

① NAME - SANTOSH - [CB.EN.UACLE20053]

DEPARTMENT - COMPUTER AND COMMUNICATION ENGINEERING

LAB TITLE AND CODE : SIGNAL PROCESSING LAB IACLE281

EXPERIMENT NUMBER : 7

DATE : 02/10/2021

TIME SHIFTING PROPERTY OF DISCRETE-TIME FOURIER SERIES (DTFS)

* AIM :

Given an input sequence $x[n]$, verify the time-shifting property and evaluate Fourier coefficients for the resulting sequence $y[n]$ using DTFS.

* SOFTWARE REQUIRED :

Spyder IDE (Anaconda3) - Python 3.8.8 (64-bit)

* THEORY :

① Discrete-time signals -

A discrete-time signal of fundamental period N can consist of frequency components $f = \frac{1}{N}, \frac{2}{N}, \dots, \frac{(N-1)}{N}$ besides $f=0$,

the DC component. Therefore, the Fourier series representation of the discrete-time periodic signal contains only N complex exponential basis functions.

② Fourier series for discrete-time periodic signals -

Given a periodic sequence $x[k]$ with period N , the Fourier series representation for $x[k]$ uses N harmonically related exponential functions -

$$e^{j2\pi kn/N}, k=0, 1, \dots, N-1$$

The Fourier series is expressed as -

$$x[k] = \sum_{n=0}^{N-1} c_n e^{j2\pi kn/N}$$

②

③ Fourier Coefficients -

The Fourier coefficients $\{c_n\}$ are given by -

$$c_n = \frac{1}{N} \sum_{k=0}^{N-1} x[k] e^{-j 2\pi k n / N}$$

④ Time shifting -

Let n_0 be any integer. If $x[n]$ is a discrete-time signal to period N , then $y[n] = x[n-n_0]$. Therefore, the Fourier series is -

$$x[n] \longleftrightarrow x[k]$$

which gives, $x[n-n_0] \xrightarrow{\text{FS}} e^{-j 2\pi k n_0 / N} x[k]$,
the time-shifted Fourier coefficients.

* GRAPH PLOTTING ALGORITHM :

The following steps are followed -

- ① Define the x-axis and corresponding y-axis values as lists.
- ② Plot them on canvas using `plot()` function.
- ③ Give a name to x-axis and y-axis using `xlabel()` and `ylabel()` functions.
- ④ Give a title to your plot using the `title()` function.
- ⑤ Finally, to view your plot, we use the `show()` function.

* THEORETICAL CALCULATION :

Given periodic sequence,

$$\underline{x[n]} = [\dots 9 18 27 36 45 \dots]$$

(Note : Boldface indicates $n=0$ index).

By keen observation, we find that fundamental period, N^5

$$\text{Angular frequency, } \omega_0 = \frac{2\pi}{N} = \frac{2\pi}{5}$$

$$x[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-jk \frac{2\pi}{5} n} = \frac{1}{5} \sum_{n=0}^4 x[n] e^{-jk \frac{2\pi}{5} n}$$

$$= \frac{1}{5} \left[x[0] e^0 + x[1] e^{-jk \frac{2\pi}{5}} + x[2] e^{-jk \frac{4\pi}{5}} + x[3] e^{-jk \frac{6\pi}{5}} + x[4] e^{-jk \frac{8\pi}{5}} \right]$$

$$= \frac{1}{5} \left[9 + 18 e^{-jk \frac{2\pi}{5}} + 27 e^{-jk \frac{4\pi}{5}} + 36 e^{-jk \frac{6\pi}{5}} + 45 e^{-jk \frac{8\pi}{5}} \right]$$

Taking 9 common,

$$x[k] = \frac{9}{5} \left[1 + 2e^{-jk \frac{2\pi}{5}} + 3e^{-jk \frac{4\pi}{5}} + 4e^{-jk \frac{6\pi}{5}} + 5e^{-jk \frac{8\pi}{5}} \right]$$

Substituting the values of k in $x[k]$, we find that -

$$x[0] = 27 + 0^\circ, \text{ for } k=0$$

$$x[1] = -4.5 + 6.2^\circ, \text{ for } k=1$$

$$x[2] = -4.5 + 1.5^\circ, \text{ for } k=2$$

$$x[3] = -4.5 - 1.5^\circ, \text{ for } k=3$$

$$x[4] = -4.5 - 6.2^\circ, \text{ for } k=4$$

Upon computing the magnitude spectrum, we find that -

$$|x[0]| = \sqrt{(27)^2 + (0)^2} = 27.0$$

$$|x[1]| = \sqrt{(-4.5)^2 + (6.2)^2} = 7.66$$

$$|x[2]| = \sqrt{(-4.5)^2 + (1.5)^2} = 4.73$$

$$|x[3]| = \sqrt{(-4.5)^2 + (-1.5)^2} = 4.73$$

$$|x[4]| = \sqrt{(-4.5)^2 + (-6.2)^2} = 7.66$$

Upon computing the phase spectrum, we find that -

$$\arg(x[n]) = \tan^{-1}\left(\frac{y}{x}\right) = -\frac{\theta}{2} = -\frac{x\pi}{180} \text{ radians}$$

Therefore,

$$\arg(x[0]) = 0 \text{ radians}$$

$$\arg(x[1]) = -0.94 \text{ radians}$$

$$\arg(x[2]) = -0.31 \text{ radians}$$

$$\arg(x[3]) = 0.31 \text{ radians}$$

$$\arg(x[4]) = 0.94 \text{ radians}$$

(4)

Given shifting factor, $n_0 = 2$.

Therefore, obtained resulting sequence,

$$y[n] = [\dots \underline{36} \ 45 \ 9 \ 18 \ 27 \ \dots]$$

(Note: Boldface indicates $n=0$ index.)

Again, fundamental period (N) = 5

$$\text{Angular frequency } (\omega_0) = \frac{2\pi}{N} = \frac{2\pi}{5}$$

Therefore, Fourier coefficients of $y[n]$ using time shifting property of DTFS are -

$$y[0] = e^{-j2\pi k n_0/N} \cdot x[0] = e^{-j2\pi(0)2/5} \cdot (27+0j) = (27+0j)$$

$$y[1] = e^{-j2\pi(1)2/5} \cdot x[1] = e^{-j2\pi(1)2/5} \cdot (-4.5+6.2j) = (7.28-2.37j)$$

$$y[2] = e^{-j2\pi(2)2/5} \cdot x[2] = e^{-j2\pi(2)2/5} \cdot (-4.5+1.5j) = (-2.78-3.83j)$$

$$y[3] = e^{-j2\pi(3)2/5} \cdot x[3] = e^{-j2\pi(3)2/5} \cdot (-4.5-1.5j) = (-2.78+3.83j)$$

$$y[4] = e^{-j2\pi(4)2/5} \cdot x[4] = e^{-j2\pi(4)2/5} \cdot (-4.5-6.2j) = (7.28+2.37j)$$

Upon computing the magnitude spectrum, we find that -

$$|y[0]| = \sqrt{(27)^2 + (0)^2} = 27.0$$

$$|y[1]| = \sqrt{(7.28)^2 + (-2.37)^2} = 7.66$$

$$|y[2]| = \sqrt{(-2.78)^2 + (-3.83)^2} = 4.73$$

$$|y[3]| = \sqrt{(-2.78)^2 + (3.83)^2} = 4.73$$

$$|y[4]| = \sqrt{(7.28)^2 + (2.37)^2} = 7.66$$

Upon computing the phase spectrum, we find that -

$$\angle(y[0]) = \tan^{-1}\left(\frac{0}{27}\right) = 0^\circ = -\times \frac{\pi}{180} \text{ radians}$$

$$\angle(y[1]) = -0.31 \text{ radians}$$

$$\angle(y[2]) = 0.94 \text{ radians}$$

$$\angle(y[3]) = -0.94 \text{ radians}$$

$$\angle(y[4]) = 0.31 \text{ radians}$$

⑧

* PROGRAM WITH COMMENTS :

```

1 # Import library source files
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 # Given Input Sequence a[n]
6 size_input = int(input("Enter the size of input a[n]: "))
7 user_defined_input = [0] * (size_input)
8 print("Enter the elements of the input a[n] one-by-one
as follows : -")
9 for sample in range(0, size_input, 1):
10     user_defined_input[sample] = input("Element " +
str(sample + 1) + ": ")
11 print("\n The entered input a[n] is : - " + str(user_defined_
input) + "\n")
12
13 # Null array Declaration for Input sequence a[n]
14 x = []
15 fourier_series_x = []
16 magnitude_spectrum_x = []
17 phase_spectrum_x = []
18
19 # Null Array Declaration for Resulting sequence y[n]
20 y = []
21 fourier_series_y = []
22 magnitude_spectrum_y = []
23 phase_spectrum_y = []
24
25 # Compute Fundamental Time Period for a[n]
26 for sample in range(1, size_input - 1):
27     x.append(user_defined_input[sample - 1])
28     if user_defined_input[0] == user_defined_input[sample]:
29         breakpoint = sample - 2 # How many times does the sequence
30         break

```

(6)

```

31
32 # Obtain frequency domain representation
33 boldface = int(input("Enter the element position that
34 indicates n=0 index : "))
35 index = int((boldface - 1) / len(x))
36
37 # sketch Input sequence x[n]
38 plt.xlabel('n')
39 plt.ylabel('x[n]')
40 plt.title("Input Sequence : ... {} ... ".format(x))
41 plt.stem(np.arange(0, index + len(x)), np.real(x)[index + 1:breakpoint], color='orange')
42 plt.stem(np.arange(index + 1, len(x)), np.real(x)[index + 1:breakpoint], color='red')
43 plt.show()
44
45 # Compute Fourier series for x[n]
46 def summation(k):
47     sum = 0
48     for n in range(len(x)):
49         exponential = np.exp(-2j * np.pi * k * n / len(x))
50         sum = sum + float(x[n]) * exponential
51     return sum
52
53 # Compute Fourier Coefficients for x[n]
54 for k in range(len(x)):
55     fourier_series_x.append(summation(k) / len(x))
56 print("The Fourier Coefficients for x[n] are as follows : ")
57 print("format(fourier_series_x))")
58
59 # sketch Fourier coefficients of x[n]
60 plt.xlabel('n')
61 plt.ylabel('x[n]')
62 plt.title("Fourier Coefficients of x[n]")
63 plt.stem(np.arange(0, len(fourier_series_x)), np.real(fourier_series_x))

```

①

```

61 plt.show()
62
63 # Compute Magnitude and phase Spectrum for x[n]
64 for sample in range (len (fourier_series - x)):
65     magnitude_spectrum - x.append (fourier_series - x[sample])
66     real ** 2 + fourier_series - x[sample].Imag ** 2) ** 0.5)
67     phase = np.arctan (fourier_series - x[sample].Imag /
68     fourier_series - x[sample].real)
69     phase_spectrum - x.append (phase)
70
71 # sketch magnitude Spectrum of x[n]
72 plt.xlabel ('k')
73 plt.ylabel ('|x[k]|')
74 plt.title ("Magnitude Spectrum of x[n]: ... {3...".format
75 (magnitude_spectrum - x))
76 plt.stem (np.arange (0, len (magnitude_spectrum - x)),
77 magnitude_spectrum - x)
78 plt.show ()
79 print ("The magnitude Spectrum of x[n] is as follows: {3".
80 format (magnitude_spectrum - x))
81
82 # sketch Phase Spectrum of x[n]
83 plt.xlabel ('k')
84 plt.ylabel ('Angle (in radians)')
85 plt.title ("Phase Spectrum of x[n]: ... {3...".format
86 (phase_spectrum - x))
87 plt.stem (np.arange (0, len (phase_spectrum - x)),
88 phase_spectrum - x)
89 plt.show ()
90 print ("The phase spectrum of x[n] is as follows: {3".format
91 (phase_spectrum - x))
92
93 intermediate = []
94
95 intermediate = []
96

```

(8)

```

87 # Copy Input sequence to Temporary sequence "intermediate"
88 for sample in range (size_input):
89     intermediate.append (user_defined_input [sample])
90
91 # Input Shifting Factor
92 shifting_factor = int (input ("Enter the shifting factor: "))
93
94 # shift Temporary sequence by "shifting-factor" of units
95 for i in range (shifting_factor):
96     intermediate.append (0) # Adds required zeros at the end
97     of list
98
99     for j in range (len (intermediate) - 1, 0, -1):
100         intermediate [j] = intermediate [j-1] # Shifts the zeros
101         at the beginning of list
102         intermediate [0] = 0 # Prepares for next iteration
103
104 # Place rest of elements in relevant position
105 temp = size_input - shifting_factor
106 for k in range (shifting_factor):
107     intermediate [k] = user_defined_input [temp]
108     temp = temp + 1
109
110 # Compute Fundamental Time Period for y[n]
111 for sample in range (1, size_input - 1):
112     y.append (intermediate [sample - 1])
113     if user_defined_input [0] == user_defined_input [sample]:
114         break
115
116 print ("The resulting sequence y[n] is :- " + str(y) + "\n")
117
118 # sketch resulting sequence y[n]
119 plt.xlabel ('n')
120 plt.ylabel ('y[n]')

```

⑨

```

118 plt.title ("Resulting sequence: {{3}} ".format(y))
119 plt.stem(np.arange(0, len(y)), len(y)+breakpoint_index,
120          y+breakpoint)
121 plt.show()
122 # Compute Fourier Coefficients for y[n]
123 for k in range (len(x)):
124     fourier_series_y.append(np.exp (-2j * np.pi * k *
125         shifting_factor) / len(x) + fourier_series_x[k])
126 print ("The Fourier Coefficients for y[n] are as follows: {{3}}"
127       "format (fourier_series_y))"
128
129 # sketch Fourier Coefficients of y[n]
130 plt.xlabel ('n')
131 plt.ylabel ('y[n]')
132 plt.title ("Fourier Coefficients of y[n]")
133 plt.stem (np.arange (0, len(fourier_series_y)),
134           np.real (fourier_series_y))
135 plt.show()
136
137 # Compute magnitude and Phase Spectrum for y[n]
138 for sample in range (len (fourier_series_y)):
139     magnitude_spectrum_y.append ((fourier_series_y[sample]
140         .real ** 2 + fourier_series_y[sample].imag ** 2) ** 0.5)
141     phase = np.arctan (fourier_series_y[sample].imag /
142                         fourier_series_y[sample].real)
143     phase_spectrum_y.append (phase)
144
145 # sketch magnitude Spectrum of y[n]
146 plt.xlabel ('k')
147 plt.ylabel ('|y[k]|')
148 plt.title ("Magnitude spectrum of y[n]: {{3}} ".format
149             (magnitude_spectrum_y))

```

(10)

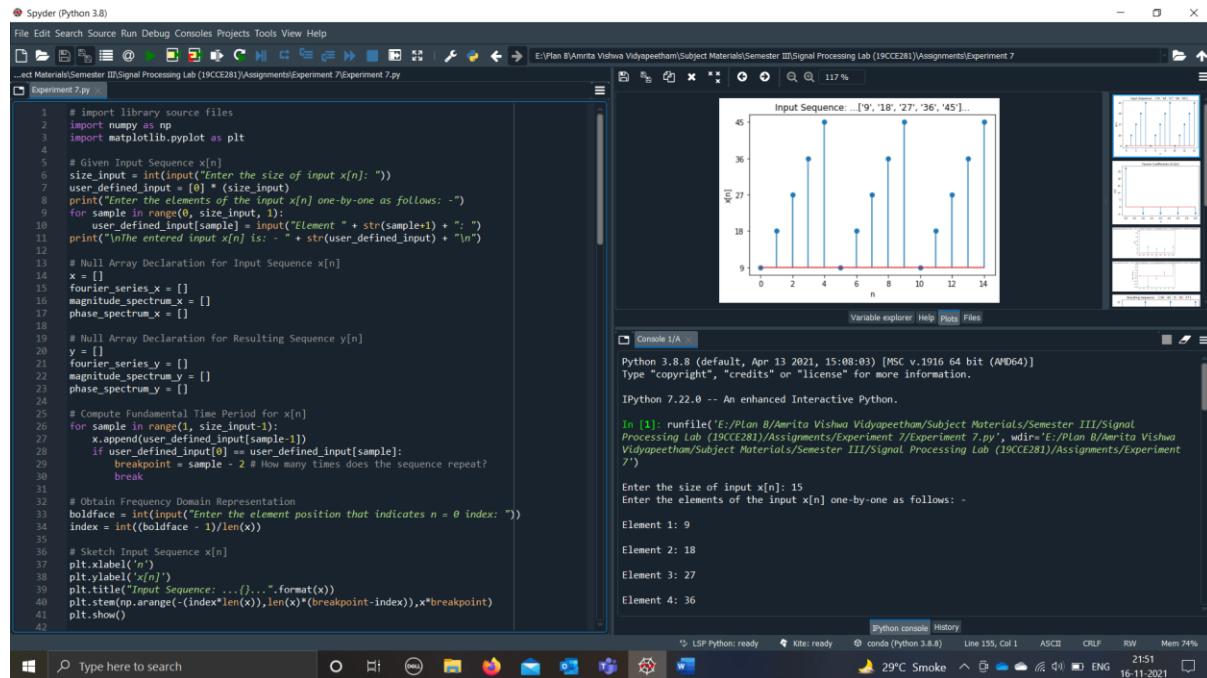
- 144 plt. stem (np.arange(0, len(magnitude_spectrum_y)), magnitude_spectrum_y))
 145 plt. show()
 146 print ("The Magnitude Spectrum of y[n] is as follows: ")
 format (magnitude_spectrum_y))
 147
 148 # sketch Phase Spectrum of y[n]
 149 plt.xlabel('k')
 150 plt.ylabel('Angle (in radians)')
 151 plt.title("Phase Spectrum of y[n]: ")
 format (phase_spectrum_y))
 152 plt.stem (np.arange(0, len(phase_spectrum_y)), phase_spectrum_y))
 153 plt.show()
 154 print ("The Phase Spectrum of y[n] is as follows: ")
 format (phase_spectrum_y))
 155

*

INFERENCES:

For the given input sequence $x[n]$, demonstrated the time-shifting property and compute Fourier coefficients for the resulting sequence $y[n]$ using DTFS.

RESULTSVERIFIED.

Given Input Sequence


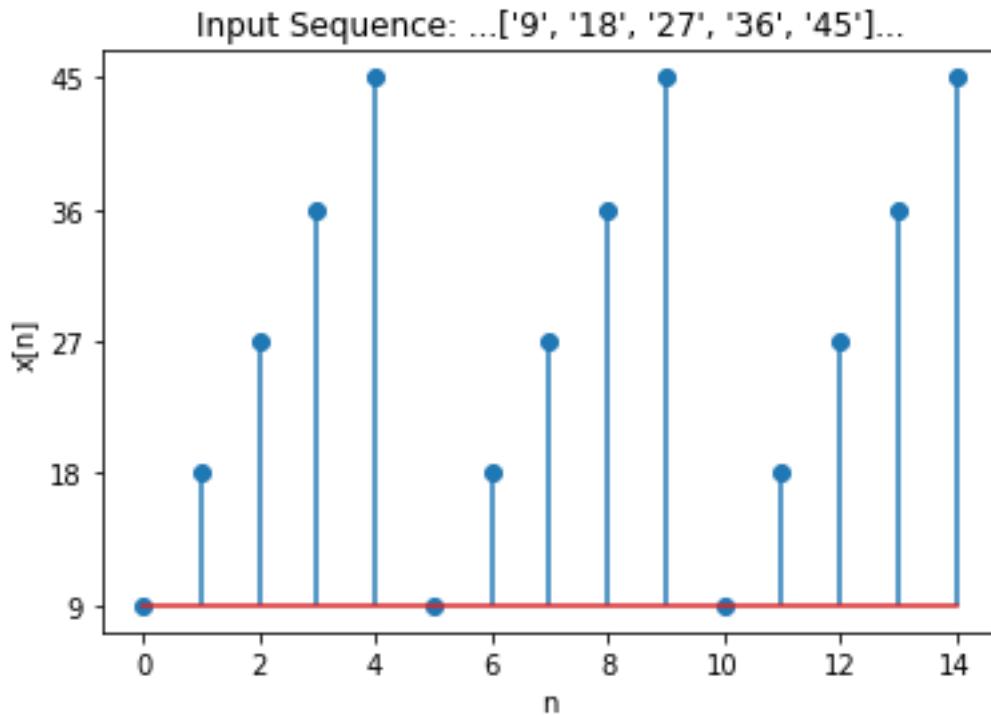
The screenshot shows the Spyder Python IDE interface. The code in the editor window is as follows:

```

1 # import library source files
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 # Given Input Sequence x[n]
6 size_input = int(input("Enter the size of input x[n]: "))
7 user_defined_input = [0] * (size_input)
8 print("Enter the elements of the input x[n] one-by-one as follows: -")
9 for sample in range(0, size_input, 1):
10     user_defined_input[sample] = input("Element " + str(sample+1) + ": ")
11     print("\nthe entered input x[n] is: - " + str(user_defined_input) + "\n")
12
13 # Null Array Declaration for Input Sequence x[n]
14 x = []
15 fourier_series_x = []
16 magnitude_spectrum_x = []
17 phase_spectrum_x = []
18
19 # Null Array Declaration for Resulting Sequence y[n]
20 y = []
21 fourier_series_y = []
22 magnitude_spectrum_y = []
23 phase_spectrum_y = []
24
25 # Compute Fundamental Time Period for x[n]
26 for sample in range(1, size_input-1):
27     x.append(user_defined_input[sample-1])
28     if user_defined_input[0] == user_defined_input[sample]:
29         breakpoint = sample - 2 # How many times does the sequence repeat?
30         break
31
32 # Obtain Frequency Domain Representation
33 boldface = int(input("Enter the element position that indicates n = 0 index: "))
34 index = int((boldface - 1)/len(x))
35
36 # Sketch Input Sequence x[n]
37 plt.xlabel('n')
38 plt.ylabel('x[n]')
39 plt.title("Input Sequence: ...{}...".format(x))
40 plt.stem(np.arange(-index*len(x)), len(x)*(breakpoint-index), x*breakpoint)
41 plt.show()
42

```

The console window shows the input sequence: `[9, 18, 27, 36, 45]`. The plot window displays the input sequence `x[n]` versus `n`, showing discrete values at `n = 0, 2, 4, 6, 8, 10, 12, 14`.



Step 1: Import library source files, numpy and matplotlib.pyplot.

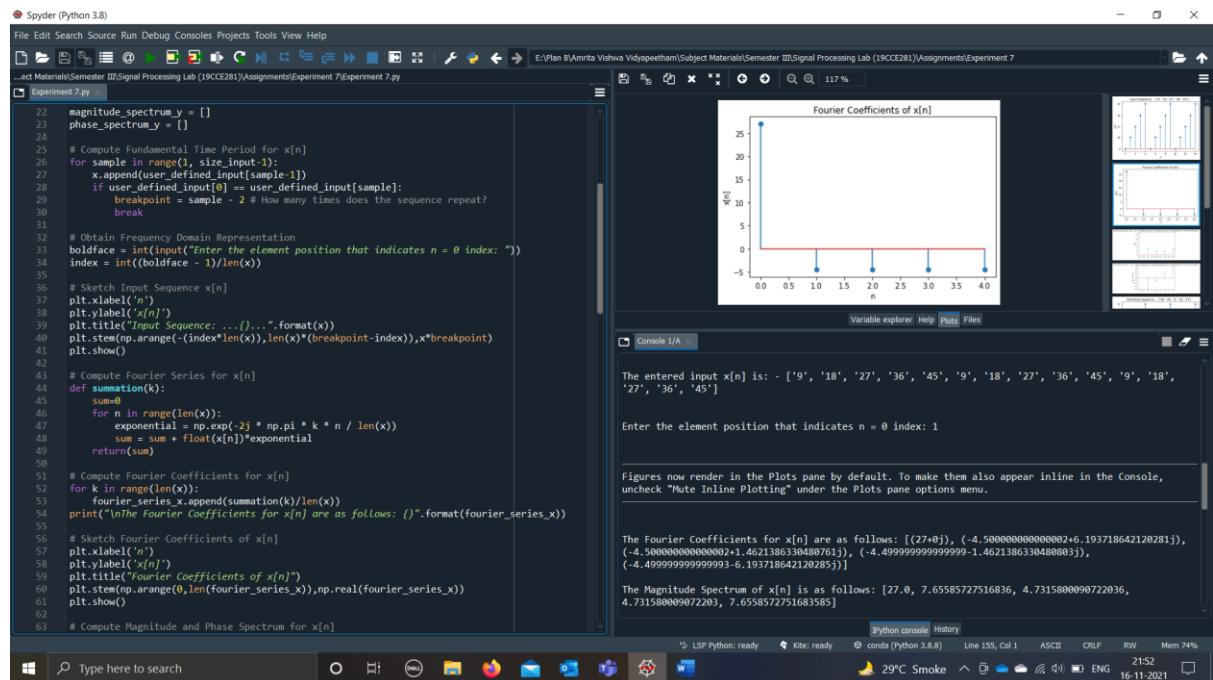
Step 2: Enter the size of the input and declare an array with the number of elements equal to the value of size.

Step 3: Enter the elements of the input along with the element position that indicates $n = 0$ index.

Step 4: Declare required null arrays required for input sequence $x[n]$ and resulting sequence $y[n]$.

Step 5: Compute fundamental time period and obtain frequency domain representation.

Step 6: Print the input sequence and show its labelled plot $x[n]$.

Plot Fourier Coefficients of $x[n]$


The screenshot shows the Spyder Python IDE interface. The code in the editor window is as follows:

```

22 magnitude_spectrum_y = []
23 phase_spectrum_y = []
24
25 # Compute Fundamental Time Period for x[n]
26 for sample in range(1, size_input-1):
27     x.append(user_defined_input[sample-1])
28     if user_defined_input[0] == user_defined_input[sample]:
29         breakpoint = sample - 2 # How many times does the sequence repeat?
30         break
31
32 # Obtain Frequency Domain Representation
33 boldface = int(input("Enter the element position that indicates n = 0 index: "))
34 index = int((boldface - 1)/len(x))
35
36 # Sketch Input Sequence x[n]
37 plt.xlabel('n')
38 plt.ylabel('x[n]')
39 plt.title("Input Sequence: ...".format(x))
40 plt.stem(np.arange(-(index+1), len(x)*(breakpoint-index)), x*breakpoint)
41 plt.show()
42
43 # Compute Fourier Series for x[n]
44 def summation(k):
45     sum=0
46     for n in range(len(x)):
47         exponential = np.exp(-2j * np.pi * k * n / len(x))
48         sum = sum + float(x[n])*exponential
49     return(sum)
50
51 # Compute Fourier Coefficients for x[n]
52 for k in range(len(x)):
53     fourier_series_x.append(summation(k)/len(x))
54 print("\nThe Fourier Coefficients for x[n] are as follows: {}".format(fourier_series_x))
55
56 # Sketch Fourier Coefficients of x[n]
57 plt.xlabel('n')
58 plt.ylabel('x[n]')
59 plt.title("Fourier Coefficients of x(n)")
60 plt.stem(np.arange(0,len(fourier_series_x)),np.real(fourier_series_x))
61 plt.show()
62
63 # Compute Magnitude and Phase Spectrum for x[n]

```

The plots pane displays the following:

- A plot titled "Fourier Coefficients of $x[n]$ " showing discrete blue vertical bars at $n = 0, 10, 20, 30, 40$ with values approximately 27, -4.5, -4.5, -4.5, and -4.5 respectively.
- A smaller inset plot titled "Fourier Coefficients of x[n]" showing a dense grid of points.
- A plot titled "Magnitude Spectrum of x[n]" showing a single sharp peak at $n = 0$ with a value of approximately 27.
- A plot titled "Phase Spectrum of x[n]" showing a single sharp peak at $n = 0$ with a value of approximately -4.5.

The console output includes:

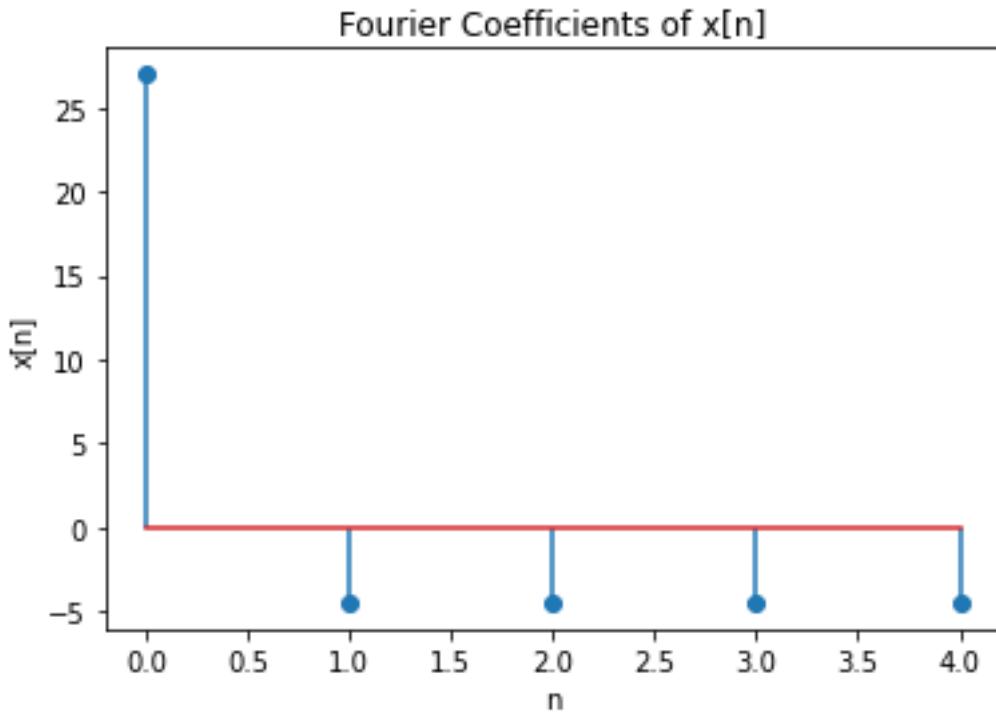
```

The entered input x[0] is: - [9, 18, 27, 36, 45, 9, 18, 27, 36, 45, 9, 18, 27, 36, 45]
Enter the element position that indicates n = 0 index: 1

The Fourier Coefficients for x[n] are as follows: [(27+0j), (-4.500000000000002+6.193718642120281j),
(-4.500000000000002+4.462138630480761j), (-4.49999999999999-6.193718642120285j),
(-4.49999999999999-6.193718642120285j)]

The Magnitude Spectrum of x[n] is as follows: [27.0, 7.65585727516836, 4.7315800090722036,
4.731580009072203, 7.6558572751683585]

```



Step 1: Compute Fourier series for $x[n]$.

Step 2: Compute Fourier coefficients for $x[n]$.

Step 3: Sketch Fourier coefficients of $x[n]$.

Plot Magnitude and Phase Spectrum of $x[n]$

The screenshot shows the Spyder Python IDE interface. The code in the editor window is as follows:

```

46     for n in range(len(x)):
47         exponential = np.exp(-2 * np.pi * k * n / len(x))
48         sum = sum + float(x[n])*exponential
49     return(sum)
50
51 # Compute Fourier Coefficients for x[n]
52 for k in range(len(x)):
53     fourier_series_x.append(summation(k)/len(x))
54 print("The Fourier Coefficients for x[n] are as follows: {}".format(fourier_series_x))
55
56 # Sketch Fourier Coefficients of x[n]
57 plt.xlabel('n')
58 plt.ylabel('x[n]')
59 plt.title("Fourier Coefficients of x[n]")
60 plt.stem(np.arange(0,len(fourier_series_x)),np.real(fourier_series_x))
61 plt.show()
62
63 # Compute Magnitude and Phase Spectrum for x[n]
64 for sample in range(len(fourier_series_x)):
65     magnitude_spectrum_x.append((fourier_series_x[sample].real**2 +
66         fourier_series_x[sample].imag**2)**0.5)
67     phase = np.arctan(fourier_series_x[sample].imag/fourier_series_x[sample].real)
68     phase_spectrum_x.append(phase)
69
70 # Sketch Magnitude Spectrum of x[n]
71 plt.xlabel('k')
72 plt.ylabel('|x[k]|')
73 plt.title("Magnitude Spectrum of x[n]: {}".format(magnitude_spectrum_x))
74 plt.stem(np.arange(0,len(magnitude_spectrum_x)),magnitude_spectrum_x)
75 plt.show()
76 print("The Magnitude Spectrum of x[n] is as follows: {}".format(magnitude_spectrum_x))
77
78 # Sketch Phase Spectrum of x[n]
79 plt.xlabel('k')
80 plt.ylabel('Angle (in radians)')
81 plt.title("Phase Spectrum of x[n]: {}".format(phase_spectrum_x))
82 plt.stem(np.arange(0,len(phase_spectrum_x)),phase_spectrum_x)
83 plt.show()
84 print("The Phase Spectrum of x[n] is as follows: {}".format(phase_spectrum_x))
85 intermediate = []

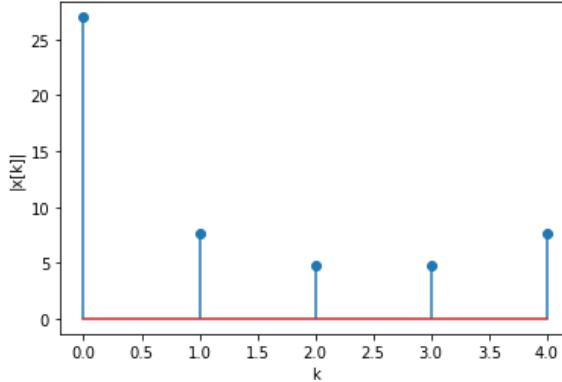
```

The execution results in the console pane show:

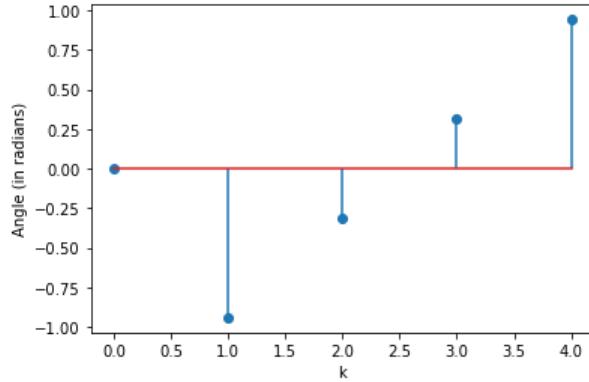
- Magnitude Spectrum of $x[n]$: $[27.0, 7.65585727516836, 4.7315800090722036, 4.731580009072203, 7.6558572751683585]$
- The Fourier Coefficients for $x[n]$ are as follows: $[(27+0j), (-4.500000000000002+6.193718642120281j), (-4.500000000000002+1.462138630480761j), (-4.49999999999999-1.462138630480803j), (-4.4999999999993-6.19371864210205j)]$
- The Magnitude Spectrum of $x[n]$ is as follows: $[27.0, 7.65585727516836, 4.7315800090722036, 4.731580009072203, 7.6558572751683585]$
- The Phase Spectrum of $x[n]$ is as follows: $[0.0, -0.9424777960769377, -0.31415926535897876, 0.31415926535897976, 0.9424777960769389]$

The variable explorer pane shows the variables `magnitude_spectrum_x`, `phase_spectrum_x`, and `fourier_series_x`.

Magnitude Spectrum of $x[n]$: $[27.0, 7.65585727516836, 4.7315800090722036, 4.731580009072203, 7.6558572751683585]$...



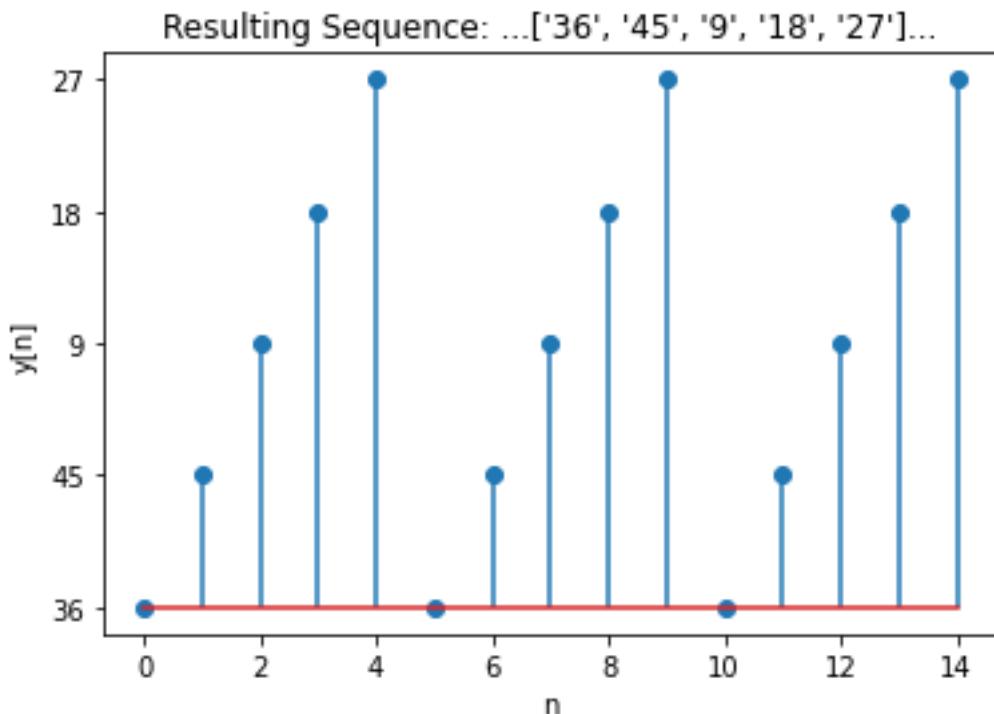
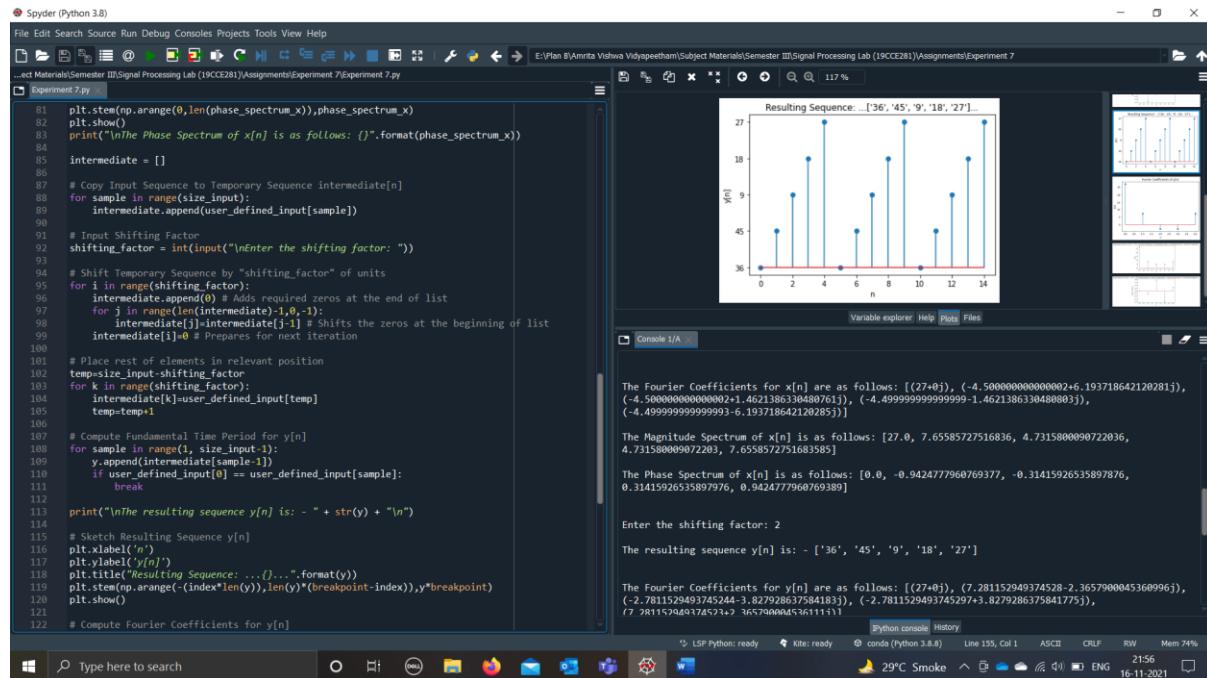
Phase Spectrum of $x[n]$: $[0.0, -0.9424777960769377, -0.31415926535897876, 0.31415926535897976, 0.9424777960769389]$...



Step 1: Compute magnitude and phase spectrum for $x[n]$.

Step 2: Sketch magnitude spectrum of $x[n]$.

Step 3: Sketch phase spectrum of $x[n]$.

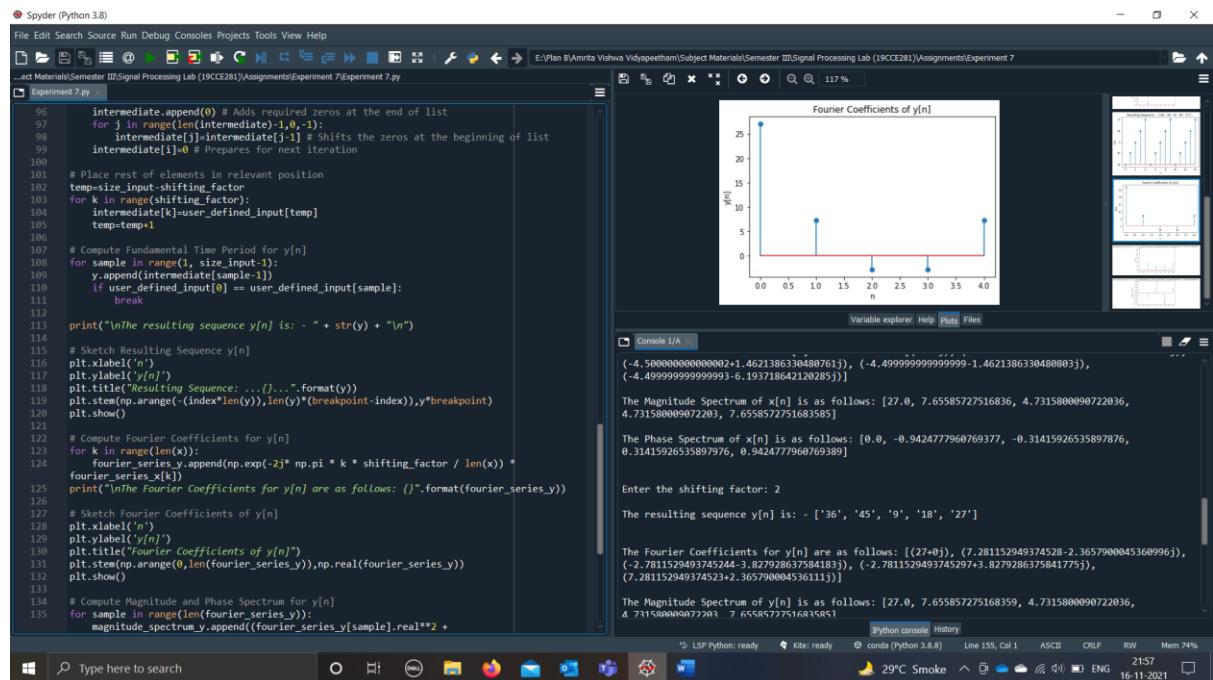
Resulting Sequence – $y[n]$ 

Step 1: Declare a NULL array “intermediate” and copy the input sequence to temporary sequence intermediate[n].

Step 2: Enter the shifting factor and shift temporary sequence by that many numbers of units.

Step 3: Place the rest of the elements in the relevant position and compute the fundamental time period for $y[n]$.

Step 4: Print and sketch the resulting sequence, $y[n]$.

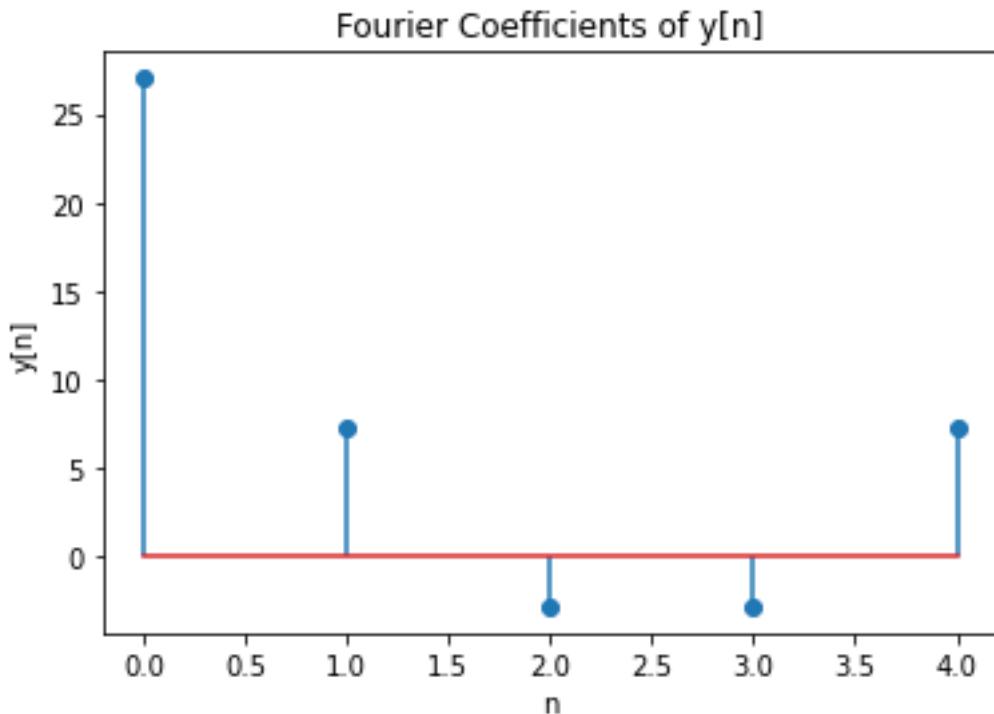
Plot Fourier Coefficients of $y[n]$


The screenshot shows the Spyder Python 3.8 IDE interface. The code in the editor window is for generating a plot of Fourier coefficients. The plot titled "Fourier Coefficients of $y[n]$ " shows discrete values at $n = 0, 10, 20, 30, 40$. The magnitude spectrum of $x[n]$ and phase spectrum of $x[n]$ are also displayed in the background plots. The console output provides the Fourier series coefficients and their magnitude and phase spectra.

```

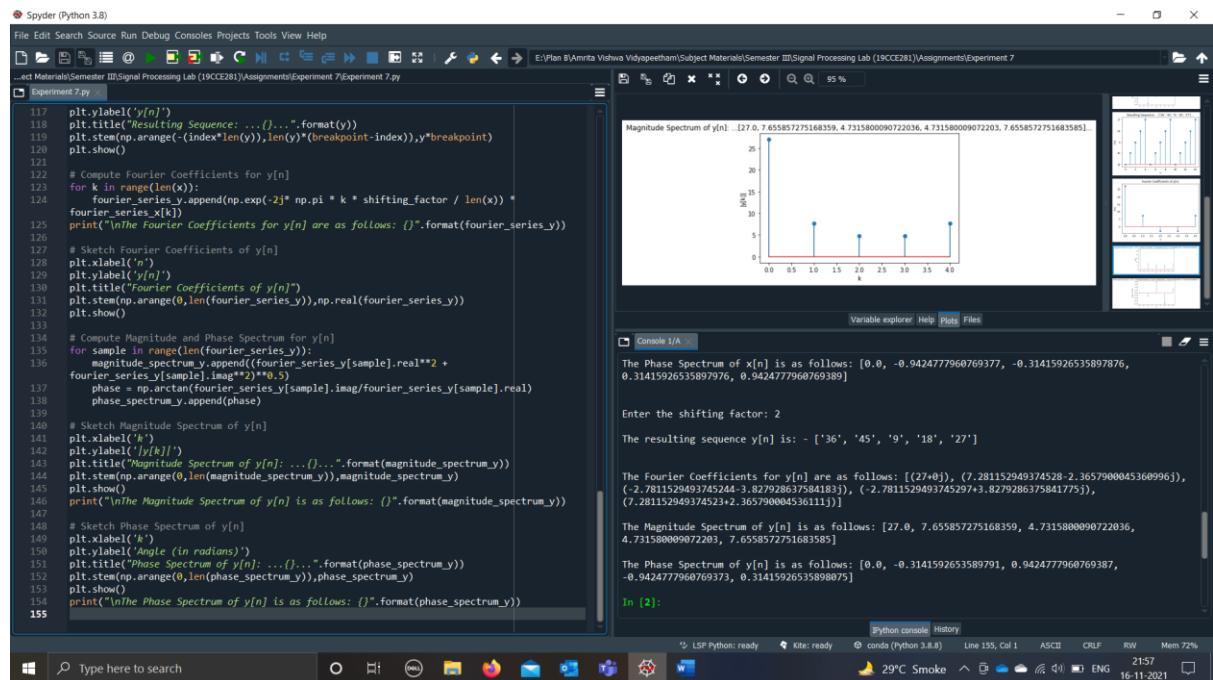
95     intermediate.append(0) # Adds required zeros at the end of list
96
97     for k in range(len(intermediate)-1,0,-1):
98         intermediate[j]=intermediate[j-1] # Shifts the zeros at the beginning of list
99
100    intermediate[0]=0 # Prepares for next iteration
101
102    # Place rest of elements in relevant position
103    temp_size_input=shifting_factor
104    for k in range(shifting_factor):
105        intermediate[k]=user_defined_input[temp]
106        temp+=1
107
108    # Compute Fundamental Time Period for  $y[n]$ 
109    for sample in range(1, size_input-1):
110        y.append(intermediate[sample-1])
111        if user_defined_input[0] == user_defined_input[sample]:
112            break
113
114    print("\n\nThe resulting sequence  $y[n]$  is: - " + str(y) + "\n")
115
116    # Sketch Resulting Sequence  $y[n]$ 
117    plt.xlabel('n')
118    plt.ylabel('y[n]')
119    plt.title("Resulting Sequence: ...".format(y))
120    plt.stem(np.arange(-(index*len(y)),len(y)*(breakpoint_index)),y,*breakpoint)
121    plt.show()
122
123    # Compute Fourier Coefficients for  $y[n]$ 
124    for k in range(len(x)):
125        fourier_series_y.append(np.exp(-2j*np.pi * k * shifting_factor / len(x)) * fourier_series_x[k])
126
127    # Sketch Fourier Coefficients of  $y[n]$ 
128    plt.xlabel('n')
129    plt.ylabel('y[n]')
130    plt.title("Fourier Coefficients of  $y[n]$ ")
131    plt.stem(np.arange(0,len(fourier_series_y)),np.real(fourier_series_y))
132    plt.show()
133
134    # Compute Magnitude and Phase Spectrum for  $y[n]$ 
135    for sample in range(len(fourier_series_y)):
136        magnitude_spectrum_y.append((fourier_series_y[sample].real)**2 +

```



Step 1: Compute Fourier coefficients for $y[n]$.

Step 2: Sketch Fourier coefficients of $y[n]$.

Plot Magnitude and Phase Spectrum of $y[n]$


The screenshot shows the Spyder Python IDE interface. The code editor contains a script named Experiment 7.py. The script performs several tasks:

- It plots the resulting sequence $y[n]$.
- It computes Fourier coefficients for $y[n]$ and prints them.
- It sketches the magnitude spectrum of $y[n]$.
- It sketches the phase spectrum of $y[n]$.

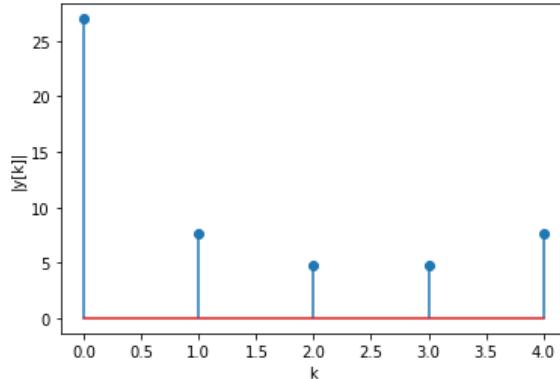
The IPython console displays the magnitude and phase spectra. The magnitude spectrum plot shows discrete values at $k=0, 1, 2, 3, 4$. The phase spectrum plot shows discrete values at the same k points.

```

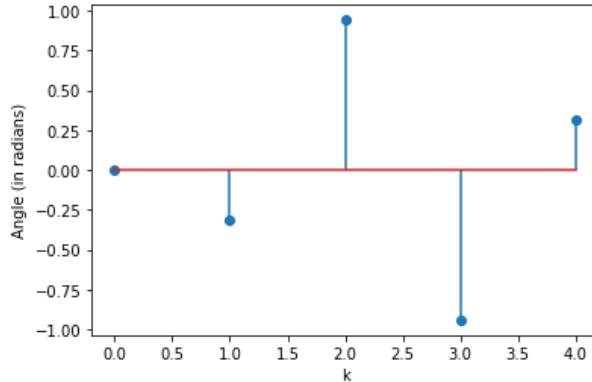
117 plt.ylabel('y[n]')
118 plt.title("Resulting Sequence: ...".format(y))
119 plt.stem(np.arange(0,index*len(y)),len(y)*(breakpoint-index),y*breakpoint)
120 plt.show()
121
122 # Compute Fourier Coefficients for y[n]
123 for k in range(len(x)):
124     fourier_series_y.append(np.exp(-2j*np.pi * k * shifting_factor / len(x)) *
125         fourier_series_x[k])
126
127 print("\nThe Fourier Coefficients for y[n] are as follows: {}".format(fourier_series_y))
128
129 # Sketch Fourier Coefficients of y[n]
130 plt.xlabel('n')
131 plt.ylabel('y[n]')
132 plt.title("Fourier Coefficients of y[n]")
133 plt.stem(np.arange(0,len(fourier_series_y)),np.real(fourier_series_y))
134 plt.show()
135
136 # Sketch Magnitude and Phase Spectrum for y[n]
137 for sample in range(len(fourier_series_y)):
138     magnitude_spectrum_y.append((fourier_series_y[sample].real)**2 +
139         fourier_series_y[sample].imag**2)**0.5
140     phase = np.arctan(fourier_series_y[sample].imag/fourier_series_y[sample].real)
141     phase_spectrum_y.append(phase)
142
143 # Sketch Magnitude Spectrum of y[n]
144 plt.xlabel('k')
145 plt.ylabel('|y[k]|')
146 plt.title("Magnitude Spectrum of y[n]: ...".format(magnitude_spectrum_y))
147 plt.stem(np.arange(0,len(magnitude_spectrum_y)),magnitude_spectrum_y)
148 plt.show()
149 print("\nThe Magnitude Spectrum of y[n] is as follows: {}".format(magnitude_spectrum_y))
150
151 # Sketch Phase Spectrum of y[n]
152 plt.xlabel('Angle (in radians)')
153 plt.title("Phase Spectrum of y[n]: ...".format(phase_spectrum_y))
154 plt.stem(np.arange(0,len(phase_spectrum_y)),phase_spectrum_y)
155 plt.show()
156 print("\nThe Phase Spectrum of y[n] is as follows: {}".format(phase_spectrum_y))

```

Magnitude Spectrum of $y[n]$: ...[27.0, 7.655857275168359, 4.7315800090722036, 4.731580009072203, 7.6558572751683585]...



Phase Spectrum of $y[n]$: ...[0.0, -0.3141592653589791, 0.9424777960769387, -0.9424777960769373, 0.31415926535898075]...



Step 1: Compute magnitude and phase spectrum for $y[n]$.

Step 2: Sketch magnitude spectrum of $y[n]$.

Step 3: Sketch phase spectrum of $y[n]$.

```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]  
Type "copyright", "credits" or "license" for more information.
```

```
IPython 7.29.0 -- An enhanced Interactive Python.
```

```
Restarting kernel...
```

```
In [1]:      'E:/Plan B/Amrita Vishwa Vidyapeetham/Subject Materials/Semester III/  
Signal Processing Lab (19CCE281)/Assignments/Experiment 7/Experiment 7.py'      = 'E:/Plan  
B/Amrita Vishwa Vidyapeetham/Subject Materials/Semester III/Signal Processing Lab  
(19CCE281)/Assignments/Experiment 7'
```

```
Enter the size of input x[n]: 15
```

```
Enter the elements of the input x[n] one-by-one as follows: -
```

```
Element 1: 9
```

```
Element 2: 18
```

```
Element 3: 27
```

```
Element 4: 36
```

```
Element 5: 45
```

```
Element 6: 9
```

```
Element 7: 18
```

```
Element 8: 27
```

```
Element 9: 36
```

```
Element 10: 45
```

```
Element 11: 9
```

```
Element 12: 18
```

```
Element 13: 27
```

```
Element 14: 36
```

```
Element 15: 45
```

```
The entered input x[n] is: - ['9', '18', '27', '36', '45', '9', '18', '27', '36', '45',  
'9', '18', '27', '36', '45']
```

```
Enter the element position that indicates n = 0 index: 1
```

```
The Fourier Coefficients for x[n] are as follows: [(27+0j),  
(-4.500000000000002+6.193718642120281j), (-4.500000000000002+1.4621386330480761j),
```

(-4.499999999999999-1.4621386330480803j), (-4.499999999999993-6.193718642120285j)]

The Magnitude Spectrum of $x[n]$ is as follows: [27.0, 7.65585727516836, 4.7315800090722036, 4.731580009072203, 7.6558572751683585]

The Phase Spectrum of $x[n]$ is as follows: [0.0, -0.9424777960769377, -0.31415926535897876, 0.31415926535897976, 0.9424777960769389]

Enter the shifting factor: 2

The resulting sequence $y[n]$ is: - ['36', '45', '9', '18', '27']

The Fourier Coefficients for $y[n]$ are as follows: [(27+0j), (7.281152949374528-2.3657900045360996j), (-2.7811529493745244-3.827928637584183j), (-2.7811529493745297+3.8279286375841775j), (7.281152949374523+2.365790004536111j)]

The Magnitude Spectrum of $y[n]$ is as follows: [27.0, 7.655857275168359, 4.7315800090722036, 4.731580009072203, 7.6558572751683585]

The Phase Spectrum of $y[n]$ is as follows: [0.0, -0.3141592653589791, 0.9424777960769387, -0.9424777960769373, 0.31415926535898075]

In [2]: