

| **Title:** Write a program to Compute linear and circular convolution of two discrete time signal sequences using Matlab. |
| --- |

**Objective:** To familiarize the beginnerto MATLAB by introducing the basic features and commands of the program.

**Expected Outcome of Experiment:**

| **CO** | **Outcome** |
| --- | --- |
| **CO3** | To understand the concept of convolution and perform different convolution operations on the given input signals. |

**Books/ Journals/ Websites referred:**

1. http://www.mathworks.com/support/
2. www.math.mtu.edu/~msgocken/intro/intro.html
3. www.mccormick.northwestern.edu/docs/efirst/matlab.pdf
4. A.Nagoor Kani “Digital Signal Processing”, 2nd Edition, TMH Education.

**Pre Lab/ Prior Concepts:**

**Convolution**

Discrete time convolution is a method of finding response of linear time invariant system. It is based on the concepts of linearity and time invariance and assumes that the system information

is known in terms of its impulse response h[n].

Convolution is defined as

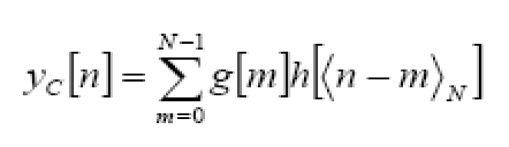
∞

Y[n] = Σ h[k]x [n-k] =h[n]\*x[n] k=-∞

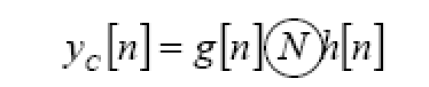
Convolution consists of folding, shifting, Multiplication and summation operations.

**Circular Convolution**

Circular convolution between two length N sequences can be carried out as shown by the expression below:

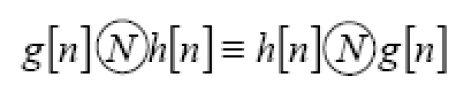


Since the above operation involves two length-N sequences it is referred to as the N-point circular convolution and denoted by:

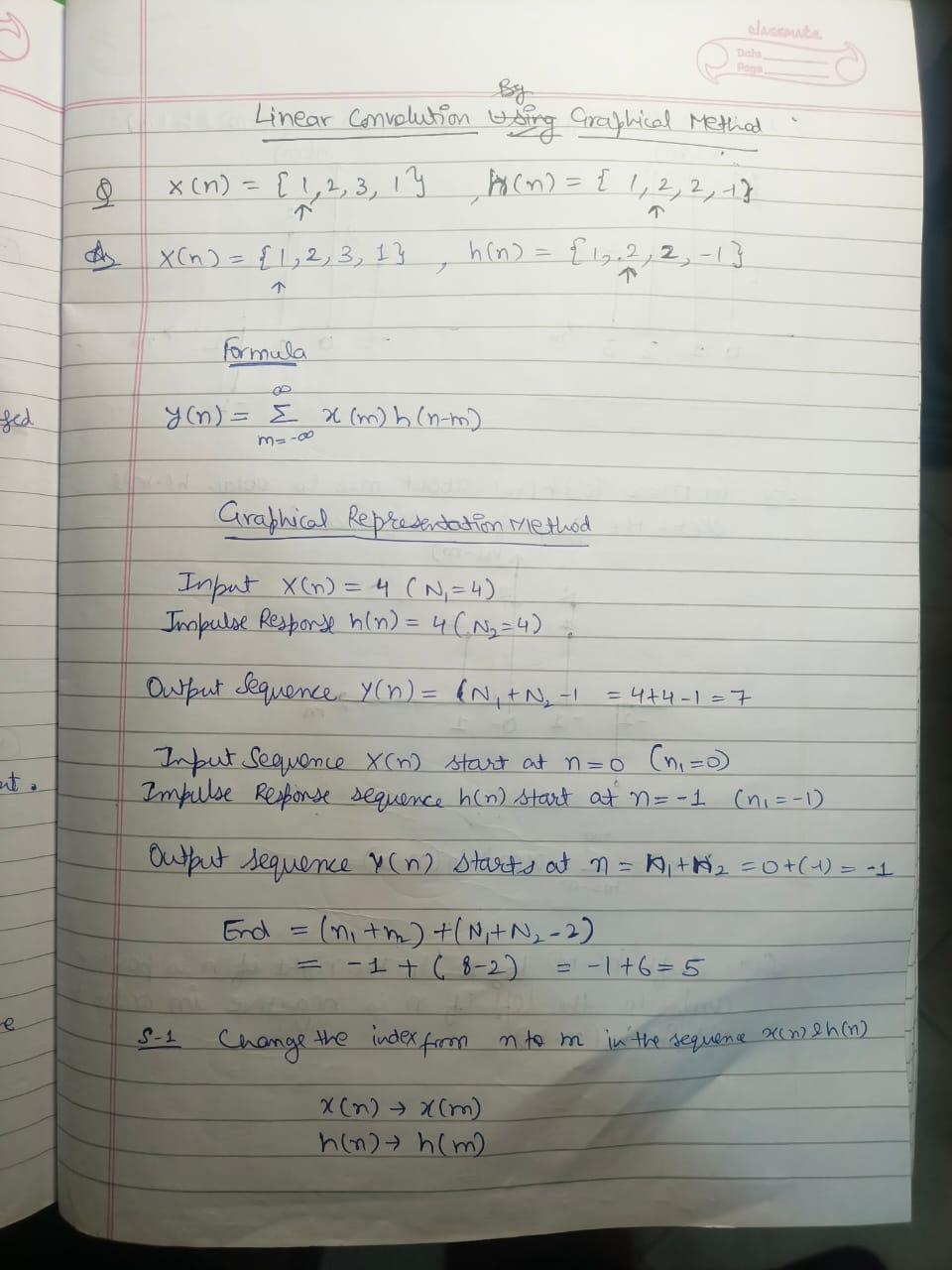


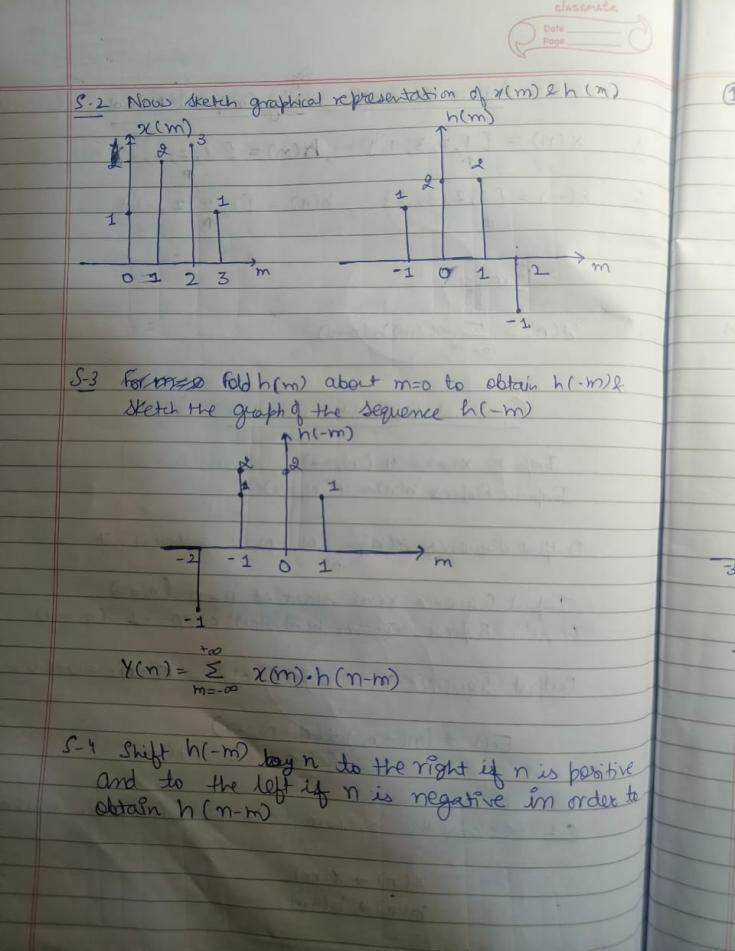
As in linear convolution circular convolution is commutative.

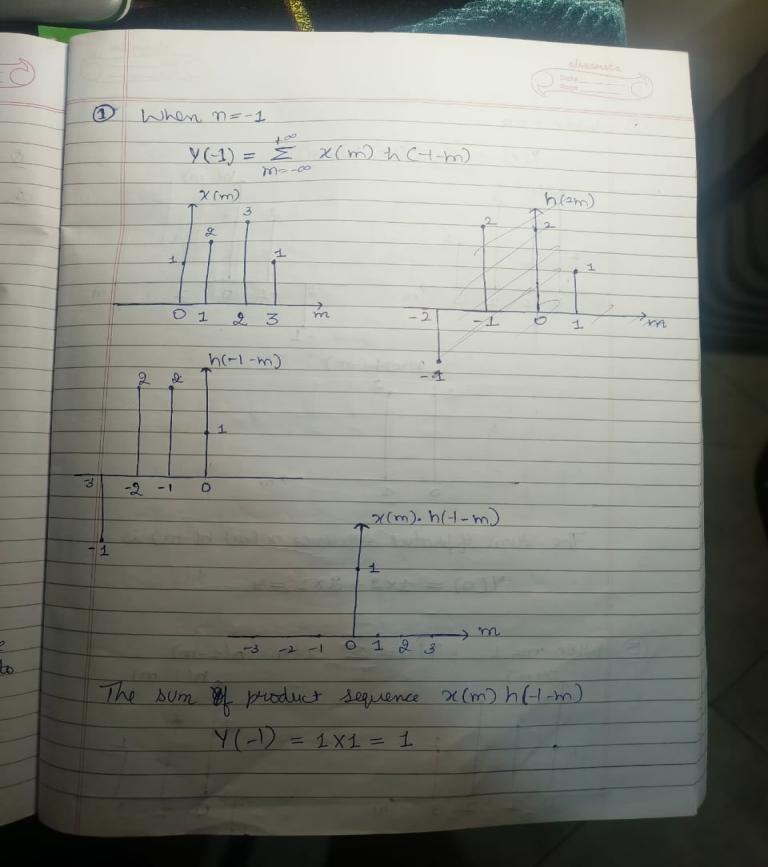
i.e.

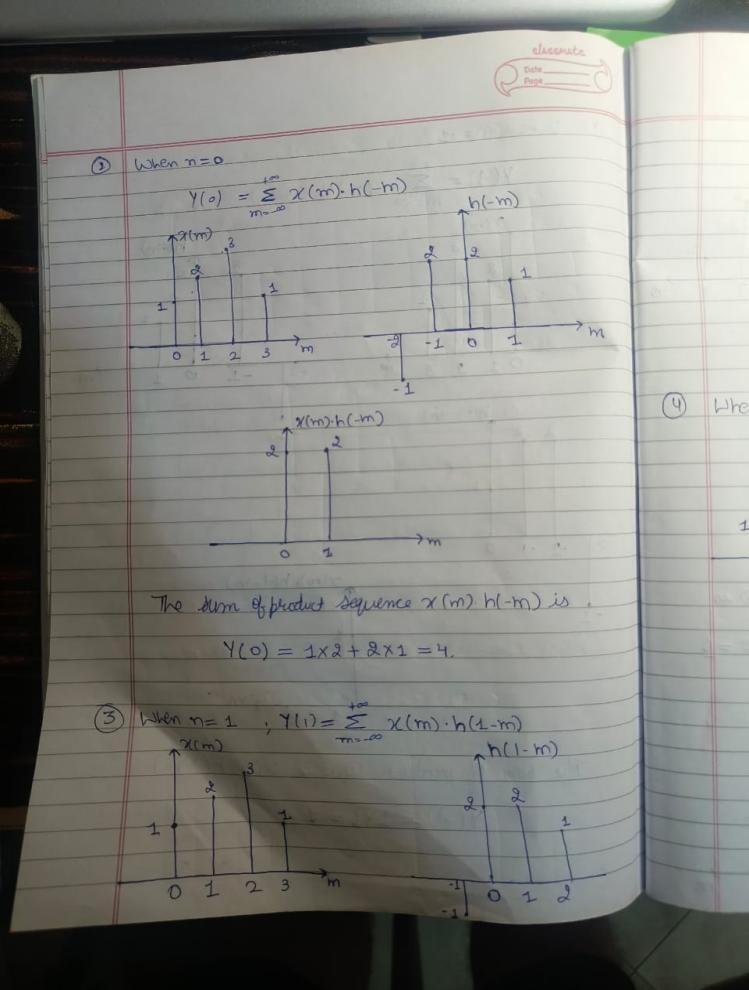


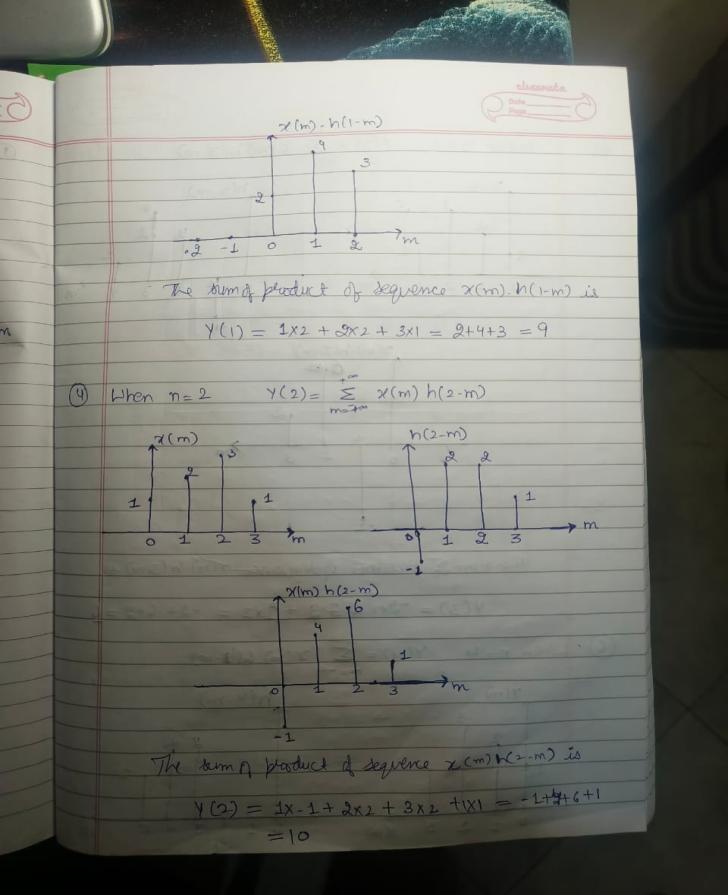
**Example Of Linear Convolution:**

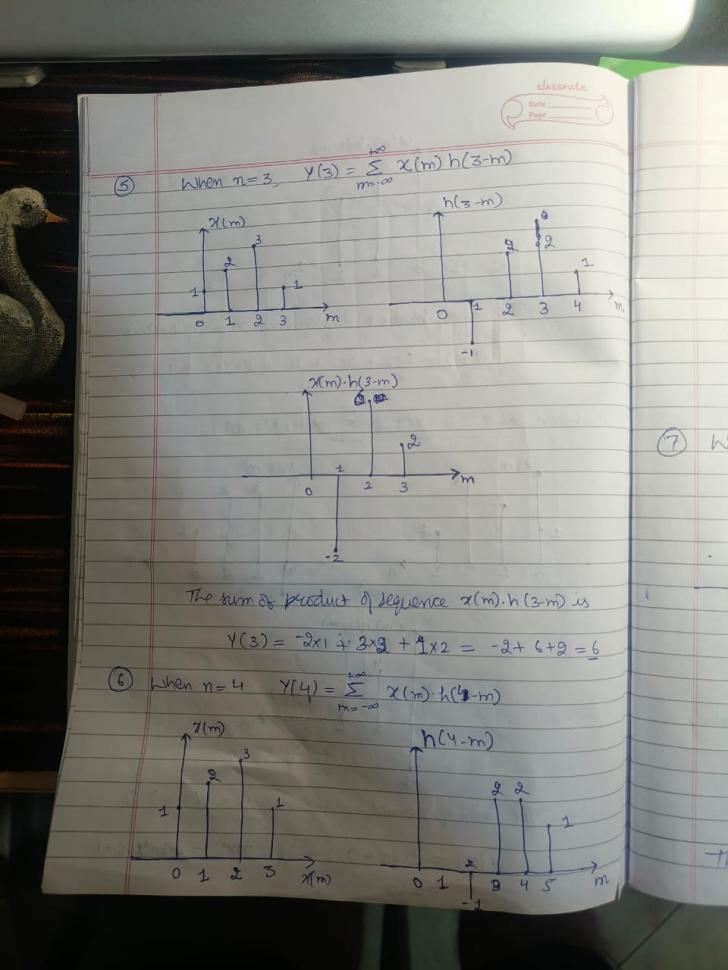


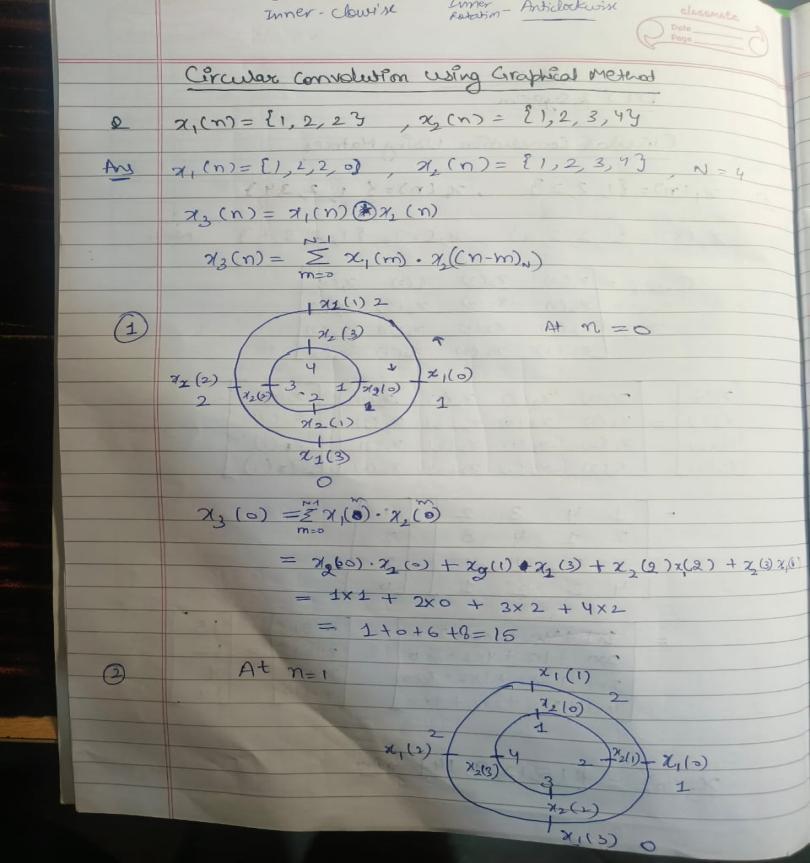
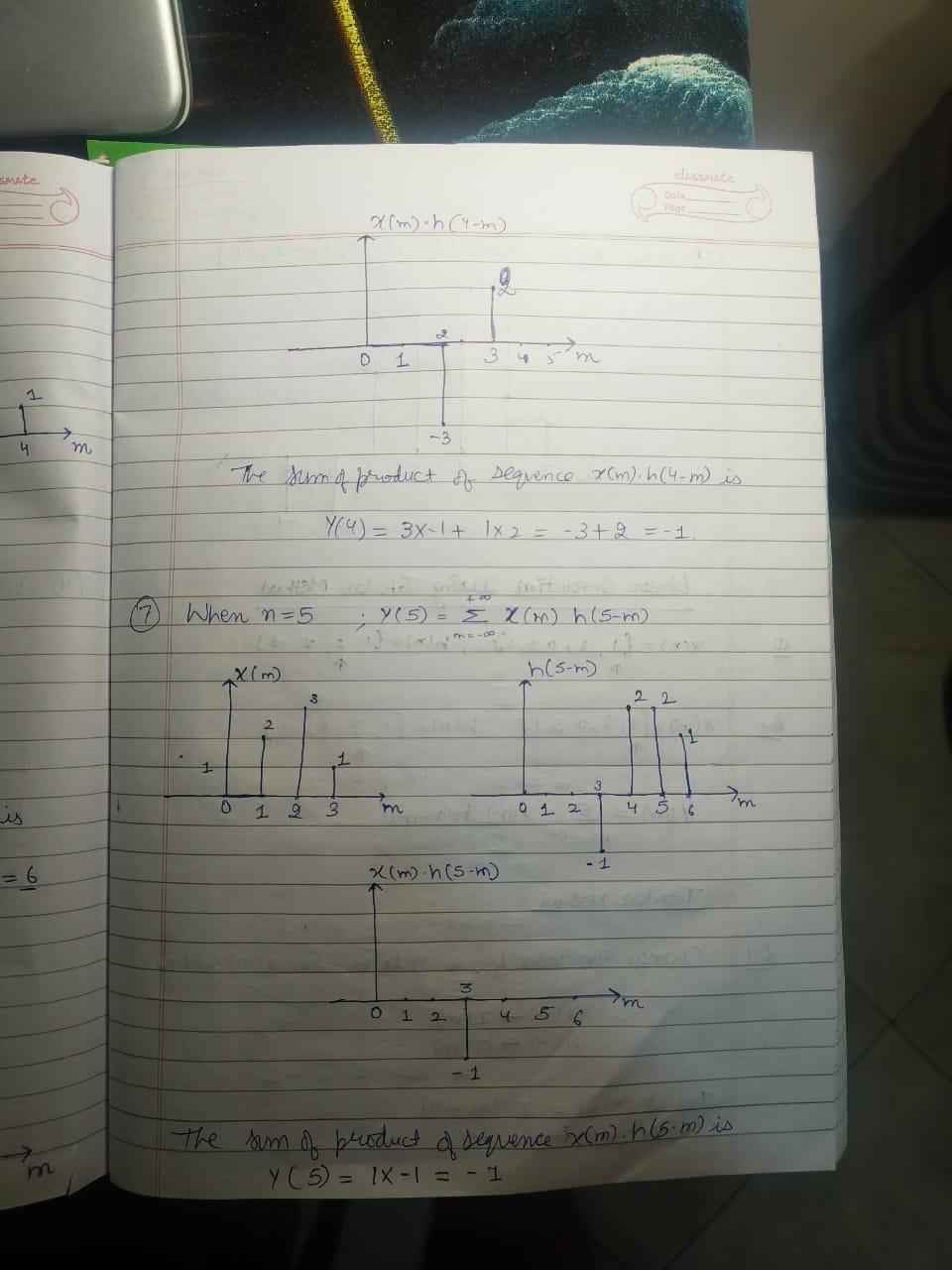




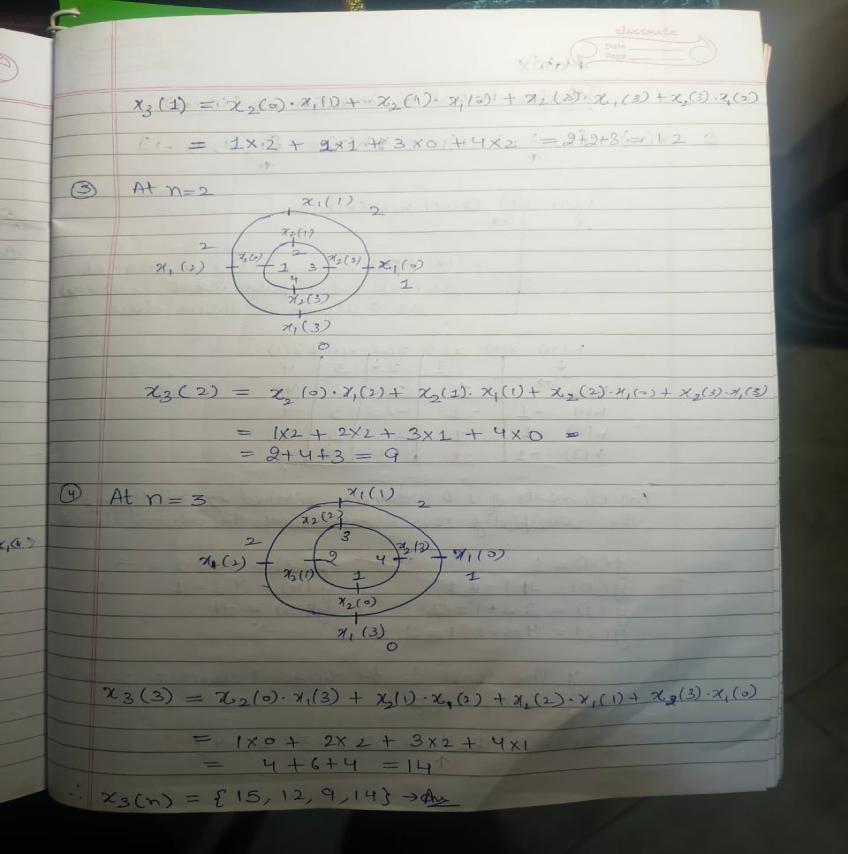








**Example Of Circular Convolution:**



**Implementation details along with screenshots:**

**x = str2double(input('Enter input signal: ', 's')); % Convert input to numeric array**

**x\_origin = input('Enter starting index of x(n): ');**

**h = (input('Enter impulse response signal: ', 's')); % Convert input to numeric array**

**h\_origin = input('Enter starting index of h(n): ');**

**[y, plots] = linear\_convolution\_with\_plots(x, x\_origin, h, h\_origin);**

**disp('The output of linear convolution y(n) is:');**

**disp(round(y, 2));**

**rows = length(plots) + 1;**

**cols = 3;**

**max\_rows\_per\_figure = 3;**

**total\_figures = ceil(rows / max\_rows\_per\_figure);**

**current\_row = 1;**

**for fig = 1:total\_figures**

**figure('Position', [100, 100, 1200, 800]);**

**for subplot\_idx = 1:(max\_rows\_per\_figure \* cols)**

**if current\_row > rows**

**break;**

**end**

**subplot\_idx\_in\_row = mod(subplot\_idx - 1, cols) + 1;**

**subplot\_idx\_in\_figure = ceil(subplot\_idx / cols);**

**subplot(max\_rows\_per\_figure, cols, subplot\_idx);**

**if current\_row == 1**

**if subplot\_idx\_in\_row == 1**

**stem(x\_origin:x\_origin+length(x)-1, x, 'filled', 'MarkerFaceColor', [0.1, 0.6, 0.8]);**

**title('x(n)', 'FontSize', 12, 'FontWeight', 'bold');**

**elseif subplot\_idx\_in\_row == 2**

**stem(h\_origin:h\_origin+length(h)-1, h, 'filled', 'MarkerFaceColor', [0.8, 0.3, 0.1]);**

**title('h(n)', 'FontSize', 12, 'FontWeight', 'bold');**

**else**

**axis off;**

**end**

**else**

**step\_idx = current\_row - 1;**

**if subplot\_idx\_in\_row == 1**

**stem(x\_origin:x\_origin+length(x)-1, x, 'filled', 'MarkerFaceColor', [0.1, 0.6, 0.8]);**

**title('x(n)', 'FontSize', 12, 'FontWeight', 'bold');**

**elseif subplot\_idx\_in\_row == 2**

**n\_vals = h\_origin + (0:length(plots{step\_idx}.h\_shifted)-1);**

**stem(n\_vals, plots{step\_idx}.h\_shifted, 'filled', 'MarkerFaceColor', [0.8, 0.3, 0.1]);**

**title(['h(-k+' num2str(step\_idx-1) ')'], 'FontSize', 12, 'FontWeight', 'bold');**

**elseif subplot\_idx\_in\_row == 3**

**n\_vals = h\_origin + (0:length(plots{step\_idx}.multiplication)-1);**

**stem(n\_vals, plots{step\_idx}.multiplication, 'filled', 'MarkerFaceColor', [0.4, 0.8, 0.4]);**

**title(['Multiplication (Step ' num2str(step\_idx-1) ')'], 'FontSize', 12, 'FontWeight', 'bold');**

**end**

**end**

**xlabel('n', 'FontSize', 10);**

**ylabel('Amplitude', 'FontSize', 10);**

**grid on;**

**if subplot\_idx\_in\_row == cols**

**current\_row = current\_row + 1;**

**end**

**end**

**end**

**figure('Position', [100, 100, 1200, 400]);**

**stem(x\_origin + h\_origin:x\_origin + h\_origin + length(y) - 1, y, 'filled', 'MarkerFaceColor', [0.6, 0.2, 0.8]);**

**title('Final Output y(n)', 'FontSize', 14, 'FontWeight', 'bold');**

**xlabel('n', 'FontSize', 12);**

**ylabel('Amplitude', 'FontSize', 12);**

**grid on;**

**% Function for linear convolution**

**function [y, plots] = linear\_convolution\_with\_plots(x, ~, h, ~)**

**len\_x = length(x);**

**len\_h = length(h);**

**N = len\_x + len\_h - 1;**

**y = zeros(1, N);**

**plots = cell(1, N);**

**for n = 1:N**

**h\_shifted = zeros(1, N);**

**multiplication = zeros(1, N);**

**for k = 1:len\_h**

**if n - k + 1 > 0 && n - k + 1 <= len\_x**

**multiplication(n - k + 1) = x(n - k + 1) \* h(k);**

**y(n) = y(n) + multiplication(n - k + 1);**

**h\_shifted(n - k + 1) = h(k);**

**end**

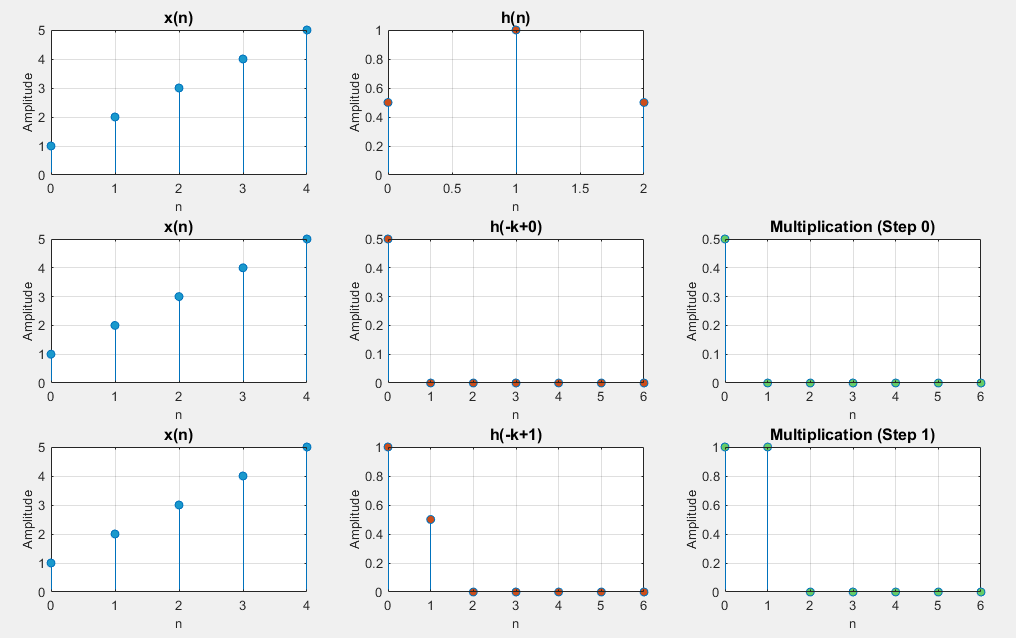
**end**

**plots{n} = struct('h\_shifted', h\_shifted, 'multiplication', multiplication, 'y\_value', y(n));**

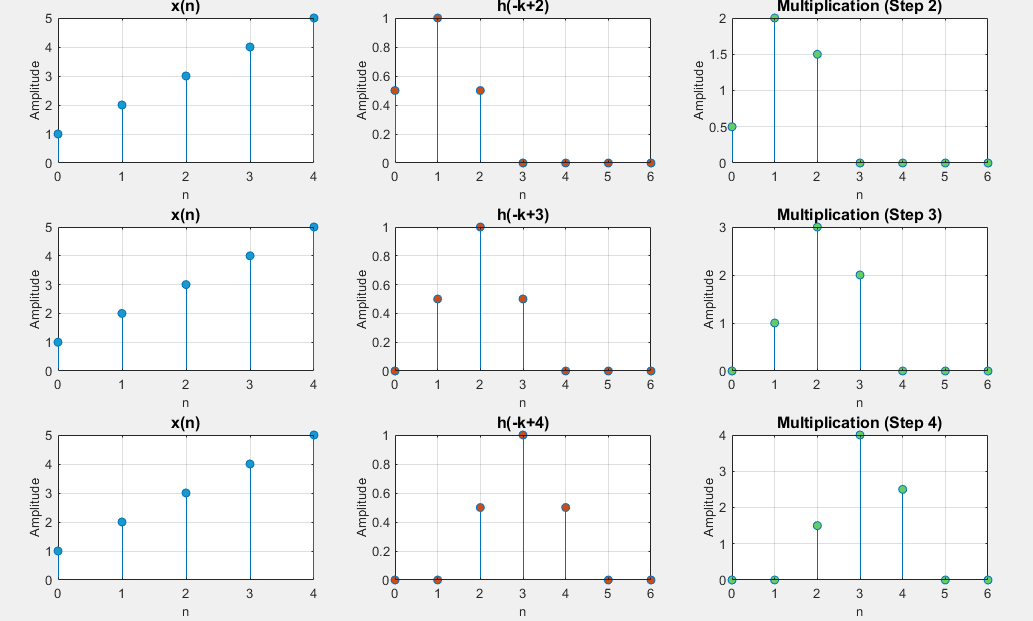
**end**

**end**

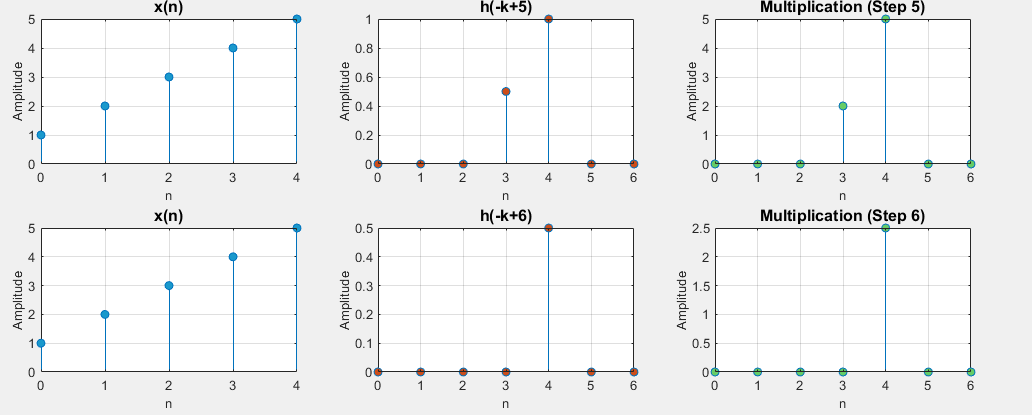
**FIG 1:**

****

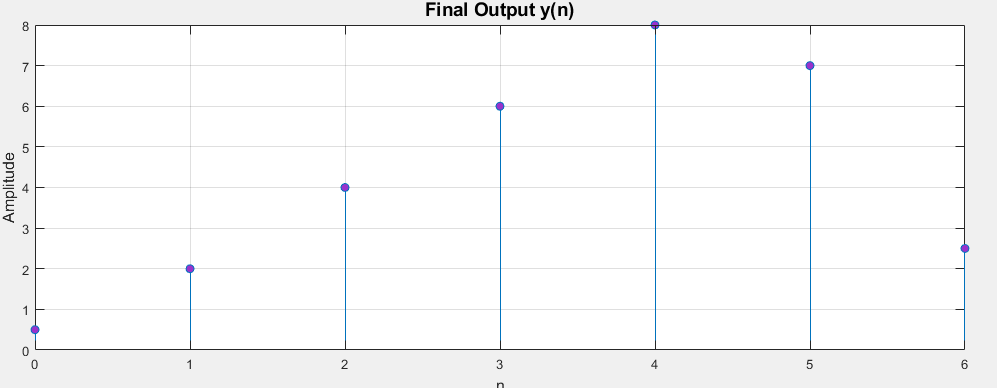
**FIG 2:**

****

**FIG 3:**

****

**FIG 4:**

****

**Conclusion:-** We learnt how to solve and implement linear convolution for generalized values.

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_ Signature of faculty in-charge**

**Post Lab Descriptive Questions**

* 1. Explain the role of convolution in signal processing.

**Ans:** Convolution is a fundamental operation in signal processing used to analyze and modify signals. Its key roles include:

● **System Response Analysis**: Determines the output of a Linear Time-Invariant (LTI) system when given an input and impulse response.

Dept. of Computer Engg. DSIP Lab Sem VI Jan-Apr 2025 Page 10



**K. J. Somaiya College of Engineering, Mumbai-77**

● **Filtering**: Used in designing digital filters to enhance or suppress specific signal components.

● **Feature Extraction**: Helps in image and audio processing by detecting patterns and smoothing signals.

● **Noise Reduction**: Eliminates unwanted noise from signals by convolving with appropriate filters.

● **Edge Detection**: In image processing, convolution with specific kernels helps detect edges and textures.

* 1. Explain the difference between linear and circular convolution?

**Ans:**

| **Feature** | **Linear Convolution** | **Circular Convolution** |
| --- | --- | --- |
| **Definition** | Computes the convolution of two sequences as if they  extend infinitely with  zero-padding. | Assumes periodic extension of sequences, making the result wrap around. |
| **Length of**  **Output** | N+M−1 (for sequences of length N and M) | Max(N, M) due to periodicity. |
| **Application**  **Handling of Data** | Used in time-domain signal processing, such as LTI system analysis.  Zero-padding is used to avoid overlapping. | Used in fast computations like FFT (Fast Fourier Transform).  The output wraps around due to periodicity. |

* 1. Explain with the help of an example the steps required to transform linear convolution with circular convolution and vice-versa.

**Ans:**

**1. To obtain the circular convolution result from a linear convolution, we must ensure the sequences are of the same length by zero-padding.**

Example: Let x(n)=[1,2,3] and h(n)=[4,5]

Compute Linear Convolution:

y(n)=[4,13,22,15](Length = 3+2−1=4)

1. Zero-pad x(n) and h(n) to the next power of two (or desired circular length): x(n)=[1,2,3,0],h(n)=[4,5,0,0]

2. Compute Circular Convolution (DFT-IDFT method or Direct formula): - Perform DFT on both sequences.

- Multiply element-wise in the frequency domain.

- Take inverse DFT to get the circularly convolved result.

**2. If a circular convolution result is given and we need to obtain linear convolution, we extend the sequences appropriately:**

1. Ensure proper zero-padding of sequences before performing circular convolution.

2. Extract only the first M+N−1 samples from the circular convolution result to match the expected length of linear convolution.