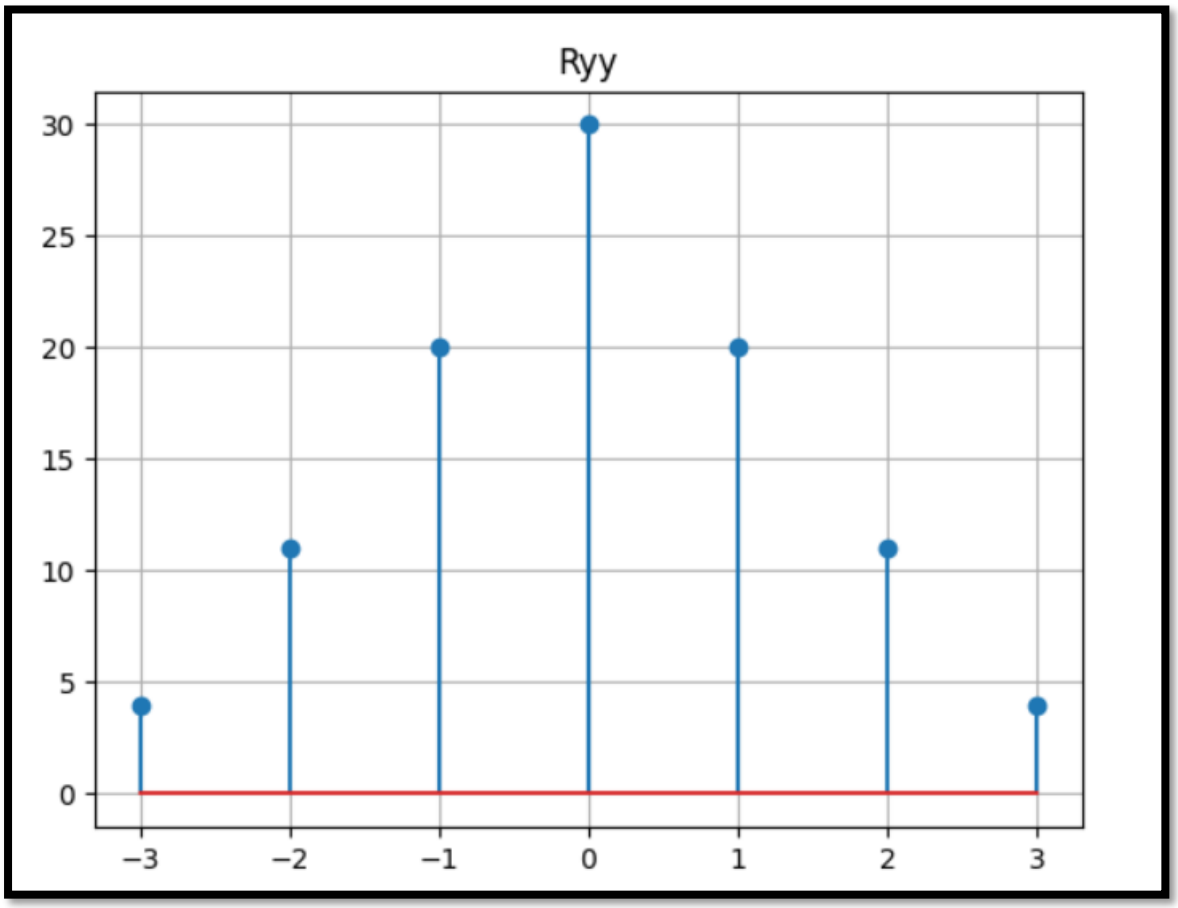

x(n)
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0
1.5
2.0
2.5
3.0


x(-n)
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−3.0
−2.5
−2.0
−1.5
−1.0
−0.5
0.0


Rxx
7
6
5
4
3
2
1
0
−3
−2
−1
0
1
2
3


y(n)
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0


y(-n)
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−3.0
−2.5
−2.0
−1.5
−1.0
−0.5
0.0


Ryy
30
25
20
15
10
5
0
−3
−2
−1
0
1
2
3

Conclusion:

1) I learnt how to perform auto correlation and cross correlation.

2)Correlation describes the mutual relationship which exists between two or more things. The same definition holds good even in the case of signals. That is, correlation between signals indicates the measure up to which the given signal resembles another signal.

Reference:

1) **https://www.tutorialspoint.com/signals_and_systems/convolution_and_correlation.htm**