QMP7.1 D/F

Channabasaveshwara Institute of Technology
(Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi)
(**NAAC Accredited & ISO 9001:2015 Certified Institution)**
NH206(B.H. Road), Gubbi, Tumkur–572216.Karnataka.

Department of Electronics & Communication Engineering

DIGITAL SIGNAL PROCESSING LAB (IPCC) (BEC502)

(NEP, Outcome Based Education (OBE) and Choice BasedCredit System (CBCS)

B.E - V Semester

Lab Manual 2024-25

Name:____

USN:____

Batch:__________________Section:_____________________

Channabasaveshwara Institute of Technology
(Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi)
(**NAAC Accredited & ISO 9001:2015 Certified Institution)**
NH206(B.H. Road), Gubbi, Tumkur–572216.Karnataka.

Department of Electronics & Communication Engineering

DIGITAL SIGNAL PROCESSING LAB (IPCC) (BEC502)

Manual

Prepared by:
1. **Dr. Veena Kumari H M**
   Professor
2. **Mr. Sreenivasa T V**
   Assistant Professor
3. **Ms. Divya Shree B S**
   Assistant Professor

Reviewed by:
Dr. Veena Kumari H M
Professor

Approved by:

Dr. Suresh D S
Professor & HOD,
Dept. Of ECE

.

INSTITUTE VISION

- To create centers of excellence in education and to serve the society by enhancing the quality of life through value based professional leadership.

INSTITUTE MISSION

- To provide high quality technical and professionally relevant education in a diverse learning environment.
- To provide the values that prepare students to lead their lives with personal integrity, professional ethics and civic responsibility in a global society.
- To prepare next generation of skilled professionals to successfully compete in the diverse global market.
- To promote campus environment that welcomes and honors women and men of all races, creeds and cultures, values and intellectual curiosity, pursuit of knowledge and academic integrity and freedom.
- To offer wide variety of off-campus education and training programmes to individuals and groups.
- To stimulate collaborative efforts with Industry, Universities, Government and Professional Societies.
- To facilitate public understanding of technical issues and achieve excellence in the operations of the institute.

QUALITY POLICY

Our organization delights customers (Student, Parents and Society) by providing value added quality education to meet the National and International requirements. We also provide necessary steps to train the students for placement and continue to improve our methods of education to the students through effective. Quality Management System, Quality Policy and Quality Objectives.

Department of Electronics & Communication Engineering

VISION OF THE DEPARTMENT
"To create globally competent Electronics and Communication Engineering professionals with ethical and moral values for the betterment of the society"

MISSION OF THE DEPARTMENT
▪ To nurture the technical/professional/engineering and entrepreneurial skills for overall self and societal upliftment through co-curricular and extra-curricular events. ▪ To orient the Faculty/Student community towards the higher education, research and development activities. ▪ To create the Centres of Excellence in the field of electronics and communication in collaboration with industries/Universities by training the faculty through latest technologies. ▪ To impart quality technical education in the field of electronics and communication engineering to meet over the current/future global industry requirements.

PROGRAM EDUCATIONAL OBJECTIVES (PEO's)

PEO1: Apply Mathematical, Scientific and Engineering skills for solving problems in the areas of Electronics and Communication Engineering.

PEO2: Expose to Emerging Technologies and excel in Industries/higher studies/research.

PEO3: Apply analytical skills in the areas of Electronics and Communication Engineering to become competent and Employable.

PEO4: Inculcate Professional ethics, human values, team work for solving Engineering problems and contribute to societal needs.

PROGRAM SPECIFIC OUTCOMES

PSO1: Build Analog and Digital Electronic systems for Multimedia Applications, VLSI and Embedded Systems in Interdisciplinary Research / Development.

PSO2: Design and Develop Communication Systems as per Real Time Application and Current Trends.

COURSE OBJECTIVES

1. Preparation: To prepare students with fundamental knowledge/ overview in the field of Digital Signal Processing
2. Core Competence: To equip students with a basic foundation of Signal Processing by delivering the basics of Discrete Fourier Transforms, their properties, efficient computations & the design of digital filters.

COURSE OUTCOMES
CO1: Analyze the different types of signals and systems used in digital signal processing.
CO2: Compute the response of an LTI system using time and frequency domain techniques.
CO3: Develop algorithms for the efficient computations of DFT and IDFT.
CO4: Design of digital FIR filters for the given specifications using different window methods.
CO5: Design of digital IIR digital filters using bilinear transformation method.

PROGRAM OUTCOMES

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs.
4. **Conduct investigations of complex problems:** An ability to design and conduct scientific and engineering experiments, as well as to analyze and interpret data to provide valid conclusions
5. **Modern tool usage:** Ability to apply appropriate techniques, modern engineering and IT tools, to engineering problems.
6. **The engineer and society:** An ability to apply reasoning to assess societal, safety, health and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** An ability to understand the impact of professional engineering solutions in societal and environmental contexts
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Ability to function effectively as an individual, and as a member or leader in a team, and in multidisciplinary tasks.
10. **Communication:** Ability to communicate effectively on engineering activities with the engineering community such as, being able to comprehend and write effective reports and design documentation, make effective presentations.
11. **Project management and finance:** An ability to apply knowledge, skills, tools, and techniques to project activities to meet the project requirements with the aim of managing project resources properly and achieving the project’s objectives.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Department of Electronics & Communication Engineering

Syllabus

VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI **B.E: Electronics & Communication Engineering** **NEP, Outcome Based Education (OBE) and Choice Based Credit System (CBCS)** **SEMESTER – V**			
DIGITAL SIGNAL PROCESSING LABORATORY (IPCC) (BEC502)			
Course Code	**BEC502**	**CIE Marks**	**50**
Teaching Hours/Week (L: T: P: S)	**(3:0:2:0)**	**SEE Marks**	**50**
Total Hours of Pedagogy	**40 hours Theory + 8-10 Lab slots**	**Total Marks**	**100**
Credits	**04**	**Exam Hours**	**03**

PRACTICAL COMPONENT OF IPCC

List of Programs to be implemented & executed using any programming languages like Moku: Go/MATLAB/OCTAVE (but not limited to)

SL. No	Experiments
1	Program to generate the following discrete time signals. a) Unit sample sequence, b) Unit step sequence, c) Exponential sequence, d) Sinusoidal sequence, e) Random sequence
2	Program to perform the following operations on signals. a) Signal addition, b) Signal multiplication, c) Scaling, d) Shifting, e) Folding
3	Program to perform convolution of two given sequences (without using built-in function) and display the signals
4	Consider a causal system $y(n) = 0.9y(n-1) + x(n)$. a) Determine $H(z)$ and sketch its pole zero plot. b) Plot $\lvert H(e^{j\omega}) \rvert$ and $\angle H(e^{j\omega})$ c) Determine the impulse response $h(n)$.
5	Computation of N point DFT of a given sequence (without using built-in function) and to plot the magnitude and phase spectrum.
6	Using the DFT and IDFT, compute the following for any two given sequences a) Circular convolution b) Linear convolution
7	Verification of Linearity property, circular time shift property & circular frequency shift property of DFT.
8	Develop decimation in time radix-2 FFT algorithm without using built-in functions.
9	Design and implementation of digital low pass FIR filter using a window to meet the given specifications
10	Design and implementation of digital high pass FIR filter using a window to meet the given specifications
11	Design and implementation of digital IIR Butterworth low pass filter to meet the given specifications
12	Design and implementation of digital IIR Butterworth high pass filter to meet the given specifications

Suggested Learning Resources

Text Books	1. Proakis & Manolakis, "Digital Signal Processing - Principles Algorithms & Applications", 4th Edition, Pearson education, New Delhi, 2007. ISBN: 81-317-1000-9.
	2. Li Tan, Jean Jiang, "Digital Signal processing Fundamentals and Applications", Academic Press, 2013, ISBN: 978- 0-12-415893.
	3. Vinay K. Ingle, John G Proakis , "Digital Signal Processing Using MATLAB, A problem Solving Companion", Cengage Learning, 2018, ISBN: 93-86668-11-4
Reference Books	1. Simon Haykin and Barry Van Veen, "Signals and Systems", 2nd Edition, 2008, Wiley India. ISBN9971-51- 239-4.
	2. Sanjit K Mitra, "Digital Signal Processing, A Computer Based Approach", 4th Edition, McGraw Hill Education, 2017, ISBN:978-1-25-909858
	3. Oppenheim & Schaffer, "Discrete Time Signal Processing", PHI, 2003.
	4. D Ganesh Rao and Vineeth P Gejji, "Digital Signal Processing" Cengage India Private Limited, 2017, ISBN: 9386858231
Web links and Video Lectures (e-Resources):	**Digital Signal processing, https://nptel.ac.in/courses/117102060**

COURSE ASSESSMENT AND EVALUATION						
Direct Assessment Methods						
What		**To whom**	**When/Where (Frequency in the course)**	**Max. Marks**	**Evidence Collected**	**Contributing to Course outcomes**
CIE	Record & Observation	Students	Every lab session (Avg. of all experiment marks)	15	Observation book written at each lab + Record submitted at each lab + Viva	CO1 – CO5
	IA Test		one	10	Blue Books	CO1 – CO5

MAPPING OF COURSE OUTCOMES WITH PROGRAM OUTCOMES															
		PO												**PSO**	
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
CO	**1**	2	1	1		2								1	
	2	2	1	1		2								1	
	3	2	1	1		2								1	
	4	2	1	1		2								1	
	5	2	1	1		2								1	
	Sum	10	5	5		10								5	
	Avg	2	1	1		2								1	

3: High correlation, 2: Medium correlation, 1: Low correlation

Instructions to the Candidates

General Lab Guidelines:

- Conduct yourself in a responsible manner at all times in the laboratory. Intentional misconduct will lead to the exclusion from the lab.
- Do not wander around, or distract other students, or interfere with the laboratory experiments of other students.
- Read the handout and procedures before starting the experiments. Follow all written and verbal instructions carefully. If you do not understand the procedures, ask the instructor or teaching assistant.
- Attendance in all the labs is mandatory, absence permitted only with prior permission from Class teacher.
- The workplace has to be tidy before, during and after the experiment.
- Do not eat food, drink beverages or chew gum in the laboratory.
- Every student should know the location and operating procedures of all Safety equipment including First Aid Kit and Fire extinguisher.

DO'S: -

- Uniform and ID card are must.
- Strictly follow the procedures for conduction of experiments.
- Records have to be submitted every week for evaluation.
- Chairs and stools should be kept under the workbenches when not in use.
- After the lab session, switch off every supply, disconnect and disintegrate the experiments and return the components.
- Keep your belongings in designated area.
- Never use damaged instruments, wires or connectors. Hand these parts to the instructor/ teaching assistant.
- Sign the log book when you enter/leave the laboratory.

DONT'S: -

- Don't touch open wires unless you are sure that there is no voltage. Always disconnect the plug by pulling on the connector body not by the cable. Switch off the supply while you make changes to the experiment.
- Don't leave the experiment table unattended when the experimental setup supply is on.
- Students are not allowed to work in laboratory alone or without presence of the teaching staff/ instructor.
- No additional material should be carried by the students during regular labs.
- Avoid stepping on electrical wires or any other computer cables

Evaluation of DIGITAL SIGNAL PROCESSING LABORATORY (IPCC) (BEC502)	University weightage Marks
Lab internal: Conduction for 50 marks **1. Write up – 10M** **Discrete Experiments:** (Aim– 1M, Circuit – 3M, Design – 3M, Waveforms – 3M) **Programming Experiments:** (Aim– 1M, Flowchart – 3M, Program –6M) **2. Program Execution – 30M** (Program/Code Output – 25M, Comments – 3M, Optimization– 2M, (Partial output – 25M, No Output – 00M) **3. Results & Viva – 10M** (Identifying & Showing the inputs and outputs – 2M and/or theoretical calculations – 2M, Output Verification – 1M)	**10**
Record and Observation: **Conduction for 30 marks (10M record and 20 M Observation)** (Aim & Apparatus, Algorithm/flowchart (each experiment should have at least one flowchart, Calculations, Input/Output observations & Result (10+20=30M)	**15**
Total Marks	**25**

CONTENTS

INTRODUCTION TO MATLAB

MATLAB stands for Matrix Laboratory. It is a high-performance language that is used for technical computing. It was developed by Cleve Molar of the company MathWorks. INC in the year 1984.It is written in C, C++, Java. It allows matrix manipulations, plotting of functions, implementation of algorithms and creation of user interfaces. It is both a programming language as well as a programming environment. It allows the computation of statements in the command window itself.

The built-in functions of MATLAB offer top-notch resources for performing calculations, including optimization, linear algebra, numerical solution of ordinary differential equations (ODEs), data analysis, quadrate, signal processing, and many other scientific tasks. Modern algorithms are used for the majority of these functions. There are many of these for both animations and 2-D and 3-D graphics. MATLAB also supports an external interface.

The user can create their own functions in the MATLAB language. Thus, they are not restricted to using only the built-in functions. Additional toolboxes are provided by MATLAB. These toolboxes were created for common uses such as neural networks, symbolic computations, image processing, control system design, and statistics.

The various uses of MATLAB are: Developing algorithms, performing linear algebra that is linear, Graph plotting for larger data sets, Data visualization and analysis, Numerical Matrix Computation

MATLAB is generally used for these types of tasks:

- Signal processing
- Optimization of functions
- Control system design
- Image and Audio processing
- Machine learning and Deep learning

Features of MATLAB

- **MATLAB is a high-level language:** MATLAB Supports Object oriented programming. It also supports different types of programming constructs like Control flow statements (IF-ELSE, FOR, WHILE). MATLAB also supports structures like in C programming, Functional programming (writing functions to contain commonly used code and later calling them). It also contains Input / Output statements like disp() and input().

- **Interactive graphics:** MATLAB has inbuilt graphics to enhance user experience. We can actually visualize whatever data is there in forms of plots and figures. It also supports processing of image and displaying them in 2D or 3D formats. We can visualize and manipulate our data across any of the three dimensions (1D, 2D, and 3D). We can plot the functions and customize them also according to our needs like changing bullet points, line color and displaying/not displaying grid.

- **A large library of Mathematical functions:** MATLAB has a huge inbuilt library of functions required for mathematical analysis of any data. It has common math functions like sqrt. factorial etc. It has functions required for statistical analysis like median, mode and std (to find standard deviation), and much more. MATLAB also has functions for signal processing like filter, butter(Butterworth filter design) audio read, Conv, xcorr, fft, fftshift etc. It also supports image processing and some common functions required for image processing in MATLAB are rgb2gray, rgb2hsv, adaptthresh etc.

- **Data access and processing:** MATLAB allows accessing of data from external sources like image files (.jpg, .PNG), audio files (.mp), and real-time data from JDBC/ ODBC. We can easily read data from external sources using the inbuilt MATLAB functions like audioread for reading audio files and imread for reading external images.

- **Interactive environment:** MATLAB offers interactive environment by providing a GUI (Graphical user interface) and different types of tools like signal analyses and tuners. MATLAB also has tools for debugging and the development of any software. Importing and exporting files becomes easy in MATLAB through the GUI. We can view the workspace data as we progress in the development of our software and modify it according to our needs.

- **MATLAB can interface with different languages:** We can write a set of codes (libraries) in languages like PERL and JAVA, and we can call those libraries from within the MATLAB itself. MATLAB also supports ActiveX and .NET libraries.

- **MATLAB and Simulink :** MATLAB has an inbuilt feature of Simulink wherein we can model the control systems and see their real-time behavior. We can design any system either using code or building blocks and see their real-time working through various inbuilt tools. It has lucid examples of basic control systems and their working.

- **MATLAB's Application programming interface (API):** MATLAB consists of an extensive API. Through this API, we can link our C/C++ programs directly to MATLAB. Some options available in MATLAB API are calling MATLAB programs, read and write M-files, and using MATLAB as an interface to run applications. MATLAB can be used both as a computation and analysis tool.
- **Machine Learning, Deep Learning, and Computer vision:** The most demanding technologies like Machine learning, Deep learning, and Computer vision can be done in MATLAB. We can create and

interconnect layers of a deep neural network, We can build custom training loops and training layers with automatic differentiation. For machine learning, we can use the DBSCAN algorithm to discover clusters and noise in DATA. For computer vision, we can do object tracking, object recognition, gesture recognition, and processing 3D point clouds.

- **Computational Biology toolbox:** This toolbox provides a great way for biologists and researchers to create and analyze new algorithms and patterns for development in biological and biochemical domains. We can build biological models and analyze them using this toolbox. Moreover, for students, this toolbox can be very much educational if they want to explore the biological domain.

Writing a MATLAB Program

1. **Using Command Window:** Only one statement can be typed and executed at a time. It executes the statement when the enter key is pressed. This is mostly used for simple calculations. Note: ans is a default variable created by MATLAB that stores the output of the given computation.

2. **Using Editor:** Multiple lines of code can be written here and only after pressing the run button (or F5) will the code be executed. It is always a good practice to write clc, clear and close all in the beginning of the program.Note: Statements ending with a semicolon will not be displayed in the command window, however, their values will be displayed in the workspace. Any statement followed by % in MATLAB is considered as a comment

3. **Vector Operations:** Operations such as addition, subtraction, multiplication and division can be done using a single command instead of multiple loops

Basic Functions in MATLAB

Function	Description
disp()	The values or the text printed within single quotes is displayed on the output screen
clear	To clear all variables
close all	To close all graphics window

Function	Description
clc	To clear the command window
exp(x)	To compute the exponential value of x to the base e
abs(x)	To compute the absolute value of x
sqrt(x)	To compute the square root of x
log(x)	To compute the logarithmic value of x to the base e
log10(x)	To compute the logarithmic value of x to the base 10
rem(x, y)	To compute the remainder of x/y
sin(x)	To compute the sine of x
cos(x)	To compute the cosine of x
tan(x)	To compute the tangent of x

Advantages of MATLAB

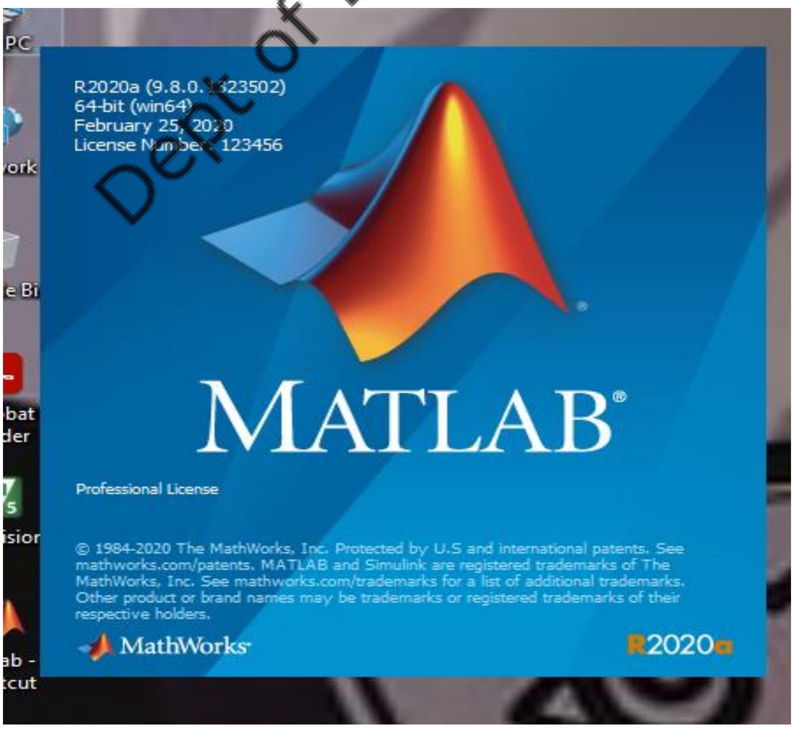
- **Easy to use interface:** A user-friendly interface with features you want to use is one click away.
- **A large inbuilt database of algorithms:** MATLAB has numerous important algorithms you want to use already built-in, and you just have to call them in your code.
- **Extensive data visualization and processing:** We can process a large amount of data in MATLAB and visualize them using plots and figures.
- **Debugging of codes easy:** There are many inbuilt tools like analyser and debugger for analysis and debugging of codes written in MATLAB.
- **Easy symbolic manipulation:** We can perform symbolic math operations in MATLAB using the symbolic manipulation algorithms and tools in MATLAB

Disadvantages of MATLAB

- MATLAB is slow since it is an interpreted language that is MATLAB programs are not converted into Machine language but are run by external software, so it can sometimes be slow.
- We cannot create the OUTPUT file in MATLAB.
- One cannot use graphics in MATLAB with -nojvm option, on doing so, we will get a runtime error.
- We cannot make functions in one single .m file as we have in the case of other programming languages. We have to create different files for different functions.
- Sometimes, the error messages are not much informative, so you have to figure out the error yourself.

PROCEDURE TO EXECUTE MATLAB PROGRAMS

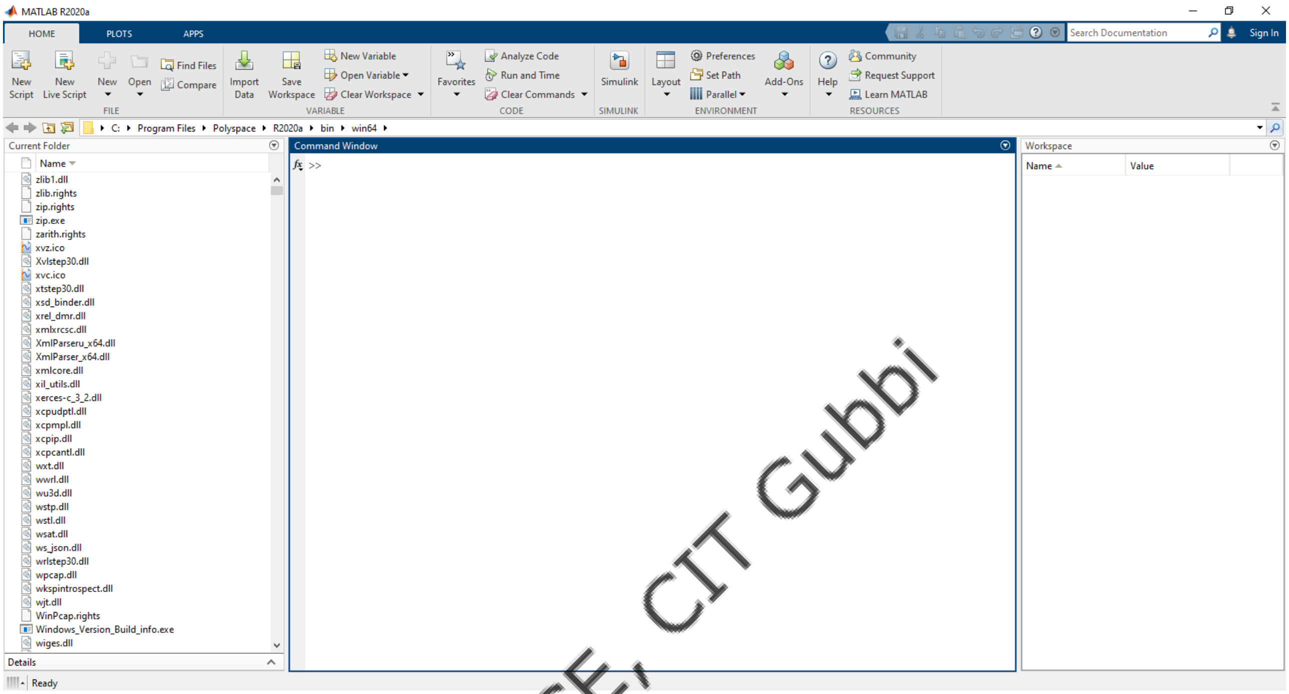
Step1: Double Click on MATLAB Icon on the Desktop



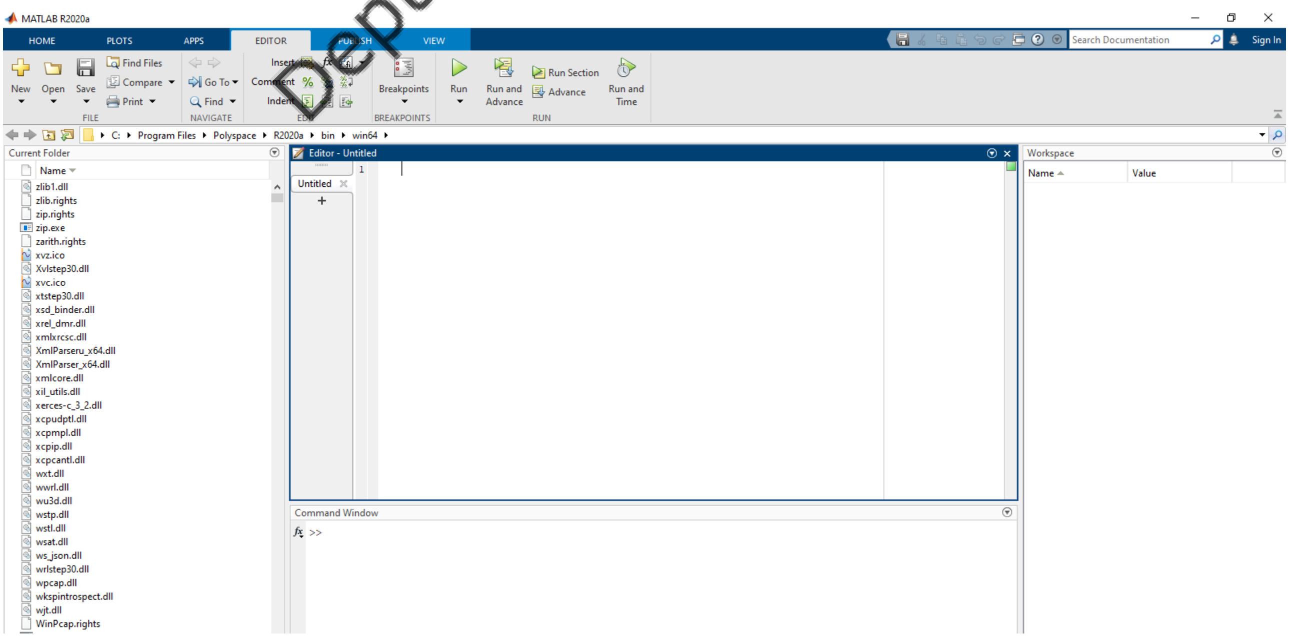
MATLAB Windows is open as follows:

it Contains three windows

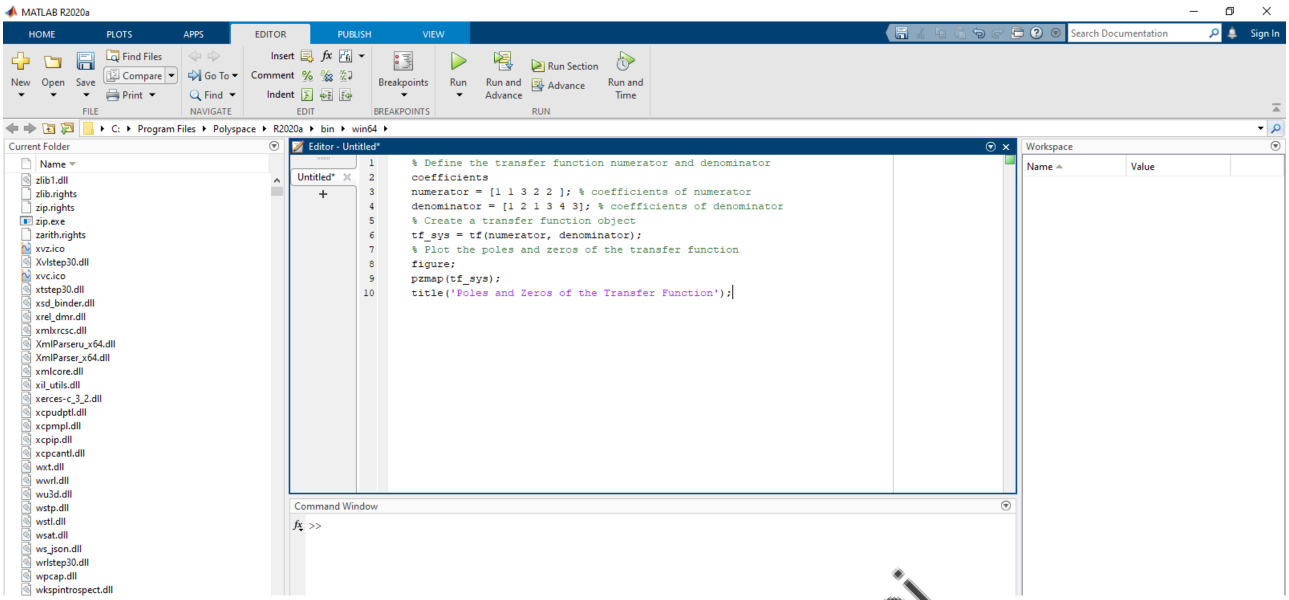
1. Current Folder Window
2. Command Window
3. Work Space



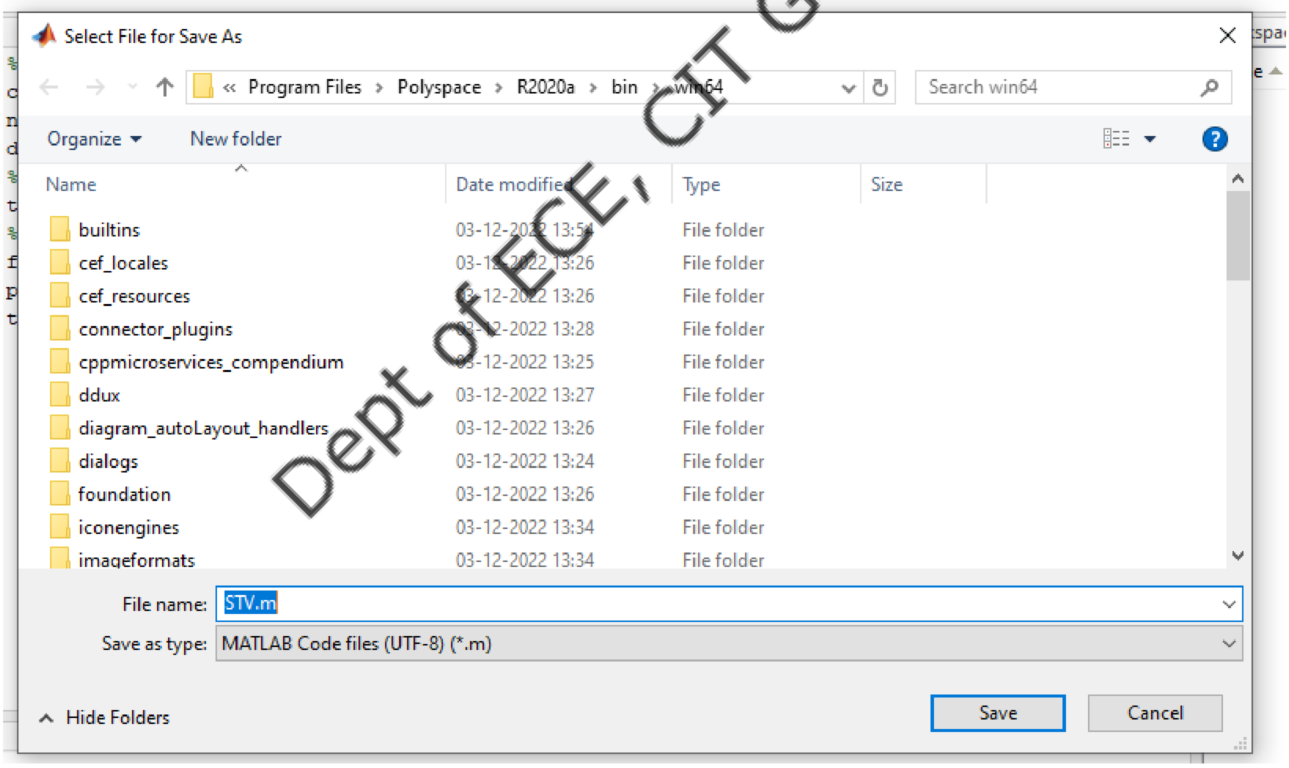
Step2: Click on New button and select M-File, An Editor window will open as follows



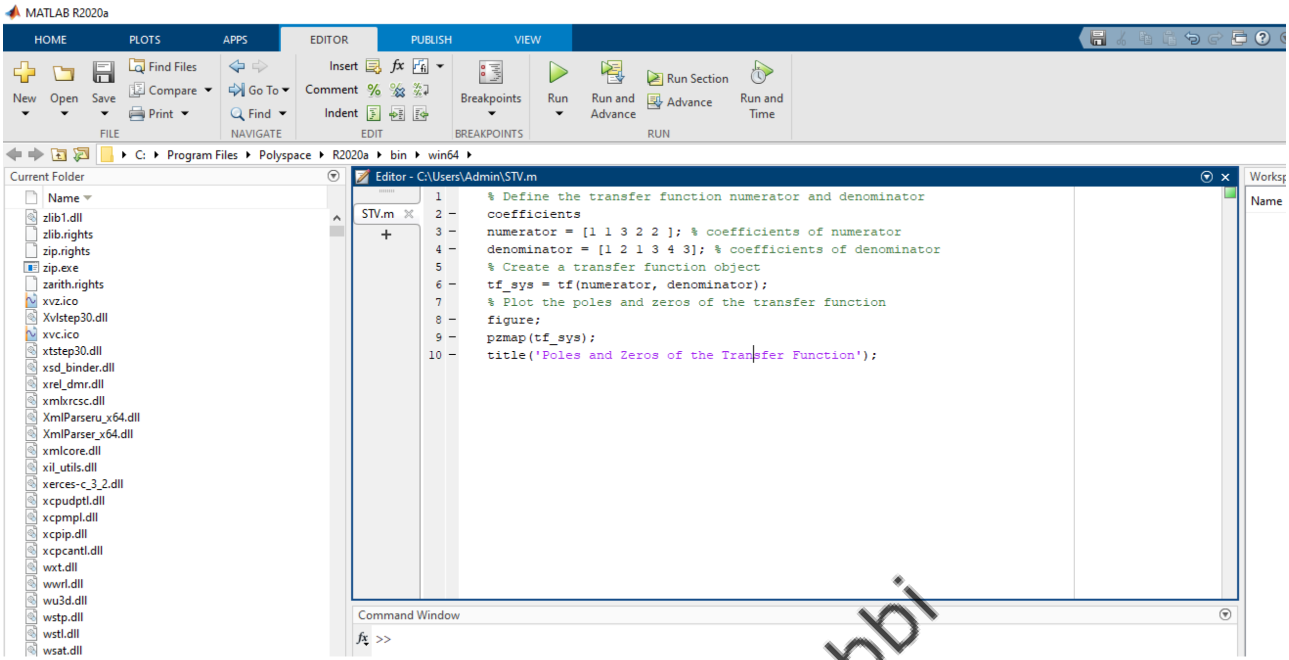
Step3: Type MATLAB Program in editor window



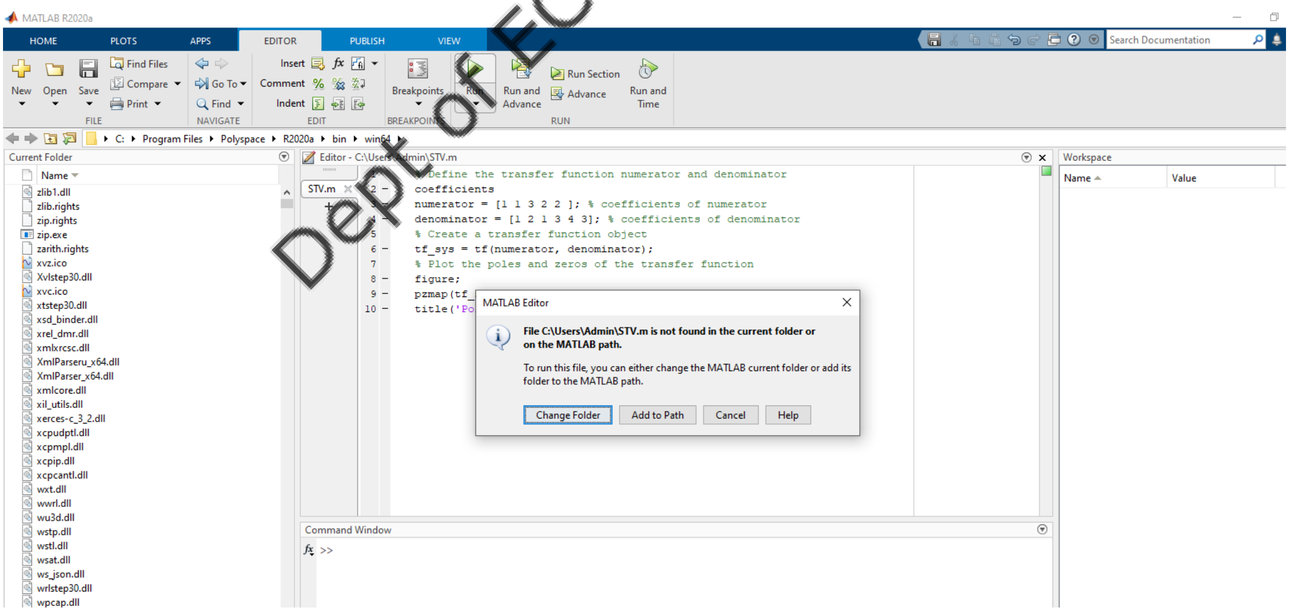
Step4: Save the MATLAB Program in a designated folder with .m Extension.



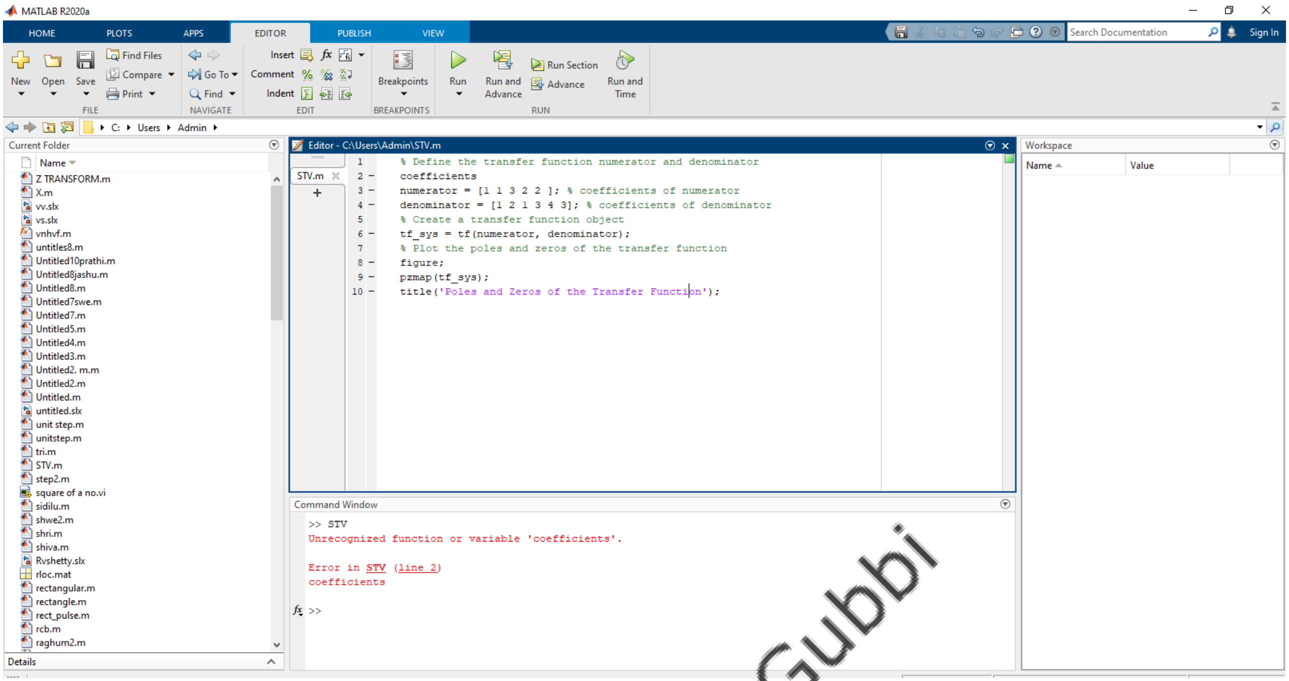
After Saving MATLAB Program Editor Window will look as follows.



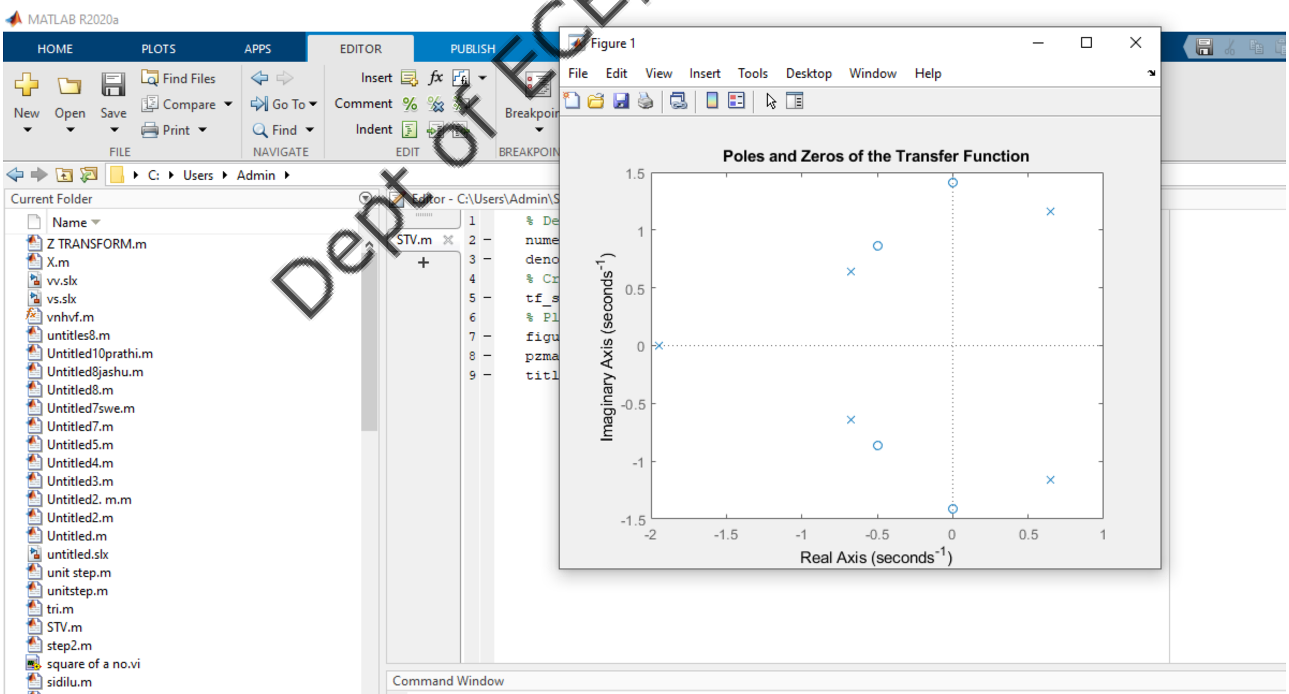
Step5: By Selecting Editor Button Click on RUN Button to Execute MATLAB Program Click on Change folder



Step6: If any error is present in MATLAB Program it will be displayed in Command window as follows. Correct the respective error.



Step7: After Correcting the error click on the RUN button again, MATLAB program will be Executed and result will be displayed in command window and respective result will be displayed.



EXPERIMENT NO 1

PROGRAM TO GENERATE THE FOLLOWING DISCRETE TIME SIGNALS.

a) UNIT SAMPLE SEQUENCE b) UNIT STEP SEQUENCE
c) EXPONENTIAL SEQUENCE d) SINUSOIDAL SEQUENCE
e) RANDOM SEQUENCE

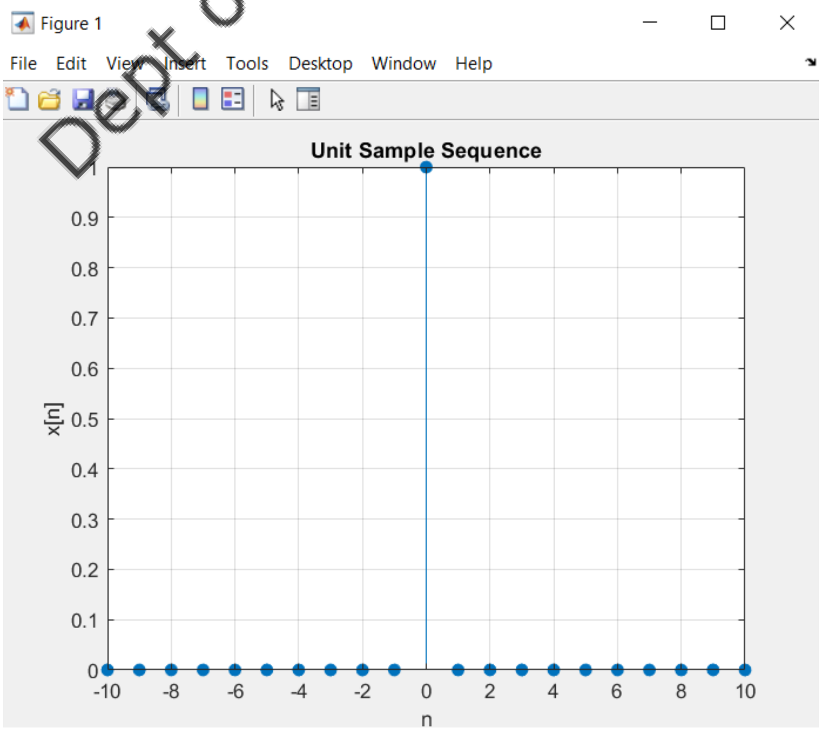
1a) Program To Generate Unit Sample Sequence

Program:

```
clc;
clear all;
N =input('enter the value of time constant'); % Length of sequence
n = -N/2:N/2; % Define the time index range
x = (n == 0);% Create the unit sample sequence (impulse function)
stem(n, x, 'filled');
xlabel('n');
ylabel('x[n]');
title('Unit Sample Sequence');
grid on;
```

Result 1a:

enter the value of time constant20



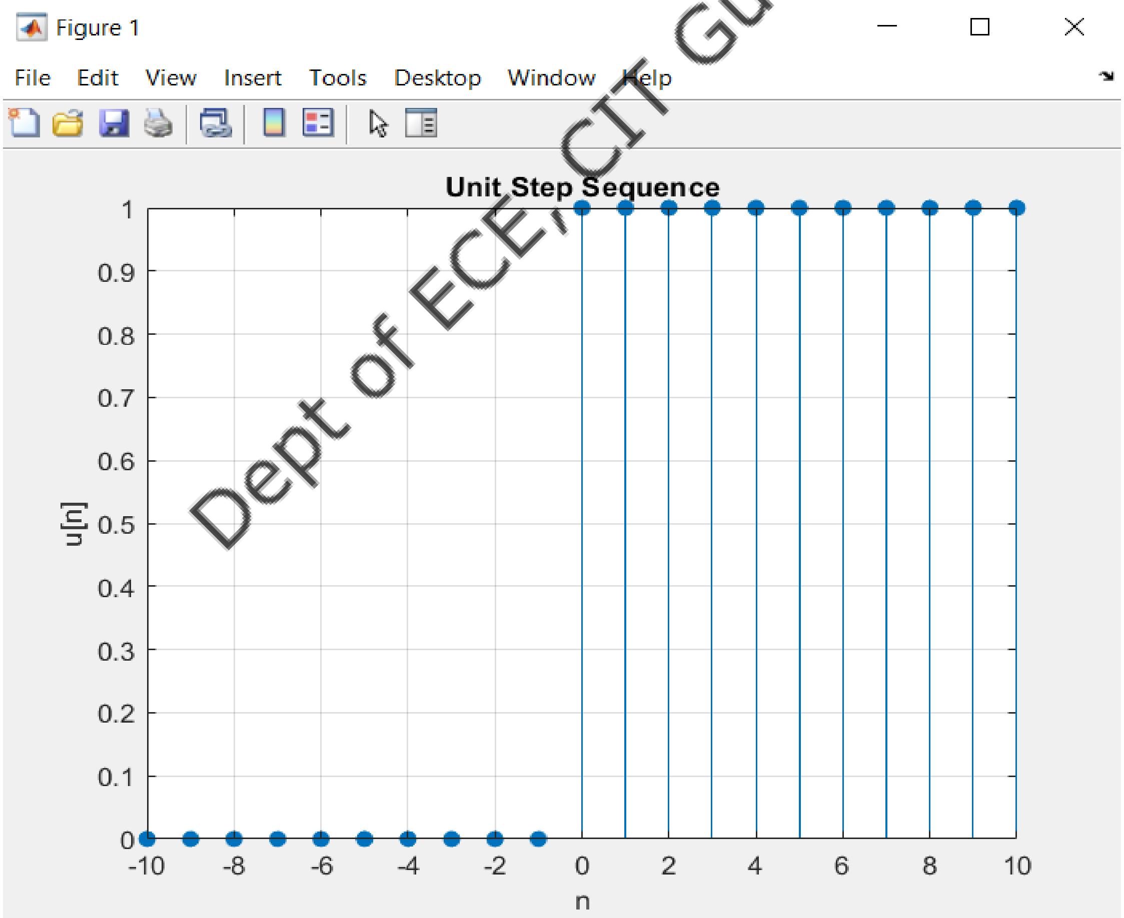
1b) Program To Generate Unit Step Sequence

Program:

```
clc;
clear all;
N =input('enter the value of time constant'); % Length of sequence
n = -N/2:N/2; % Define the time index range
u = (n >= 0);% Create the unit step sequence
stem(n, u, 'filled');
xlabel('n');
ylabel('u[n]');
title('Unit Step Sequence');
grid on;
```

Result 1b:

enter the value of time constant20



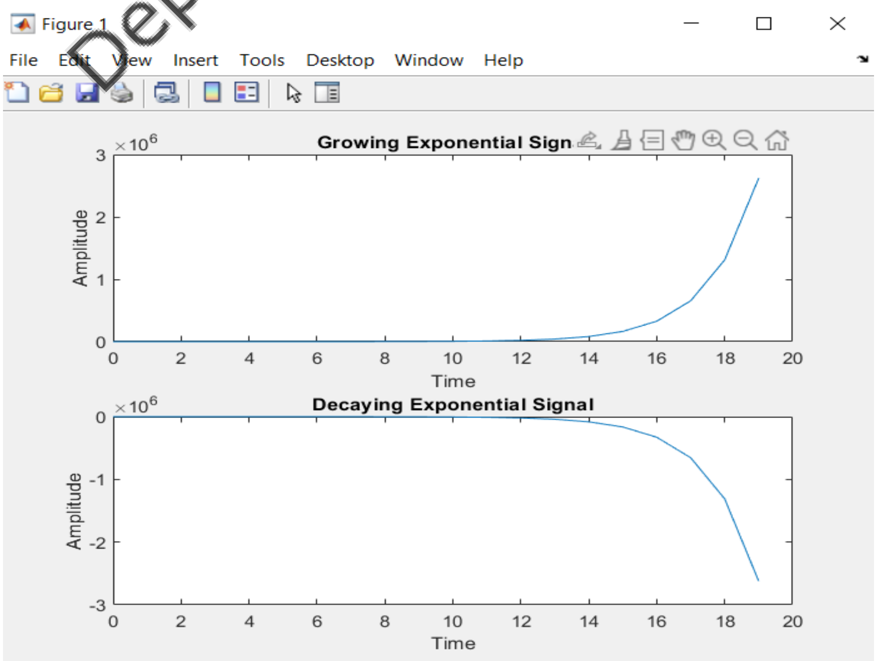
1c) Program To Generate Exponential Sequence

Program:

```
clc;
clear all;
N =input('enter the value of time constant'); % Length of sequence
n = 0:N-1; % Time index ranging from 0 to N-1
A = input('enter the value of a Amplitude');
a = input('enter the value of a Growth constant');
b = input('enter the value of a Decay constant');
% Define the exponential factor
x = A * a.^n; % Growing signal
y = A * -b.^n;% Decaying signal
% Plot the generated exponential signals
figure;
subplot(2, 1, 1);
plot(n, x);
xlabel('Time');
ylabel('Amplitude');
title('Growing Exponential Signal');
subplot(2, 1, 2);
plot(n, y);
xlabel('Time');
ylabel('Amplitude');
title('Decaying Exponential Signal');
```

Result 1c:

enter the value of time constant20
enter the value of a Amplitude5
enter the value of a Growth constant2
enter the value of a Decay constant2



1d) Program To Generate Random Sequence

Program:

```
clc;
clear all;
N =input('enter the value Length of the random sequence');
x = rand(1, N);  % Generates a 1xN array of random numbers between
0 and 1
stem(x, 'filled');
title('Random Sequence');
xlabel('Index');
ylabel('x[n]');
grid on;
```

Result 1d:

enter the value Length of the random sequence20

EXPERIMENT NO 2

PROGRAM TO PERFORM THE FOLLOWING OPERATIONS ON SIGNALS.

a) SIGNAL ADDITION b) SIGNAL MULTIPLICATION c) SCALING d) SHIFTING, e) FOLDING

2a) Program To Perform the Signal Addition operation

Program:

```
clc;
clear all;
x =input('enter the value of first signal Sequence');
y =input('enter the value of Second signal Sequence');
z=x+y;
figure;
subplot(3, 1, 1);
stem(x);
xlabel('Time');
ylabel('Amplitude');
title('First Signal');
subplot(3, 1, 2);
stem(y);
xlabel('Time');
ylabel('Amplitude');
title('Second Signal');
subplot(3, 1, 3);
stem(z);
xlabel('Time');
ylabel('Amplitude');
title('Addition of Two Signal Z=X+Y');
```


Result 2a:

enter the value of first signal Sequence [1 2 3 4]

enter the value of Second signal Sequence [1 1 1 1]

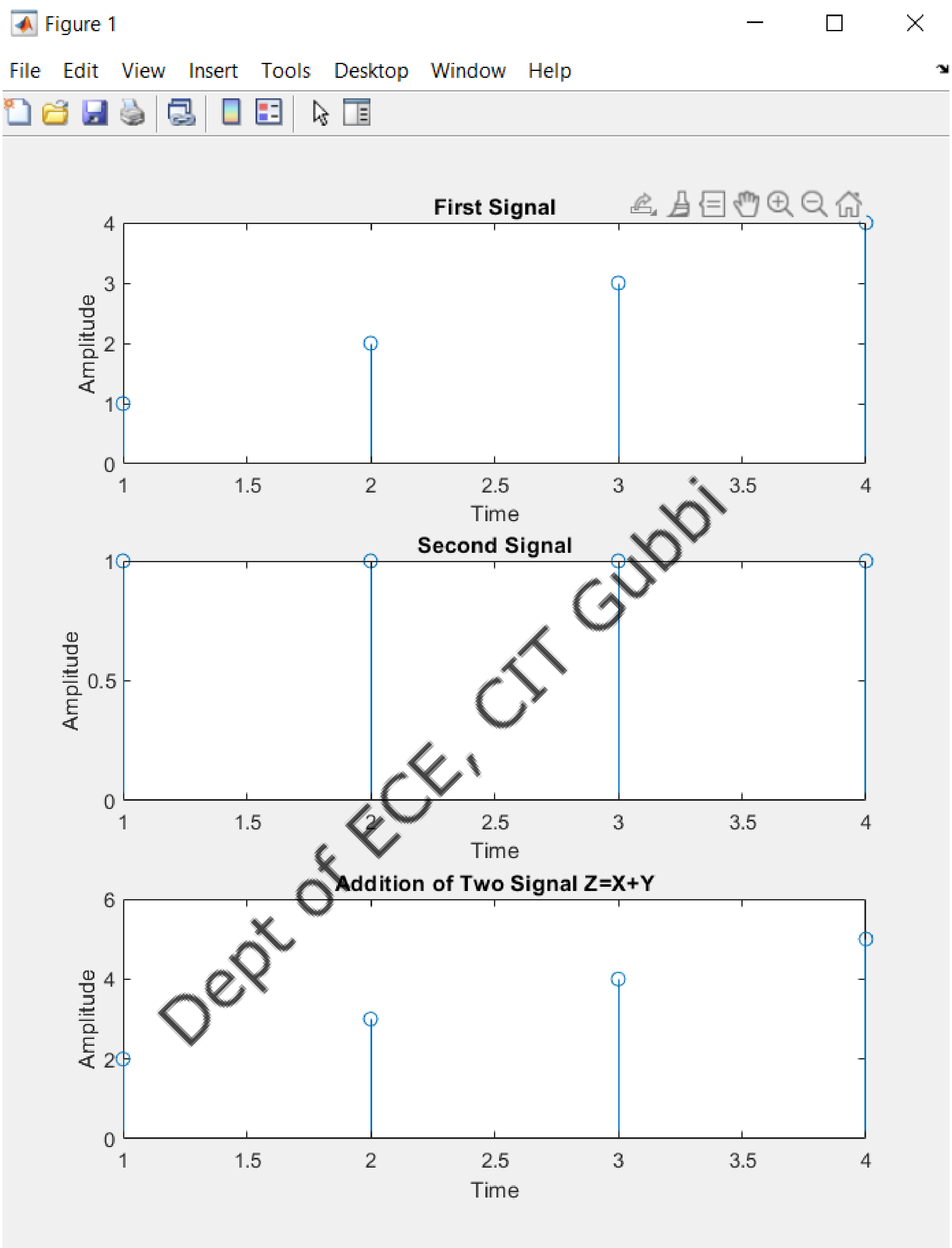


Figure 1
File Edit View Insert Tools Desktop Window Help
First Signal
Amplitude
Time
Second Signal
Amplitude
Time
Addition of Two Signal Z=X+Y
Amplitude
Time

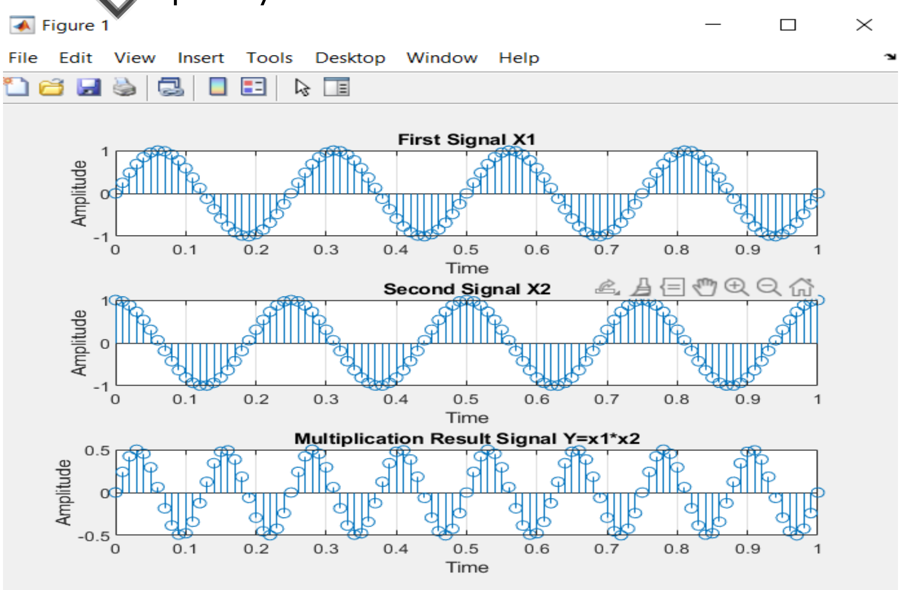
2b) Program To Perform the Multiplication operation on Signals

Program:

```
clc;
clear all;
T=input('enter the length of Sequence');
t=0:0.01:T-1;
f=input('enter the value of Frequency');
t1=2*pi*f*t;
x1=sin(t1);
subplot(3,1,1)
stem(t,x1)
title('First Signal X1')
xlabel('Time');
ylabel('Amplitude')
grid on;
x2=cos(t1);
subplot(3,1,2)
stem(t,x2)
title('Second Signal X2')
xlabel('Time');
ylabel('Amplitude')
grid on;
y=x1.*x2; % Multiplication
subplot(3,1,3)
stem(t,y)
title('Multiplication Result Signal Y=x1*x2')
xlabel('Time')
ylabel('Amplitude')
grid on;
```


Result 2b:

enter the length of Sequence2
enter the value of Frequency4



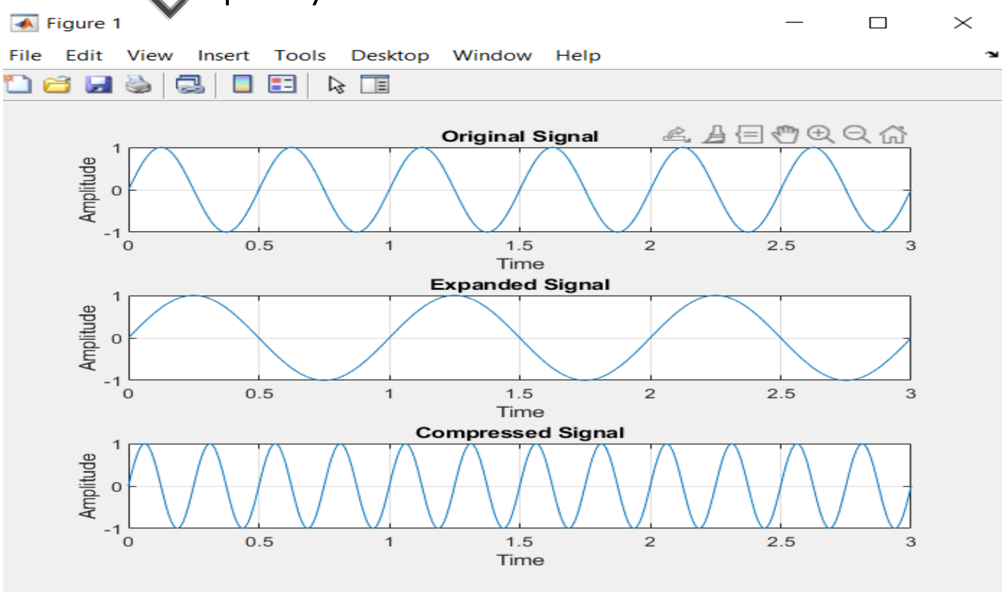
2c) Program To Perform the Scaling operation on Signals.

Program:

```
clc;
clear all;
T=input('enter the length of Sequence');
t=0:0.01:T-1;
f=input('enter the value of Frequency');
t1=2*pi*f*t;
x=sin(t1);
subplot(3,1,1)
plot(t,x)
title('Original Signal')
xlabel('Time')
ylabel('Amplitude')
grid on;
y=sin(t1/2);
subplot(3,1,2)
plot(t,y)
title('Expanded Signal')
xlabel('Time')
ylabel('Amplitude')
grid on;
z=sin(t1*2);
subplot(3,1,3)
plot(t,z)
title('Compressed Signal')
xlabel('Time')
ylabel('Amplitude')
grid on;
```


Result 2c:

enter the length of Sequence4
enter the value of Frequency2



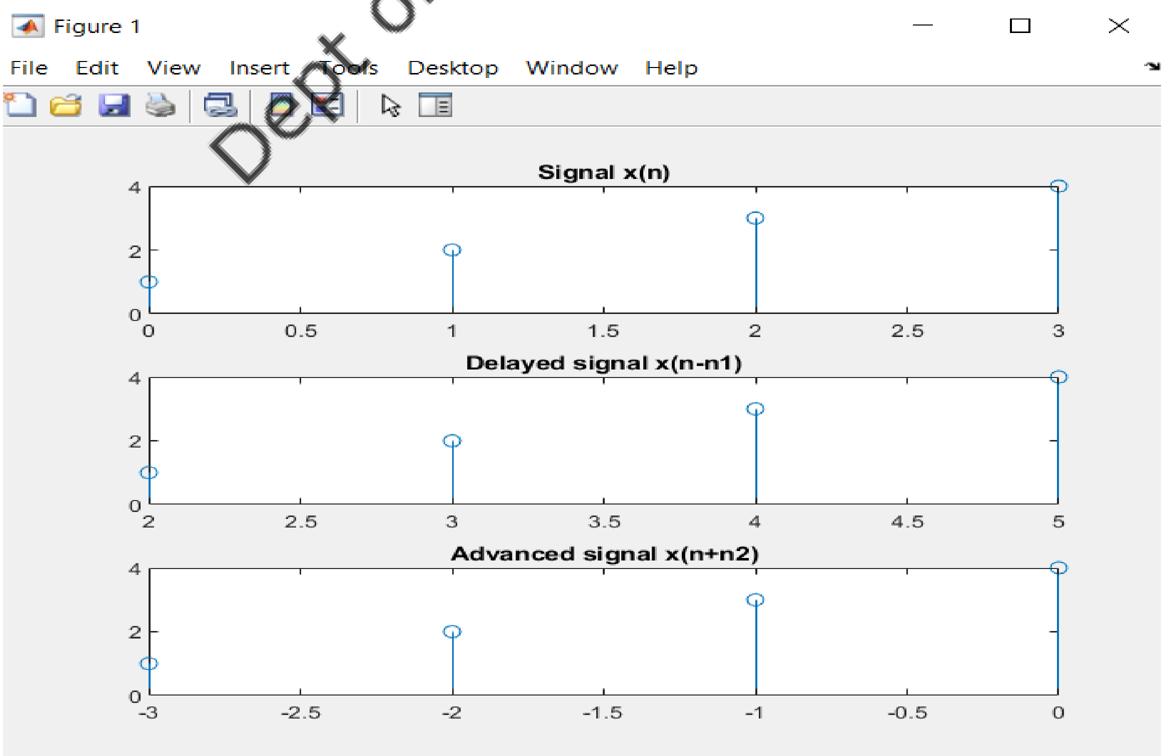
2d) Program To Perform the Shifting operation on Signals.

Program:

```
clc;
clear all;
N=input('Enter the length of Sequence');
x =input('Enter the value of Signal Sequence of Length N');
n1=input('Enter the amount to be delayed');
n2=input('Enter the amount to be advanced');
n=0:N-1;
subplot(3,1,1);
stem(n,x);
title('Signal x(n)');
m=n+n1;
y=x;
subplot(3,1,2);
stem(m,y);
title('Delayed signal x(n-n1)');
t=n-n2;
z=x;
subplot(3,1,3);
stem(t,z);
title('Advanced signal x(n+n2)');
```

Result 2d:

Enter the length of Sequence4
Enter the value of Signal Sequence of Length N[1 2 3 4]
Enter the amount to be delayed2
Enter the amount to be advanced3



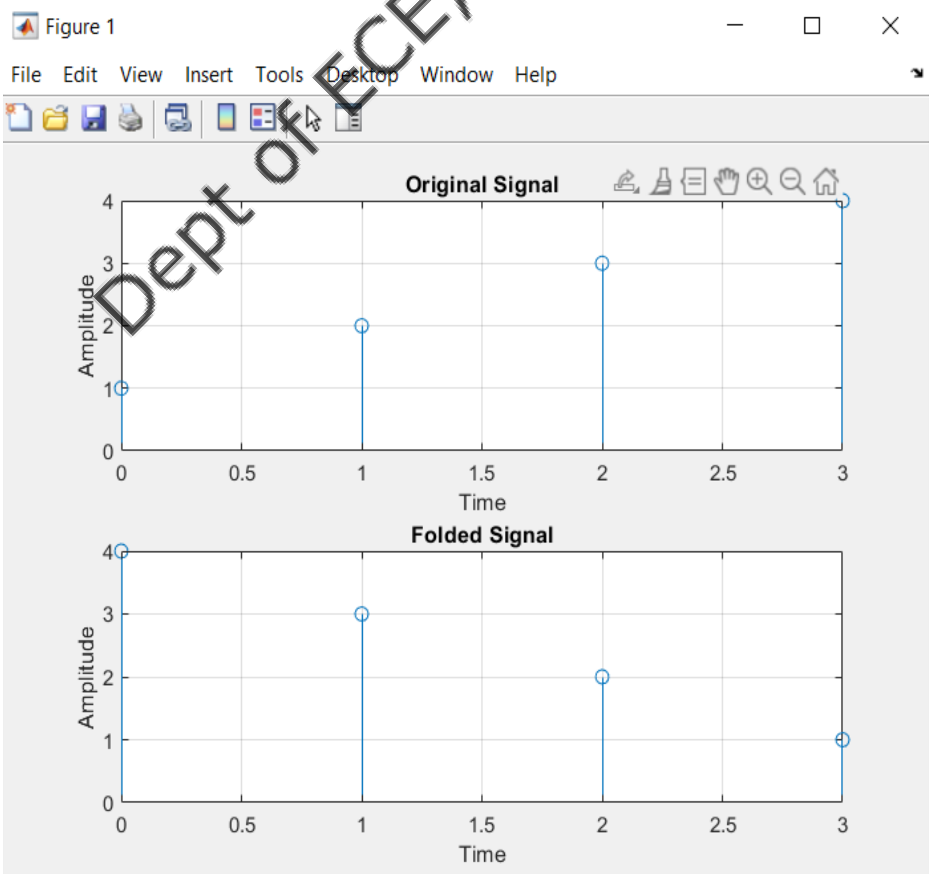
2e) Program to Perform the Folding operation on Signals.

Program:

```
clc;
clear all;
N=input('Enter the length of Sequence');
x =input('Enter the value of Signal Sequence of Length N');
n=0:N-1;
subplot(2,1,1)
stem(n,x)
title('Original Signal')
xlabel('Time')
ylabel('Amplitude')
grid on;
y=fliplr(x);
subplot(2,1,2)
stem(n,y)
title('Folded Signal')
xlabel('Time')
ylabel('Amplitude')
grid on;
```

Result 2e:

Enter the length of Sequence4
Enter the value of Signal Sequence of Length N[1 2 3 4]



EXPERIMENT NO 3

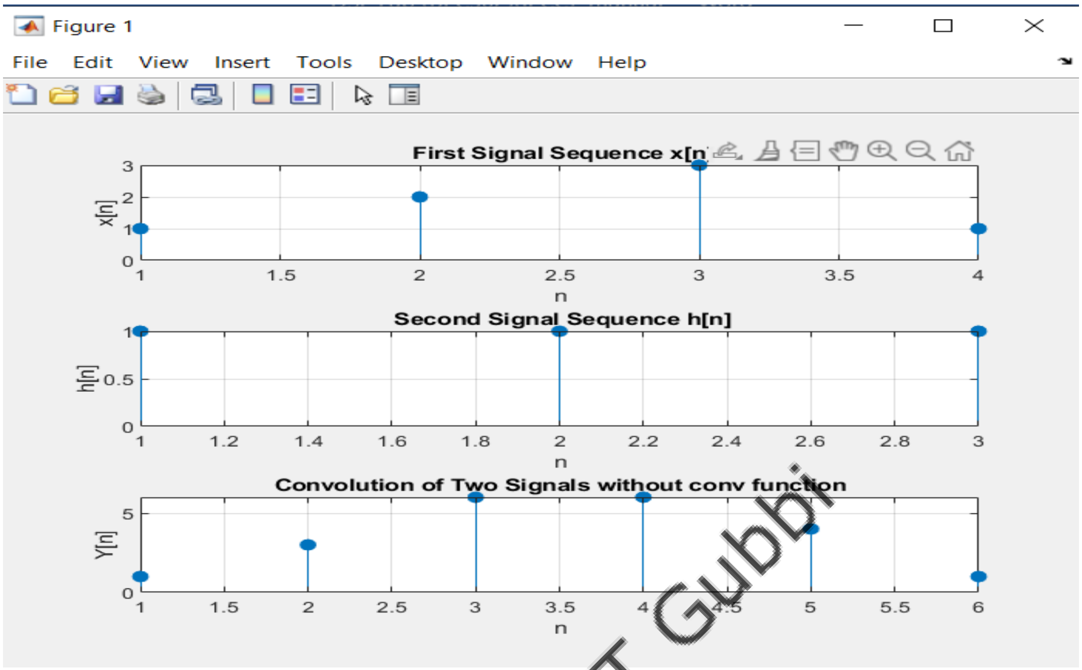
PROGRAM TO PERFORM CONVOLUTION OF TWO GIVEN SEQUENCES (WITHOUT USING BUILT-IN FUNCTION) AND DISPLAY THE SIGNALS

Program:

```
clc;
clear all;
x =input('Enter the value of First Signal Sequence x[n]');
h =input('Enter the value of Second Signal Sequence h[n]');
% convolution
m=length(x);
n=length(h);
X=[x,zeros(1,n)];
H=[h,zeros(1,m)];
for i=1:n+m-1
    Y(i)=0;
    for j=1:m
        if(i-j+1>0)
            Y(i)=Y(i)+X(j)*H(i-j+1);
        else
        end
    end
end

figure;
subplot(3,1,1);
stem(x, 'filled');
xlabel('n');
ylabel('x[n]');
title(' First Signal Sequence x[n]');
grid on;

subplot(3,1,2);
stem(h, 'filled');
xlabel('n');
ylabel('h[n]');
title(' Second Signal Sequence h[n]');
grid on;

subplot(3,1,3);
stem(Y,'filled');
ylabel('Y[n]');
xlabel('n');
grid on;
title('Convolution of Two Signals without conv function');
```

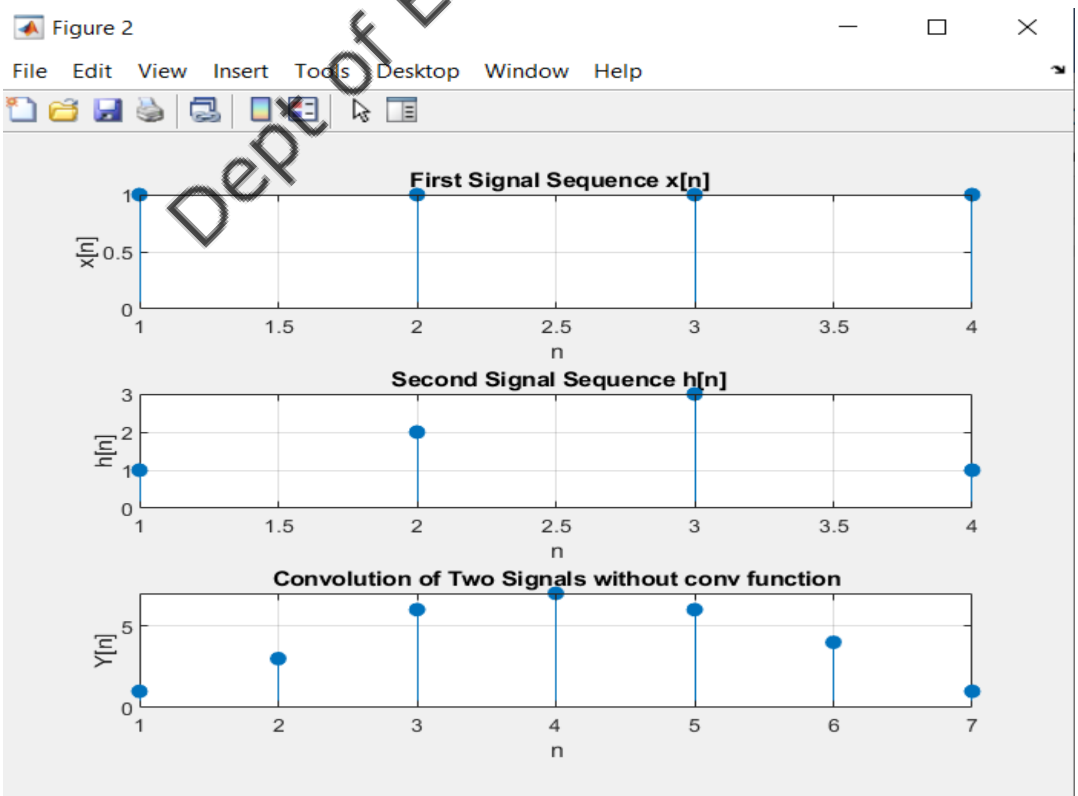
Result 3a:

Enter the value of First Signal Sequence x[n][1 2 3 1]
Enter the value of Second Signal Sequence h[n][1 1 1]



Result 3b:

Enter the value of First Signal Sequence x[n][1 1 1 1]
Enter the value of Second Signal Sequence h[n][1 2 3 1]



EXPERIMENT NO 4

CONSIDER A CAUSAL SYSTEM y(n) = 0.9y(n-1) +x(n).
a) DETERMINE H(z) & SKETCH ITS POLE ZERO PLOT.
b) PLOT $|H(e^{j\omega})|$ AND $\angle H(e^{j\omega})$
c) DETERMINE THE IMPULSE RESPONSE h(n).

Program:

```
clc;
clear all;
numerator = [1]; % Numerator of H(z)
denominator = [1 -0.9]; % Denominator of H(z)

% Create a pole-zero plot
figure;
zplane(numerator, denominator);
title('Pole-Zero Plot of H(z)');
omega = linspace(-pi, pi, 1000); % Define frequency range
H = exp(1j * omega) ./ (exp(1j * omega) - 0.9);% Calculate H(e^j?)
magnitude_H = abs(H);% Magnitude response
phase_H = angle(H);% Phase response

% Plot Magnitude Response
figure;
plot(omega, magnitude_H);
title('|H(e^{j\omega})|');
xlabel('\omega');
ylabel('|H(e^{j\omega})|');

% Plot Phase Response
figure;
plot(omega, unwrap(phase_H));
title('\angle H(e^{j\omega})');
xlabel('\omega');
ylabel('\angle H(e^{j\omega})');

% Define the impulse response
n = 0:50; % Time index
h = (0.9 .^ n); % Impulse response

% Plot impulse response
figure;
stem(n, h, 'filled');
title('Impulse Response h(n)');
xlabel('n');
ylabel('h(n)');
```

Result :

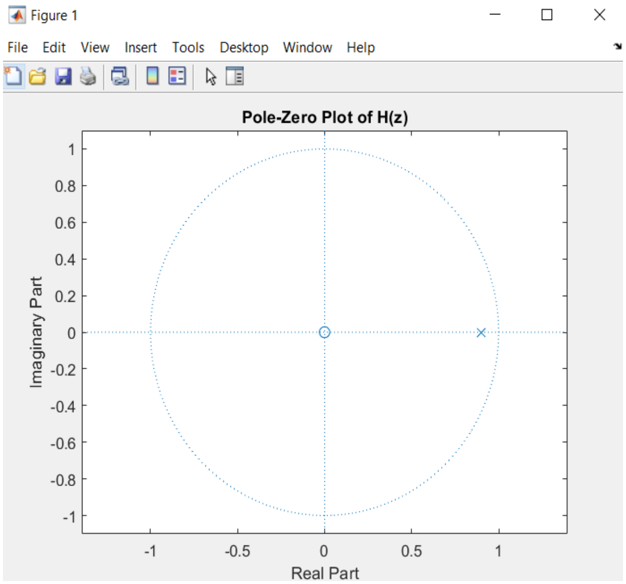


Figure 1
Pole-Zero Plot of H(z)
Imaginary Part
Real Part

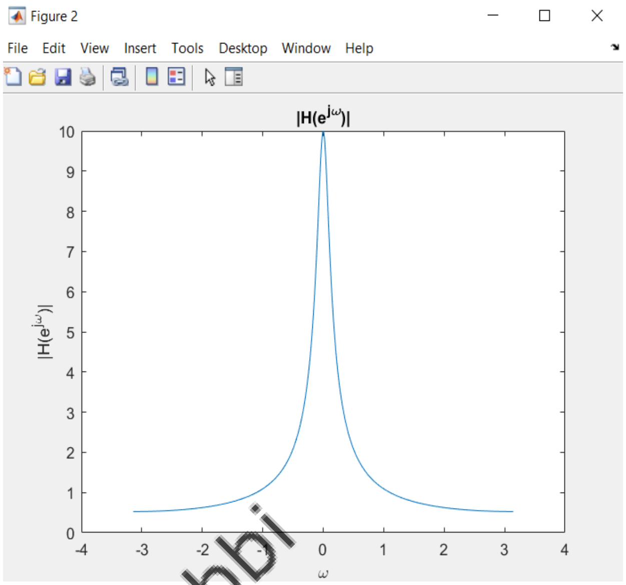


Figure 2
$|H(e^{j\omega})|$
$|H(e^{j\omega})|$
ω

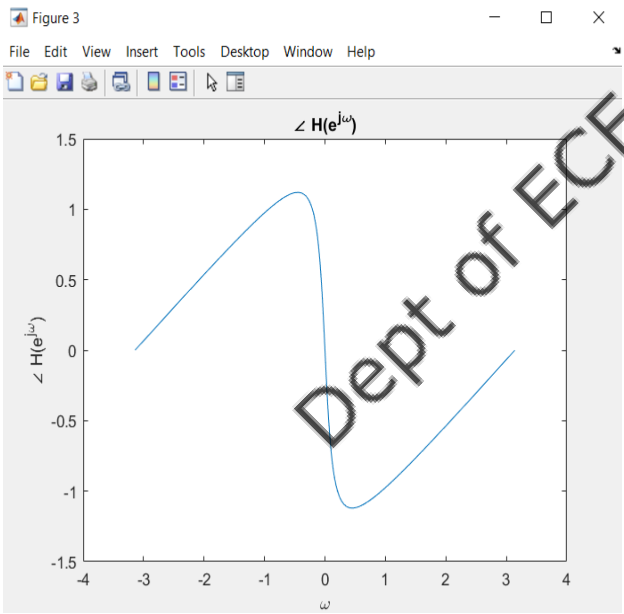


Figure 3
$\angle H(e^{j\omega})$
$\angle H(e^{j\omega})$
ω

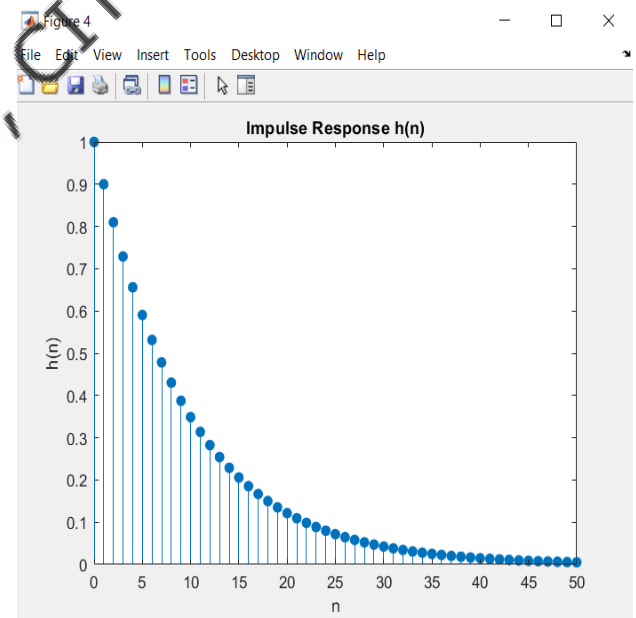


Figure 4
Impulse Response h(n)
h(n)
n

EXPERIMENT NO 5

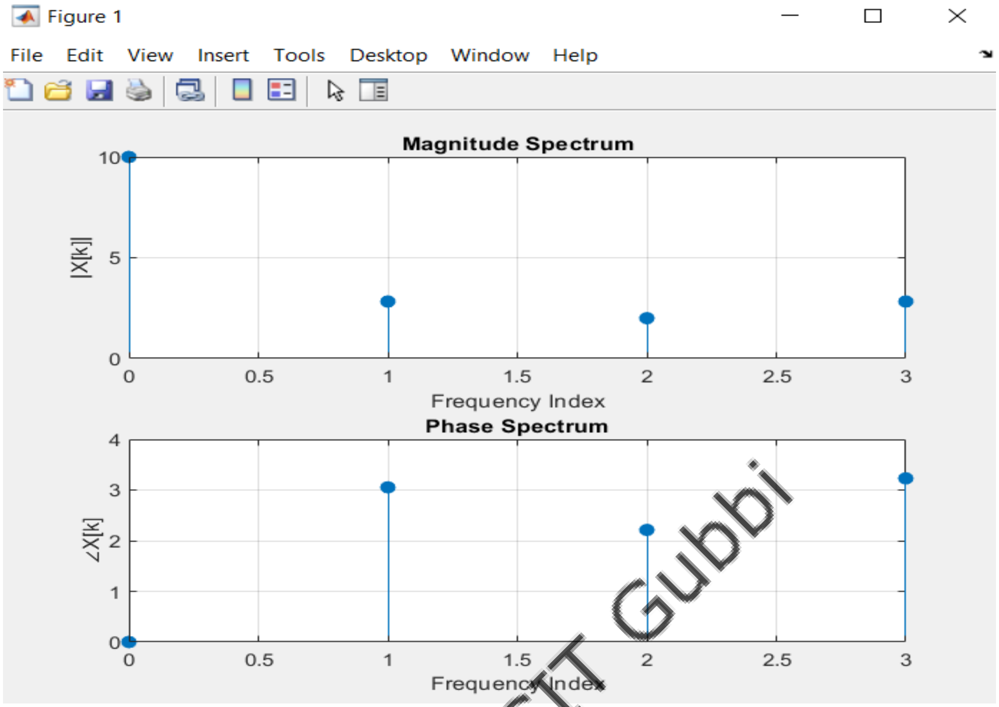
COMPUTATION OF N POINT DFT OF A GIVEN SEQUENCE (WITHOUT USING BUILT-IN FUNCTION) AND TO PLOT THE MAGNITUDE & PHASE SPECTRUM.

Program:

```
clc;
clear all;
x = input('Enter the value of Signal Sequence x[n]');
N = length(x); % Length of the sequence
X = zeros(1, N);

% Compute the N-point DFT
for k = 0:N-1
    sum = 0;
    for n = 0:N-1
        % Compute the DFT using the formula
        sum = sum + x(n+1) * exp(-1j * 2 * pi * k * n / N);
    end
    X(k+1) = sum; % Store the result
end

% Compute magnitude and phase spectra
magnitude_X = abs(X);
phase_X = angle(X);

% Plot the magnitude spectrum
figure;
subplot(2, 1, 1);
stem(0:N-1, magnitude_X, 'filled');
title('Magnitude Spectrum');
xlabel('Frequency Index');
ylabel('|X[k]|');
grid on;

% Plot the phase spectrum
subplot(2, 1, 2);
stem(0:N-1, unwrap(phase_X), 'filled');
title('Phase Spectrum');
xlabel('Frequency Index');
ylabel('?X[k]');
grid on;
```

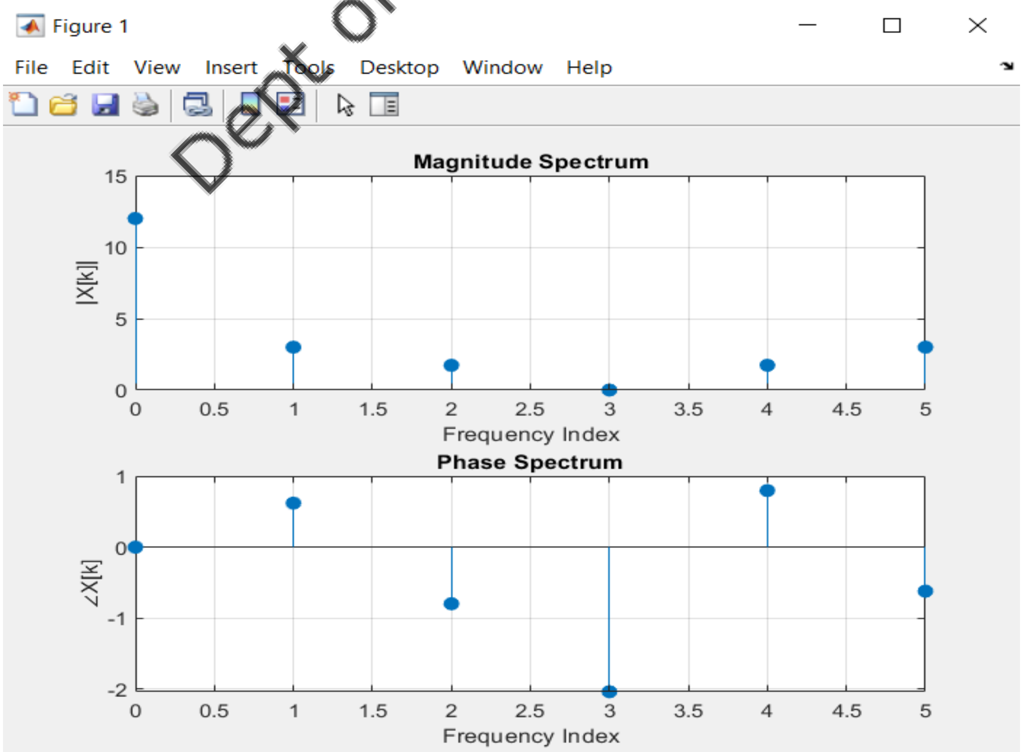
Result 5a:

Enter the value of Signal Sequence x[n][1 2 3 4]
magnitude_X = 10.0000 2.8284 2.0000 2.8284
phase_X = 0 135.0000 -180.0000 -135.0000



Result 5b:

Enter the value of Signal Sequence x[n][1 1 2 2 3 3]
magnitude_X = 12.0000 3.0000 1.7321 0.0000 1.7321 3.0000
phase_X = 0 120.0000 150.0000 -90.0000 -150.0000 -120.0000



EXPERIMENT NO 6

USING THE DFT AND IDFT, COMPUTE THE FOLLOWING FOR ANY TWO GIVEN SEQUENCES
a) CIRCULAR CONVOLUTION
b) LINEAR CONVOLUTION

6a) Program to compute circular convolution

Program:

```
clc;
clear all;
x = input('Enter the value of Signal Sequence x[n]');
h = input('Enter the value of Signal Sequence h[n]');
N = max(length(x), length(h));
x_padded = [x, zeros(1, N - length(x))];
h_padded = [h, zeros(1, N - length(h))];
% Compute the DFT of both sequences
X = fft(x_padded)
H = fft(h_padded)
Y = X .* H % Multiply the DFTs (element-wise)
y = ifft(Y);% Compute the IDFT to get the circular convolution
disp('Circular Convolution Result: y');
disp(y);
```


Result 6a1:

Enter the value of Signal Sequence x[n][1 2 3 4]
Enter the value of Signal Sequence h[n][1 1 1 1]
X =10.0000 + 0.0000i -2.0000 + 2.0000i -2.0000 + 0.0000i -2.0000 - 2.0000i
H = 4 0 0 0
Y = 40 0 0 0
Circular Convolution Result:
10 10 10 10

Result 6a2:

Enter the value of Signal Sequence x[n][2 3 1 1]
Enter the value of Signal Sequence h[n][1 3 5 3]
X =7.0000 + 0.0000i 1.0000 - 2.0000i -1.0000 + 0.0000i 1.0000 + 2.0000i
H = 12 -4 0 -4
Y =84.0000 + 0.0000i -4.0000 + 8.0000i 0.0000 + 0.0000i -4.0000 - 8.0000i
Circular Convolution Result: y
19 17 23 25

6b) Program to compute linear convolution

Program:

```
clc;
clear all;
x = input('Enter the value of Signal Sequence x[n]');
h = input('Enter the value of Signal Sequence h[n]');
Lx = length(x);
Lh = length(h);
N = Lx + Lh - 1; % Length of the linear convolution result
x_padded = [x, zeros(1, N - Lx)];
h_padded = [h, zeros(1, N - Lh)];
% Compute the DFT of both sequences
X = fft(x_padded)
H = fft(h_padded)
Y = X .* H % Multiply the DFTs (element-wise)
y = ifft(Y); % Compute the IDFT to get the linear convolution
disp('Linear Convolution Result: y');
disp(y);
```

Result 6b:

```
Enter the value of Signal Sequence x[n][1 2 3 4]
Enter the value of Signal Sequence h[n][2 3 1 1]
X =
  Columns 1 through 5

  10.0000 + 0.0000i  -2.0245 - 6.2240i   0.3460 + 2.4791i   0.1784 -
2.4220i   0.1784 + 2.4220i

  Columns 6 through 7

   0.3460 - 2.4791i  -2.0245 + 6.2240i
H =
  Columns 1 through 5

   7.0000 + 0.0000i   2.7470 - 3.7543i   1.0550 - 1.7091i  -0.3019 -
1.4947i  -0.3019 + 1.4947i

  Columns 6 through 7

   1.0550 + 1.7091i   2.7470 + 3.7543i
Y =
  Columns 1 through 5

  70.0000 + 0.0000i -28.9279 - 9.4967i   4.6020 + 2.0240i  -3.6741 +
0.4646i  -3.6741 - 0.4646i

  Columns 6 through 7

   4.6020 - 2.0240i -28.9279 + 9.4967i

Linear Convolution Result: y
    2.0000    7.0000   13.0000   20.0000   17.0000    7.0000    4.0000
```

EXPERIMENT NO 7

VERIFICATION OF LINEARITY PROPERTY, CIRCULAR TIME SHIFT PROPERTY AND CIRCULAR FREQUENCY SHIFT PROPERTY OF DFT.

7a) Verification of Linearity property of DFT

Linearity property: DFT {a.x1(n)+b.x2(n)} = a.X1(K)+ b.X2(K)

Program:

```
clc;
clear all;
x1 = input('Enter the value of Signal Sequence x1[n]');
x2 = input('Enter the value of Signal Sequence x2[n]');

a = input('Enter the value of constants a ');
b = input('Enter the value of constants b ');

% Compute the DFTs of x1 and x2
X1 = fft(x1);
X2 = fft(x2);

% Compute the linear combination of x1 and x2
x_combined = a * x1 + b * x2;

% Compute the DFT of the linear combination
X_combined = fft(x_combined);

% Compute the linear combination of the DFTs
X_combined_direct = a * X1 + b * X2;

% Display results
disp('DFT of Linear Combination:');
disp(X_combined);

disp('Linear Combination of DFTs:');
disp(X_combined_direct);

% Verify if they are approximately equal
if isequal(round(X_combined, 10), round(X_combined_direct,
10))
disp('Linearity property is verified: DFT(a*x1[n] + b*x2[n])
equals a*X1[k] + b*X2[k].');
else
    disp('Linearity property is not verified.');
end
```

Result 7a1:

Enter the value of Signal Sequence x1[n][1 2 3 4]
Enter the value of Signal Sequence x2[n][2 3 1 1]
Enter the value of constants a 2
Enter the value of constants b 3
DFT of Linear Combination:
  41.0000 + 0.0000i -1.0000 - 2.0000i -7.0000 + 0.0000i -1.0000 + 2.0000i

Linear Combination of DFTs:
  41.0000 + 0.0000i -1.0000 - 2.0000i -7.0000 + 0.0000i -1.0000 + 2.0000i

Linearity property is verified: DFT(a*x1[n] + b*x2[n]) equals a*X1[k] + b*X2[k].

Result 7a2:

Enter the value of Signal Sequence x1[n][1 2 3 4]
Enter the value of Signal Sequence x2[n][1 1 1 1]
Enter the value of constants a 5
Enter the value of constants b 10
DFT of Linear Combination:
  90.0000 + 0.0000i -10.0000 +10.0000i -10.0000 + 0.0000i -10.0000 -10.0000i

Linear Combination of DFTs:
  90.0000 + 0.0000i -10.0000 +10.0000i -10.0000 + 0.0000i -10.0000 -10.0000i

Linearity property is verified: DFT(a*x1[n] + b*x2[n]) equals a*X1[k] + b*X2[k].

7b) Verification of circular time shift property of DFT

Circular time shift property if x(n)↔X(K) then DFT [x(n-m)] = $e^{-j2\pi km/N}$ X(K)

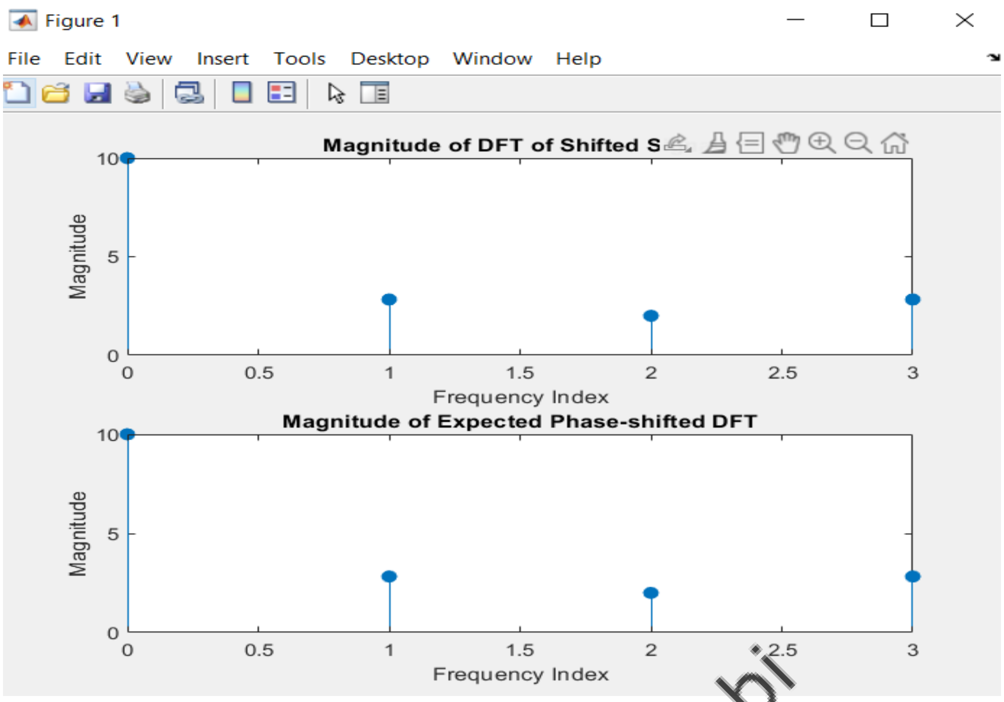
Program:

```
clc;
clear all;
N = input('Enter the Length of the signal');
x= input('Enter the input sequence=');
m = input('Enter the Number of circular shifts=');
X = fft(x);% Compute DFT of the original signal
x_shifted = circshift(x, m);% Circularly shift the signal by m
X_shifted = fft(x_shifted)% Compute DFT of the shifted signal

% Compute the expected phase-shifted DFT
k = 0:N-1; % Frequency indices
expected_X_shifted = X .* exp(-1j * 2 * pi * k .* m / N)

subplot(2,1,1);
stem(0:N-1, abs(X_shifted), 'filled');
title('Magnitude of DFT of Shifted Signal');
xlabel('Frequency Index');
ylabel('Magnitude');

subplot(2,1,2);
stem(0:N-1, abs(expected_X_shifted), 'filled');
title('Magnitude of Expected Phase-shifted DFT');
xlabel('Frequency Index');
ylabel('Magnitude');
```


Result 7b1:

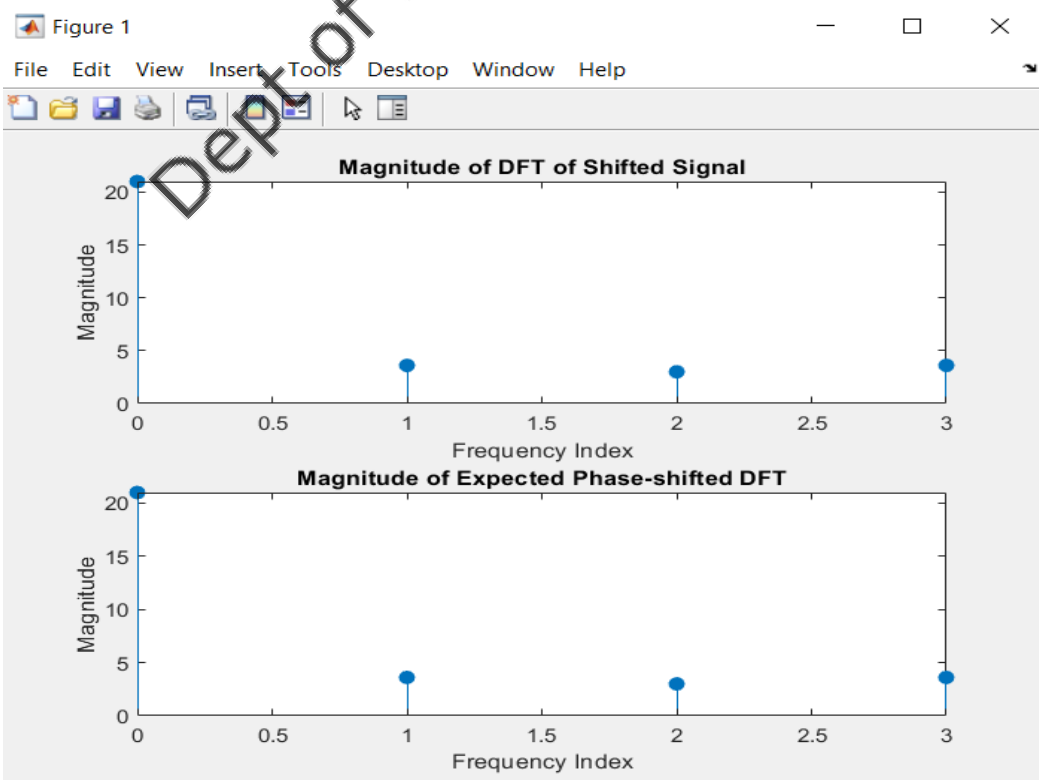
Enter the Length of the signal4
Enter the input sequence=[1 2 3 4]
Enter the Number of circular shifts=2
X_shifted =
 10.0000 + 0.0000i   2.0000 - 2.0000i -2.0000 + 0.0000i   2.0000 + 2.0000i

expected_X_shifted =
 10.0000 + 0.0000i   2.0000 - 2.0000i -2.0000 - 0.0000i   2.0000 + 2.0000i



Result 7b2:

```
Enter the Length of the signal4
Enter the input sequence=[ 5 6 7 3]
Enter the Number of circular shifts=3
X_shifted =
  21.0000 + 0.0000i   3.0000 - 2.0000i  -3.0000 + 0.0000i   3.0000 + 2.0000i
expected_X_shifted =
  21.0000 + 0.0000i   3.0000 - 2.0000i  -3.0000 - 0.0000i   3.0000 + 2.0000i
```



7c) Verification of circular frequency shift property of DFT

Circular frequency shift property if x(n)↔X(K) then x(n) $e^{-j2\pi kl/N}$ = X((K-l)) $_N$

Program:

```
clc;
clear all;
N = input('Enter the Length of the signal');
x= input('Enter the input sequence=');
l = input('Enter the Number of circular shifts=');
X = fft(x);% Compute DFT of the original signal

% Frequency shift multiplier
n = 0:N-1;
e_jw = exp(1j * 2 * pi * l * n / N); % Complex exponential
x_shifted = x .* e_jw;% Modified signal
X_shifted = fft(x_shifted)% Compute DFT of the modified signal

expected_X_shifted = circshift(X, -l)% Circular frequency-shifted
DFT

subplot(2,1,1);
stem(0:N-1, abs(X_shifted), 'filled');
title('Magnitude of DFT of Frequency-shifted Signal');
xlabel('Frequency Index');
ylabel('Magnitude');

subplot(2,1,2);
stem(0:N-1, abs(expected_X_shifted), 'filled');
title('Magnitude of Circular frequency-shifted DFT');
xlabel('Frequency Index');
ylabel('Magnitude');
```

Result 7c1:

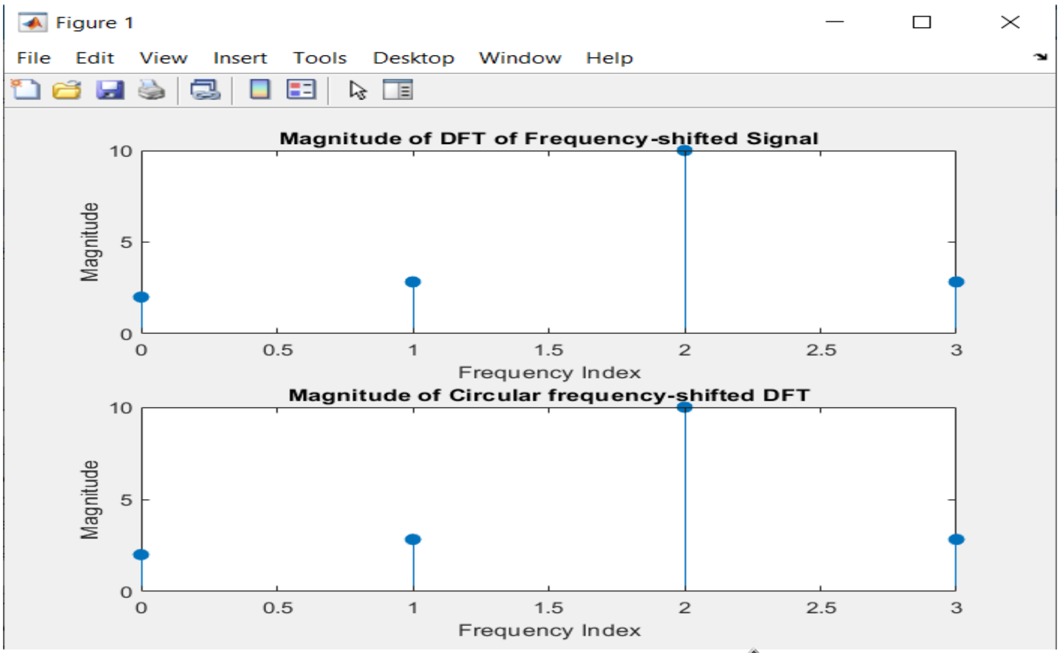
Enter the Length of the signal4
Enter the input sequence=[1 2 3 4]
Enter the Number of circular shifts=2

X_shifted =

-2.0000 + 0.0000i -2.0000 - 2.0000i 10.0000 - 0.0000i -2.0000 + 2.0000i

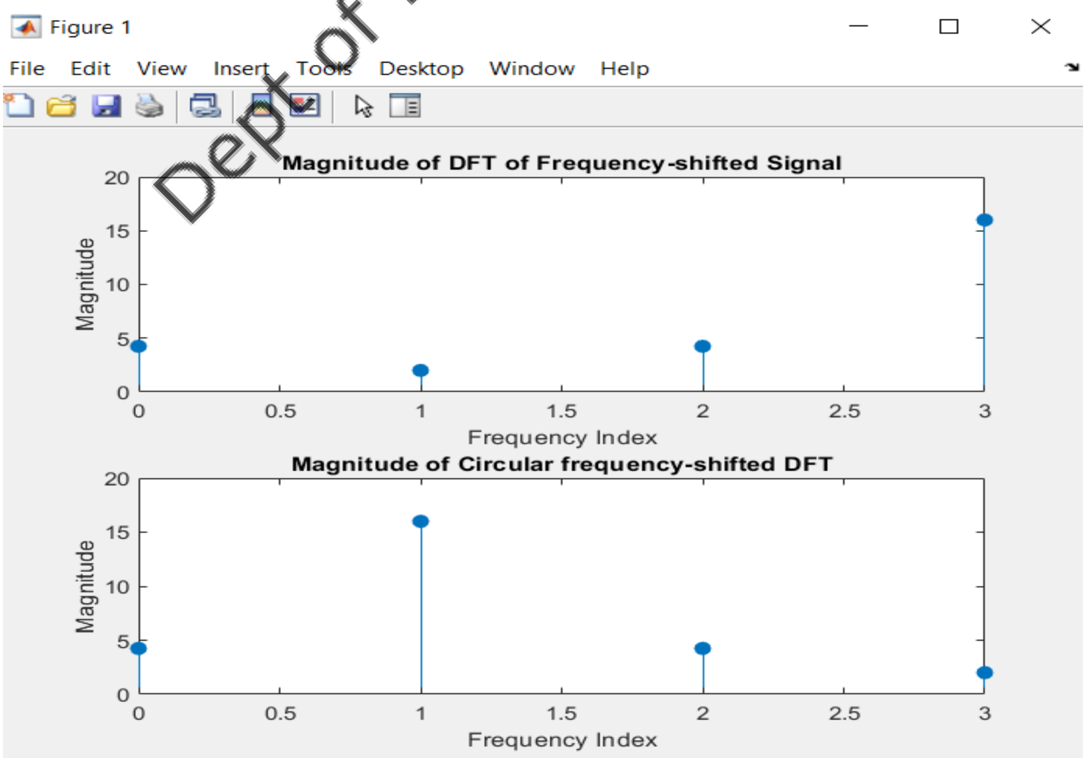
expected_X_shifted =

-2.0000 + 0.0000i -2.0000 - 2.0000i 10.0000 + 0.0000i -2.0000 + 2.0000i



Result 7c1:

Enter the Length of the signal4
Enter the input sequence=[5 6 2 3]
Enter the Number of circular shifts=3
X_shifted =
   3.0000 - 3.0000i -2.0000 + 0.0000i   3.0000 + 3.0000i 16.0000 - 0.0000i
expected_X_shifted =
   3.0000 + 3.0000i 16.0000 + 0.0000i   3.0000 - 3.0000i -2.0000 + 0.0000i



EXPERIMENT NO 8

DEVELOP DECIMATION IN TIME RADIX-2 FFT ALGORITHM WITHOUT USING BUILT-IN FUNCTIONS

Program:

```
clc;
clear all;
x= input('Enter the input sequence=');
N = length(x);

% Bit-reversal permutation
num_bits = log2(N);
reversed_indices = bit_reverse(0:N-1, num_bits);
x = x(reversed_indices + 1);

% FFT computation
% Initial length of the FFT
current_len = 2;

while current_len <= N
    half_len = current_len / 2;
    twiddle_factor = exp(-2i * pi * (0:half_len-1) / current_len);

    for k = 1:current_len:N
        even = x(k:k+half_len-1);
        odd = x(k+half_len:k+current_len-1);

        x(k:k+current_len-1) = [even + twiddle_factor .* odd, even
- twiddle_factor .* odd];
    end

    current_len = current_len * 2;
end

% Display the result
disp('FFT of the input:');
disp(x);

% Bit reversal function
function reversed_indices = bit_reverse(indices, num_bits)
    % Reverse the bits of the indices
    reversed_indices = zeros(size(indices));
    for k = 1:length(indices)
        bin_str = dec2bin(indices(k), num_bits);
        reversed_bin_str = bin_str(end:-1:1);
        reversed_indices(k) = bin2dec(reversed_bin_str);
    end
end
```

Result 8a:

Enter the input sequence=[1 2 3 4 5 6 7 8]
FFT of the input:
Columns 1 through 5

36.0000 + 0.0000i -4.0000 + 9.6569i -4.0000 + 4.0000i -4.0000 + 1.6569i -4.0000 + 0.0000i

Columns 6 through 8

-4.0000 - 1.6569i -4.0000 - 4.0000i -4.0000 - 9.6569i

Result 8b:

Enter the input sequence=[1 1 0 0 -1 -1 0 0]
FFT of the input:
Columns 1 through 5

0.0000 + 0.0000i   3.4142 - 1.4142i   0.0000 + 0.0000i   0.5858 - 1.4142i   0.0000 + 0.0000i

Columns 6 through 8

0.5858 + 1.4142i   0.0000 + 0.0000i   3.4142 + 1.4142i

EXPERIMENT NO 9

DESIGN AND IMPLEMENTATION OF DIGITAL LOW PASS FIR FILTER USING A WINDOW TO MEET THE GIVEN SPECIFICATIONS

Summary of Parameters for Common Windows:

SL No	Window	Window Equation	Stop band attenuation	K	Transition Window $\Delta W = W_s - W_p$	MATLAB Function
1	Rectangular (Boxcar) window	$w[n]=\begin{cases}1, & 0\le n\le N-1\\ 0, & \text{otherwise}\end{cases}$	21 dB	2	$\frac{4\pi}{N}$	rectwin
2	Triangular (Bartlett) window	$w[n]=\begin{cases}\frac{2n}{N-1}, & 0\le n\le \frac{N-1}{2}\\ \frac{2(N-1-n)}{N-1}, & \frac{N-1}{2}<n\le N-1\end{cases}$	27 dB	4	$\frac{8\pi}{N}$	artlett
3	Kaiser Window	$w[n]=I_0\left(\beta\sqrt{1-\left(\frac{2n}{N-1}-1\right)^2}\right)/I_0(\beta),\quad 0\le n\le N-1$	40 dB			kaiser
4	Hanning window	$w[n]=0.54-0.46\cos\left(\frac{2\pi n}{N-1}\right),\quad 0\le n\le N-1$	44 dB	4	$\frac{8\pi}{N}$	hann
5	Hamming window	$w[n]=0.5\left(1-\cos\left(\frac{2\pi n}{N-1}\right)\right),\quad 0\le n\le N-1$	53 dB	4	$\frac{8\pi}{N}$	hamming
6	Blackman window	$w[n]=0.42-0.5\cos\left(\frac{2\pi n}{N-1}\right)+0.08\cos\left(\frac{4\pi n}{N-1}\right),\quad 0\le n\le N-1$	75 dB	6	$\frac{12\pi}{N}$	blackman

Program:

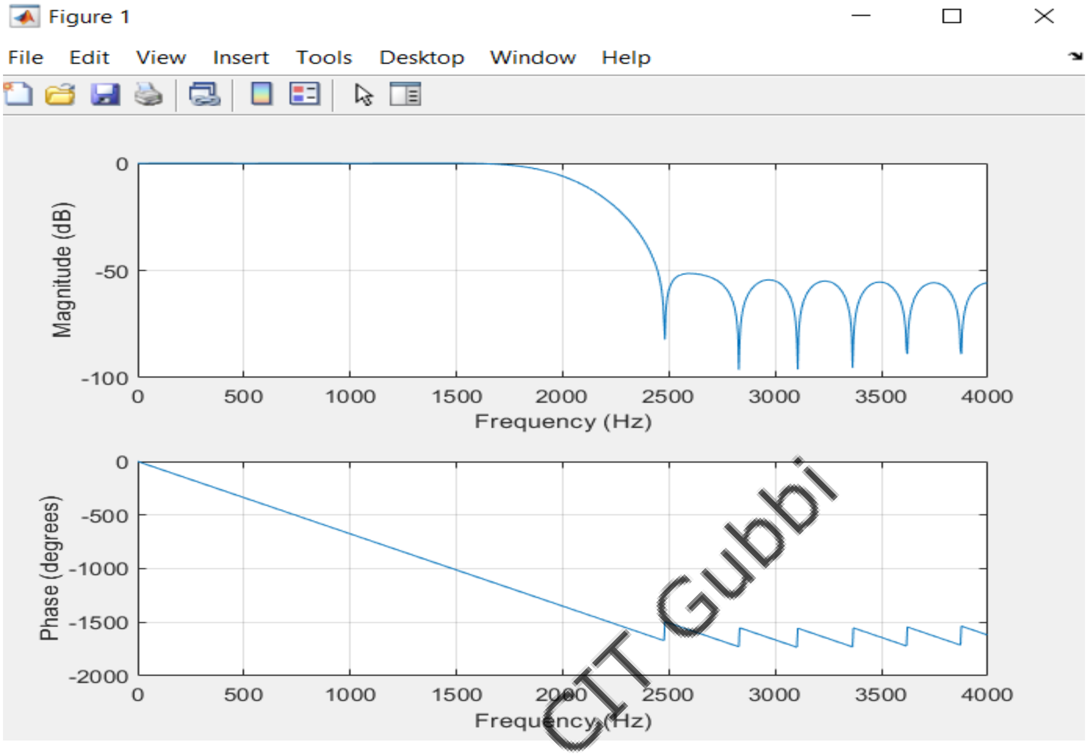
```
clc;
clear all;
close all;
fs= input('Enter the Value of Sampling frequency (fs) in Hz=');
fc= input('Enter the Value of Cutoff frequency (fc)in Hz=');
N= input('Enter the Order (ODD Order)')
wc=fc/(fs/2);
h=fir1(N,wc,'low',hamming(N+1));
freqz(h,1,1024,fs)
```

Result 9a:

Enter the Value of Sampling frequency (fs) in Hz=8000

Enter the Value of Cutoff frequency (fc)in Hz=2000

Enter the Order (ODD Order)30

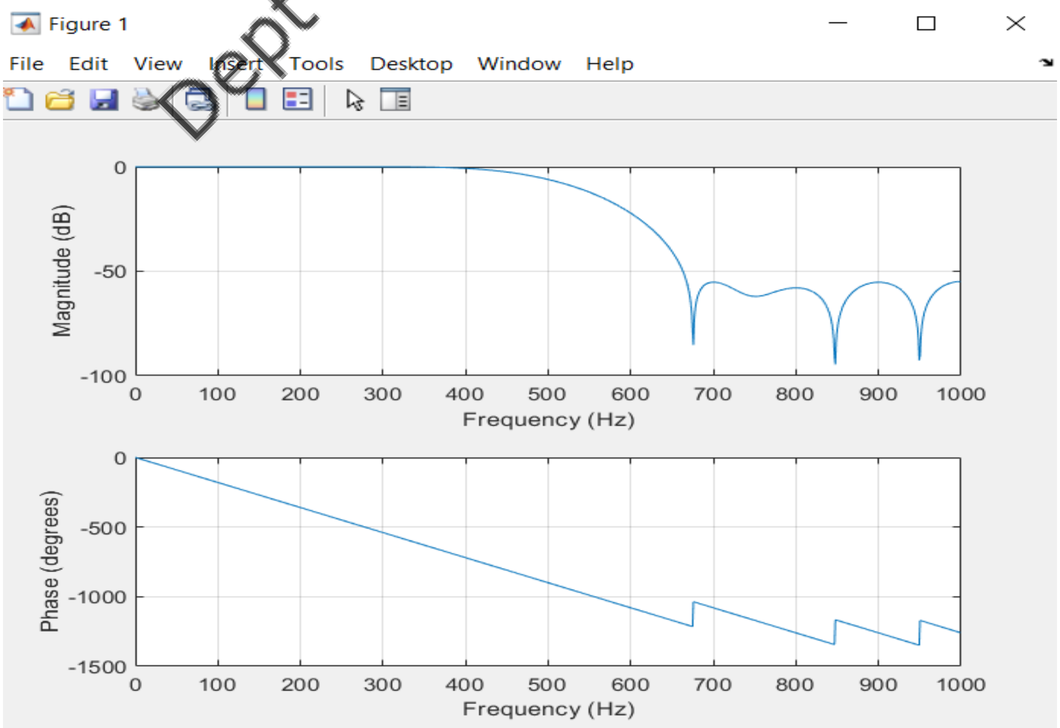


Result 9a:

Enter the Value of Sampling frequency (fs) in Hz=2000

Enter the Value of Cutoff frequency (fc)in Hz=500

Enter the Order (ODD Order)20



EXPERIMENT NO 10

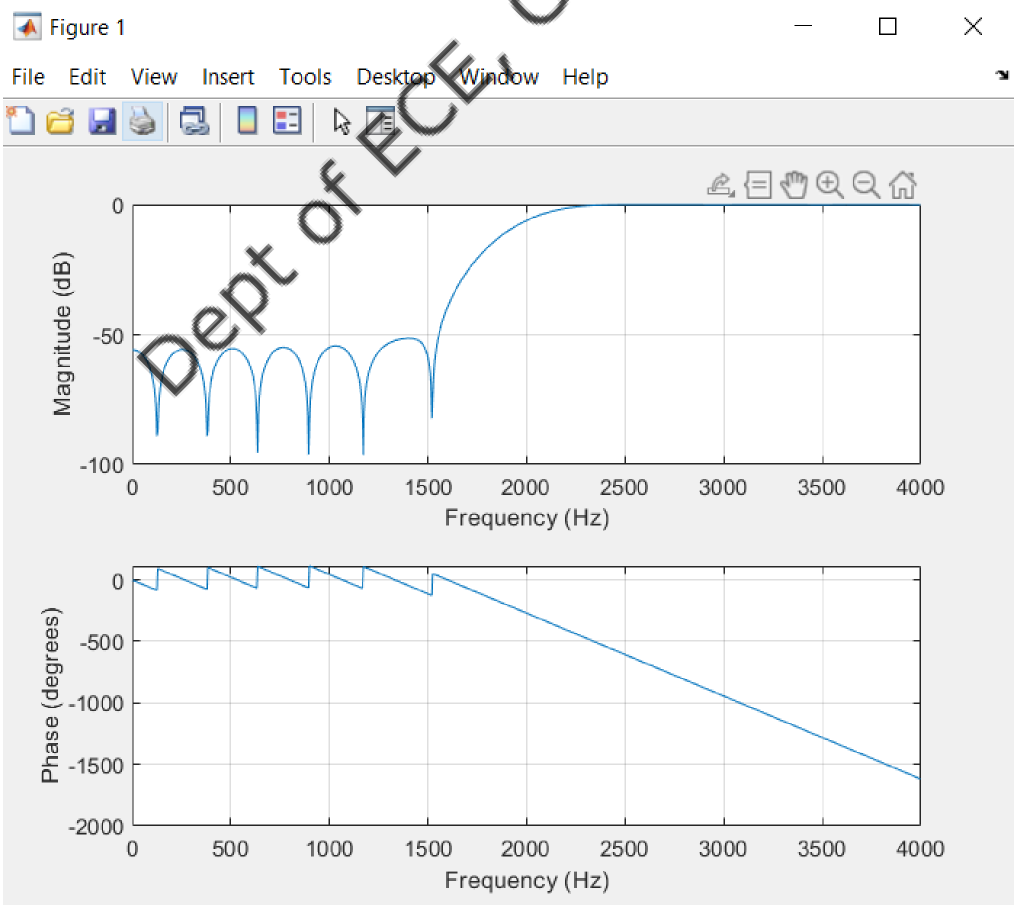
DESIGN AND IMPLEMENTATION OF DIGITAL HIGH PASS FIR FILTER USING A WINDOW TO MEET THE GIVEN SPECIFICATIONS

Program:

```
clc;
clear all;
close all;
fs= input('Enter the Value of Sampling frequency (fs) in Hz=');
fc= input('Enter the Value of Cutoff frequency (fc)in Hz=');
N= input('Enter the Order (ODD Order)');
wc=fc/(fs/2);
h=fir1(N,wc,'high',hamming(N+1));
freqz(h,1,1024,fs)
```

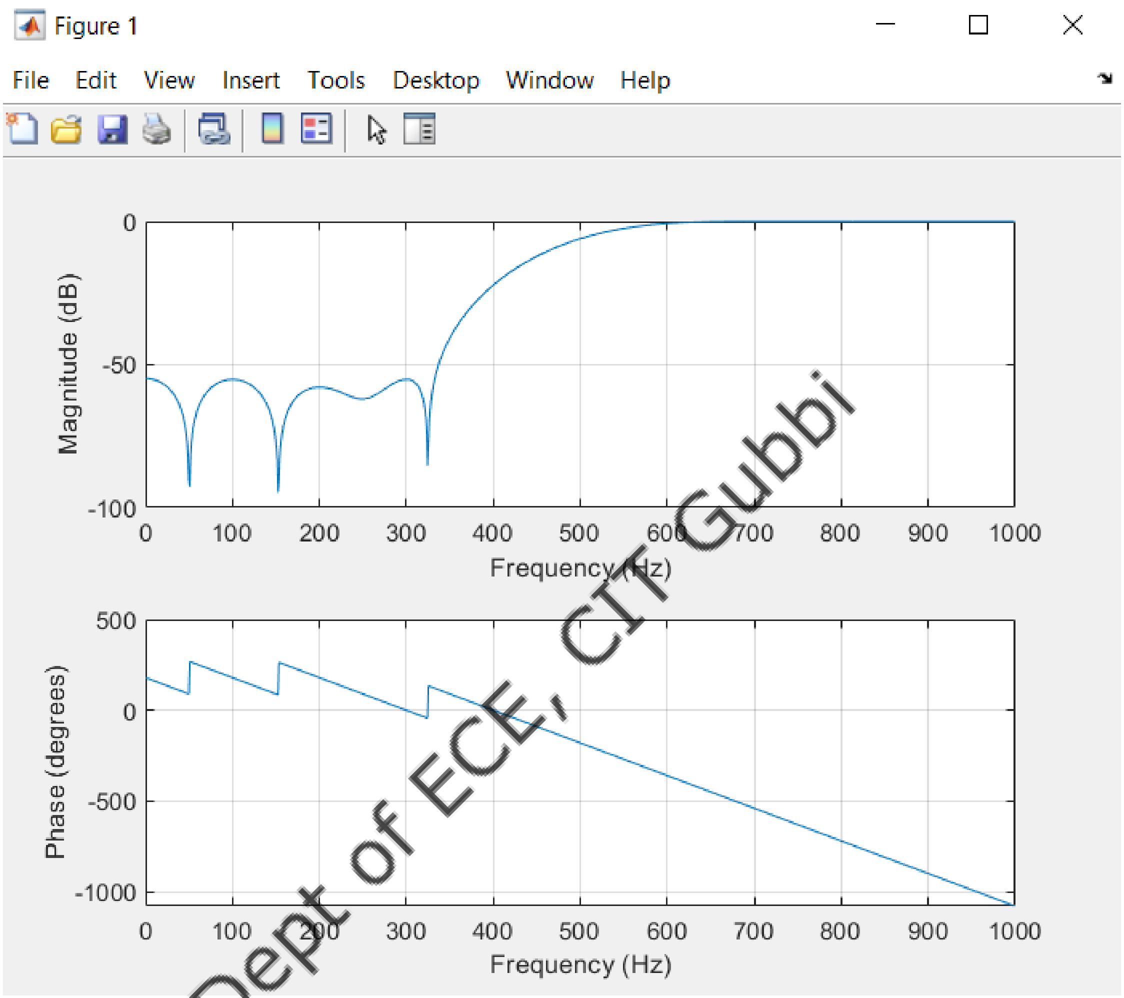
Result 10a:

Enter the Value of Sampling frequency (fs) in Hz=8000
Enter the Value of Cutoff frequency (fc)in Hz=2000
Enter the Order (ODD Order)30



Result 10b:

Enter the Value of Sampling frequency (fs) in Hz=2000
Enter the Value of Cutoff frequency (fc)in Hz=500
Enter the Order (ODD Order)20



EXPERIMENT NO 11

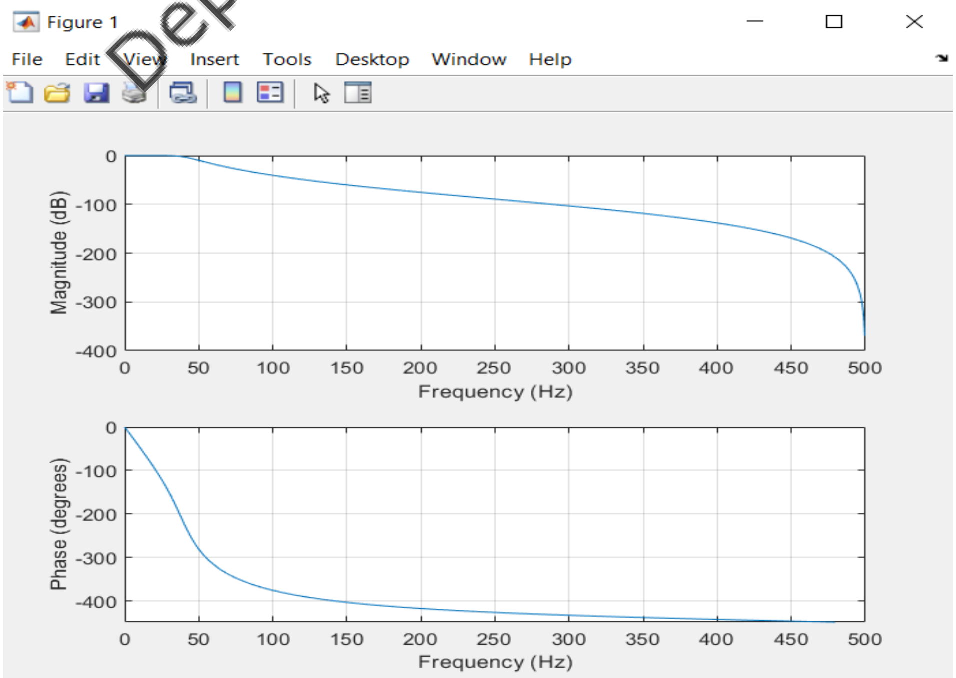
DESIGN AND IMPLEMENTATION OF DIGITAL IIR BUTTERWORTH LOW PASS FILTER TO MEET THE GIVEN SPECIFICATIONS.

Program:

```
clc;
clear all
fs=input('Enter the Value of Sampling frequency (fs) in Hz=');
kp=input('Enter the Value of Pass band Attenuation in dB=');
ks=input('Enter the Value of Stop band Attenuation in dB=');
fp=input('Enter the Value of Pass band Frequency in Hz=');
fstop=input('Enter the Value of Stop band Frequency in Hz=');
wp=fp/(fs/2);
wstop=fstop/(fs/2);
[N, wc]=buttord(wp,wstop,kp,ks)
[b, a]=butter(N,wc,'low')
freqz(b,a,1000,fs)
```

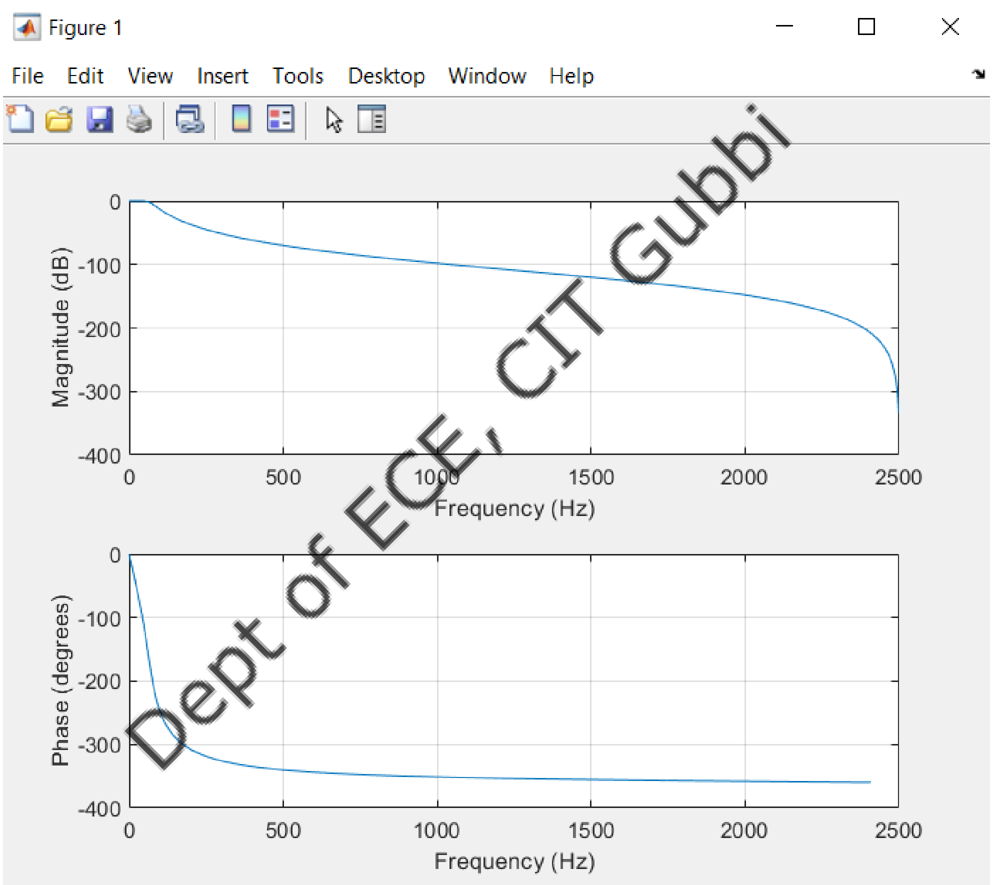
Result 11a:

Enter the Value of Sampling frequency (fs) in Hz=1000
Enter the Value of Pass band Attenuation in dB=3
Enter the Value of Stop band Attenuation in dB=60
Enter the Value of Pass band Frequency in Hz=40
Enter the Value of Stop band Frequency in Hz=150
N =5
wc =0.0810
b = 1.0e-03 *0.0227 0.1136 0.2272 0.2272 0.1136 0.0227
a =1.0000 -4.1768 7.0358 -5.9686 2.5478 -0.4375



Result 11b:

Enter the Value of Sampling frequency (fs) in Hz=5000
Enter the Value of Pass band Attenuation in dB=7
Enter the Value of Stop band Attenuation in dB=70
Enter the Value of Pass band Frequency in Hz=50
Enter the Value of Stop band Frequency in Hz=500
N =   4
wc = 0.0276
b =1.0e-04 *0.0315   0.1259   0.1888   0.1259   0.0315
a =1.0000   -3.7737   5.3464   -3.3701   0.7974



EXPERIMENT NO 12

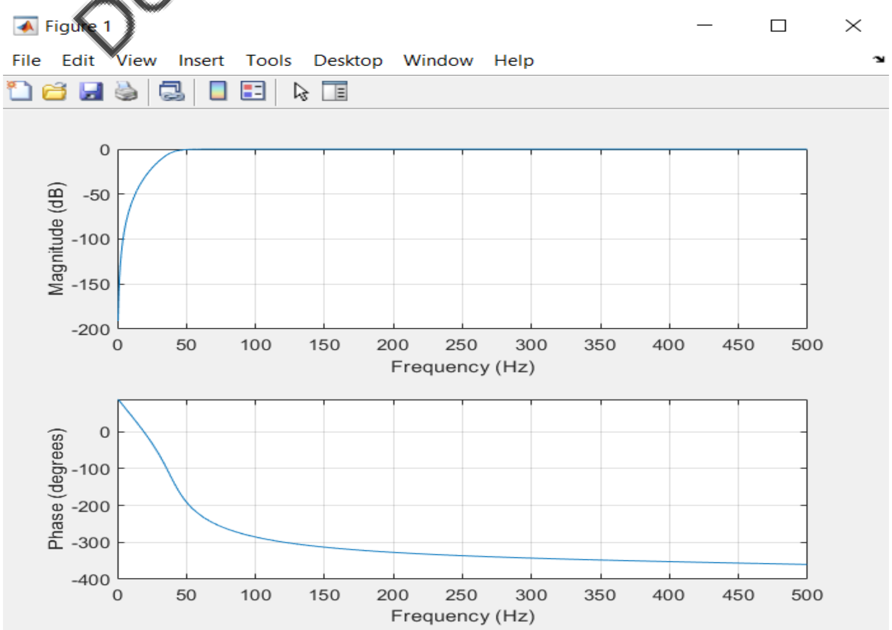
DESIGN AND IMPLEMENTATION OF DIGITAL IIR BUTTERWORTH HIGH PASS FILTER TO MEET THE GIVEN SPECIFICATIONS.

Program:

```
clc;
clear all
fs=input('Enter the Value of Sampling frequency (fs) in Hz=');
kp=input('Enter the Value of Pass band Attenuation in dB=');
ks=input('Enter the Value of Stop band Attenuation in dB=');
fp=input('Enter the Value of Pass band Frequency in Hz=');
fstop=input('Enter the Value of Stop band Frequency in Hz=');
wp=fp/(fs/2);
wstop=fstop/(fs/2);
[N, wc]=buttord(wp,wstop,kp,ks)
[b, a]=butter(N,wc,'high')
freqz(b,a,1000,fs)
```

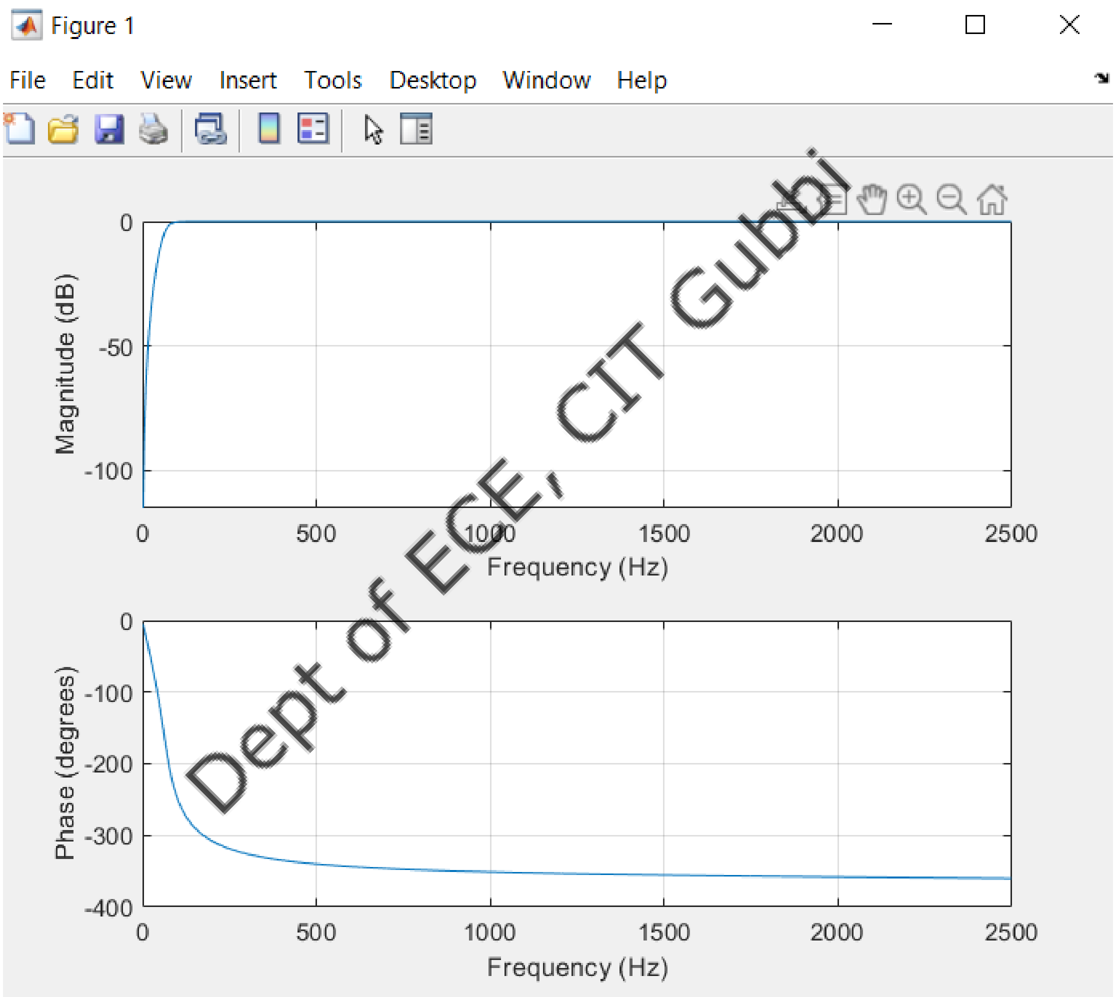

Result 12a:

Enter the Value of Sampling frequency (fs) in Hz=1000
Enter the Value of Pass band Attenuation in dB=3
Enter the Value of Stop band Attenuation in dB=60
Enter the Value of Pass band Frequency in Hz=40
Enter the Value of Stop band Frequency in Hz=150
N = 5
wc =0.0810
b =0.6615   -3.3073   6.6145   -6.6145   3.3073   -0.6615
a =1.0000   -4.1768   7.0358   -5.9686   2.5478   -0.4375



Result 12a:

Enter the Value of Sampling frequency (fs) in Hz=5000
Enter the Value of Pass band Attenuation in dB=7
Enter the Value of Stop band Attenuation in dB=70
Enter the Value of Pass band Frequency in Hz=50
Enter the Value of Stop band Frequency in Hz=500
N = 4
wc =0.0276
b = 0.8930 -3.5719 5.3579 -3.5719 0.8930
a = 1.0000 -3.7737 5.3464 -3.3701 0.7974



Beyond Syllabus Experiments

EXPERIMENT NO A

COMPUTE AUTOCORRELATION AND CROSS CORRELATION OF SEQUENCES

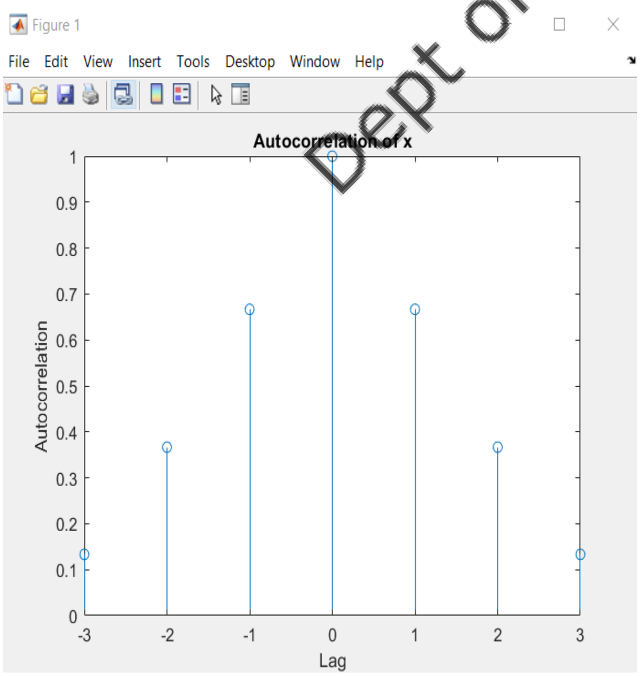
Program:

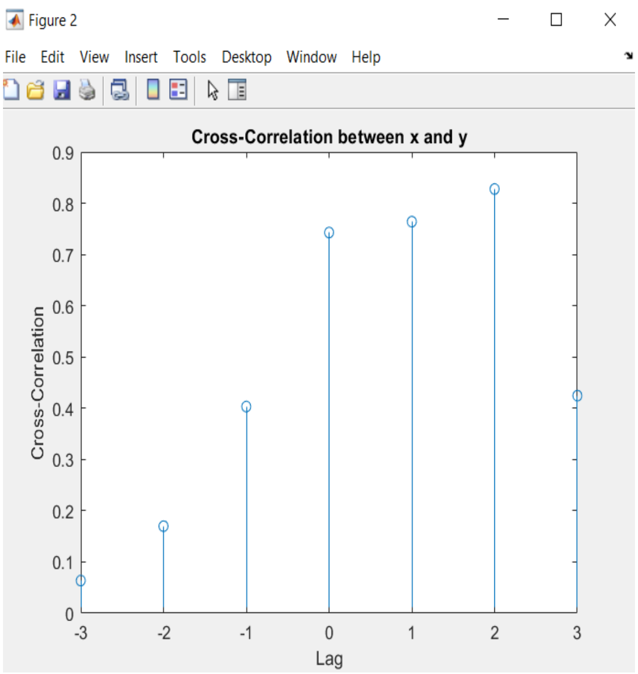
```
clc;
clear all
x =input('Enter the Value of First sequence=');
y =input('Enter the Value of Second sequence=');

[r, lags] = xcorr(x, 'coeff'); % Compute autocorrelation
figure;
stem(lags, r);
title('Autocorrelation of x');
xlabel('Lag');
ylabel('Autocorrelation');

[r, lags] = xcorr(x, y, 'coeff');% Compute cross-correlation
figure;
stem(lags, r);
title('Cross-Correlation between x and y');
xlabel('Lag');
ylabel('Cross-Correlation');
```

Result A:

Enter the Value of First sequence=[1 2 3 4]

Enter the Value of Second sequence=[5 6 2 3]





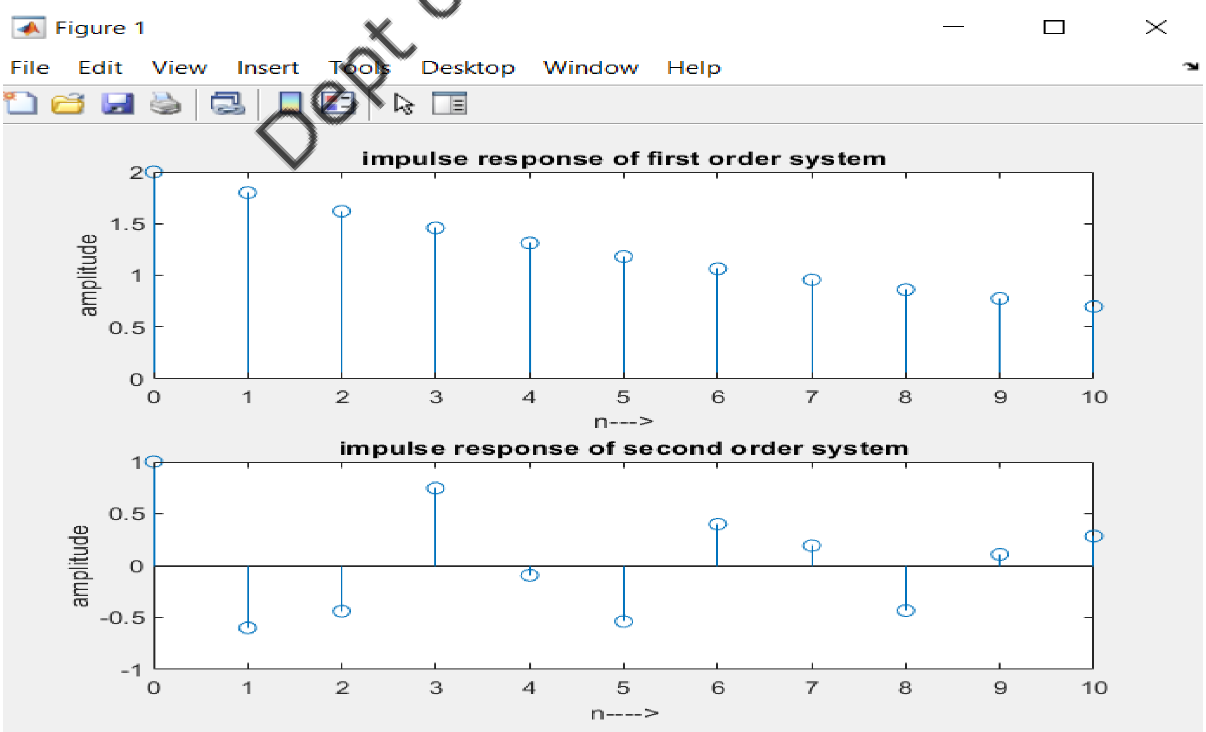
EXPERIMENT NO B

IMPULSE RESPONSE OF FIRST AND SECOND ORDER SYSTEMS

Program:

```
clc;
close all;
clear all;
n=0:10;
%impulse response of first order system
b=[2 0 0];
a=[1 -0.9 0];
y=dimpulse(b,a,length(n));
subplot(2,1,1);
stem(n,y);
xlabel('n--->');
ylabel('amplitude');
title('impulse response of first order system');
b=[1 0 0];
a=[1 0.6 0.8];
y1=dimpulse(b,a,length(n));
subplot(2,1,2);
stem(n,y1);
xlabel('n---->');
ylabel('amplitude');
title('impulse response of second order system');
```

Result B:



EXPERIMENT NO C

PROGRAM TO PROVE THE PARSEVALS THEOREM

Parseval's theorem can be expressed as

$$\int_{-\infty}^{\infty} |x(t)|^2 \, dt = \int_{-\infty}^{\infty} |X(f)|^2 \, df$$

For discrete-time signals, the theorem can be written as:

$$\sum_{n=-\infty}^{\infty} |x[n]|^2 = \sum_{k=-\infty}^{\infty} |X[k]|^2$$

Program:

```
clc;
clear all;
close all;
x1=input('enter the 1st input seq=')
x2=input('enter the 2nd input seq=')
N=4;
X1=fft(x1,N)
X2=fft(x2,N)
y1=sum(x1.*conj(x2))
y2=sum(X1.*conj(X2))./N
if  y1==y2
    disp('parsevals theorem proved');
else
disp('parsevals theorem not proved')
end
```

Result C:

enter the 1st input seq=[1 2 3 4]

x1 =1 2 3 4

enter the 2nd input seq=[5 6 2 3]

x2 =5 6 2 3

X1 =

10.0000 + 0.0000i -2.0000 + 2.0000i -2.0000 + 0.0000i -2.0000 - 2.0000i

X2 =

16.0000 + 0.0000i 3.0000 - 3.0000i -2.0000 + 0.0000i 3.0000 + 3.0000i

y1 =

35

y2 =

35

Parseval's theorem proved

EXPERIMENT NO D

COMPUTATION OF N POINT DFT OF A GIVEN SEQUENCE BY USING BUILT-IN FUNCTION AND TO PLOT THE MAGNITUDE & PHASE SPECTRUM.

Program:

```
clc;
clear all;
close all;
N=input('enter the value of N=');
n=0:1:(N-1);
x = input('enter the input signal, x(n)=');
X=fft(x,N)
mag=abs(X)
phase=angle(X)
subplot(3,1,1), stem(n,x);
xlabel('Time (s)');
ylabel('Amplitude');
title('Time domain - Input sequence');
subplot(3,1,2);
stem(n,mag)
xlabel('Frequency');
ylabel('|X(k)|');
title('Frequency domain - Magnitude response');
subplot(3,1,3);
stem(n,unwrap(phase));
ylabel('phase(X(k))');
title('Frequency domain - phase response');
```


Result D1:

enter the value of N=4
enter the input signal, x(n)=[1 2 3 4]
X =

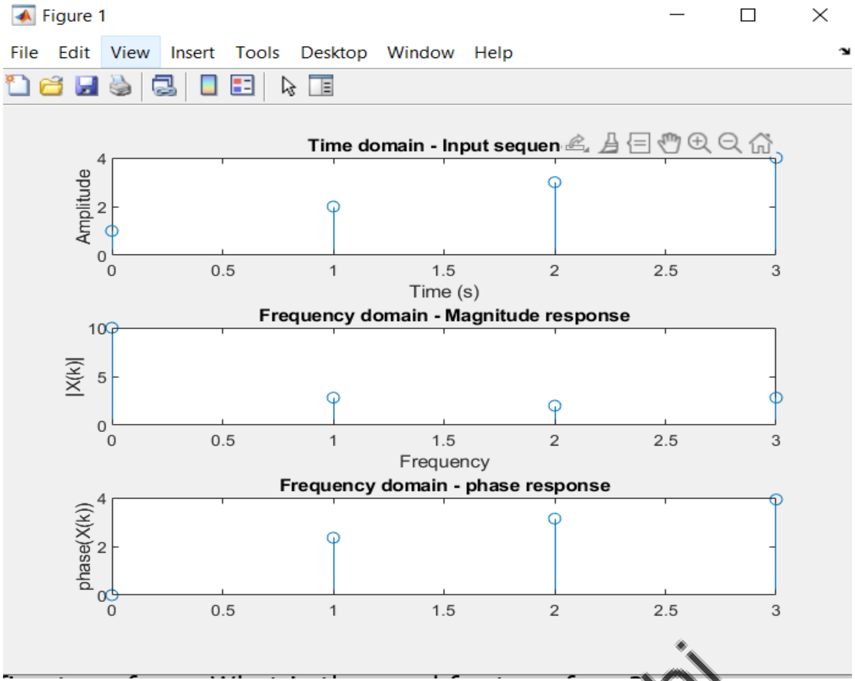
10.0000 + 0.0000i -2.0000 + 2.0000i -2.0000 + 0.0000i -2.0000 - 2.0000i

mag =

10.0000 2.8284 2.0000 2.8284

phase =

0 2.3562 3.1416 -2.3562



Result D2:

enter the value of N=6

enter the input signal, x(n)= [1 1 2 2 3 3]

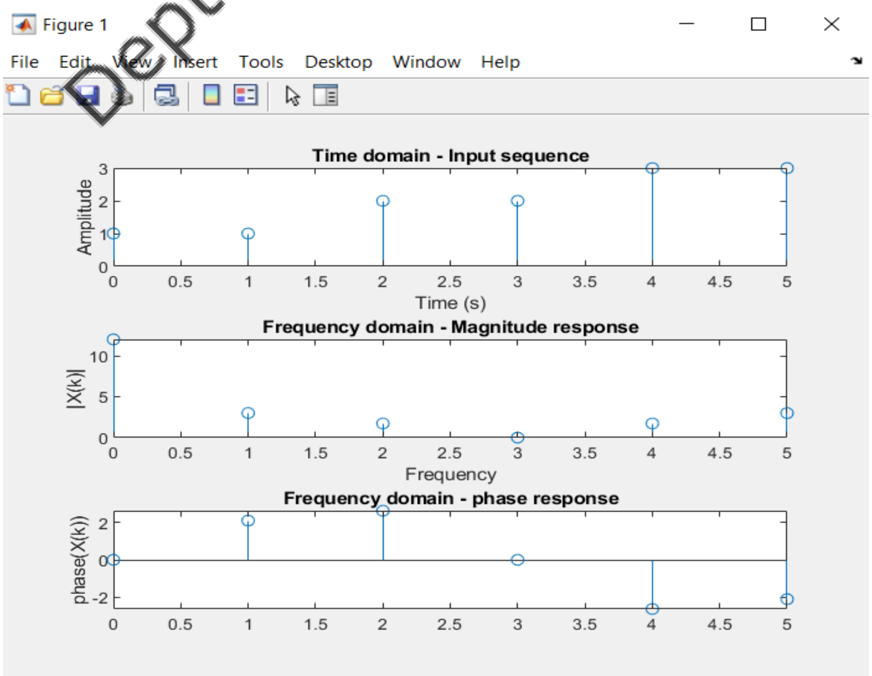
X =12.0000 + 0.0000i -1.5000 + 2.5981i -1.5000 + 0.8660i 0.0000 + 0.0000i -1.5000 - 0.8660i -1.5000 - 2.5981i

mag =

12.0000 3.0000 1.7321 0 1.7321 3.0000

phase =

0 2.0944 2.6180 0 -2.6180 -2.0944



EXPERIMENT NO E

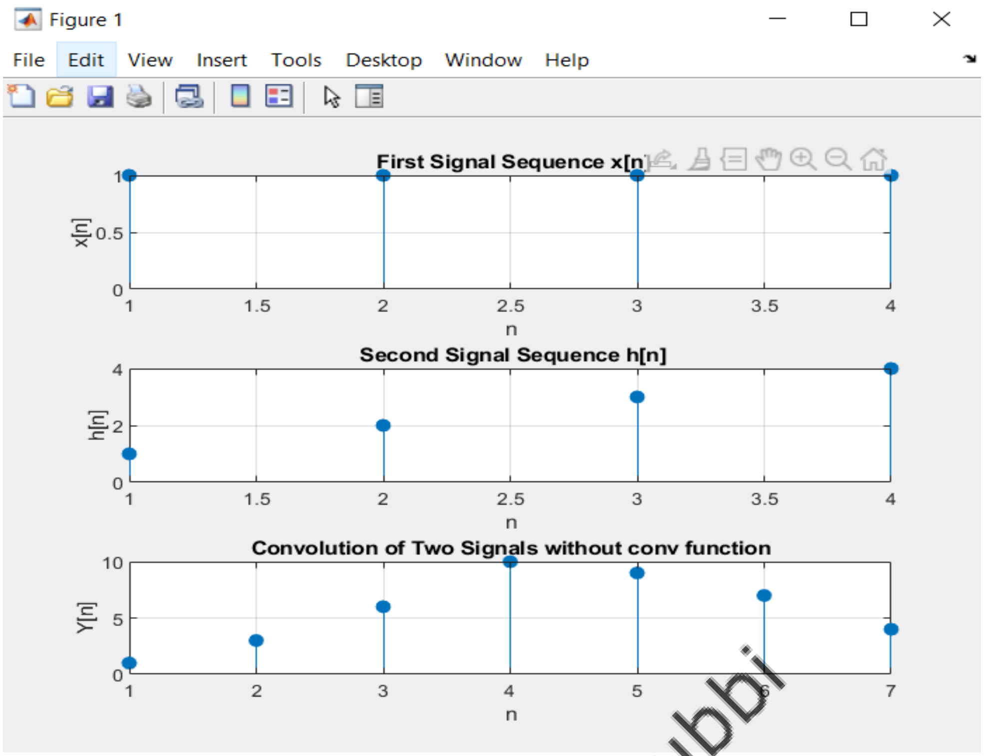
PROGRAM TO PERFORM CONVOLUTION OF TWO GIVEN SEQUENCES BY USING BUILT-IN FUNCTION AND DISPLAY THE SIGNALS.

Program:

```
clc;
clear all;
x =input('Enter the value of First Signal Sequence x[n]');
h =input('Enter the value of Second Signal Sequence h[n]');
y = conv(x, h);% convolution
figure;
subplot(3,1,1);
stem(x, 'filled');
xlabel('n');
ylabel('x[n]');
title(' First Signal Sequence x[n]');
grid on;
subplot(3,1,2);
stem(h, 'filled');
xlabel('n');
ylabel('h[n]');
title(' Second Signal Sequence h[n]');
grid on;
subplot(3,1,3);
stem(y,'filled');
ylabel('Y[n]');
xlabel('n');
grid on;
title('Convolution of Two Signals without conv function');
```

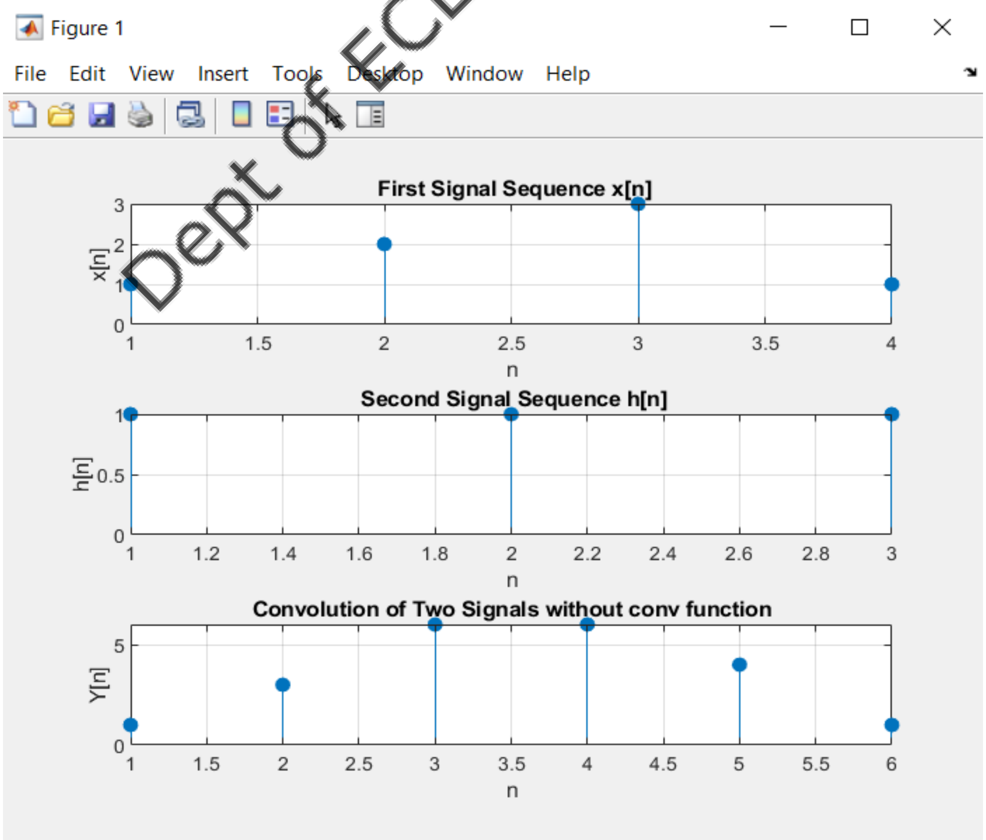

Result E1:

Enter the value of First Signal Sequence x[n][1 1 1 1]
Enter the value of Second Signal Sequence h[n][1 2 3 4]



Result E2:

Enter the value of First Signal Sequence x[n][1 2 3 1]
Enter the value of Second Signal Sequence h[n][1 1 1]



VIVA QUESTIONS

1. Define signal, Give Examples for 1-D, 2-D, 3-D signals.
2. Define transform. What is the need for transform?
3. Differentiate Fourier transform and discrete Fourier transform.
4. Differentiate DFT and DTFT
5. Explain mathematical formula for calculation of DFT.
6. Explain mathematical formula for calculation of IDFT.
7. How to calculate FT for 1-D signal?
8. What is meant by magnitude plot, phase plot, power spectrum?
9. Explain the applications of DFT.
10. What are separable transforms?
11. Explain the significance of convolution.
12. Define linear convolution.
13. Why linear convolution is called as a periodic convolution?
14. Why zero padding is used in linear convolution?
15. What are the four steps to find linear convolution?
16. What is the length of the resultant sequence in linear convolution?
17. How linear convolution will be used in calculation of LTI system response?
18. List few applications of linear convolution in LTI system design.
19. Give the properties of linear convolution.
20. How the linear convolution will be used to calculate the DFT of a signal?
21. Define transform. What is the need for transform?
22. Differentiate Fourier transform and discrete Fourier transform.
23. Differentiate DFT and DTFT.
24. What are the advantages of FFT over DFT?
25. Differentiate DITFFT and DIFFFT algorithms.
26. What is meant by radix?
27. What is meant by twiddle factor and give its properties?
28. How FFT is useful to represent a signal?
29. Compare FFT and DFT with respect to number of calculations required?
30. How the original signal is reconstructed from the FFT of a signal?
31. Define filter.
32. What are the different types of filters?
33. Why are FIR filters generally preferred over IIR filters in multirate (decimating and interpolating) systems/
34. Difference between IIR and FIR filters?
35. Differentiate ideal filter and practical filter responses.
36. What is the filter specifications required to design the analog filters?
37. What is meant by frequency response of filter?
38. What is meant by magnitude response?
39. What is meant by phase response?

40. Difference between FIR low pass filter and high pass filter
41. List some advantages of digital filters over analog filters.
42. Write some differences between FIR and IIR filters.
43. What are the different methods to design IIR filters?
44. Why IIR filters are not reliable?
45. What are different applications of IIR filters?
46. What are advantages of IIR filters?
47. What are disadvantages of IIR filters?
48. Differentiate Butterworth and Chebyshev approximations.
49. What is meant by impulse response?
50. Difference between IIR low pass and High pass filters
51. What is sampling theorem?
52. What do you mean by process of reconstruction.
53. What are techniques of reconstructions.
54. What do you mean Aliasing? What is the condition to avoid aliasing for sampling?
55. Write the conditions of sampling.
56. How many types of sampling there?
57. Explain the statement-t= 0:0.000005:0.058. In the above example what does colon (:) and semicolon (;) denotes.
58. What is a) Under sampling b) nyquist plot c) Oversampling.
59. Write the MATLAB program for Oversampling.
60. What is the use of command 'legend'?
61. Write the difference between built in function, plot and stem describe the function.
62. What is the function of built in function and subplot?
63. What is linear convolution?
64. Explain how convolution syntax built in function works.
65. How to calculate the beginning and end of the sequence for the two sided controlled output?
66. What is the total output length of linear convolution sum.
67. What is an LTI system? Describe impulse response of a function.20. What is the difference between convolution and filter?
68. How to calculate output length of the linear and circular convolution
69. What is window method? How you will design an FIR filter using window method.
70. What do you mean by cut-off frequency?