SVR ENGINEERING COLLEGE

Approved by AICTE & Permanently Affiliated to JNTUA

Ayyalurmetta, Nandyal – 518503. Website: www.svrec.ac.in

Department of Electronics and Communication Engineering

DIGITAL SIGNAL PROCESSING LABORATORY MANUAL

III B.Tech (ECE) I Semester (R - 20)

2022-2023

INDEX

ECE DEPT VISION & MISSION PEOs and PSOs

Vision: To produce highly skilled, creative and competitive Electronics and Communication Engineers to meet the emerging needs of the society.

Mission:

- Impart core knowledge and necessary skills in Electronics and Communication Engineering through innovative teaching and learning.
- Inculcate critical thinking, ethics, lifelong learning and creativity needed for industry and society
- Cultivate the students with all-round competencies, for career, higher education and self-employability

I. PROGRAMME EDUCATIONAL OBJECTIVES (PEOS)

PEO1: Graduates apply their knowledge of mathematics and science to identify, analyze and solve problems in the field of Electronics and develop sophisticated communication systems.

PEO2: Graduates embody a commitment to professional ethics, diversity and social awareness in their professional career.

PEO3: Graduates exhibit a desire for life-long learning through technical training and professional activities.

II. PROGRAM SPECIFIC OUTCOMES (PSOS)

PSO1: Apply the fundamental concepts of electronics and communication engineering to design a variety of components and systems for applications including signal processing, image processing, communication, networking, embedded systems, VLSI and control system

PSO2: Select and apply cutting-edge engineering hardware and software tools to solve complex Electronics and Communication Engineering problems.

III. PROGRAMME OUTCOMES (PO'S)

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, review research literature, and analyze 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 with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

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.

IV. COURSE OBJECTIVES

Students can learn the basics of using DSP chips to perform real-time digital signal processing.

Ability to apply knowledge of mathematics, science and engineering: Construction of tools for visualizing the basic concepts of discrete signal representation such as Fourier transforms, discrete time representations.

Students will learn numerous programming tools for design and implementations of filtering algorithms.

Understand the concept of Multi-rate signal processing and sample rate conversion.

Develop and Implement DSP algorithms in software using CCS with DSP floating pointProcessor.

V. COURSE OUTCOMES

After the completion of the course students will be able to

Course Outcomes	Course Outcome statements	BTL
CO1	Ability to design-test, to verify, to evaluate, and to benchmark a real-time DSP system.	L1
CO2	Ability to calculate discrete time domain and frequency domain of signals using discreteFourier series and Fourier transform.	L3
CO3	Ability to design, using MATLAB-based filter design techniques, FIR and IIR digital filters and Determine the frequency response of filters.	L4
CO4	Implementation of basic signal processing algorithms such as convolution, difference equation implementation and application of them in the construction of FIR and IIR filters.	L2
CO5	Design DSP based real time processing systems to meet desired needs of the society.	L5

VI.COURSE MAPPING WITH PO'S AND PEO'S

Course Title	P01	P02	P03	P04	P05	P06	P07	P08	P09	P010	P011	P012	PE01	PE02
DSP Lab	2.2	2.6	2.2	2.4	2.6	2.2	1.4	2	2.0	2.2	2.2	2.6	2.6	2.4

V MAPPING OF COURSE OUTCOMES WITH PEO'S AND PO'S

Course Title	P01	P02	P03	P04	P05	P06	P07	P08	P09	P010	P011	P012	PE01	PE02
CO1	2	3	2	2	3	2	2	1	3	3	3	3	3	3
CO2	1	2	3	2	2	2	1	2	1	2	3	2	2	3
CO3	3	3	2	3	2	3	2	2	2	1	1	3	2	2
CO4	3	2	2	2	3	2	1	3	2	2	2	2	3	2
CO5	2	3	2	3	3	2	1	2	2	3	2	3	3	2

LABORATORY INSTRUCTIONS

1. While entering the Laboratory, the students should follow the dress code. (Wear shoes and White apron, Female Students should tie their hair back).

2. The students should bring their observation book, record, calculator, necessary stationery items and graph sheets if any for the lab classes without which the students will not be allowed for doing the experiment.

3. All the Equipment and components should be handled with utmost care. Any breakage or damage will be charged.

4. If any damage or breakage is noticed, it should be reported to the concerned in charge immediately.

5. The theoretical calculations and the updated register values should be noted down in the observation book and should be corrected by the lab in-charge on the same day of the laboratory session.

6. Each experiment should be written in the record note book only after getting signature from the lab in-charge in the observation notebook.

7. Record book must be submitted in the successive lab session after completion of experiment.

8. 100% attendance should be maintained for the laboratory classes.

WORKING PROCEDURE WITH MATLAB:

1) Double click on Matlab icon. -> Then Matlab will be opened

2) To write the Matlab Program

Goto file menu-> New -> Script(Mfile) -> In the opened Script file write the Matlab code and save the file with an extension of .m

Ex: “linear.m”

3)To execute Matlab Program Select the all lines in matlab program(ctrl+A) of mfile and press “F9”

to execute the matlab code

4)Entering the inputs in command window

 If the command window is displaying the message like “enter the input sequence” then

enter the sequence with square brackets and each sample values is spaced with single space

Ex: Enter input sequence [1 2 3 4] If it is asking a value input write the value without brackets

Ex: “enter length of sequence 4” After entering inputs It displays the Output Graphs.

PROCEDURE TO WORK ON CODE COMPOSER STUDIO

Test the USB port by running DSK Port test from the start menu

Use Start□Programs□Texas Instruments□Code Composer Studio□Code Composer Studio

CDSK6713 Tools□ DSK6713 Diagnostic Utilities

□ *Select*□ *Start*□*Select DSK*6713 *Diagnostic Utility Icon from Desktop*

□ *Select* ***Start*** *Option*

□ *Utility Program will test the board*

□ *After testing Diagnostic Status you will get* ***PASS***

To create the New Project

Project □ New (File Name. pjt , Eg: **Vectors.pjt**)

To Create a Source file

File □ New □ Type the code (Save & give file name, Eg: **sum.c**).

To Add Source files to Project

Project □ Add files to Project □ c/ccs studio3.1/my projects/your project name/

sum.c(select the file type as c/c++ source files)

To Add rts.lib file & hello.cmd:

Project □ Add files to Project □rts6700.lib

(Path:c/ccs studio3.1/cg tools/c6000/lib/ rts6700.lib**)**

Note: Select Object & Library in(*.o,*.l) in Type of files

Project Add files to Project hello.cmd

CMD file – Which is common for all non real time programs.

(Path: c/ccs studio3.1\tutorial\dsk 6713 \hello1\hello.cmd)

Note: Select Linker Command file(*.cmd) in Type of files

Compile:

To Compile: Project Compile project

To Build: Project build project,

To Rebuild: Project rebuild,

Which will create the final .out executable file.(Eg. Vectors.out).

Procedure to Load and Run program:

Load the program to DSK: File Load program Vectors. out

To Execute project: Debug Run.

1. Execution should halt at break point.

2. Now press F10. See the changes happening in the watch window.

3. Similarly go to view & select CPU registers to view the changes happening in CPU registers.

Configure the graphical window as shown below

INPUT

x[n] = {1, 2, 3, 4,0,0,0,0} h[k] = {1, 2, 3, 4,0,0,0,0}

CONNECTING DSP PROCESSOR TO PC

 Connect the dsp processor to the pc using usb cable connector.

 Check the DSK6713 diagnostics (IF you get the “pass”then click on ok).

 Click on ccs studio3.1 desk top icon. Then the window will be opened.

 Go to debug click on connect (then target device will be connected to pc)

TO CREATE PROJECT

 Project new given project name and select the family’TMS320C67XX’Then click ok

 File new source file write down the ‘c’program and save it with.’c’ extension in current project file

 File new dsp/bios.config file select dsk67xx click on dsk6713 and save it in current project.

 Project add files to project add source file

 Project add files to project add library file by following the given path

- *c/ccs studio3.1/cgtools/c6000/dsk6713/DSK6713.bs/file*.
- Project add files to the project .Add the configuration file.
- Now files are generated and included in generated files . in that open the 3rd file, and copy the header file and paste it in source file. Copy the include files named as "**dsk6713.h" and "dsk6713_aic23.h"** paste it in current project folder.
- Now compile project.(project compile)
- Project build. Project rebuild all.
- File load program project name.pjt debug "project name .out" file click on open debug click on run
- Now apply the input sine wave to line in of dsk6713 kit.
- Observe the output at line out of dsk6713 by using CRO.

INTRODUCTION

MATLAB: MATLAB is a software package for high performance numerical computation and visualization provides an interactive environment with hundreds of built in functions for technical computation, graphics and animation .The MATLAB name stands for Matrix Laboratory.

MATLAB Windows : MATLAB works with through three basic windows

Command Window : This is the main window .it is characterized by MATLAB command prompt >> when you launch the application program MATLAB puts you in this window all commands including those for user-written programs ,are typed in this window at the MATLAB prompt

Graphics window: the output of all graphics commands typed in the command window are flushed to the graphics or figure window, a separate gray window with white background color the user can create as many windows as the system memory will allow

Edit window: This is where you write edit, create and save your own programs in files called M files.

Input-output:

MATLAB supports interactive computation taking the input from the screen and flushing, the output to the screen. In addition it can read input files and write output files

Data Type: the fundamental data –type in MATLAB is the array. It encompasses several distinct data objects- integers, real n umbers, matrices, charcter strings, structures and cells.There is no need to declare variables as real or complex,

MATLAB automatically sets the variable to be real.

Dimensioning: Dimensioning is automatic in MATLAB. No dimension statements are required for vectors or arrays .we can find the dimensions of an existing matrix or a vector with the size and length commands.

Where to work in MATLAB?

All programs and commands can be entered either in the a)Command window

b) As an M file using Matlab editor

Note: Save all M files in the folder 'work' in the current directory. Otherwise you have to locate the file during compiling.

BASIC INSTRUCTIONS

T = 0: 1:10

This instruction indicates a vector T which as initial value 0 and final value 10 with an increment of 1 Therefore T = [0 1 2 3 4 5 6 7 8 9 10]

F = 20: 1: 100 Therefore F = [20 21 22 23 24 …… 100]

T = 0:1/pi: 1 Therefore T= [0, 0.3183, 0.6366, 0.9549]

zeros (1, 3)

The above instruction creates a vector of one row and three columns whose values are zero

Output= [0 0 0]

Syntax: w = conv(u,v)

Description: w = conv(u,v) convolves vectors u and v. Algebraically, convolution is the same operation as multiplying the polynomials whose coefficients are the elements of u and v.

Disp Syntax: disp(X)

Description: disp(X) displays an array, without printing the array name. If X contains a text string, the string is displayed.Another way to display an array on the screen is to type its name, but this prints a leading "X=," which is not always desirable.Note that disp does not display empty arrays.

xlabel

Syntax**: xlabel('string')**

Description: xlabel('string') labels the x-axis of the current axes.

ylabel

Syntax : ylabel('string')

Description: ylabel('string') labels the y-axis of the current axes.

Title

Syntax **: title('string')**

Description: title('string') outputs the string at the top and in the center of the current axes. .grid on

Syntax **: grid on** Description: grid on adds major grid lines to the current axes.

FFT

FFT(X) is the discrete Fourier transform (DFT) of vector X . For matrices, the FFT operation is applied to each column. For N -D arrays, the FFT operation operates on the first non-singleton dimension.

FFT(X,N) is the N-point FFT, padded with zeros if X has less than N points and truncated if it has more.

Starting MATLAB:

After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut *icon* (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains *other* windows. The major tools within or accessible from the desktop are:

- The Command Window
- The Command History
- The Workspace
- The Current Directory
- The Help Browser

- The Start button

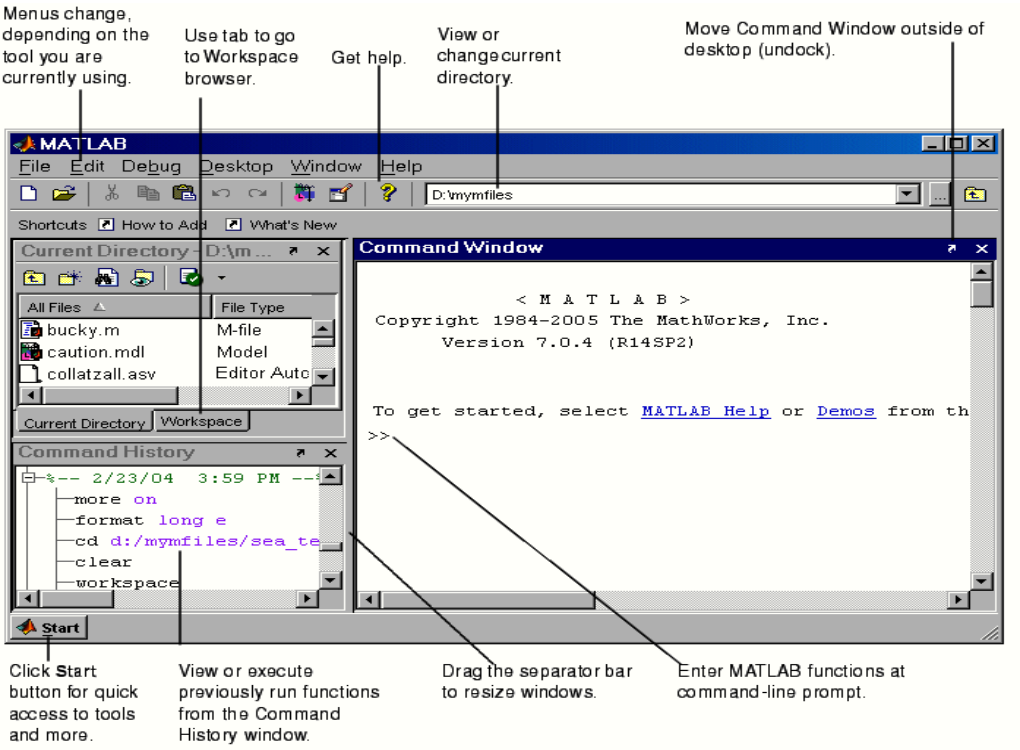


Figure : The graphical interface to the MATLAB workspace

When MATLAB is started for the firsttime, the screen looks like the one that shown in the Figure. This illustration also shows the default configuration of the MATLAB desktop. You cancustomize the arrangement of tools and documents to suit your needs.

Now, we are interested in doing some simple calculations. We will assume that you have sufficient understanding of your computer under which MATLAB is being run. You are now faced with the MATLAB desktop on your computer, which contains the prompt (>>) in the Command Window.

Note: To simplify the notation, we will use this prompt, >>, as a standard prompt sign.

Using MATLAB as a calculator

As an example of a simple interactive calculation, just type the expression you want to evaluate. Let's start at the very beginning. For example, let's suppose you want to calculate the expression, 1+ 2 × 3. You type it at the prompt command (>>) as follows,

```
>> 1+2*3

ans = 7
```

You will have noticed that if you do not specify an output variable, MATLAB uses a default variable ans, short for answer, to store the results of the current calculation. Note that the variable ans is created (or overwritten, if it is already existed). To avoid this, you may assign a value to a variable or outputargument name. For example

```
>> x = 1+2*3

x =7
```

Will result in x being given the value 1 + 2*3=7. This variable name can always

be used to refer to the results of the previous computations. Therefore, computing 4 result in

```
>> 4*x

ans =28.0000
```

3 Quitting MATLAB

To end your MATLAB session, type quit in the Command

Window, or select FileMATLAB in the desktop main menu.

1. Generate the following standard discrete time signals.
i) Unit Impulse ii) Unit step iii) Ramp iv) Exponential v) Sawtooth

AIM: To generate sin signal and plot the same as a waveform showing all the specifications.

APPARATUS: PC with MATLAB Software.

PROCEDURE:

1. Click on the MATLAB Icon on the desktop.
2. MATLAB window open.
3. Click on the 'FILE' Menu on menu bar.
4. Click on NEW M-File from the file Menu.
5. An editor window open, start typing commands.
6. Now SAVE the file in directory.
7. Then Click on DEBUG from Menu bar and Click Run.

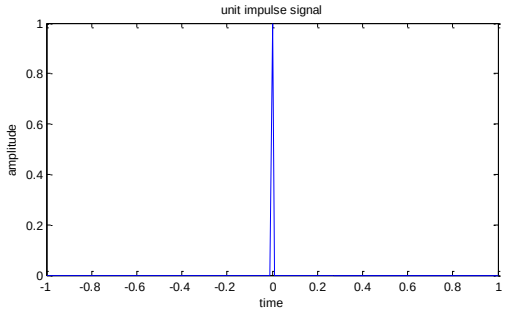
PROGRAM:

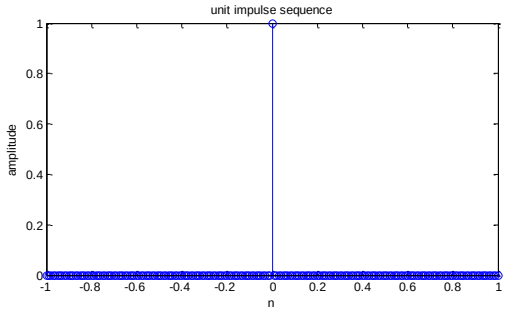
```
% Generation of signals and sequences
clc;
clear all;
close all;
%generation of unit impulse signal
t1=-1:0.01:1
y1=(t1==0);
subplot(2,2,1);
plot(t1,y1);
xlabel('time');
ylabel('amplitude');
title('unit impulse signal');
%generation of impulse sequence
subplot(2,2,2);
stem(t1,y1);
xlabel('n');
ylabel('amplitude');
title('unit impulse sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of unit step signal
t2=-10:0.1:10;
y2=(t2>0);
subplot(2,2,3);
plot(t2,y2);
xlabel('time');
ylabel('amplitude');
title('unit step signal');
%generation of unit step sequence
subplot(2,2,4);
stem(t2,y2);
xlabel('n');
ylabel('amplitude');
title('unit step sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of square wave signal
t=0:0.002:0.1;
y3=square(2*pi*50*t);
```

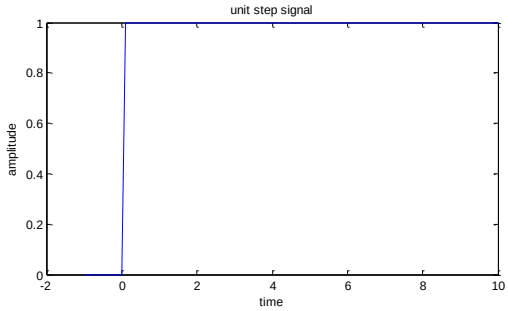
```
figure;
subplot(2,2,1);
plot(t,y3);
axis([0 0.1 -2 2]);
xlabel('time');
ylabel('amplitude');
title('square wave signal');
%generation of square wave sequence
subplot(2,2,2);
stem(t,y3);
axis([0 0.1 -2 2]);
xlabel('n');
ylabel('amplitude');
title('square wave sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of sawtooth signal
y4=sawtooth(2*pi*50*t);
subplot(2,2,3);
plot(t,y4);
axis([0 0.1 -2 2]);
xlabel('time');
ylabel('amplitude');
title('sawtooth wave signal');
%generation of sawtooth sequence
subplot(2,2,4);
stem(t,y4);
axis([0 0.1 -2 2]);
xlabel('n');
ylabel('amplitude');
title('sawtooth wave sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of triangular wave signal
y5=sawtooth(2*pi*50*t,.5);
figure;
subplot(2,2,1);
plot(t,y5);
axis([0 0.1 -2 2]);
xlabel('time');
ylabel('amplitude');
title(' triangular wave signal');
%generation of triangular wave sequence
subplot(2,2,2);
stem(t,y5);
axis([0 0.1 -2 2]);
xlabel('n');
ylabel('amplitude');
title('triangular wave sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of sinsoidal wave signal
y6=sin(2*pi*40*t);
subplot(2,2,3);
plot(t,y6);
axis([0 0.1 -2 2]);
xlabel('time');
ylabel('amplitude');
title(' sinsoidal wave signal');
%generation of sin wave sequence
subplot(2,2,4);
stem(t,y6);
```

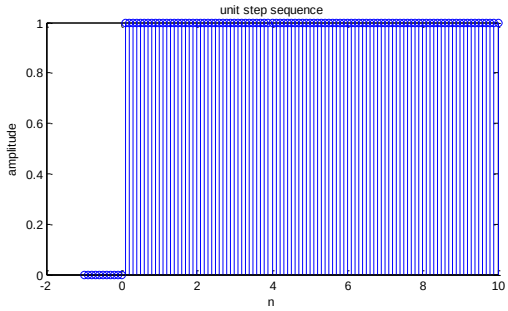
```
axis([0 0.1 -2 2]);
xlabel('n');
ylabel('amplitude');
title('sin wave sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of ramp signal
y7=t;
figure;
subplot(2,2,1);
plot(t,y7);
xlabel('time');
ylabel('amplitude');
title('ramp signal');
%generation of ramp sequence
subplot(2,2,2);
stem(t,y7);
xlabel('n');
ylabel('amplitude');
title('ramp sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of sinc signal
t3=linspace(-5,5);
y8=sinc(t3);
subplot(2,2,1);
plot(t3,y8);
xlabel('time');
ylabel('amplitude');
title(' sinc signal');
%generation of sinc sequence
subplot(2,2,2);
stem(y8);
xlabel('n');
ylabel('amplitude');
title('sinc sequence');
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%generation of exponential signal
x = -3 : 0.5 : 20;
y9= exp (x / 3);
subplot(2,2,3);
plot(x,y9)
xlabel('time');
ylabel('amplitude');
title('exponential signal');
subplot(2,2,4);
stem(x,y9);
xlabel('n');
ylabel('amplitude');
title('exponential sequence');
```

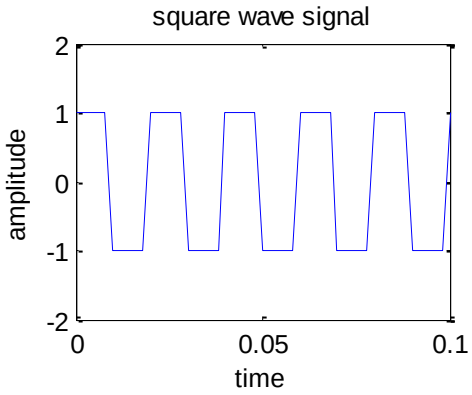
FIGURES : -

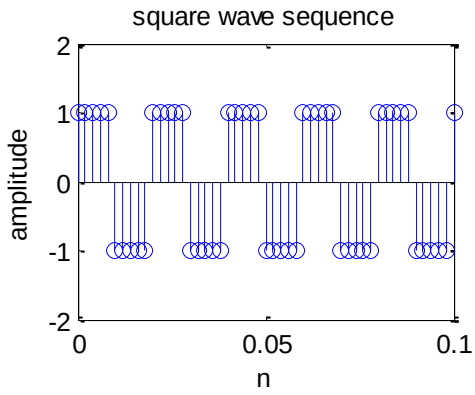

unit impulse signal
amplitude
time

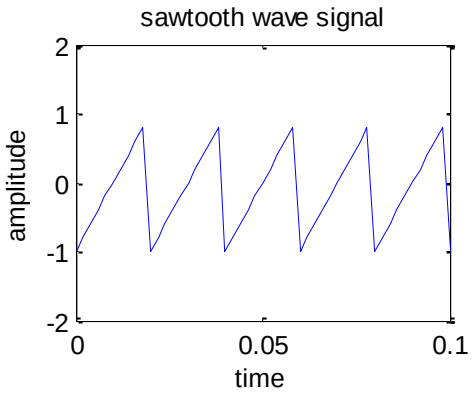

unit impulse sequence
amplitude
n

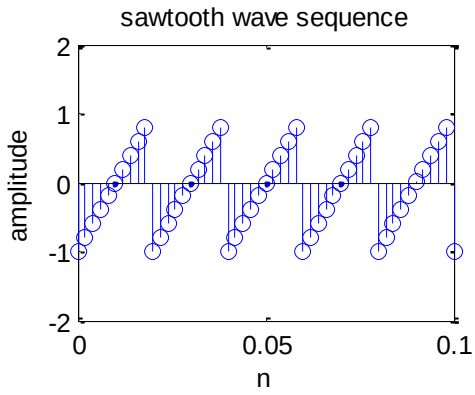

unit step signal
amplitude
time

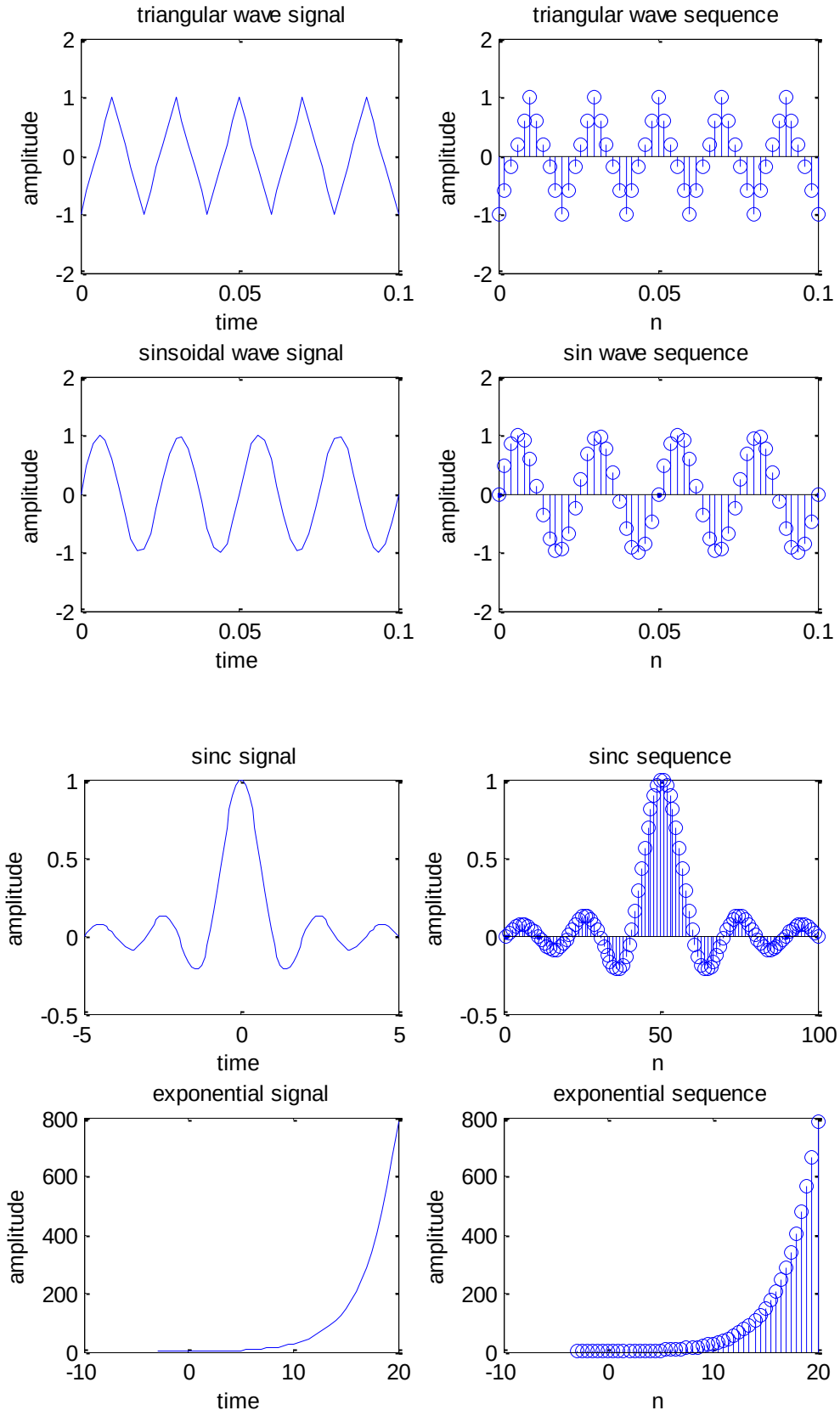

unit step sequence
amplitude
n


square wave signal
amplitude
time


square wave sequence
amplitude
n


sawtooth wave signal
amplitude
time


sawtooth wave sequence
amplitude
n



Result: Various signals & sequences generated using Matlab software.

2) Generate sum of two sinusoidal signals and find the frequency response (magnitude and phase).

AIM: To generate sum of two sinusoidal signals and find the frequency response (magnitude and phase)

APPARATUS: PC with MATLAB Software.

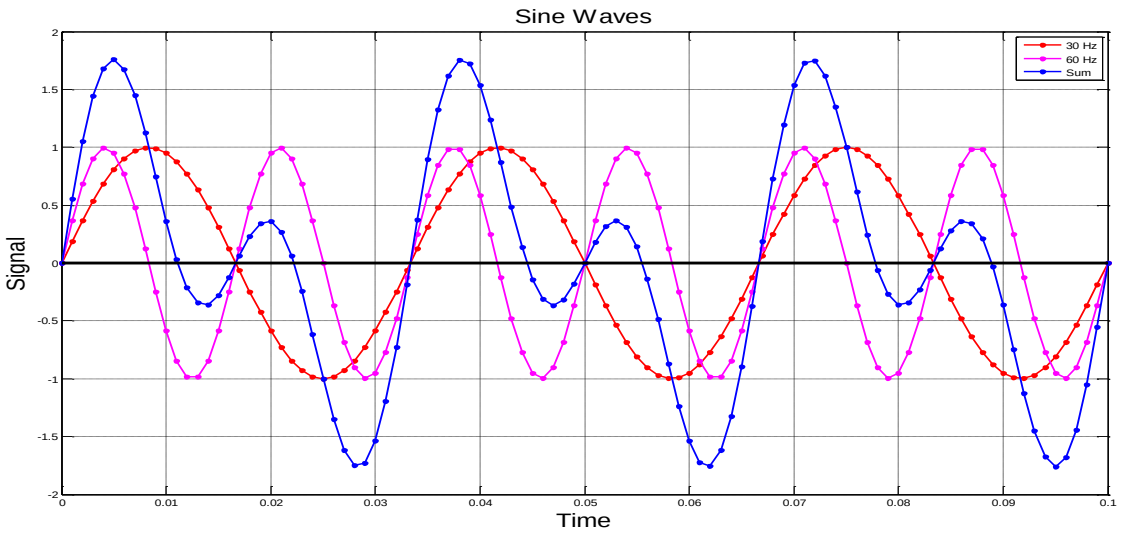
PROCEDURE:

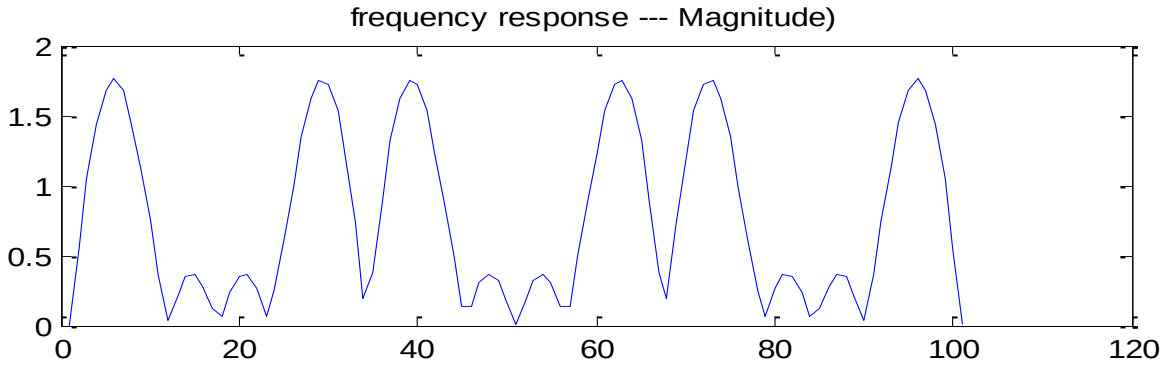
1. Click on the MATLAB Icon on the desktop.
2. MATLAB window open.
3. Click on the 'FILE' Menu on menu bar.
4. Click on NEW M-File from the file Menu.
5. An editor window open, start typing commands.
6. Now SAVE the file in directory.
7. Then Click on DEBUG from Menu bar and Click Run.

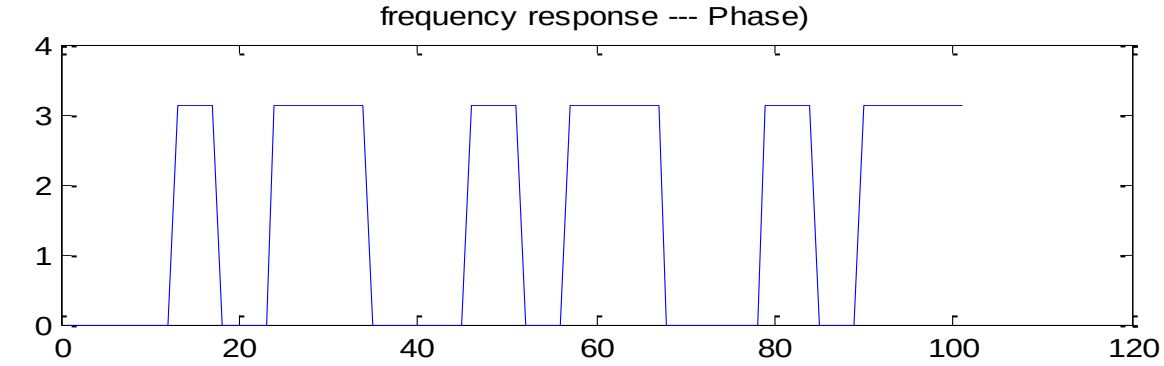
PROGRAM 1:

```
clc; % Clear the command window.
close all; % Close all figures (except those of imtool.)
clear; % Erase all existing variables. Or clearvars if you want.
workspace; % Make sure the workspace panel is showing.
format long g;
format compact;
fontSize = 20;

% Make 0.1 seconds sampled every 1/1000 of a second
t = 0 : 1/1000 : 0.1;

% Define sine wave parameters.
f1 = 30; % per second
T1 = 1/f1; % period, seconds
amp1 = 1; % amplitude
f2 = 60; % per second
T2 = 1/f2; % period, seconds
amp2 = 1; % amplitude
 % Make signals.
signal1 = amp1 * sin(2*pi*t/T1);
signal2 = amp2 * sin(2*pi*t/T2);
signal = signal1 + signal2;
plot(t, signal1, 'r.-', 'LineWidth', 2, 'MarkerSize', 16);
hold on;
plot(t, signal2, 'm.-', 'LineWidth', 2, 'MarkerSize', 16);
plot(t, signal, 'b.-', 'LineWidth', 2, 'MarkerSize', 16);
grid on;
title('Sine Waves', 'FontSize', fontSize);
xlabel('Time', 'FontSize', fontSize);
ylabel('Signal', 'FontSize', fontSize);
% Make bolder x axis
line(xlim, [0,0], 'Color', 'k', 'LineWidth', 3);
legend('30 Hz', '60 Hz', 'Sum');
```

```
Z=abs(signal)
A=angle(signal)
figure
subplot(2,1,1);
plot(Z)
title('frequency response --- Magnitude)');
subplot(2,1,2);
plot(A)
title('frequency response --- Phase)');
```

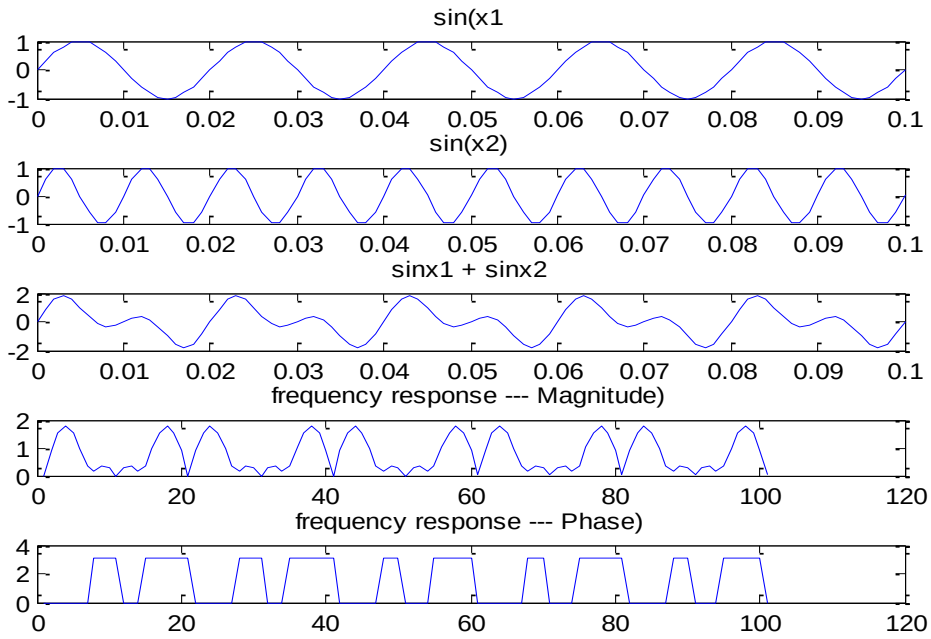






PROGRAM 2:

```
clc;
clear all;
close all;
t=0:0.001:0.1;
f1=50;
x1=2*pi*f1*t;
y1=sin(x1);
subplot (5,1,1);
plot (t,y1);
title('sin(x1');
f2=100;
x2=2*pi*f2*t;
y2=sin(x2);
subplot(5,1,2);
plot(t,y2);
title('sin(x2)');
y=y1+y2;
subplot(5,1,3);
plot(t,y);
title('sinx1 + sinx2')
Z=abs(y)
A=angle(y)
subplot(5,1,4);
plot(Z)
title('frequency response --- Magnitude)');
subplot(5,1,5);
plot(A)
title('frequency response --- Phase)');
```



Result: Generated sum of two sinusoidal signals and find the frequency response (magnitude and phase).

3. Implement and verify linear and circular convolution between two given signals.

AIM: - To Implement and verify linear and circular convolution between two given signals.

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:- Linear convolution.

```
clc;  clear all;  close all;
x=input ('enter the input sequence');
h=input ('enter the impulse response');
n1=length(x); disp (n1)
n2=length (h); disp (n2)
n3=n1+n2-1;
disp('the resultant length is:');n3
y=conv(x,h);
subplot (3,1,1);
t1=0: 1: (n1-1);
stem(t1,x);  xlabel('n');  ylabel('x(n)');  title ('input sequence');
subplot (3,1,2);
t2=0: 1: (n2-1);
stem (t2,h);
xlabel('n');  ylabel ('h(n)');  title ('impulse response');
subplot (3,1,3);
t3=0 : 1 : ( n3-1);
stem (t3,y);  xlabel('n');  ylabel('y(n)');  title('linear convoluted response');
disp('the resultant signal is:');y
```

enter the input sequence[2 4 5 6 8]

enter the impulse response[4 2 3 6 5]

5

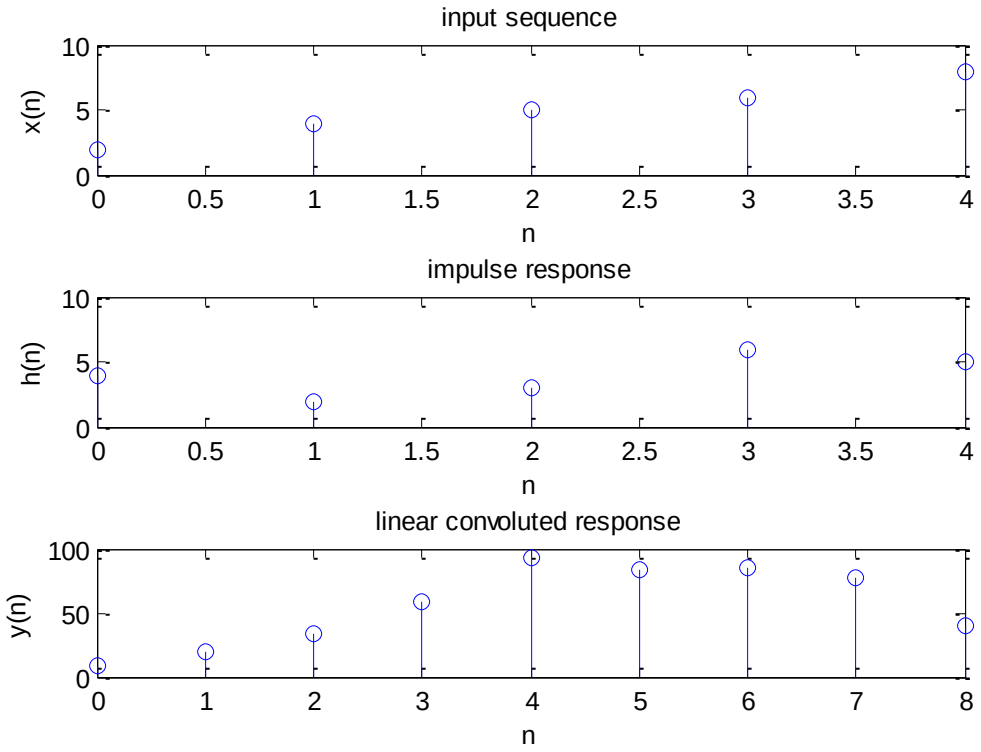
5

The resultant length is:

n3 = 9

The resultant signal is:

y = 8 20 34 58 93 84 85 78 40



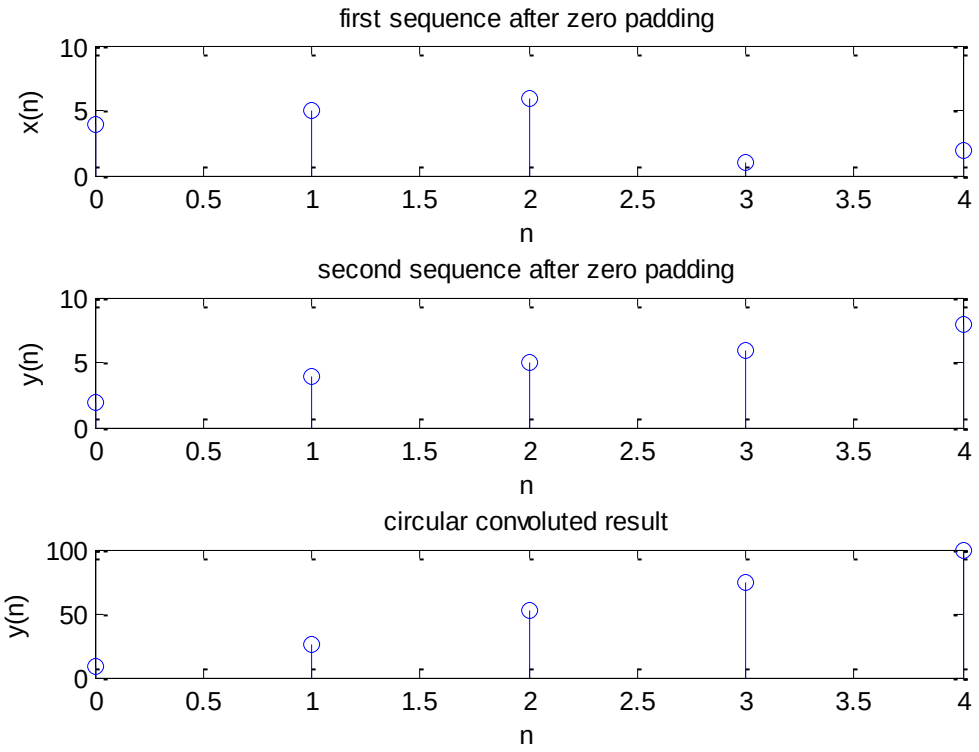
RESULT: - Thus the program for linear convolution is written using MATLAB and verified.

CIRCULAR CONVOLUTION USING MATLAB

PROGRAM:-

```
clc;  clear all;
x=input ('enter first sequence');
y=input ('enter second sequence');
n1=length(x);
n2=length(y);
N=max (n1, n2);
n3=n1-n2;
if n3>=0
y= [y,zeros(1,n3)];
else
 x=[x,zeros(1,n3)];
end
a=length(x);
subplot (3,1,1);
t1=0: 1: (a-1);
stem (t1,x);  xlabel('n');  ylabel('x(n)');
title ('first sequence after zero padding');
subplot (3,1,2);
```

```
b=length(y);
t2=0:1 :( b-1);
stem (t2,y);  xlabel('n');  ylabel('y(n)');
title ('second sequence after zero padding');
for n=1:N
k (n)=0;
for i=1:N
j=n-i+1;
if j<=0
j=N+j;
else
k(n)=k(n)+x(i)*y(j)
end;
end;
end;
subplot (3,1,3);
t3=0: 1: (a-1);
stem (t3,k);  xlabel('n');  ylabel('y(n)');  title('circular convoluted result');
```




```
enter first sequence[4 5 6 1 2]

enter second sequence[2 4 5 6 8]

k = 8    26    52    75   100
```

RESULT:- Thus the program for circular convolution is written using MATLAB and verified.

4 .Auto Correlation and Cross Correlation

AIM: To Implement and verify autocorrelation for the given sequence and cross correlation between two given signals.

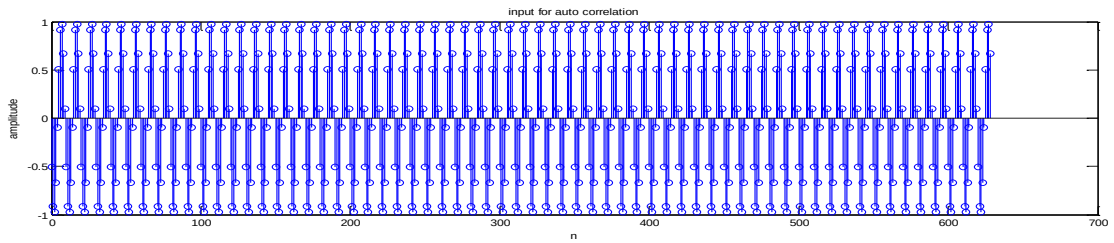
APPARATUS: PC with MATLAB Software.

PROCEDURE:

1 Click on the MATLAB Icon on the desktop.
2 MATLAB window open.
3 Click on the 'FILE' Menu on menu bar.
4 Click on NEW M-File from the file Menu.
5 An editor window open, start typing commands.
6 Now SAVE the file in directory.
7 Then Click on DEBUG from Menu bar and Click Run.

Program for Auto Correlation of a sequence

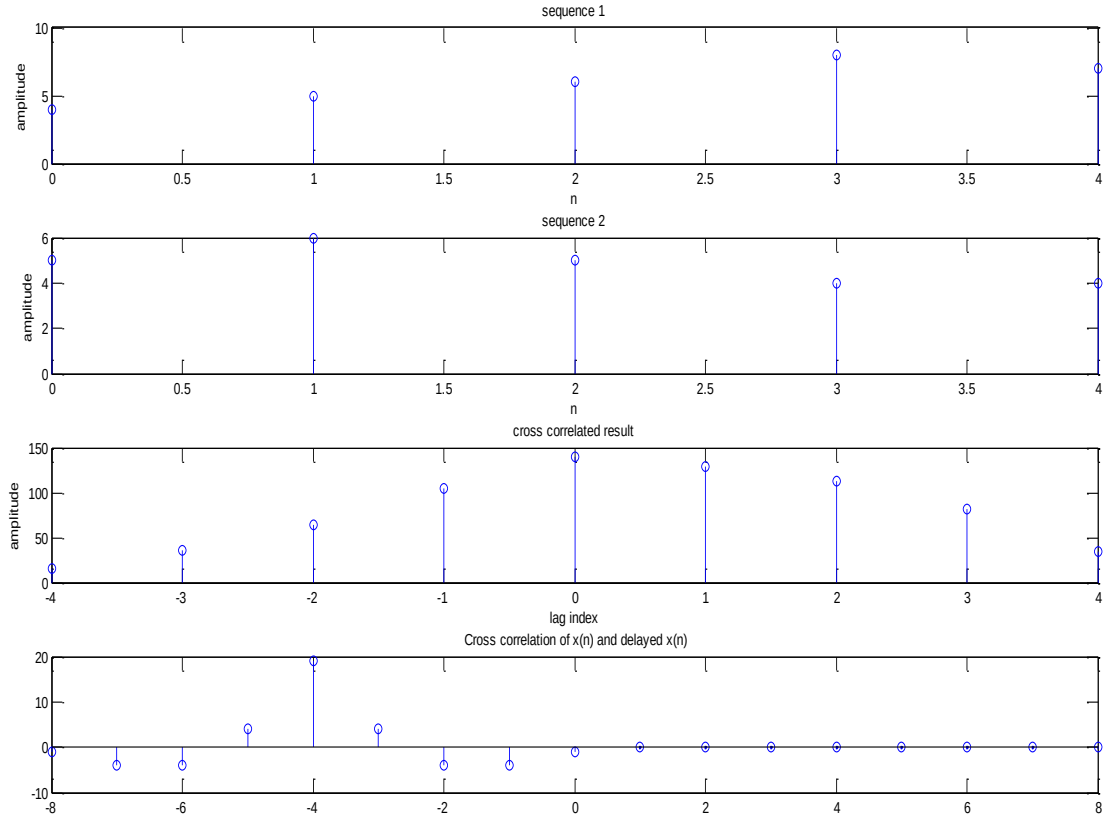
```
clc;  clear all;  close all;
t= -pi: 0.01: pi;
f=input ('enter the fundamental freq:');
x=sin (2*pi*f*t);
n2= (length(x)-1)
k= (-n2):n2';
subplot (2,1,1);
t1=0:1:n2;
stem (t1,x);
xlabel('n');    ylabel('amplitude');
title ('input for auto correlation');
r=xcorr(x,x);
subplot (2,1,2);
stem (k,r);     xlabel('lag index');    ylabel('amplitude');
 title ('auto correlated result');

OUTPUT:--
enter the fundamental freq:10
```



Program for Cross correlation of sequences:

```
clc;  clear all;  close all;
x=input ('enter the first seq');    y=input ('enter the second seq');
n1= (length(y)-1)
n2= (length(x)-1)
k= (-n1):n2';
subplot (4,1,1);
t1=0:1:n2;
stem (t1,x);
xlabel('n');  ylabel('amplitude');  title ('sequence 1');
subplot (4,1,2);
t2=0:1:n1;
stem(t2,y);
xlabel('n');  ylabel('amplitude');  title('sequence 2');
r=xcorr(x,y);
subplot (4,1,3);
stem (k , r);
xlabel('lag index');  ylabel('amplitude');
title ('cross correlated result');
disp('cross correlated result is:');r
%Program for cross correlation of signal x( n) and delayed x(n)
x = [1 2 3 -2 -1];
D= input (' Enter the delay ');
xd = [zeros(1,D),x];
[r,lag] = xcorr(x,xd);
subplot(4,1,4)
stem(lag,r);
title(' Cross correlation of x(n) and delayed x(n)');
```

```
enter the first seq[4 5 6 8 7]

enter the second seq[5 6 5 4 4]

n1 =

     4

n2 =

     4

cross correlated result is:

r =

   16.0000   36.0000   64.0000  105.0000  140.0000  129.0000  113.0000   82.0000   35.0000

 Enter the delay 4
```



RESULT: Autocorrelation for the given sequence and cross correlation between two given signals verified.

5. Compute and implement the N-point DFT of a given sequence and compute the power density spectrum of the sequence.

AIM: To find the DFT of given signal.

APPARATUS: PC with MATLAB Software.

PROCEDURE:

1 Click on the MATLAB Icon on the desktop.
2 MATLAB window open.
3 Click on the 'FILE' Menu on menu bar.
4 Click on NEW M-File from the file Menu.
5 An editor window open, start typing commands.
6 Now SAVE the file in directory.
7 Then Click on DEBUG from Menu bar and Click Run.

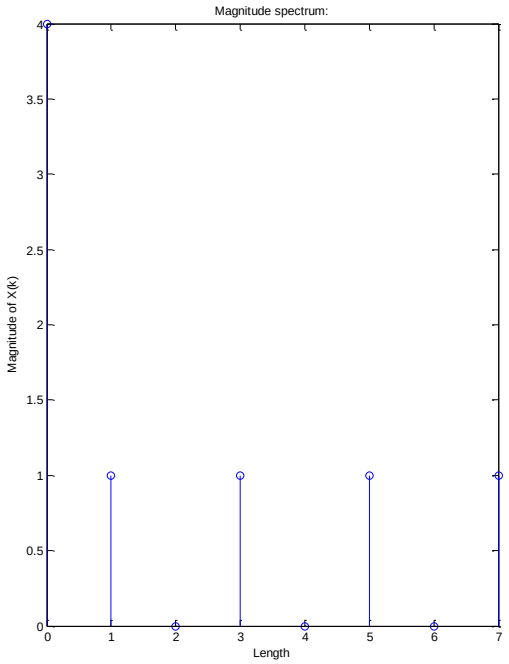
PROGRAM: WITH BUILT IN FUNCTION:

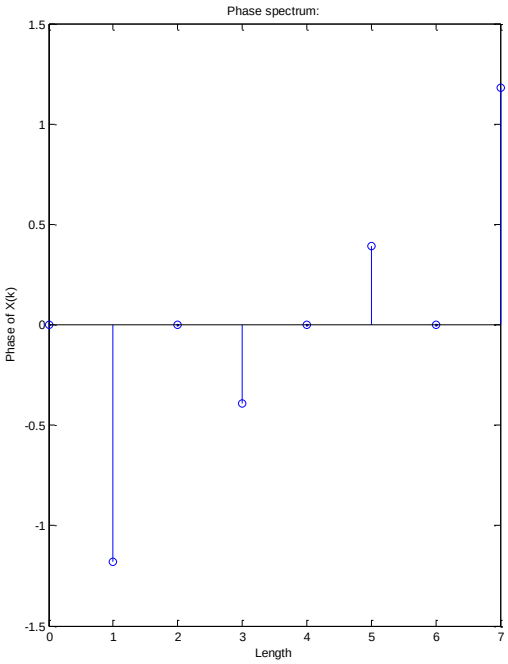
```
clc;
x1 = input('Enter the sequence:');
n = input('Enter the length:');
m = fft(x1,n);
disp('N-point DFT of a given sequence:');disp(m);
N = 0:1:n-1; subplot(1,2,1);
stem(N,m); xlabel('Length');
ylabel('Magnitude of X(k)');
title('Magnitude spectrum:');
an = angle(m);
subplot(1,2,2);
stem(N, an); xlabel('Length');
ylabel('Phase of X(k)');
title('Phase spectrum:');
```

OUTPUT:

```
Enter the sequence:[1 1 1 1 0 0 0
0]
Enter the length:8
N-point DFT of a given sequence:
  Columns 1 through 5

   4.0000            1.0000 - 2.4142i
0            1.0000 - 0.4142i        0

  Columns 6 through 8
```

1.0000 + 0.4142i 0
1.0000 + 2.4142i





PROGRAM: WITH OUT BUILTIN FUNCTION:

```
N = input('Enter the the value of N(Value of N in N-Point DFT)');

x = input('Enter the sequence for which DFT is to be calculated');

n=[0:1:N-1];

k=[0:1:N-1];

WN=exp(-1j*2*pi/N); % twiddle factor

nk=n'*k;

WNnk=WN.^nk;

Xk=x*WNnk;

disp(Xk);

MagX=abs(Xk) % Magnitude of calculated DFT

PhaseX=angle(Xk)*180/pi % Phase of the calculated DFT

figure(1);

subplot(2,1,1);
```

```
title('MAGNITUDE PLOT ');
plot(k,MagX);
subplot(2,1,2);
plot(k,PhaseX)
title('PHASE PLOT ')
```

OUTPUT

Enter the the value of N(Value of N in N-Point DFT)8

Enter the sequence for which DFT is to be calculated[1 1 1 1 0 0 0 0]

Columns 1 through 5

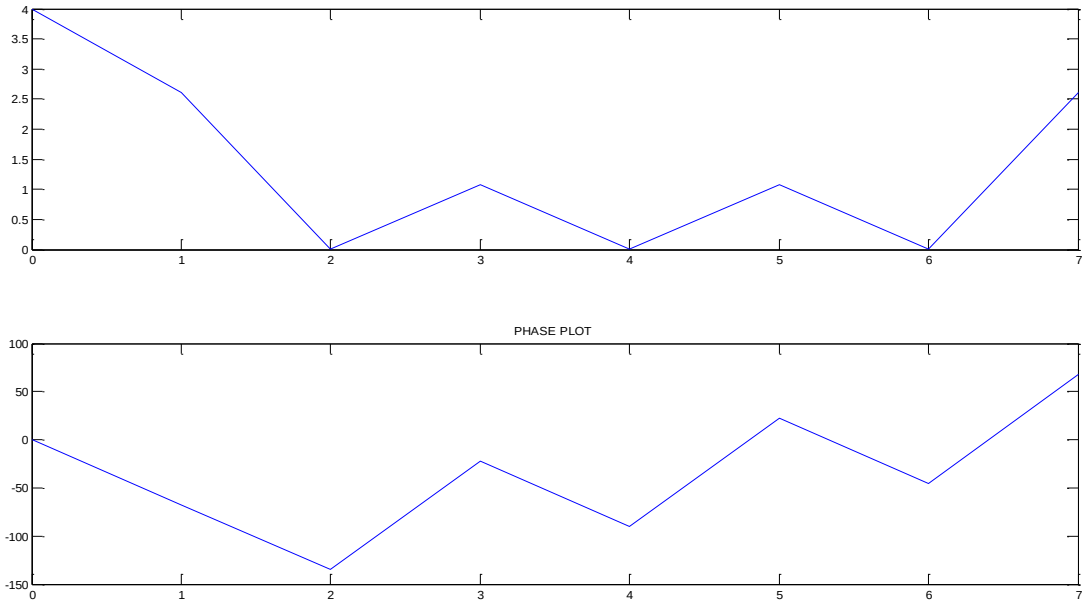
4.0000 1.0000 - 2.4142i -0.0000 - 0.0000i 1.0000 - 0.4142i 0 - 0.0000i

Columns 6 through 8

1.0000 + 0.4142i 0.0000 - 0.0000i 1.0000 + 2.4142i

MagX = 4.0000 2.6131 0.0000 1.0824 0.0000 1.0824 0.0000 2.6131

PhaseX = 0 -67.5000 -135.0000 -22.5000 -90.0000 22.5000 -45.0000 67.5000



RESULT: The DFT of given sequence is obtained. Hence the theory and practical value are proved

6. Implement and verify N-point DIT-FFT of a given sequence and find the frequency response (magnitude and phase).

AIM: To perform the FFT of signal x(n) using Mat lab.

APPARATUS: PC with MATLAB Software.

PROCEDURE:

1 Click on the MATLAB Icon on the desktop.

2 MATLAB window open.

3 Click on the 'FILE' Menu on menu bar.

4 Click on NEW M-File from the file Menu.

5 An editor window open, start typing commands.

6 Now SAVE the file in directory.

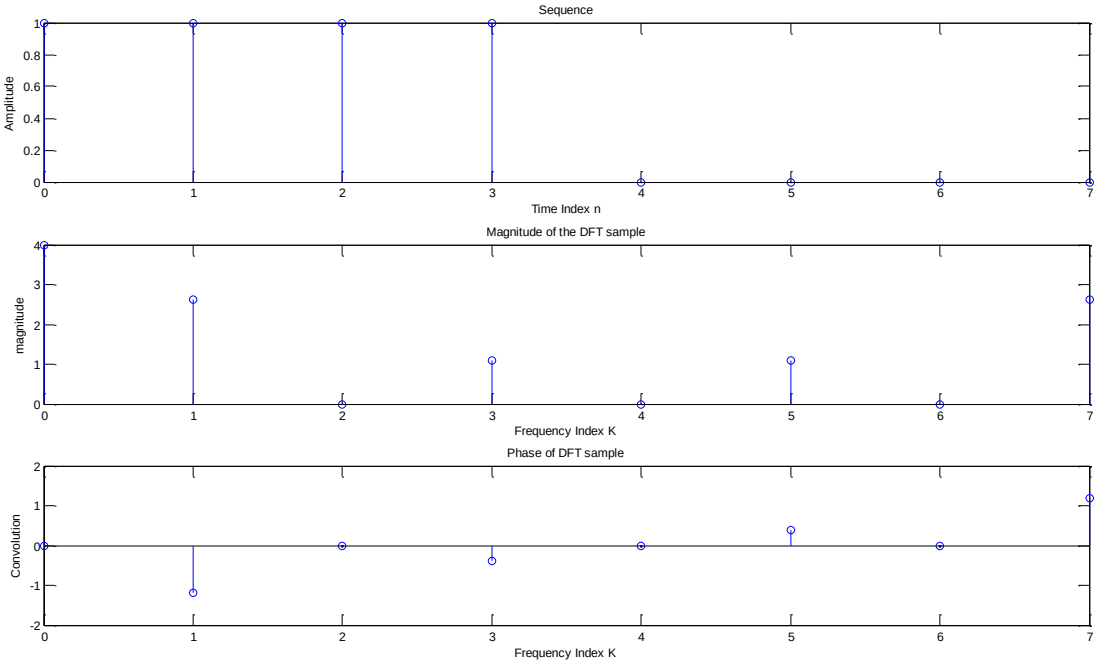
7 Then Click on DEBUG from Menu bar and Click Run.

PROGRAM:

```
clear all; N=8; m=8;
a=input('Enter the input sequence'); n=0:1:N-1;
subplot(3,1,1);
stem(n,a); xlabel('Time Index n'); ylabel('Amplitude'); title('Sequence');
x=fft(a,m);
k=0:1:N-1;
subplot(3,1,2);
stem(k,abs(x)); ylabel('magnitude'); xlabel('Frequency Index K');
title('Magnitude of the DFT sample'); subplot(3,1,3);
stem(k,angle(x)); xlabel('Frequency Index K'); ylabel('Phase');
title('Phase of DFT sample');
```

OUTPUT:-

Enter the input sequence[1 1 1 1 0 0 0 0]



RESULT: - Thus Fast Fourier Transform is Performed using Matlab.

7. Implement and verify N-point IFFT of a given sequence.

AIM: To perform the IFFT of signal x(n) using Mat lab.

APPARATUS: PC with MATLAB Software.

PROCEDURE:

1 Click on the MATLAB Icon on the desktop.

2 MATLAB window open.

3 Click on the 'FILE' Menu on menu bar.

4 Click on NEW M-File from the file Menu.

5 An editor window open, start typing commands.

6 Now SAVE the file in directory.

7 Then Click on DEBUG from Menu bar and Click Run.

PROGRAM :

```
clear all; N=8; m=8;
a=input('Enter the input sequence'); n=0:1:N-1;
subplot(3,1,1);
stem(n,a); xlabel('Time Index n'); ylabel('Amplitude'); title('Sequence');
x=ifft(a,m);
k=0:1:N-1;
subplot(3,1,2);
stem(k,abs(x)); ylabel('magnitude'); xlabel('Frequency Index K');
title('Magnitude of the IFFT sample'); subplot(3,1,3);
stem(k,angle(x)); xlabel('Frequency Index K'); ylabel('Phase');
title('Phase of IFFT sample');
disp(x)
```

OUTPUT

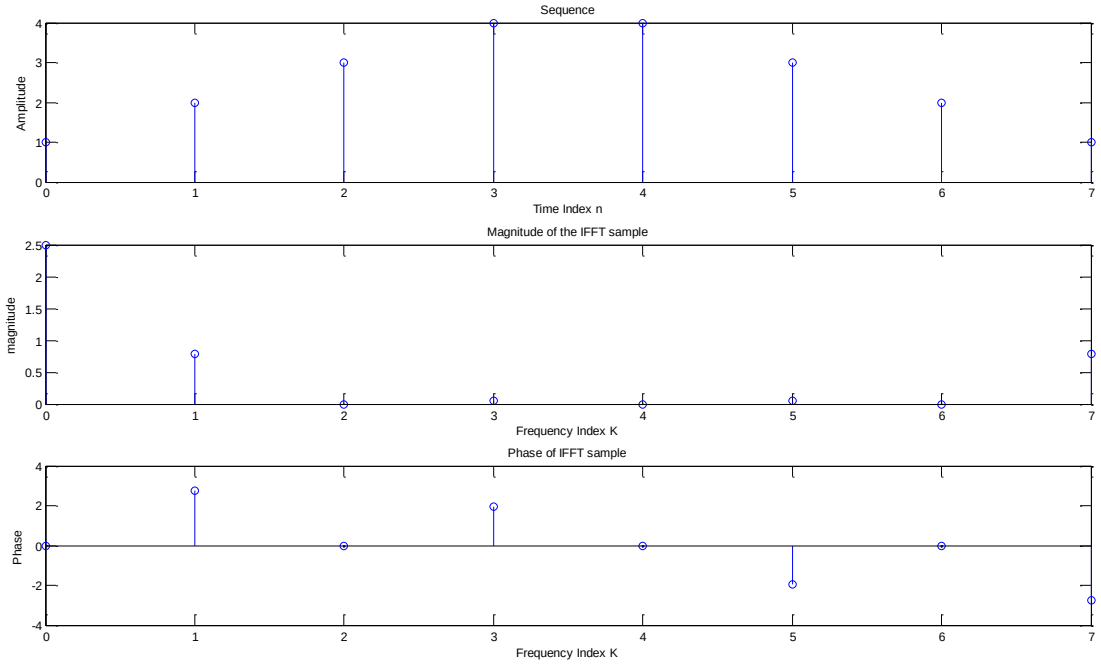
Enter the input sequence[1 2 3 4 4 3 2 1]

Columns 1 through 5

2.5000 -0.7286 + 0.3018i 0 -0.0214 + 0.0518i 0

Columns 6 through 8

-0.0214 - 0.0518i 0 -0.7286 - 0.3018i


Sequence
Amplitude
Time Index n
Magnitude of the IFFT sample
magnitude
Frequency Index K
Phase of IFFT sample
Phase
Frequency Index K

8) Design And Implementation of Iir Butterworth (LP/HP) Filter.

AIM: To write a matlab program for design IIR Butterworth filter and compare their performances with different orders (Low Pass Filter /High Pass Filter).

APPARATUS: PC with MATLAB Software.
PROCEDURE:

1 Click on the MATLAB Icon on the desktop.
2 MATLAB window open.
3 Click on the 'FILE' Menu on menu bar.
4 Click on NEW M-File from the file Menu.
5 An editor window open, start typing commands.
6 Now SAVE the file in directory.
7 Then Click on DEBUG from Menu bar and Click Run.

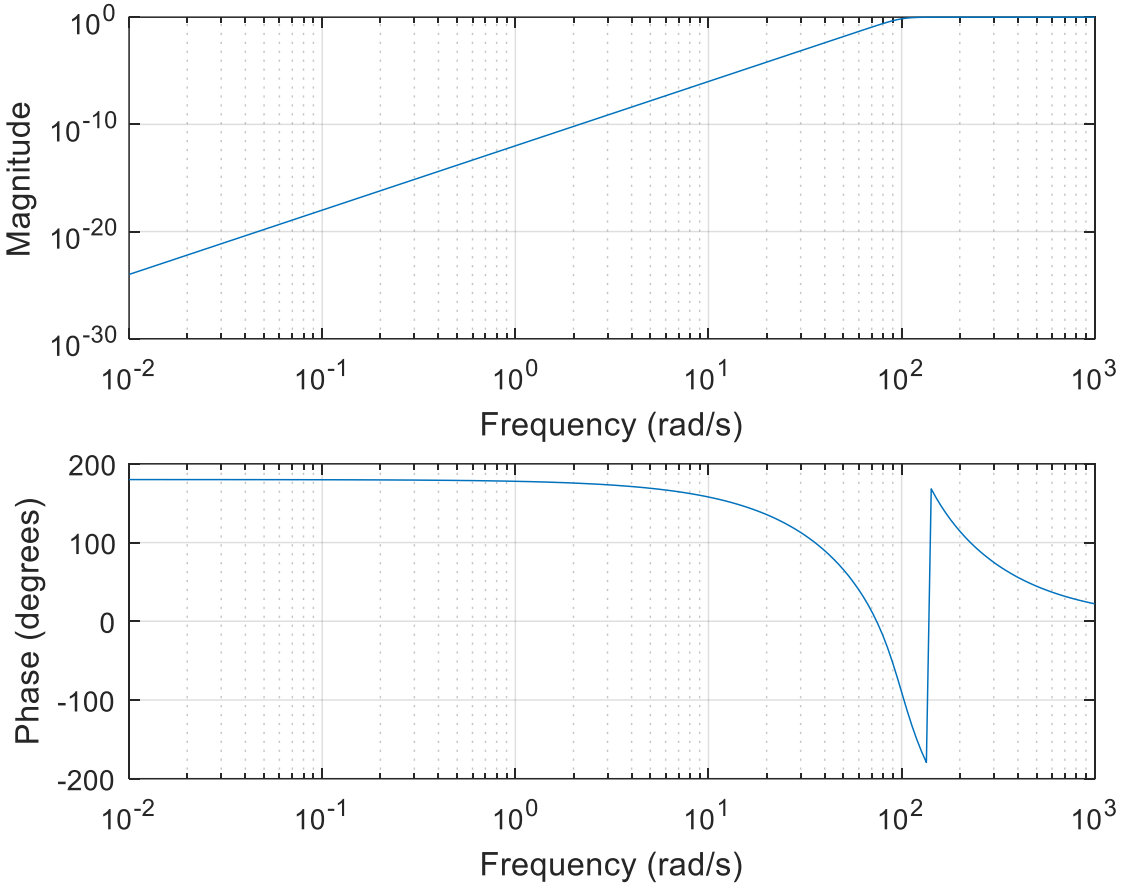
PROGRAM:

```
clc;

clear all;

close all;

disp('Enter the Analog filter design specifications');

N=input('Enter the order of the filter');

c=input('Enter the choice of filter 1. LPF 2. HPF 3.BPF 4.BSF \n ');

if(c==1)

disp('Frequency response of Analog LPF is:');

Cf=100;

[b,a]=butter(N,Cf,'S');

freqs(b,a);

end

if(c==2)

disp('Frequency response of Analog HPF is:');

Cf=100;

[b,a]=butter(N,Cf,'HIGH','S');
```

```
freqs(b,a);

end

if(c==3)

disp('Frequency response of Analog BPF is:');

Cf1=[10 100];

[b,a]=butter(N,Cf1,'S');

freqs(b,a);

end

if(c==4)

disp('Frequency response of Analog BPF is:');

Cf1=[10 100];

[b,a]=butter(N,Cf1,'STOP','S');

freqs(b,a);

end
```

OUTPUT :

```
Enter the Analog filter design specifications

Enter the order of the filter6

Enter the choice of filter 1. LPF 2. HPF 3.BPF 4.BSF

 2

Frequency response of Analog HPF is:
```



RESULT: -- Analog filters using Matlab is designed.

9) Design And Implementationof Iir Chebyshev(Lp/Hp) Filter

AIM: To writa a matlab program for design IIR Chebyshev filter and compare their performances with different orders (Low Pass Filter /High Pass Filter).

APPARATUS: PC with MATLAB Software.

PROCEDURE:

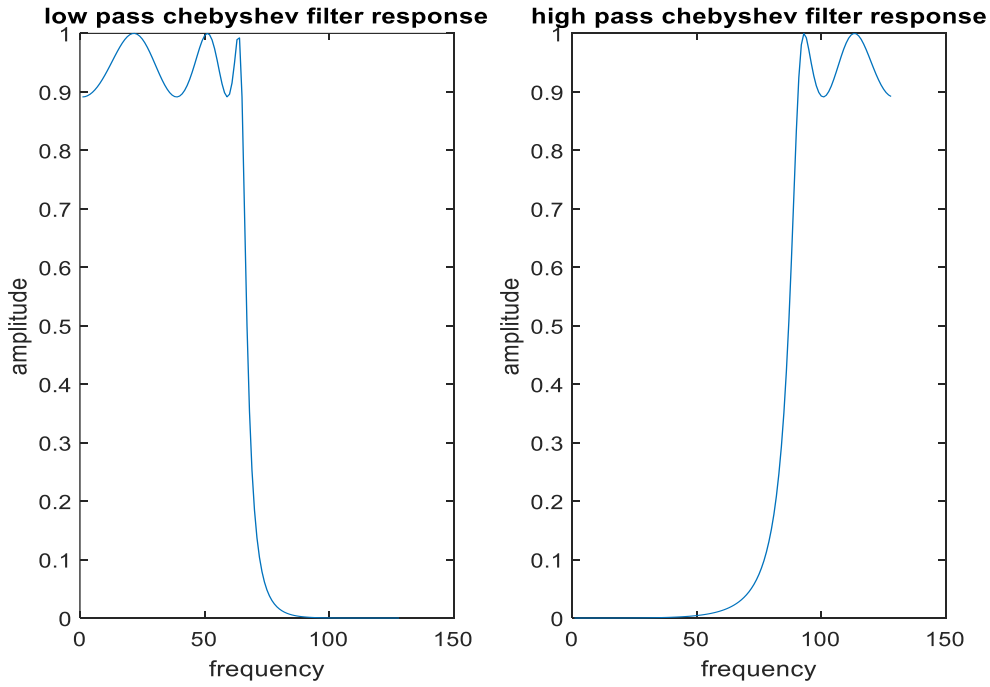
1 Click on the MATLAB Icon on the desktop.
2 MATLAB window open.
3 Click on the 'FILE' Menu on menu bar.
4 Click on NEW M-File from the file Menu.
5 An editor window open, start typing commands.
6 Now SAVE the file in directory.
7 Then Click on DEBUG from Menu bar and Click Run.

PROGRAM :

```
% chebyshev low pass filter
clc;

clear all;

wp=0.5;

%%% chebyshev low pass filter;

ws=0.7;

rp=1;

rs=50;

[n,wn]=cheb1ord(wp,ws,rp,rs);

[b,a]=cheby1(n,rp,wn);

[h,w]=freqz(b,a,128);

subplot(1,2,1);

plot(abs(h)); xlabel('frequency'); ylabel('amplitude');

title('low pass chebyshev filter response');

%%% chebyshev high pass filter

wp=0.7;

ws=0.5;

rp=1;

rs=30;
```

```
[n,wn]=cheb1ord(wp,ws,rp,rs);

[b,a]=cheby1(n,rp,wn, 'high');

[h,w]=freqz(b,a,128);

subplot(1,2,2);

plot(abs(h));

xlabel('frequency');

ylabel('amplitude');

title('high pass chebyshev filter response')
```

Output :



RESULT : Design and implementation of iir chebyshev (lp/hp) filter using matlab is verified.

VIVA QUESTION:

1. What do you mean by cut-off frequency?
2. Give the difference between analog and digital filter?
3. What is the difference between type 1 and type 2 filter structure?
4. what is the role of delay element in filter design?
5. Explain how the frequency is filter in filters?
6. Differences between Butterworth chebyshev filters?
7. Can IIR filters be Linear phase? how to make it linear Phase?

10) Design FIR filter (Low Pass Filter /High Pass Filter) using windowing technique.

i. **Using rectangular window**
ii. **Using hamming window**
iii. **Using Kaiser window**

AIM :- To Design FIR LP Filter using Rectangular/Hamming/kaiser Windowing Technique

APPARATUS: PC with MATLAB Software.

PROCEDURE:

1 Click on the MATLAB Icon on the desktop.

2 MATLAB window open.

3 Click on the 'FILE' Menu on menu bar.

4 Click on NEW M-File from the file Menu.

5 An editor window open, start typing commands.

6 Now SAVE the file in directory.

7 Then Click on DEBUG from Menu bar and Click Run.

ALGORITHM:-

1) Enter the pass band ripple (rp) and stop band ripple (rs).
2) Enter the pass band frequency (fp) and stop band frequency (fs).
3) Get the sampling frequency (f), beta value.
4) Calculate the analog pass band edge frequencies, w1 and w2.

w1 = 2*fp/f

w2 = 2*fs/f

5) calculate the numerator and denominator
6) Use an If condition and ask the user to choose either Rectangular Window or Triangular

window or Kaiser window..

7) use rectwin,triang,kaiser commands
8) Calculate the magnitude of the frequency response in decibels (dB

m=20*log10(abs(h))

9) Plot the magnitude response [magnitude in dB Vs normalized frequency (om/pi)]

10) Give relevant names to x and y axes and give an appropriate title for the plot.

11) Plot all the responses in a single figure window.[Make use ofsubplot]

PROGRAM:-1. LOWPASS FILTER

```
clc;
clear all; close all;
rp=input('enter passband ripple'); rs=input('enter the stopband ripple');
fp=input('enter passband freq'); fs=input('enter stopband freq');
f=input('enter sampling freq '); beta=input('enter beta value');
wp=2*fp/f; ws=2*fs/f;
num=-20*log10(sqrt(rp*rs))-13; dem=14.6*(fs- fp)/f; n=ceil(num/dem);
n1=n+1;
if(rem(n,2)~=0) n1=n;
n=n-1; end
c=input('Enter your choice of window function 1. rectangular 2. Hamming 3.kaiser: \n
'); if(c==1)
y=rectwin(n1);
disp('Rectangular window filter response'); end
if (c==2) y=hamming(n1);
disp('Hamming window filter response'); end
if(c==3) y=kaiser(n1,beta);
disp('kaiser window filter response'); end
%LPF
b=fir1(n,wp,y); [h,o]=freqz(b,1,256); m=20*log10(abs(h)); plot(o/pi,m);
title('LPF');
ylabel('Gain in dB-->');
xlabel('(a) Normalized frequency-->');
```

Output:-

enter passband ripple0.02

enter the stopband ripple0.01

enter passband freq1000

enter stopband freq2000

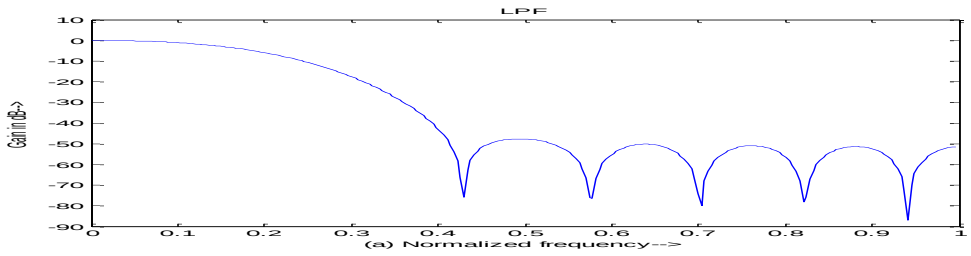
enter sampling freq 10000

enter beta value0.2

Enter your choice of window function 1. Rectangular 2. Hamming 3.kaiser:

2

Hamming window filter response



PROGRAM:-2. HIGH PASS FILTER

ALGORITHM:-

1) Enter the pass band ripple (rp) and stop band ripple (rs).
2) Enter the pass band frequency (fp) and stop band frequency (fs).
3) Get the sampling frequency (f), beta value.
4) Calculate the analog pass band edge frequencies, w1 and w2.

$$w1 = 2*fp/f$$

$$w2 = 2*fs/f$$

5) calculate the numerator and denominator
6) Use an If condition and ask the user to choose either Rectangular Window or Triangular window or Kaiser window..
7) use rectwin,triang,kaiser commands
8) Calculate the magnitude of the frequency response in decibels (dB

$$m=20*\log10(abs(h))$$

9) Plot the magnitude response [magnitude in dB Vs normalized frequency (om/pi)]
10) Give relevant names to x and y axes and give an appropriate title for the plot.

11) Plot all the responses in a single figure window.[Make use of subplot]

PROGRAM:-

```
%FIR Filter design window techniques (HIGH PASS FILTER )
clc;
clear all; close all;
rp=input('enter passband ripple'); rs=input('enter the stopband ripple');
fp=input('enter passband freq'); fs=input('enter stopband freq'); f=input('enter
sampling freq '); beta=input('enter beta value');
wp=2*fp/f; ws=2*fs/f;
num=-20*log10(sqrt(rp*rs))-13; dem=14.6*(fs- fp)/f; n=ceil(num/dem);
n1=n+1;
if(rem(n,2)~=0) n1=n;
n=n-1; end
c=input('enter your choice of window function 1. rectangular 2. Hamming 3.kaiser: \n
'); if(c==1)
y=rectwin(n1);
disp('Rectangular window filter response'); end
if (c==2) y=triang(n1);
disp('Hamming window filter response'); end
if(c==3) y=kaiser(n1,beta);
disp('kaiser window filter response'); end
%HPF
b=fir1(n,wp,'high',y);
[h,o]=freqz(b,1,256); m=20*log10(abs(h)); plot(o/pi,m);
title('HPF');
ylabel('Gain in dB-->');
xlabel('(b) Normalized frequency-->');
```

OUTPUT:-

enter passband ripple0.02

enter the stopband ripple0.01

enter passband freq1000

enter stopband freq2000

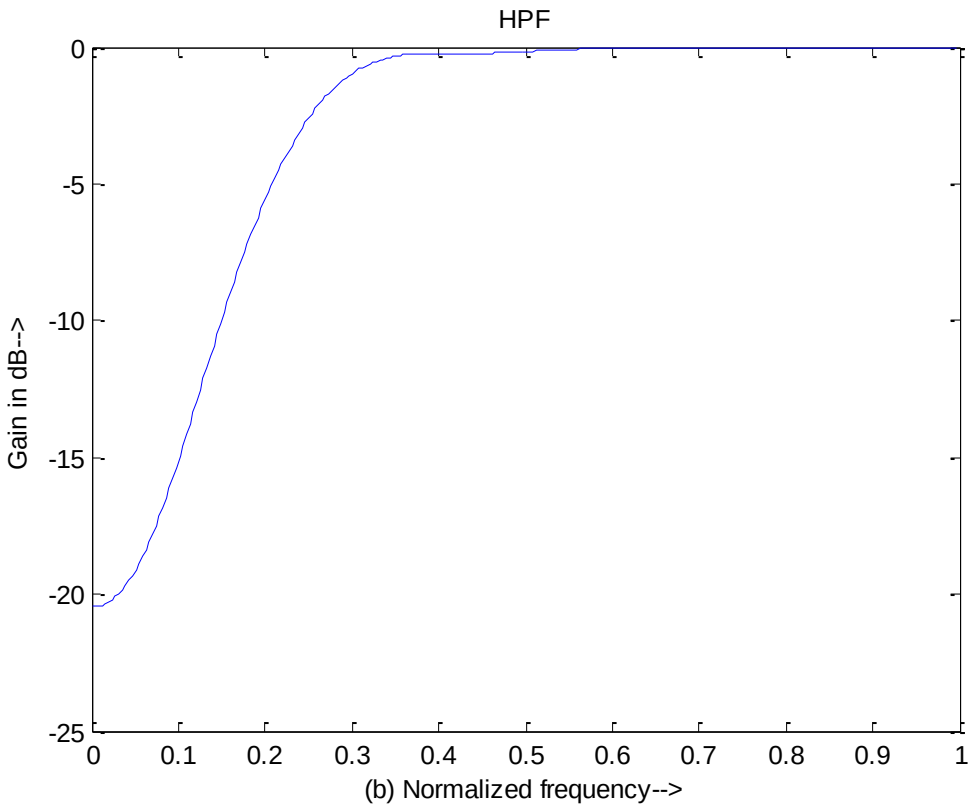
enter sampling freq 10000

enter beta value0.2

enter your choice of window function 1. rectangular 2. Hamming 3.kaiser:

2

Hamming window filter response



RESULT:- Thus FIR HP Filter is designed for Rectangular/triangular/kaiser windowing techniquesusing MATLAB.

MATLAB CODE FOR COMPARISION OF WINDOW METHODS

```
clc;
clear all;
close all;
passripple=0.04; %pass band ripple
stopripple=0.05; %stop band ripple
passfreq=2000; %pass band ripple
stopfreq=3000; %pass band ripple
freq=9000; %sampling freq
wp=2*passfreq/freq;
ws=2*stopfreq/freq;
s=passripple*stopripple;
r=stopfreq-passfreq;
p=r/freq;
num=-20*log10(sqrt(s))-13;
dem=14.6*p;
n=ceil(num/dem);
n1=n+1;
if(rem(n,2)~=0)
n1=n;
n=n-1;
end
beta=4;
y=kaiser(n1,beta);
b=fir1(n,wp,'low',y); %low pass filter
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
figure(2)
subplot(2,2,1);
```

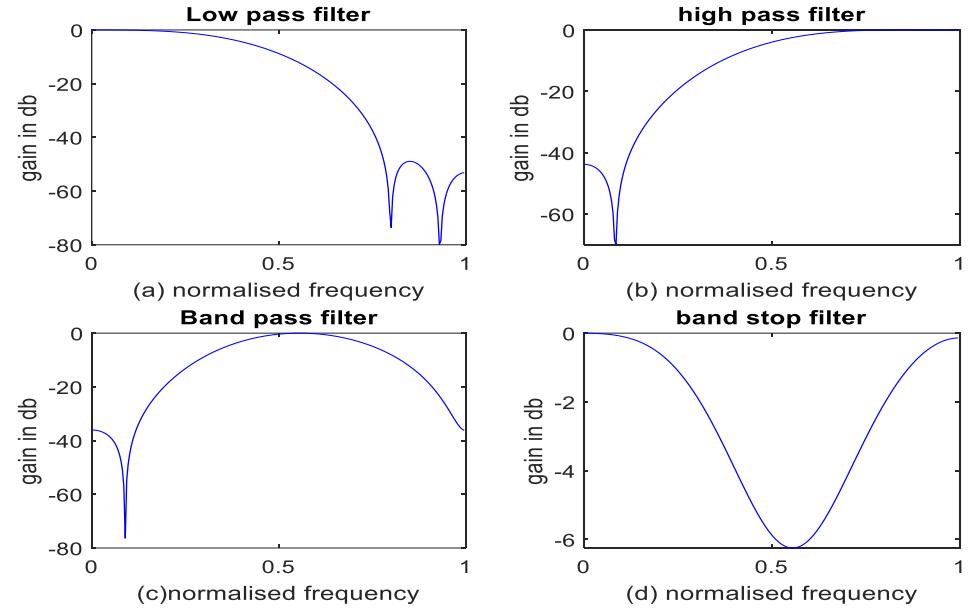
```
plot(o/pi,m,'b');


title('Low pass filter');
ylabel('gain in db');
xlabel('(a) normalised frequency');
b=fir1(n,wp,'high',y); %high pass filter
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,2);
plot(o/pi,m,'b');
title('high pass filter');
ylabel( 'gain in db');
xlabel('(b) normalised frequency');
wn=[wp,ws]; %band pass filter
b=fir1(n,wn,y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,3);
plot(o/pi,m,'b');
title('Band pass filter');
ylabel('gain in db');
xlabel('(c)normalised frequency');
b=fir1(n,wn,'stop',y); %band stop filter
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,4);
plot(o/pi,m,'b');
title('band stop filter');
ylabel('gain in db');
xlabel('(d) normalised frequency');
hold on
```

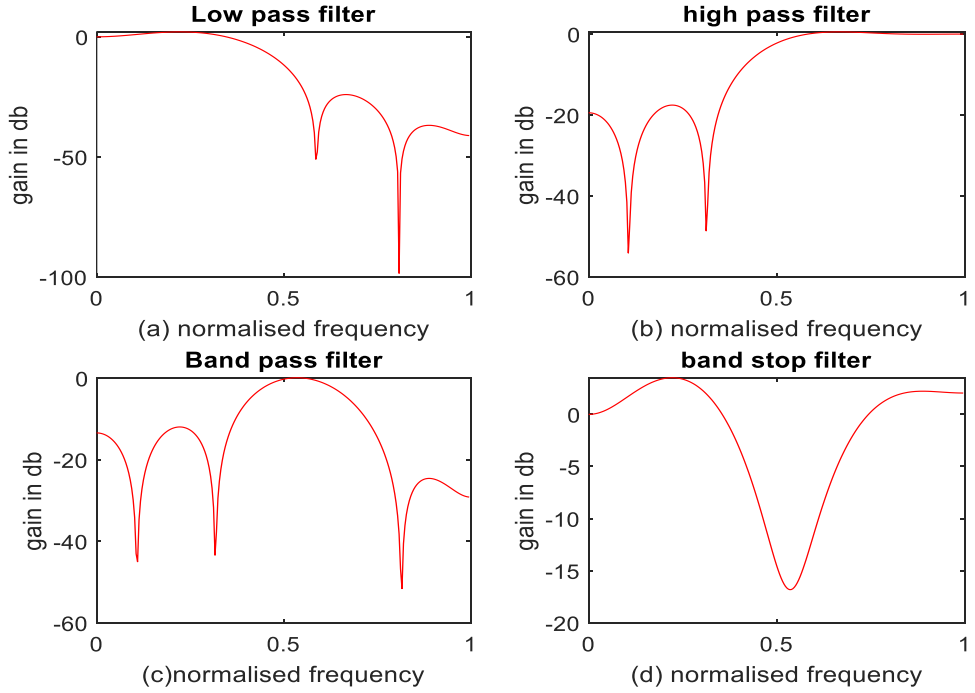
```
ys=rectwin(n1);


b=fir1(n,wp,'low',ys); %low pass filter
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
figure(1)
subplot(2,2,1);
plot(o/pi,m,'r');
title('Low pass filter');
ylabel('gain in db');
xlabel('(a) normalised frequency');
b=fir1(n,wp,'high',ys); %high pass filter
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,2);
plot(o/pi,m,'r');
title('high pass filter');
ylabel( 'gain in db');
xlabel('(b) normalised frequency');
wn=[wp,ws]; %band pass filter
b=fir1(n,wn,ys);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,2,3);
plot(o/pi,m,'r');
title('Band pass filter');
ylabel('gain in db');
xlabel('(c)normalised frequency');
b=fir1(n,wn,'stop',ys); %band stop filter
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
```

```
subplot(2,2,4);


plot(o/pi,m,'r');
title('band stop filter');
ylabel('gain in db'); xlabel('(d) normalised frequency');
```

OUTPUT :

Blue color- Kaiser Window, Red color-Rectangular window





11) Design and verify Filter (IIR and FIR) frequency response by using Filter design and Analysis Tool.

AIM: To write a matlab program for design of IIR filter using any of the available methods and verify the frequency response of the filter.

APPARATUS: PC with MATLAB Software.

PROCEDURE:

1 Click on the MATLAB Icon on the desktop.
2 MATLAB window open.
3 Click on the 'FILE' Menu on menu bar.
4 Click on NEW M-File from the file Menu.
5 An editor window open, start typing commands.
6 Now SAVE the file in directory.
7 Then Click on DEBUG from Menu bar and Click Run.

PROGRAM:

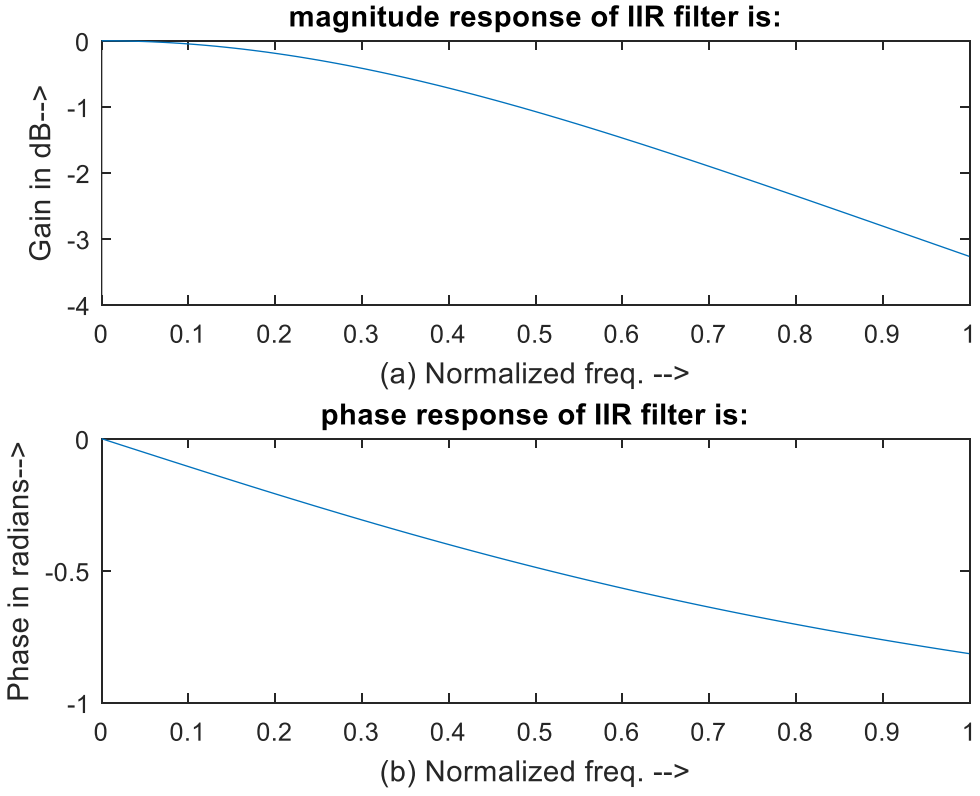
```
% IIR FILTERS LPF & HPF USING MATLAB

clc;

clear all;

close all;

disp('enter the IIR filter design specifications');

rp=input('enter the passband ripple');

rs=input('enter the stopband ripple');

wp=input('enter the passband freq');

ws=input('enter the stopband freq');

fs=input('enter the sampling freq');

w1=2*wp/fs;w2=2*ws/fs;

[n,wn]=buttord(w1,w2,rp,rs,'s');

% IIR FILTERS LPF & HPF USING MATLAB

clc;
```

```
clear all;
close all;
disp('enter the IIR filter design specifications');



rp=input('enter the passband ripple');
rs=input('enter the stopband ripple');
wp=input('enter the passband freq');
ws=input('enter the stopband freq');
fs=input('enter the sampling freq');
w1=2*wp/fs;w2=2*ws/fs;
[n,wn]=buttord(w1,w2,rp,rs,'s');
c=input('enter choice of filter 1. LPF 2. HPF \n ');
if(c==1)
disp('Frequency response of IIR LPF is:');
[b,a]=butter(n,wn,'low','s');
end
if(c==2)
disp('Frequency response of IIR HPF is:');
[b,a]=butter(n,wn,'high','s');
end
w=0:.01:pi;
[h,om]=freqs(b,a,w);
m=20*log10(abs(h));
an=angle(h);
figure,subplot(2,1,1);plot(om/pi,m);
title('magnitude response of IIR filter is:');
xlabel('(a) Normalized freq. -->');
```

```
ylabel('Gain in dB-->');

subplot(2,1,2);plot(om/pi,an);

title('phase response of IIR filter is:');

xlabel('(b) Normalized freq. -->');

ylabel('Phase in radians-->');
```

OUTPUT :--

```
enter the IIR filter design specifications

enter the passband ripple.5

enter the stopband ripple.8

enter the passband freq1000

enter the stopband freq2000

enter the sampling freq3000

enter choice of filter 1. LPF 2. HPF

 1

Frequency response of IIR LPF is:
```



RESULT:-- IIR filters using matlab is designed.

VIVA QUESTION:

1. What is filter?
2. What is FIR and IIR filter define, and distinguish between these two?
3. What is window method? How you will design an FIR filter using window method?
4. What are low-pass and band-pass filter and what is the difference between these two?
5. What is the matlab command for Hamming window? Explain.
6. What do you mean by built in function 'abs' and where it is used?
7. Explain how the FIR filter are stable?
8. Why is the impulse response "finite"?
9. What does "FIR" mean?
10. What are the advantages of FIR Filters (compared to IIR filters)?
11. What are the disadvantages of FIR Filters (compared to IIR filters)?
12. What terms are used in describing FIR filters?
13. What is the delay of a linear-phase FIR?
14. What is the Z transform of a FIR filter?
15. What is the frequency response formula for a FIR filter?
16. How Can I calculate the frequency response of a FIR using the Discrete Fourier Transform (DFT)?
17. What is the DC gain of a FIR filter?

12) Compute the Decimation and Interpolation for the given signal.

Aim : To write a matlab program to Compute the Decimation and Interpolation for the given signal.

APPARATUS: PC with MATLAB Software.

PROCEDURE:

1 Click on the MATLAB Icon on the desktop.
2 MATLAB window open.
3 Click on the 'FILE' Menu on menu bar.
4 Click on NEW M-File from the file Menu.
5 An editor window open, start typing commands.
6 Now SAVE the file in directory.
7 Then Click on DEBUG from Menu bar and Click Run.

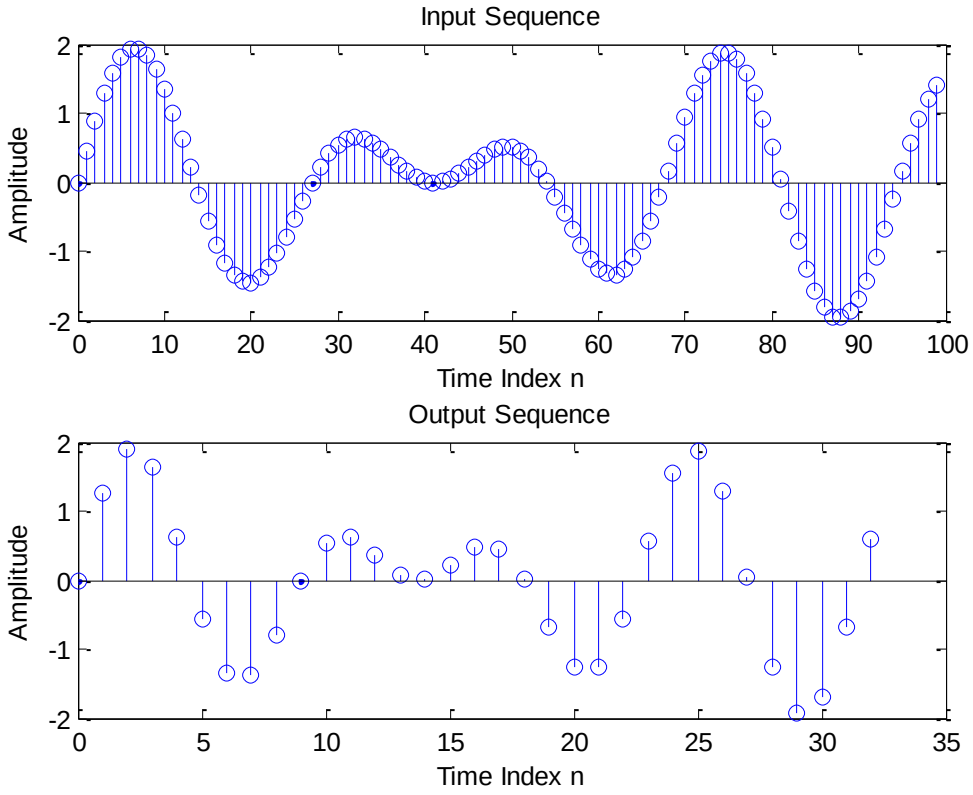
PROGRAM: Decimation

```
clc;
clear all;
close all;
M=input('Enter Down Sampling Factor:');
N=input('Enter Number of Samples:');
n=0:N-1;
x=sin(2*pi*0.043*n)+sin(2*pi*0.031*n);
y=decimate(x,M,'fir');
subplot(2,1,1);
stem(n,x(1:N));
title('Input Sequence');
xlabel('Time Index n');
ylabel('Amplitude');
subplot(2,1,2);
m=0:(N/M)-1;
stem(m,y(1:N/M));
title('Output Sequence');
xlabel('Time Index n');
ylabel('Amplitude');
```

OUTPUT :

Enter Down Sampling Factor:3

Enter Number of Samples:100



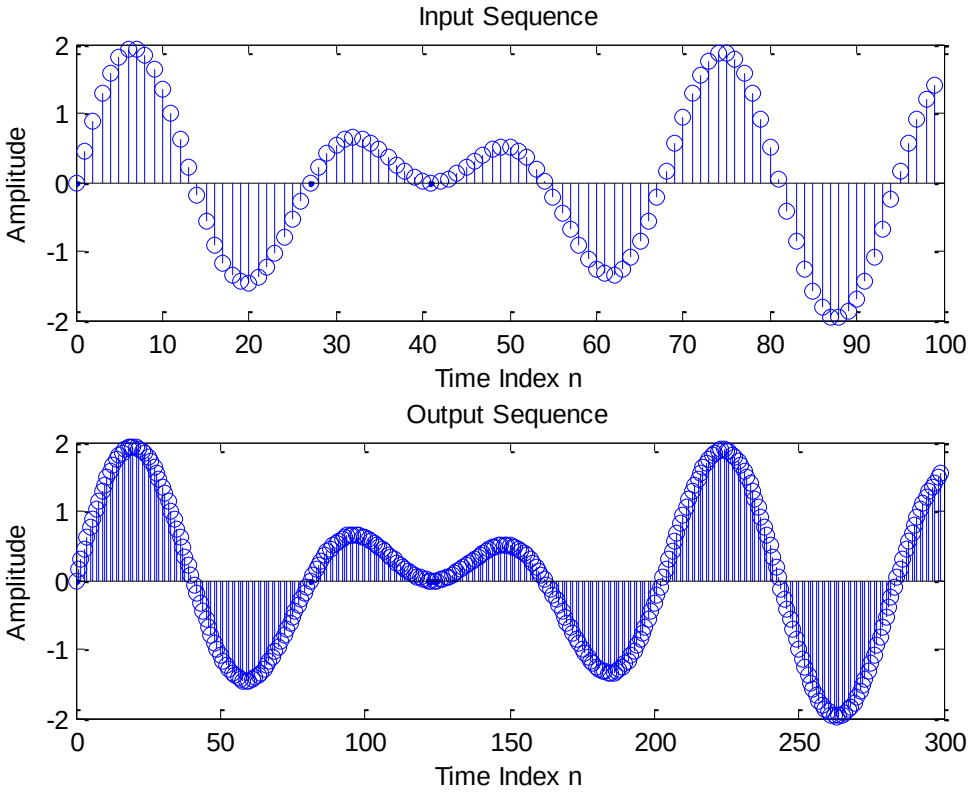
PROGRAM: Interpolation

```
clc;
clear all;
close all;
L=input('Up Sampling Factor:');
N=input('Enter Number of Samples:');
n=0:N-1;
x=sin(2*pi*0.043*n)+sin(2*pi*0.031*n);
y=interp(x,L);
subplot(2,1,1);
stem(n,x(1:N));
title('Input Sequence');
xlabel('Time Index n');
ylabel('Amplitude');
subplot(2,1,2);
m=0:(N*L)-1;
stem(m,y(1:N*L));
title('Output Sequence');
xlabel('Time Index n');
ylabel('Amplitude');
```

OUTPUT :

Up Sampling Factor:3

Enter Number of Samples:100



Result: Decimation and Interpolation for the given signal are computed.