Signals and Systems Laboratory Manual

For 3rd Semester

Electronics and Telecommunications Engineering Department

Electronics and Telecommunications Engineering Department
Assam Engineering College, Jalukabri, Guwahati, Assam, Pin- 781013

S

Index

Aim: **1.** Introduction to Matlab.
2. Plotting of various continuous time signal.
a. Sinusoidal b. Cosinusoidal

Software used: Matlab R2016a

Theory:

MATLAB is a high-performance language for technical computing. The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects, which together represent the state-of-the-art in software for matrix computation. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Math and computation
- Algorithm development
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including Graphical User Interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows us to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

- **Sinusoidal signal**

The sinusoidal signal is defined as,

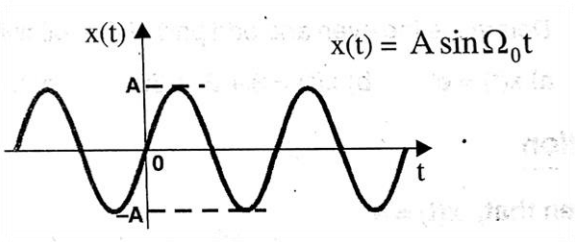
$$X(t) = A \sin(\Omega_o t + \phi)$$

where, $\Omega_o = 2\pi F_o = 2\pi/T$ = Angular frequency in rad/sec

F_o = Frequency in cycles/sec or Hz

T = Time period in sec

When $\phi = 0$, $x(t) = A \sin \Omega_o t$
When ϕ = positive , $x(t) = A \sin(\Omega_o t + \phi)$
When ϕ = negative , $x(t) = A \sin(\Omega_o t - \phi)$



- **Cosinusoidal signal**

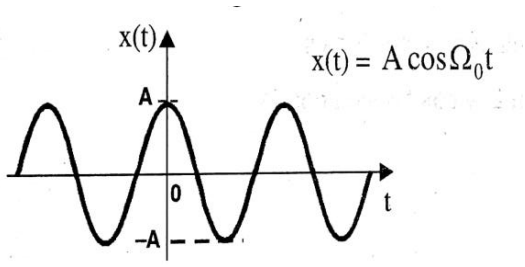
  The sinusoidal signal is defined as,

  $X(t) = A \cos(\Omega_o t + \phi)$

  where, $\Omega_o = 2\pi F_o = 2\pi/T$ = Angular frequency in rad/sec

  F_o = Frequency in cycles/sec or Hz

  T = Time period in sec



When $\phi = 0$, $x(t) = A \cos \Omega_o t$

When ϕ = positive , $x(t) = A \cos(\Omega_o t + \phi)$

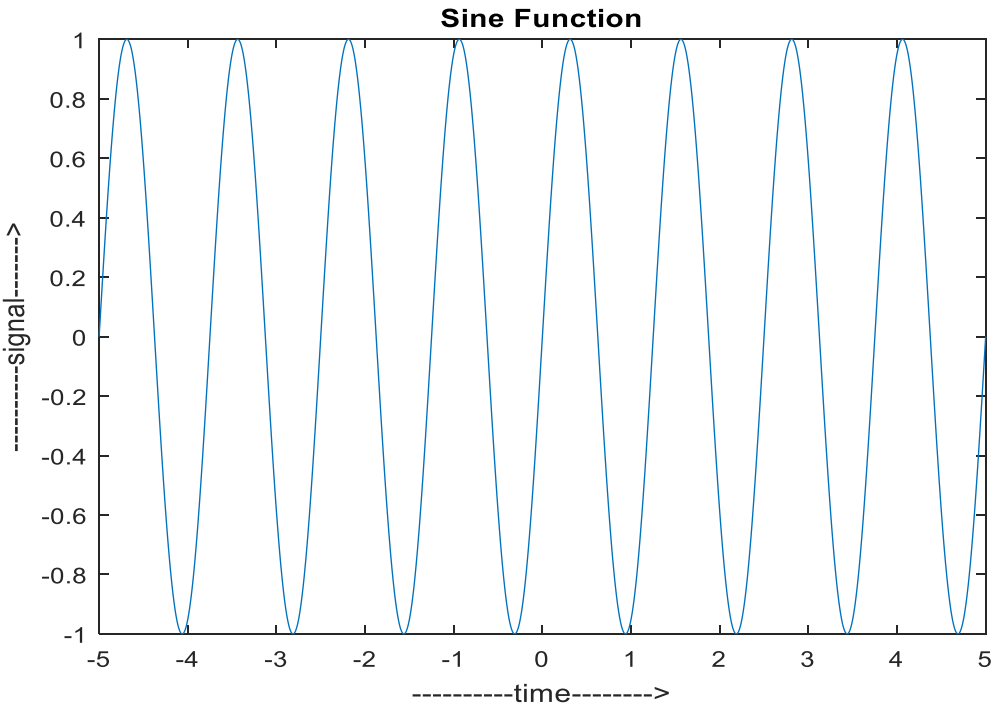
When ϕ = negative , $x(t) = A \cos(\Omega_o t - \phi)$

Program:

- Plotting of sinuisoidal signal

```
clc;
clear all;
close all
t=-5:0.001:5;
a=1;
f=0.8;
xt=a*sin(2*pi*f*t)
plot(t,xt)
title('Sine Function');
xlabel('----------time-------->');
ylabel('----------signal------->');
```

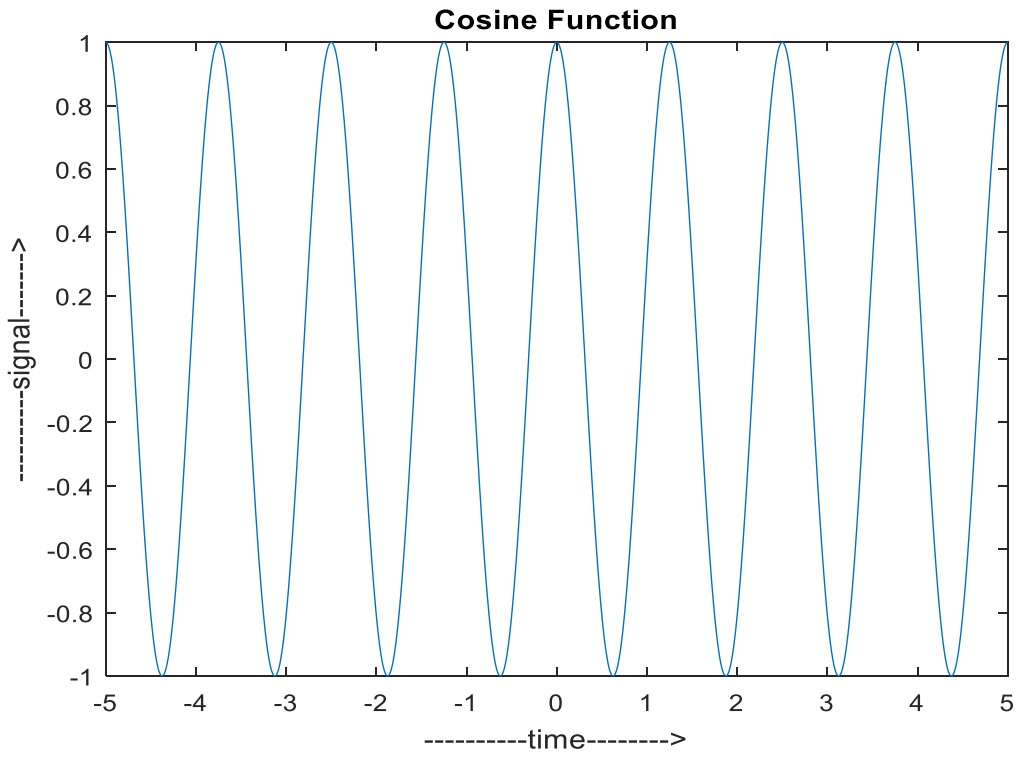
Output:


Sine Function
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-5
-4
-3
-2
-1
0
1
2
3
4
5
--------signal------>
----------time-------->
Sine Function

- Plotting of cosinuisoidal signal

```
clc;
clear all;
close all
t=-5:0.001:5;
a=1;
f=0.8;
xt=a*cos(2*pi*f*t)
plot(t,xt)
title('Cosine Function');
xlabel('----------time-------->');
ylabel('----------signal------->');
```

Output:



Result: With the help of MATLAB program the desired output is obtained and the result is verified.

Aim: Plotting of various continuous time signal.

Software used: Matlab R2016a

Theory:

A continuous-time real (or complex) signal is any real-valued (or complex-valued) function which is defined for all time t in an interval, most commonly an infinite interval.

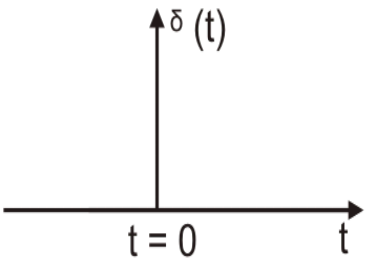
A continuous signal or a continuous-time signal is a varying quantity (a signal) that is expressed as a function of a real-valued domain, usually time. The function of time need not be continuous.

The signal is defined over a domain, which may or may not be finite, and there is a functional mapping from the domain to the value of the signal.

Impulse signal: The impulse signal is a signal with infinite magnitude and zero duration, but with an area of A. Mathematically, impulse signal is defined as

Impulse signal, $\delta(t) = \infty$; t=0 and $\int_{-\infty}^{+\infty} \delta(t)dt$ =A

$= 0$; $t \neq 0$



The impulse signal is a signal with infinite magnitude and zero duration, but with unit area . Mathematically, impulse signal is defined as

Impulse signal, $\delta(t) = \infty$; t=0 and $\int_{-\infty}^{+\infty} \delta(t)dt$ =1

$= 0$; $t \neq 0$

Step signal: The step signal is defined as,

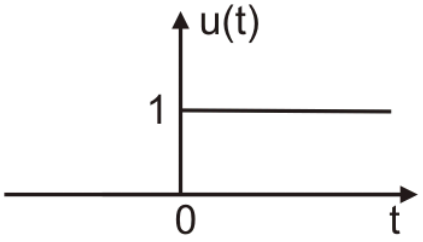
$x(t) = A$; $t \geq 0$

$= 0$; $t < 0$

The unit step signal is defined as,

$x(t) = u(t) = 1$; $t \geq 0$

$= 0$; $t < 0$



Ramp Signal: The ramp signal is defined as,

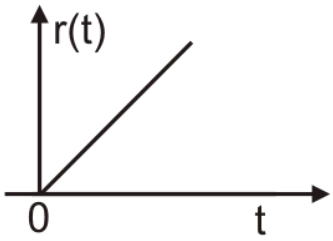
$x(t) = At$; $t \geq 0$

$= 0$; $t < 0$

The unit ramp signal is defined as,

$x(t) = 1.t$; $t \geq 0$

$= 0$; $t < 0$

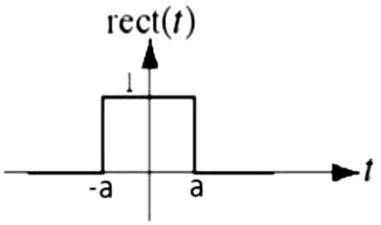


Unit rectangular or gate signal: given by,

$\pi(t) = 0 \; ; -a > t > a$

$= 1 \; ; -a < t < a$

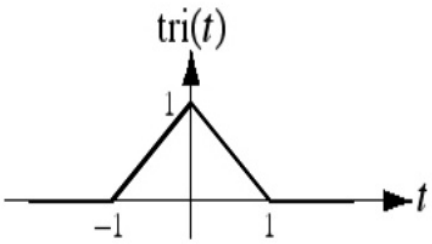
or, $\text{rect}(t) = u(t+a) - u(t-a)$



Triangular pulse signal: given by,

$$X(t) = \Delta_a(t) = 1 - \frac{|t|}{a} \quad ; \quad |t| \le a$$

$$= 0 \quad ; \quad |t| > a$$

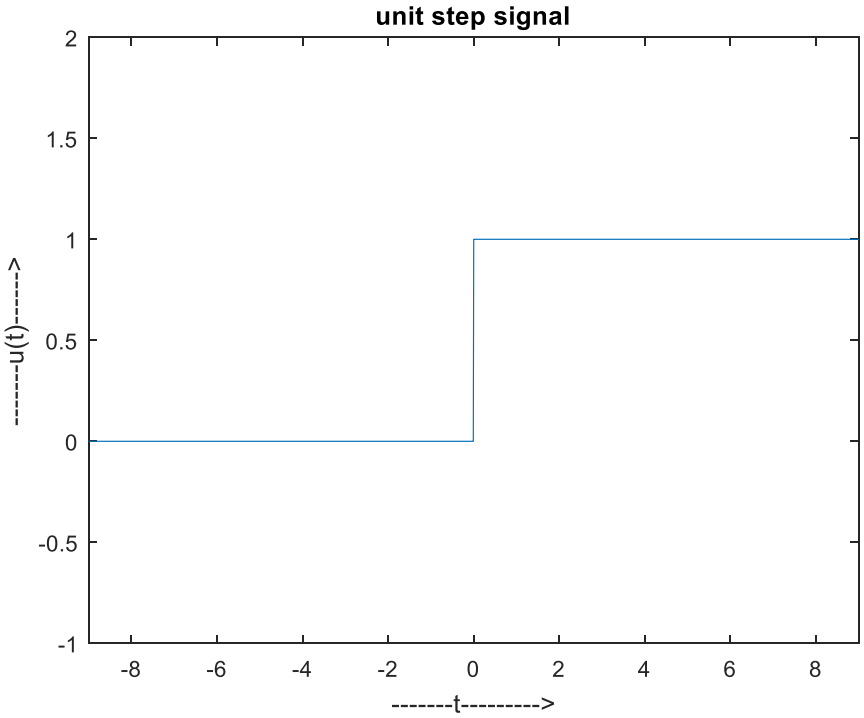


Program:

- Plotting of unit step signal ,u(t)

```
clc
clear all
close all
t = -10:0.01:10
u(t>=0)= 1;
u(t<0)=0;
plot(t,u);
title('unit step signal');
xlabel('-------t--------->');
ylabel('-------u(t)------>');
axis([-9 9 -1 2 ])
```

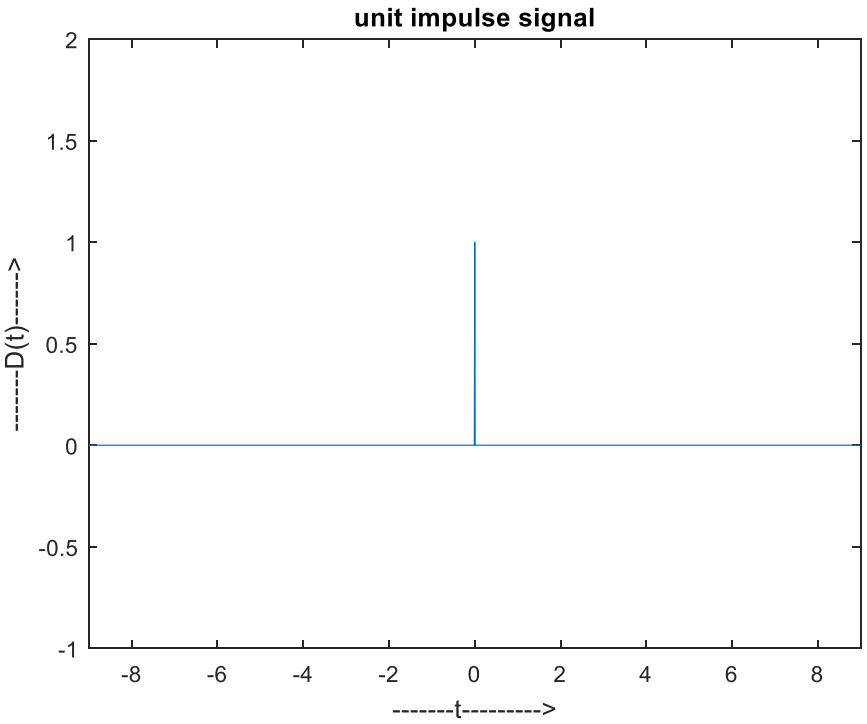
Output:



- Plotting of unit impulse signal, δ(t)

```
clc;
clear all;
close all;
t = -10:0.01:10;
D(t==0)= 1;
D(t~=0)= 0;
plot(t,D);
title('unit impulse signal');
xlabel('-------t--------->');
ylabel('-------D(t)------>');
axis([-9 9 -1 2 ])
```

Output:

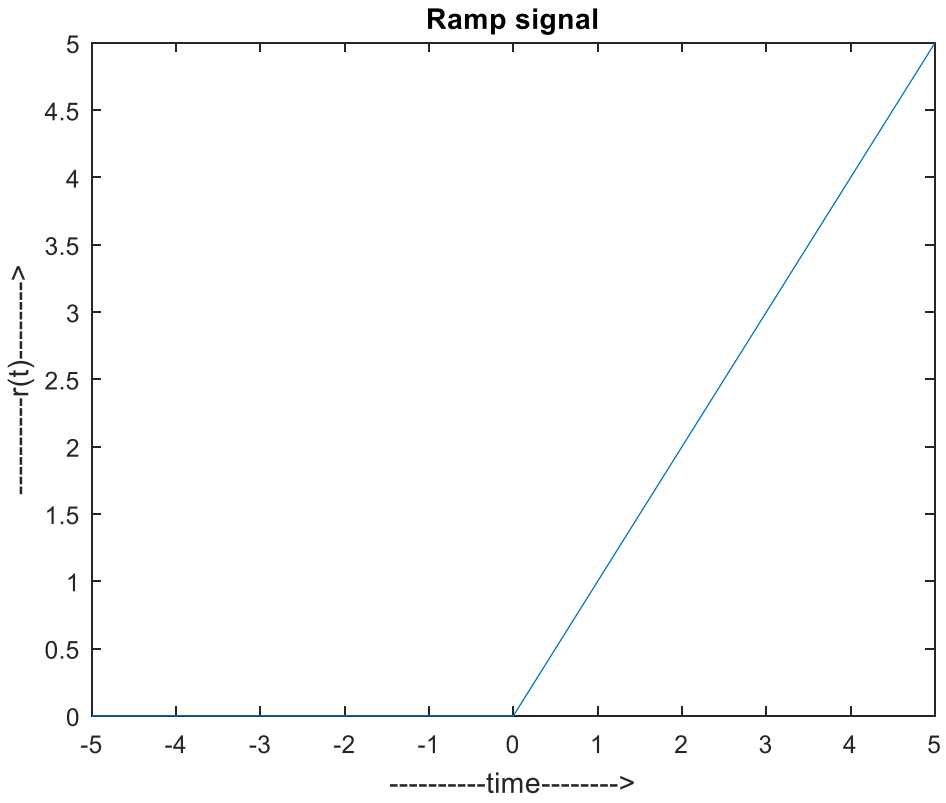


- Plotting of ramp signal, r(t)

```
clc;
clear all;
close all;
t=-5:0.0001:5;
r=((t).*1);
r(t<=0)=0;
plot(t,r);
xlabel('----------time-------->');
ylabel('----------r(t)-------->');
```

```
title('Ramp signal');
```

Output:



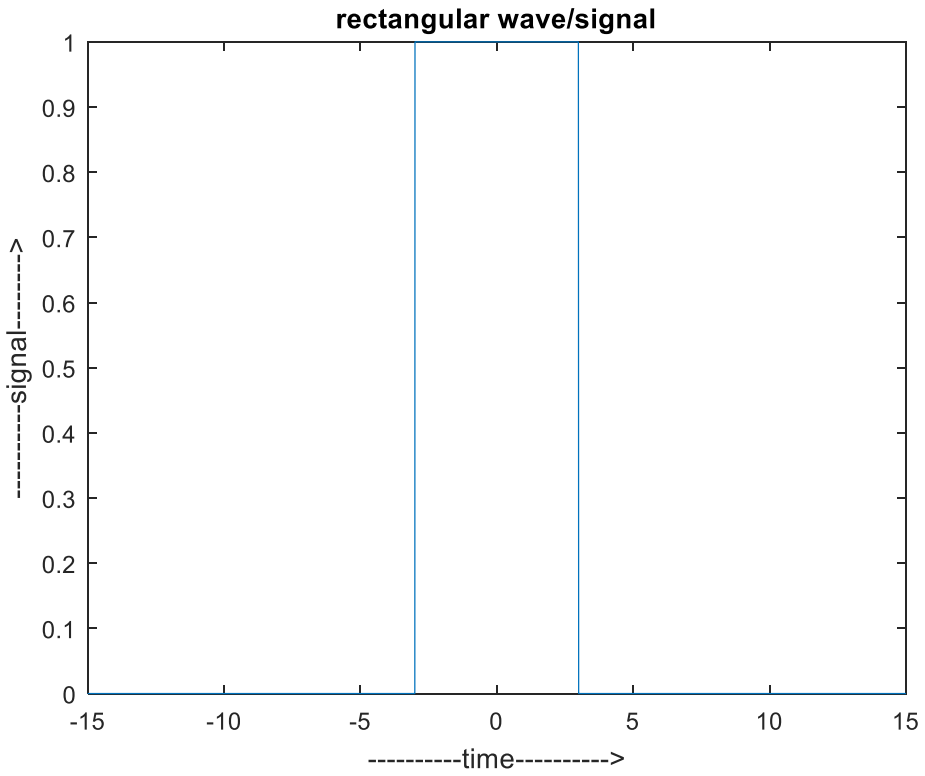
- Plotting of rectangular wave/signal

```
clc;
clear all;
close all;
t=-15:0.001:15;
a= input('a= ')

x1(t>=-a)=1;
x2(t<a)=0;
x2(t>=a)=1;
x3=x1-x2;
plot(t,x3)
title('rectangular wave/signal');
xlabel('----------time---------->')
ylabel('----------signal-------->')
```

Command window:

```
a= 3

a =

     3
```

Output:

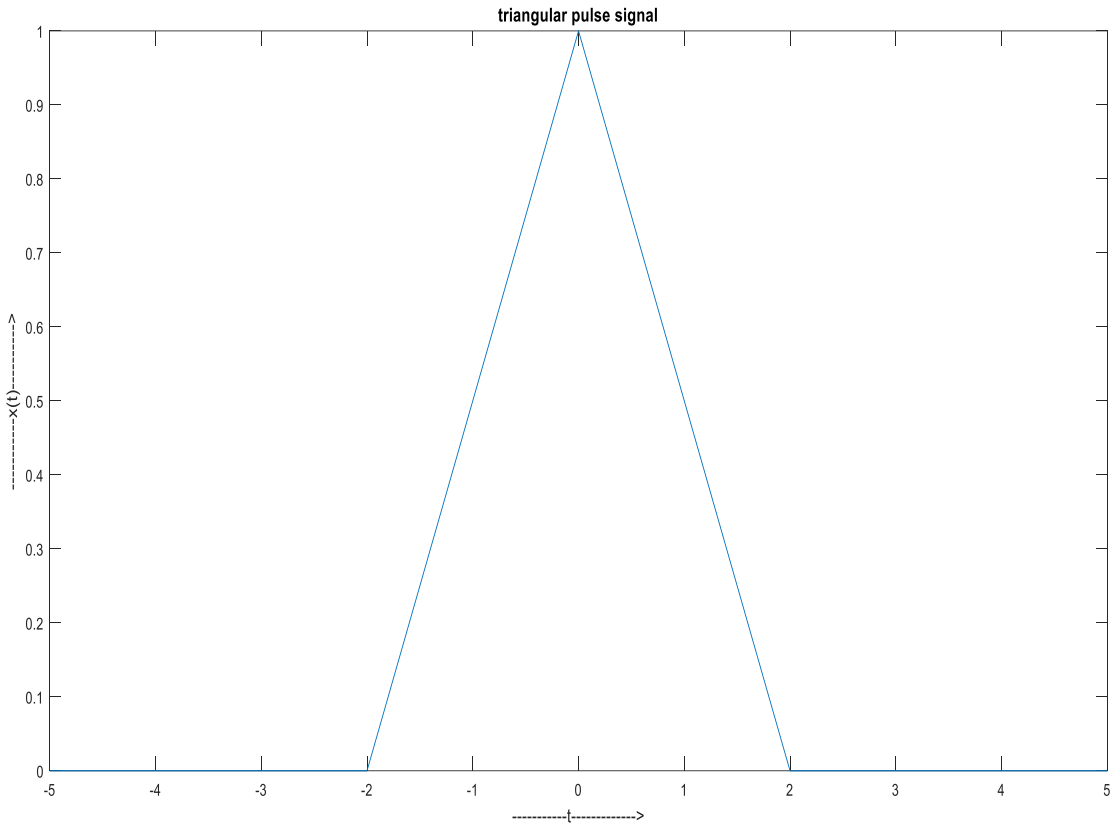


- Plotting of triangular pulse/signal

```
clc
clear all
close all
t=-5:0.01:5;
a=2;
x1=1-abs(t)/a;
x2=1-abs(t)/a;
x=x1.*(abs(t)<=a+x2.*(abs(t)>a));
plot(t,x);
xlabel('----------t---------->');
```

```
ylabel('-----------x(t)---------->');
title('triangular pulse signal');
```

Output:



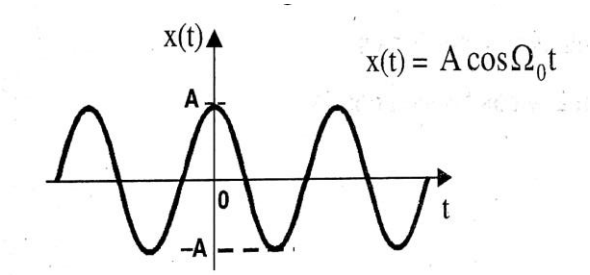
Result:

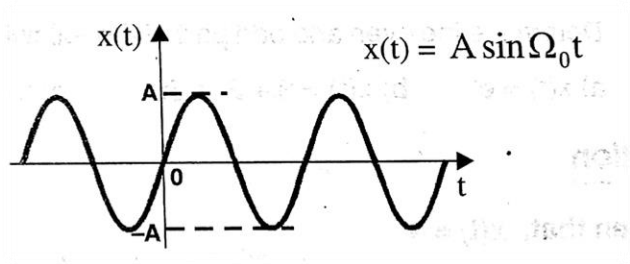
With the help of MATLAB program the desired output is obtained and the result is verified.

Aim: To determine and plot odd and even function signals.

Software used: Matlab R2016a

Theory: When a signal exhibits symmetry with respect to t = 0, then it is called an even signal.
Therefore, the even signals satisfies the condition, x(-t) = x(t) .
When a signal exhibits antisymmetry with respect to t = 0, then it is called an odd signal.
Therefore, the odd signals satisfies the condition, x(-t) = - x(t) .





Even signal Odd signal

A continuous time signal x(t) which is neither even nor odd can be expressed as a sum of even and odd signal.
If, $x(t) = x_e(t) + x_o(t)$
where , $x_e(t)$ = Even part of x(t) and $x_o(t)$ = Odd part of x(t)
then,

$$x_e(t) = \frac{1}{2}[x(t) + x(-t)] \quad \text{and} \quad x_o(t) = \frac{1}{2}[x(t) + x(-t)]$$

Program:

- To find whether the function is even or odd:

```
clc;
clear all;
close all;
t=0:0.01:10;
a=cos(t);
b=cos(-t);
```

```
if a==b
disp('even')
else if a==(-b)
disp('odd');
    else
        disp('none');
    end
end
```

Command Window Output:

```
even
>>
```

➢
```
clc;
clear all;
close all;
t=0:0.01:10;
a=sin(2*t)+cos(t);
b=sin(-2*t)+cos(t);

if a==b
disp('even')
else if a==(-b)
disp('odd');
    else
        disp('none');
    end
end
```

Command Window Output:

```
none
>>
```

- To find odd and even components of continuous time signals:

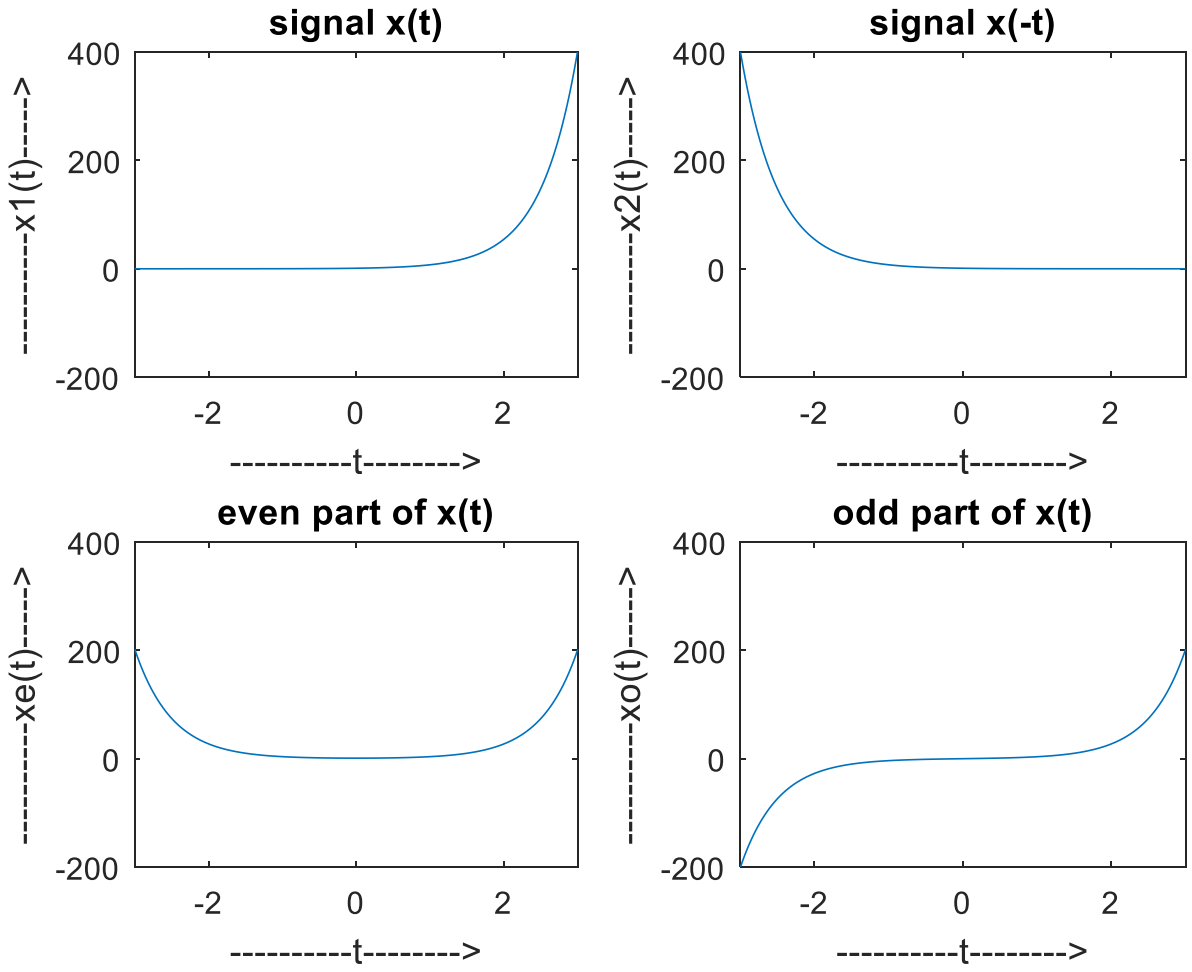
```
clc;
clear all;
close all;
% x(t)=exp(2*t)     <------- signal
t=-4:0.001:4;
x1=exp(2*t);
x2=exp(-2*t);
if(x2==x1)
```

```
    disp('the signal is even');
else if(x2==(-x1))
       disp('the signal is odd');
    else
        disp('the signal is neither even nor odd');
    end
end
xe=(x1+x2)/2;
xo=(x1-x2)/2;

subplot(2,2,1);plot(t,x1);
xlabel('----------t-------->');
ylabel('----------x1(t)----->');
title('signal x(t)');
axis([-3 3 -200 400]);

subplot(2,2,2);plot(t,x2);
xlabel('----------t-------->');
ylabel('----------x2(t)----->');
title('signal x(-t)');
axis([-3 3 -200 400]);

subplot(2,2,3);plot(t,xe);
xlabel('----------t-------->');
ylabel('----------xe(t)----->');
title('even part of x(t)');
axis([-3 3 -200 400]);

subplot(2,2,4);plot(t,xo);
xlabel('----------t-------->');
ylabel('----------xo(t)----->');
title('odd part of x(t)');
axis([-3 3 -200 400]);
```

Command Window Output:

```
the signal is neither even nor odd
```

Output graphs:



Result:

With the help of MATLAB program the desired output is obtained and the result is verified.

Aim: Introdution to discrete time signals.

Software used: Matlab R2016a

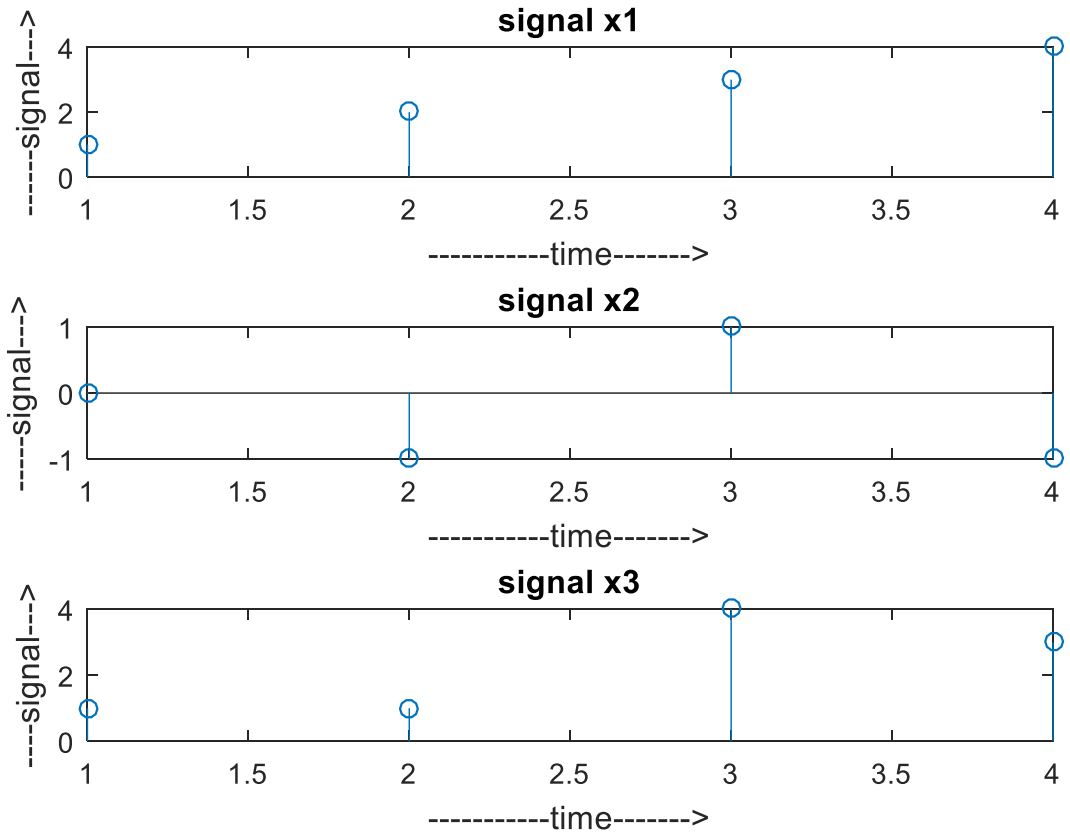
Theory: In a signal with time as independent variable, if the signal is defined only for discrete instants of the independent variable time, then the signal is called discrete time signal. In discrete time signal the independent variable time “t” is uniformely divided into discrete intervals of time and each interval of time is denoted by an integer "n", where "n" stands for discrete interval of time and "n" can take any integer value in the range -∞ to +∞. Therefore, for a discrete time signal the independent variable is "n” and the magnitude of the discrete time signal is defined only for integer values of independent variable "n”. The discrete time signal is denoted by "x(n)”.

Program:

- Plotting of discrete time signals

```
clc;
clear all;
close all;
x1=[1 2 3 4];
subplot(3,1,1);
stem(x1);
title('signal x1');
xlabel('-----------time------->');
ylabel('------signal--->');

x2=[0 -1 1 -1];
subplot(3,1,2);
stem(x2);
title('signal x2');
xlabel('-----------time------->');
ylabel('-----signal--->');

x3= x1+x2;
subplot(3 ,1, 3);
stem(x3);
title('signal x3');
xlabel('-----------time------->');
ylabel('----signal--->');
```

Output:



Result:

With the help of MATLAB program the desired output is obtained and the result is verified.

Aim: Operations on signals.

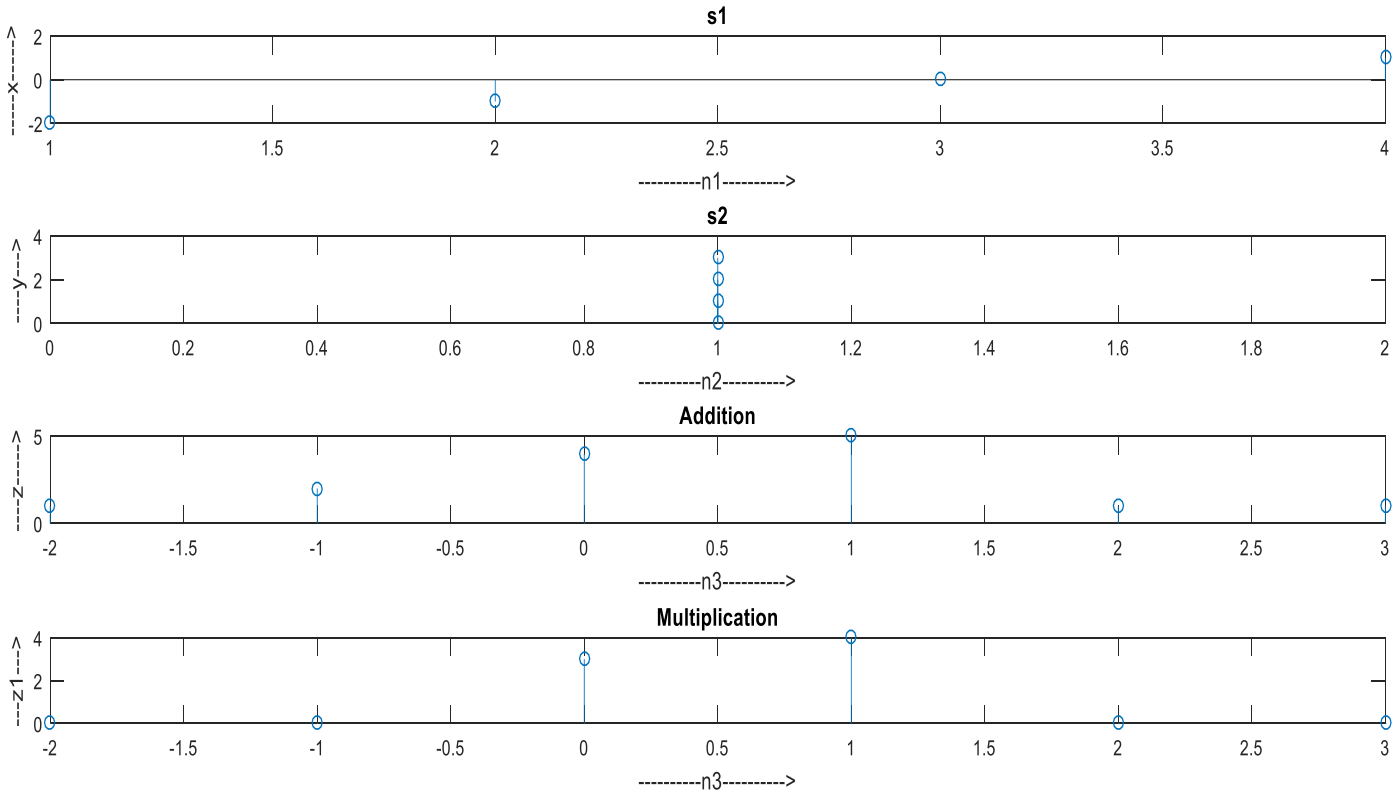
Software used: Matlab R2016a

Program:

- To perform addition and multiplication of two discrete signals

```
clc;
clear all;
close all;
x=[1 2 3 4];
y=[1 1 1 1];
n1=-2:1;
n2=0:3;
subplot(5,1,1)
stem(x,n1)
xlabel('----------n1---------->')
ylabel('-----x----->')
title('s1')
subplot(5,1,2)
stem(y,n2)
title('s2')
xlabel('----------n2---------->')
ylabel('----y--->')
n3=min(min(n1),min(n2)):max(max(n1),max(n2));
s1=zeros(1,length(n3));
s2=s1;
s1(find((n3>=min(n1))&(n3<=max(n1))==1))=x;
s2(find((n3>=min(n2))&(n3<=max(n2))==1))=y;
z=s1+s2;
subplot(5,1,3)
stem(n3,z);
title('Addition')
xlabel('----------n3---------->')
ylabel('----z----->')

z1=s1.*s2;
subplot(5,1,4)
stem(n3,z1);
title('Multiplication')
xlabel('----------n3---------->')
ylabel('---z1--->')
```

Output:



- To perform amplitude scaling, time scaling and time shift on a given signal, y(t)

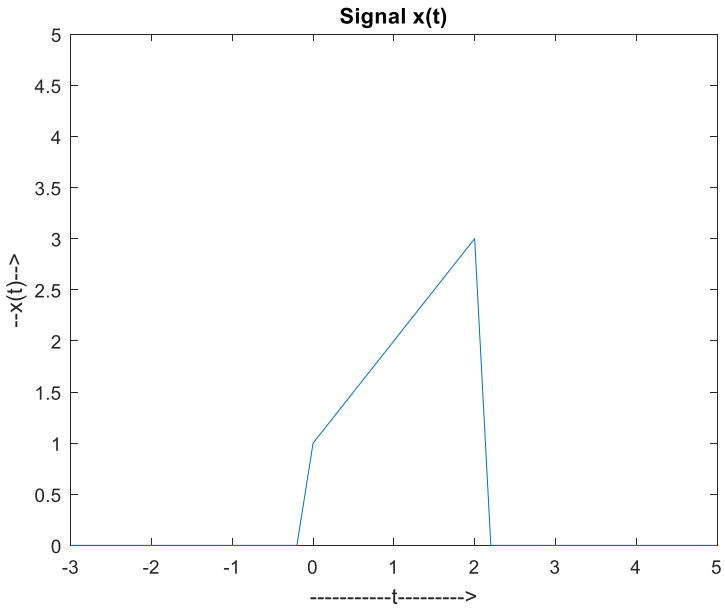
```
%on the signal x(t)=1.0+t; for t= 0 to 2
```

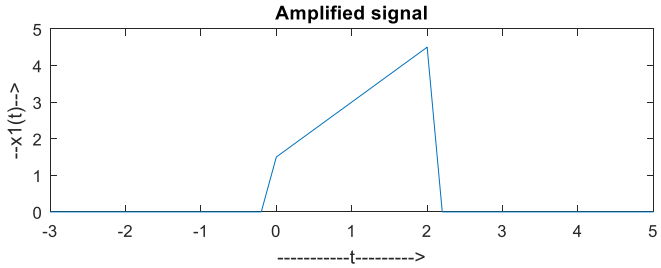
```
function x=y(t)                 %this program is saved as a separate
                                %file in work directory and the
                                %signal x(t) is declared as
                                %function y(t)
x=(1 + t).*(t>=0 & t<=2);
```

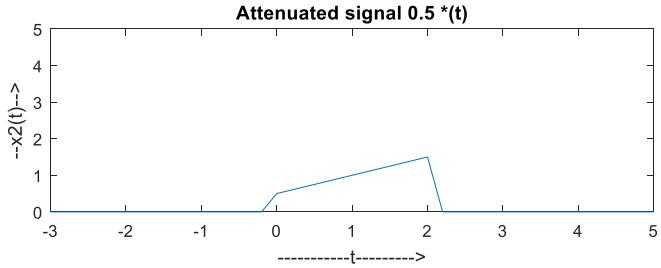
```
clc;
clear all;
close all;
t=-3:0.2:5;

y0=y(t);
y1=1.5*y(t);
y2=0.5*y(t);
y3=y(2*t);
y4=y(0.5*t);
y5=y(t-2);
y6=y(t+2);
```

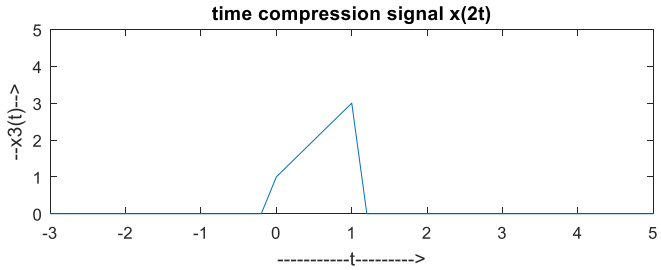
```
figure(1)
plot(t,y0);
axis([-3 5 0 5]);
xlabel('----------t--------->');
ylabel('--x(t)-->')
title('Signal x(t)')

figure(2)
subplot(3,2,1);plot(t,y1);
axis([-3 5 0 5]);
xlabel('----------t--------->');
ylabel('--x1(t)-->')
title('Amplified signal')

subplot(3,2,2);plot(t,y2);
axis([-3 5 0 5]);
xlabel('----------t--------->');
ylabel('--x2(t)-->')
title('Attenuated signal 0.5 *(t)')

subplot(3,2,3);plot(t,y3);
axis([-3 5 0 5]);
xlabel('----------t--------->');
ylabel('--x3(t)-->')
title('time compression signal x(2t)')

subplot(3,2,4);plot(t,y4);
axis([-3 5 0 5]);
xlabel('----------t--------->');
ylabel('--x4(t)-->')
title('time expansion signal x(0.5t)')

subplot(3,2,5);plot(t,y5);
axis([-3 5 0 5]);
xlabel('----------t--------->');
ylabel('--x5(t)-->')
title('Delayed signal x(t-2)')

subplot(3,2,6);plot(t,y6);
axis([-3 5 0 5]);
xlabel('----------t--------->');
ylabel('--x6(t)-->')
title('Advanced signal x(t+2)')
```

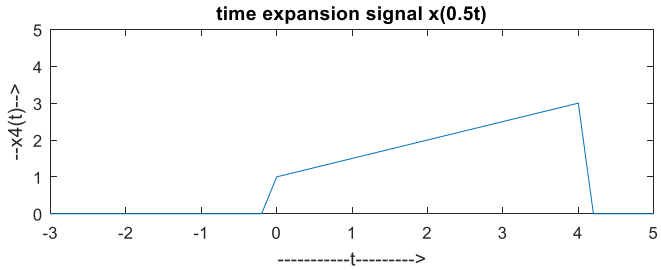
Output graphs:

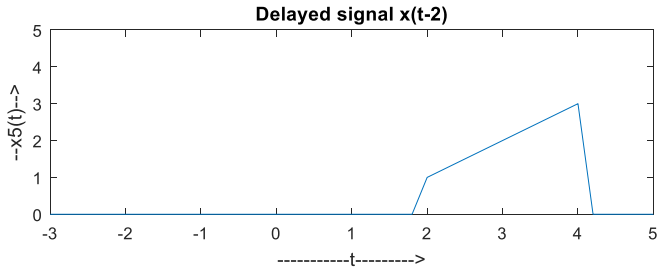


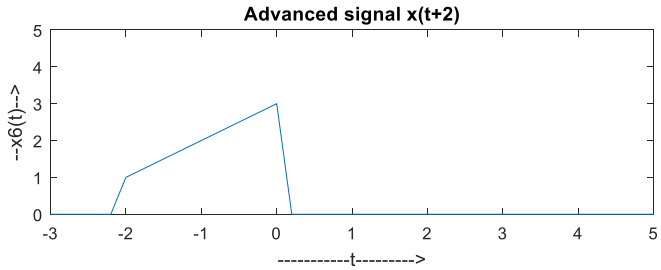












Result:

With the help of MATLAB program the desired output is obtained and the result is verified.

Aim: Mathematical operations on continuous signals.

Software used: Matlab R2016a

Theory:

Time shifting of continuous signals:

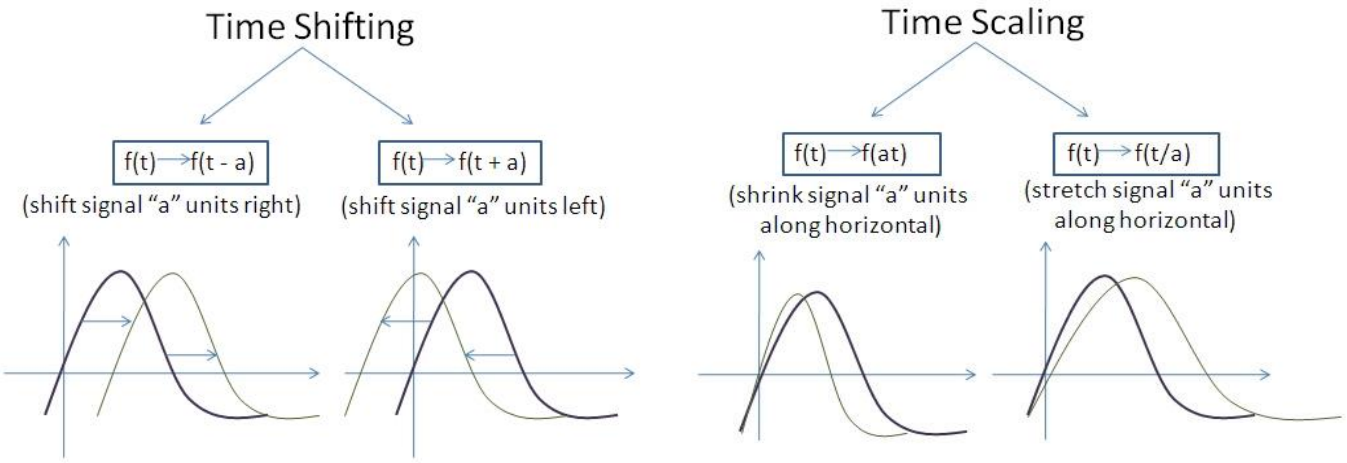
The time shifting of a continuous time signal x(t) is performed by replacing the independent variable t by t- m, to get the time shifted signal x(t- m), where m represents the time shift in seconds.

- In x(t- m), if m is positive, then the time shift results in a delay by m seconds. The delay results in shifting the original signal x(t) to right, to generate the time shifted signal x(t-m).
- In x(t - m), if m is negative, then the time shift results in an advance of the signal by |m| seconds. The advance results in shifting the original signal x(1) to left, to generate the time shifted signal x(t-m).

Time scaling of continuous signals:

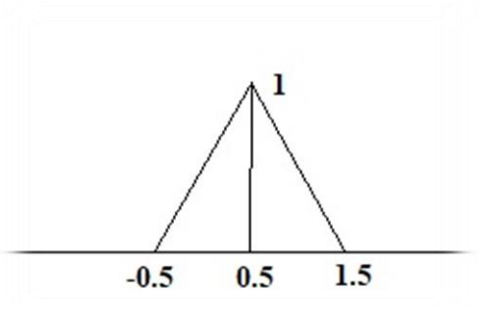
The time scaling is performed by multiplying the variable time by a constant. lf x(t) 1s a continuous time signal, then x(At) is the time scaled version of x(t), where A is a constant.

- When A > 1, then x(At) is the time compressed version of x() and when |A| < 1, then x(At) is the time expanded version of x(t).



Program:

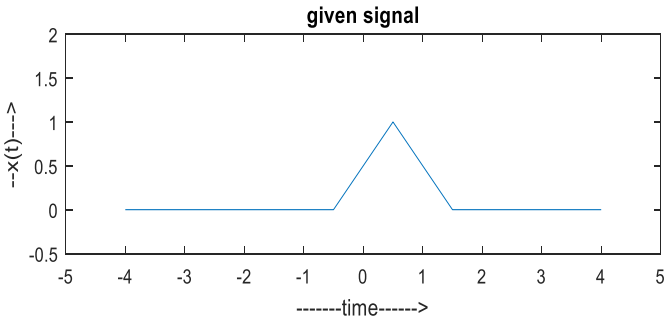
- To write a program to generate the plot. Also to plot the output for time advanced, time delayed, time expanded and time compressed signal.

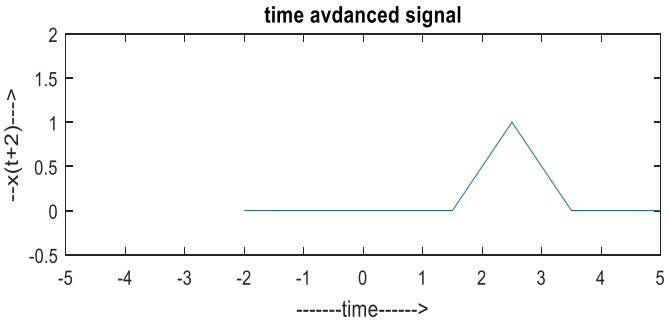


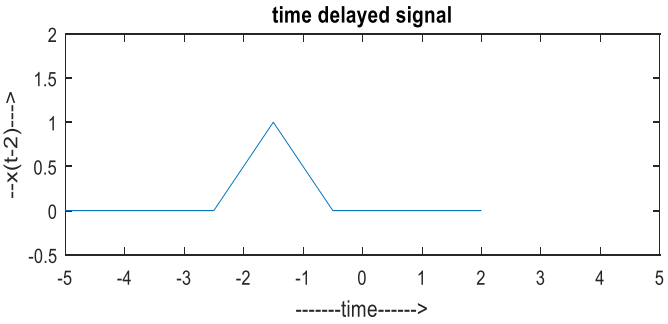

```
clc;
clear all;
close all;
t=-4:0.01:4;
```

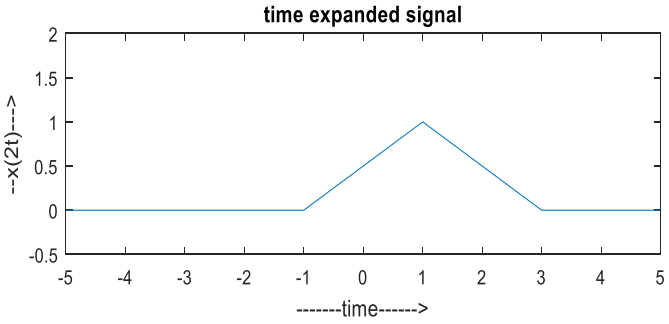
```
lt=length(t);
x=zeros(1,lt);

for i=1:lt
    if((t(i)<=0.5)&&(t(i)>=-0.5))
        x(i)=t(i)+0.5;
        end;
        if((t(i)<=1.5)&&(t(i)>=0.5))
        x(i)=1.5-t(i);
        end;
         subplot(3,2,1)
        plot(t,x)
        axis([-5 5 -0.5 2])
        title('given signal');
        xlabel('-------time------>');
        ylabel('--x(t)--->')

        subplot(3,2,3)
        plot(t+2,x)
        axis([-5 5 -0.5 2])
        title('time avdanced signal');
        xlabel('-------time------>');
        ylabel('--x(t+2)--->')

        subplot(3,2,4)
        plot(t-2,x)
        axis([-5 5 -0.5 2])
        title('time delayed signal');
        xlabel('-------time------>');
        ylabel('--x(t-2)--->')

        subplot(3,2,5)
        plot(2*t,x)
        axis([-5 5 -0.5 2])
        title('time expanded signal');
        xlabel('-------time------>');
        ylabel('--x(2t)--->')

        subplot(3,2,6)
        plot(0.5*t,x)
        axis([-5 5 -0.5 2])
        title('time compressed signal');
        xlabel('-------time------>');
        ylabel('--x(0.5t)--->')
end;
```

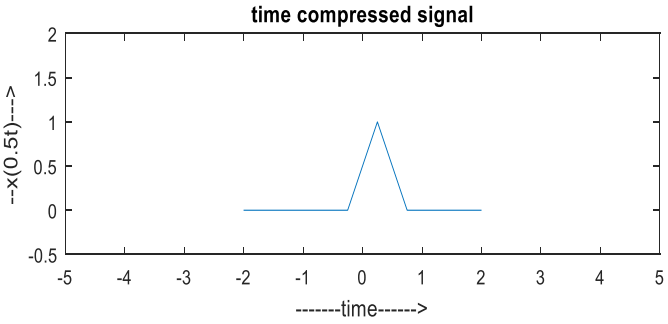
Output graphs:











Result:

With the help of MATLAB program the desired output is obtained and the result is verified.

Aim: Problems related to plotting of discrete and continuous signals and operations.

Software used: Matlab R2016a

Program:

- To plot the given signal y[n] = u[n] – u[n-10] and find:
  a) whether the signal is odd or even or none. b) even and odd components.

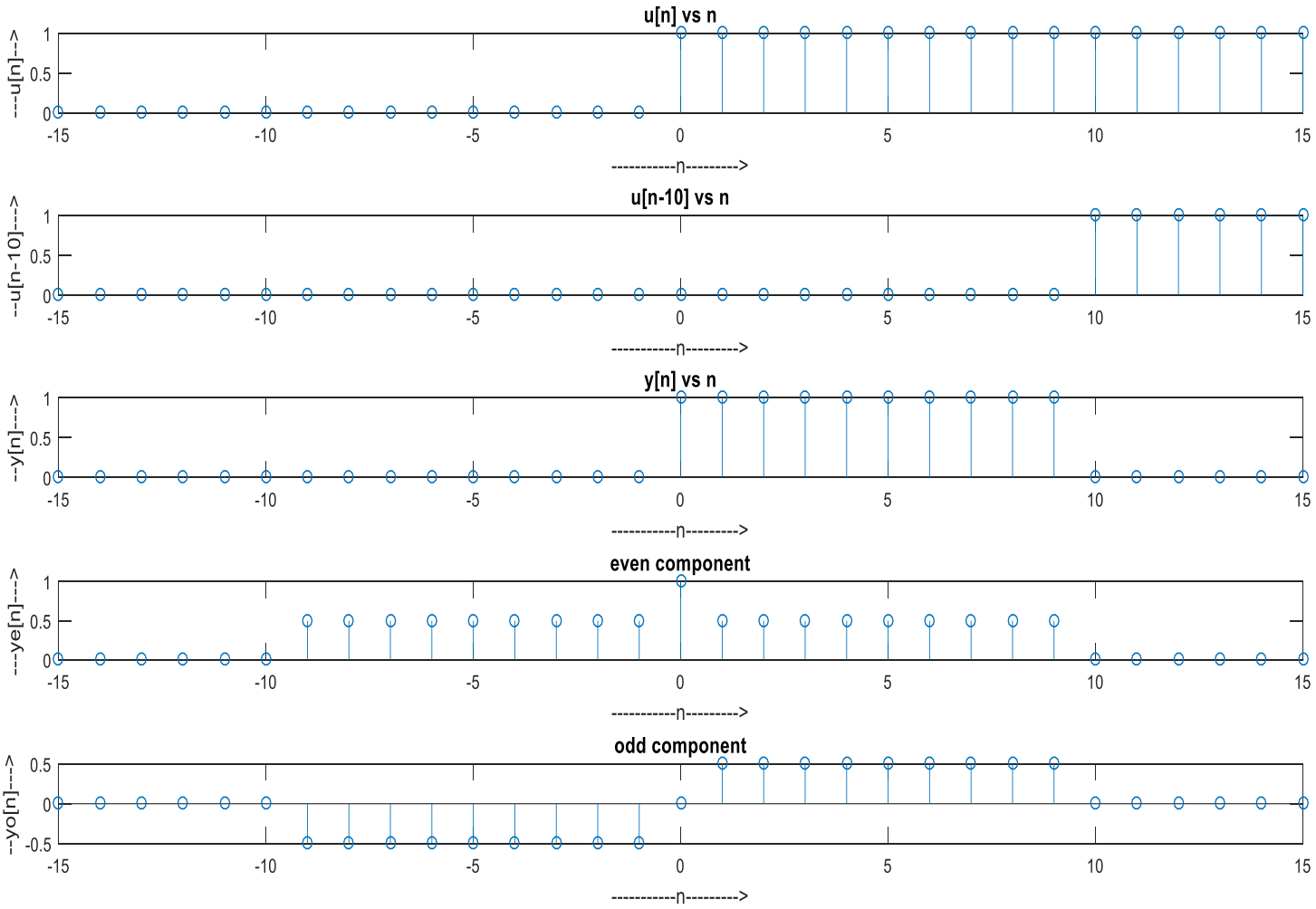
```
clc;
clear all;
close all;
n=-15:1:15;

y1=[zeros(1,15),ones(1,16)];
subplot(5,1,1);
stem(n,y1);
title('u[n] vs n')
xlabel('----------n--------->')
ylabel('---u[n]-->')

y2=[zeros(1,25),ones(1,6)];
subplot(5,1,2);
stem(n,y2);
title('u[n-10] vs n')
xlabel('----------n--------->')
ylabel('--u[n-10]--->')

y=y1-y2;
subplot(5,1,3);
stem(n,y);
title('y[n] vs n')
xlabel('----------n--------->')
ylabel('--y[n]--->')

x=fliplr(y);
if y==x
    disp('function is even');
else if y==(-x)
    disp('function is odd');
    else
        disp('function is neither odd nor even');
    end
end

e=(y+x)*0.5;
subplot(5,1,4);
```

```
stem(n,e);
title('even component');
xlabel('-----------n--------->')
ylabel('---ye[n]--->')

o=(y-x)*0.5;
subplot(5,1,5);
stem(n,o);
title('odd component');
xlabel('-----------n--------->')
ylabel('--yo[n]--->')
```

Command Window Output:

```
function is neither odd nor even
```

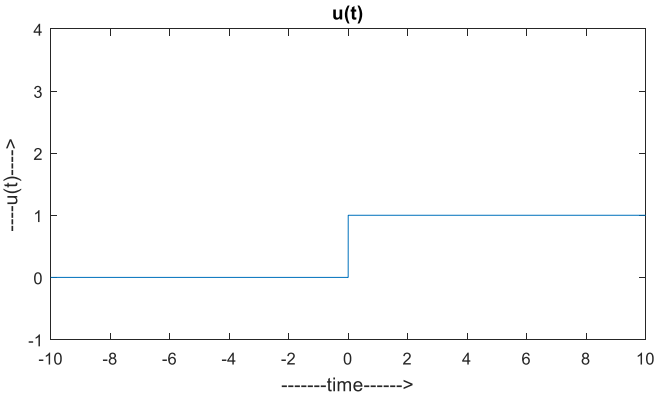
Output Graphs:

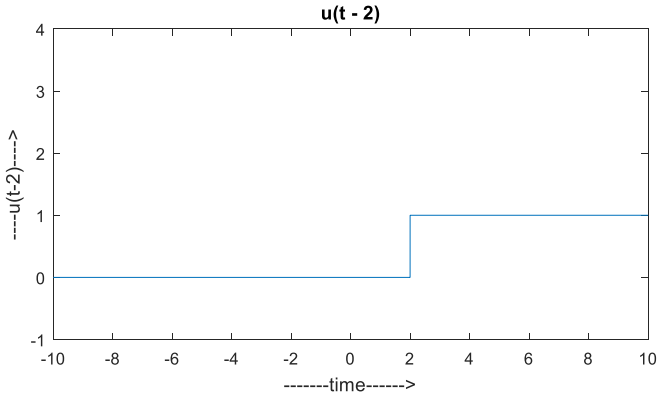


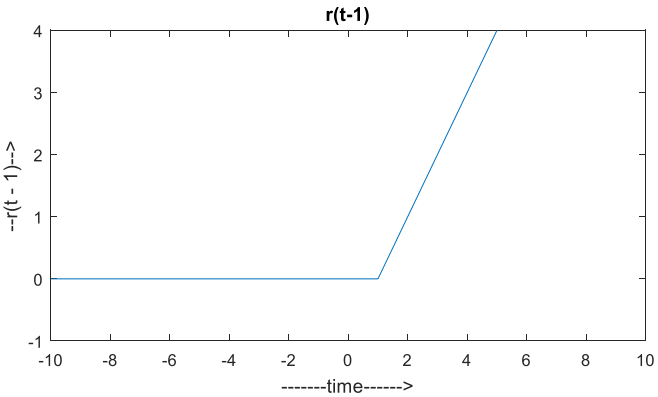
- To sketch the signal, x(t)= 2u(t) + t u(t)- (t-1) u (t-1)- 3u (t-2)

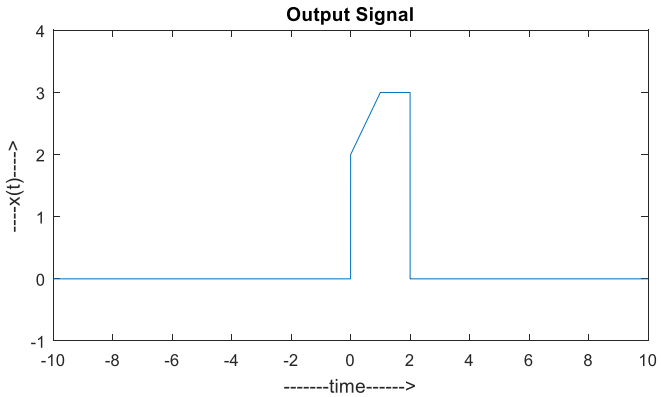
```
clc;
clear all;
close all;
t=-10:0.001:10;

u1(t>=0)=1;
u1(t<0)=0;
subplot(2,2,1);
plot(t,u1);                 %....plotting signal u(t)
axis([-10 10 -1 4])
xlabel('-------time------>');
ylabel('----u(t)---->');
title('u(t)');

u2(t>=2)=1;
u2(t<2)=0;
subplot(2,2,2);
plot(t,u2);                 %....plotting signal u(t-2)
axis([-10 10 -1 4])
xlabel('-------time------>');
ylabel('----u(t-2)---->');
title('u(t-2)');

R =( (t-1).*1);
R(t<=1) = 0;
subplot(2,2,3);
plot(t,R);                  %....plotting signal r(t-1)
axis([-10 10 -1 4])         % r(t-1)=(t-1)*u(t-1)
title('r(t-1)')
xlabel('-------time------>')
ylabel('--r(t-1)-->')

y=(2.*u1) + (t.*u1) - (R) -(3.*u2);
subplot(2,2,4)
plot(t,y)
axis([-10 10 -1 4])
xlabel('-------time------>');
ylabel('----x(t)---->');
title('Output Signal');
```

Output:







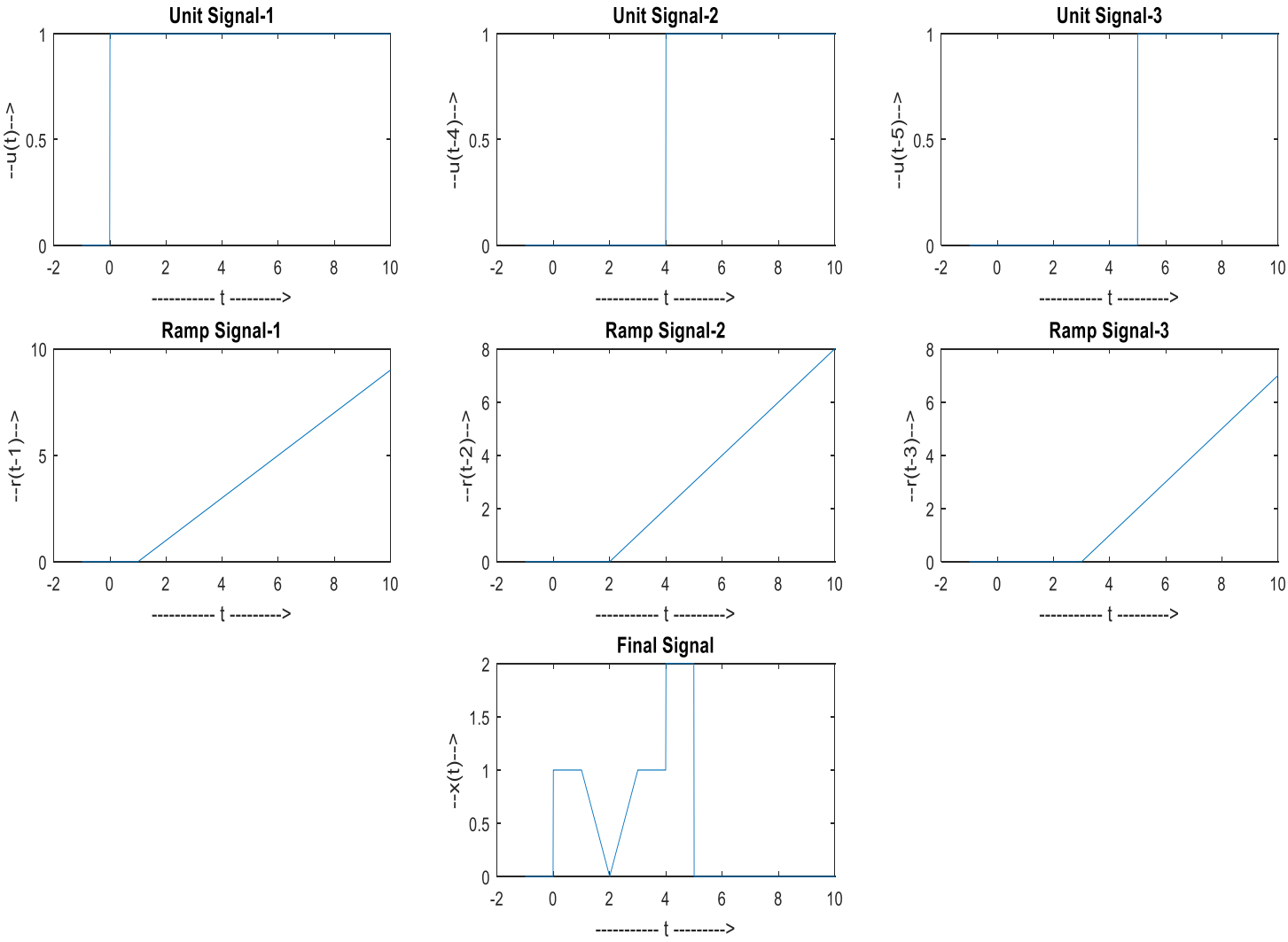


- To plot the signal with respect to the time,
  $x(t)= u(t) – r(t – 1) + 2r (t - 2) – r (t – 3) + u (t - 4) – 2 u (t – 5).$

```
clc;
close all;
clear all;

t= -1:0.01:10;

U1(t>=0) = 1;
U1(t<0) =0;
subplot(3,3,1);
plot(t,U1);                    %....plotting signal u(t)
title('Unit Signal-1');
xlabel('----------- t --------->')
ylabel('--u(t)-->')

U2(t>=4) = 1;
U2(t<4) =0;
subplot(3,3,2);
plot(t,U2);                     %....plotting signal u(t-4)
```

```
title('Unit Signal-2')
xlabel('---------- t --------->')
ylabel('--u(t-4)-->')

U3(t>=5) = 1;
U3(t<5) =0;
subplot(3,3,3);
plot(t,U3);                  %....plotting signal u(t-5)
title('Unit Signal-3')
xlabel('---------- t --------->')
ylabel('--u(t-5)-->')

R1 =( (t-1).*1);
R1(t<=1) = 0;
subplot(3,3,4);
plot(t,R1);                  %....plotting signal r(t-1)
title('Ramp Signal-1')
xlabel('---------- t --------->')
ylabel('--r(t-1)-->')

R2 =( (t-2).*1);
R2(t<=2) = 0;
subplot(3,3,5);
plot(t,R2);               %....plotting signal r(t-2)
title('Ramp Signal-2')
xlabel('---------- t --------->')
ylabel('--r(t-2)-->')

R3 =( (t-3).*1);
R3(t<=3) = 0;
subplot(3,3,6);
plot(t,R3);               %....plotting signal r(t-3)
title('Ramp Signal-3')
xlabel('---------- t --------->')
ylabel('--r(t-3)-->')

Y1 = (U1)-(R1)+( 2*R2)-(R3)+(U2)-(2*U3);
subplot(3,3,8)
plot(t,Y1)
title('Final Signal')
xlabel('---------- t --------->')
ylabel('--x(t)-->')
```

Output:



Result:

With the help of MATLAB program the desired output is obtained and the result is verified.

Aim: To perform linear convulation and circular convulation of two signals.

Software used: Matlab R2016a

Theory:

Convolution is a mathematical way of combining two signals to form a third signal. It is the single most important technique in Digital Signal Processing. Using the strategy of impulse decomposition, systems are described by a signal called the impulse response. Convolution is important because it relates the three signals of interest: the input signal, the output signal, and the impulse response.

The convulation of two continuous time signals x1(t) and x2(t) is defined as ,

$$x_3(t) = \int_{-\infty}^{\infty} x1(t).\ x2(t-\lambda)\, d\lambda \qquad \text{.....(a)}$$

where, $x_3(t)$ is the signal obtained by convolution x1(t) and x2(t),
and λ is a dummy variable used for integration;

the convolution relation of equation(a) can be symbolically expressed as,

$$x_3(t) = x1(t) * x(t)$$

where the symbol * indicates convolution operation.

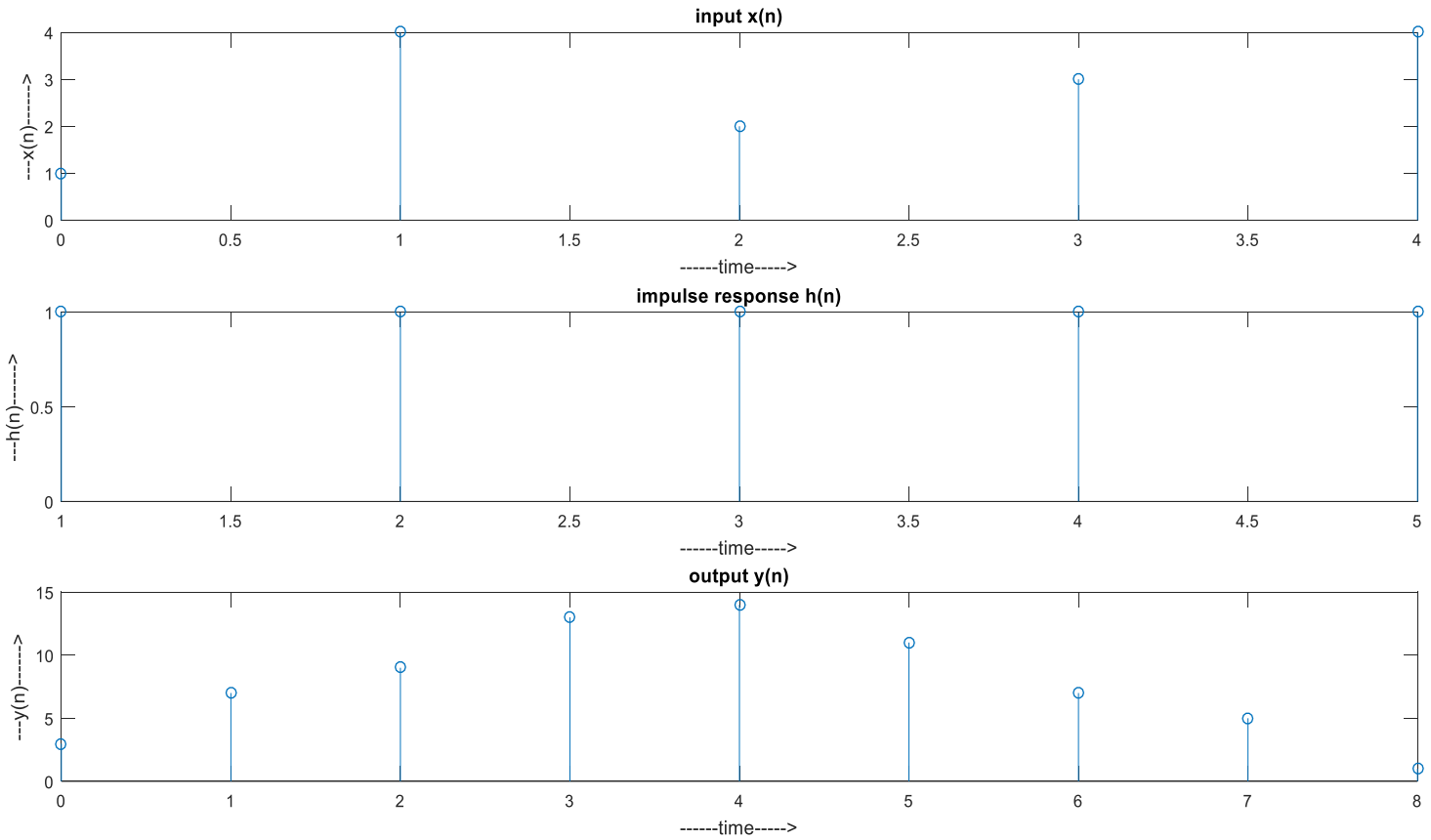
Program:

- To perform convolution of two discrete signals

```
clc;
clear all;
close all;
x=[3 4 2 4 1];
xn=[3 4 2 1 0];
h=[1 1 1 1 1];
hn=[3 4 5 2 1];
y=conv(x,h);
ny=0:length(y)-1;

subplot(3,1,1);
stem(xn,x);
title('input x(n)')
xlabel('------time----->');
ylabel('---x(n)------>')

subplot(3,1,2);
stem(hn,h);
title('impulse response h(n)')
xlabel('------time----->');
ylabel('---h(n)------>')
```

```
subplot(3,1,3);
stem(ny,y);
title('output y(n)')
xlabel('------time----->');
ylabel('---y(n)------>')
```

Output:

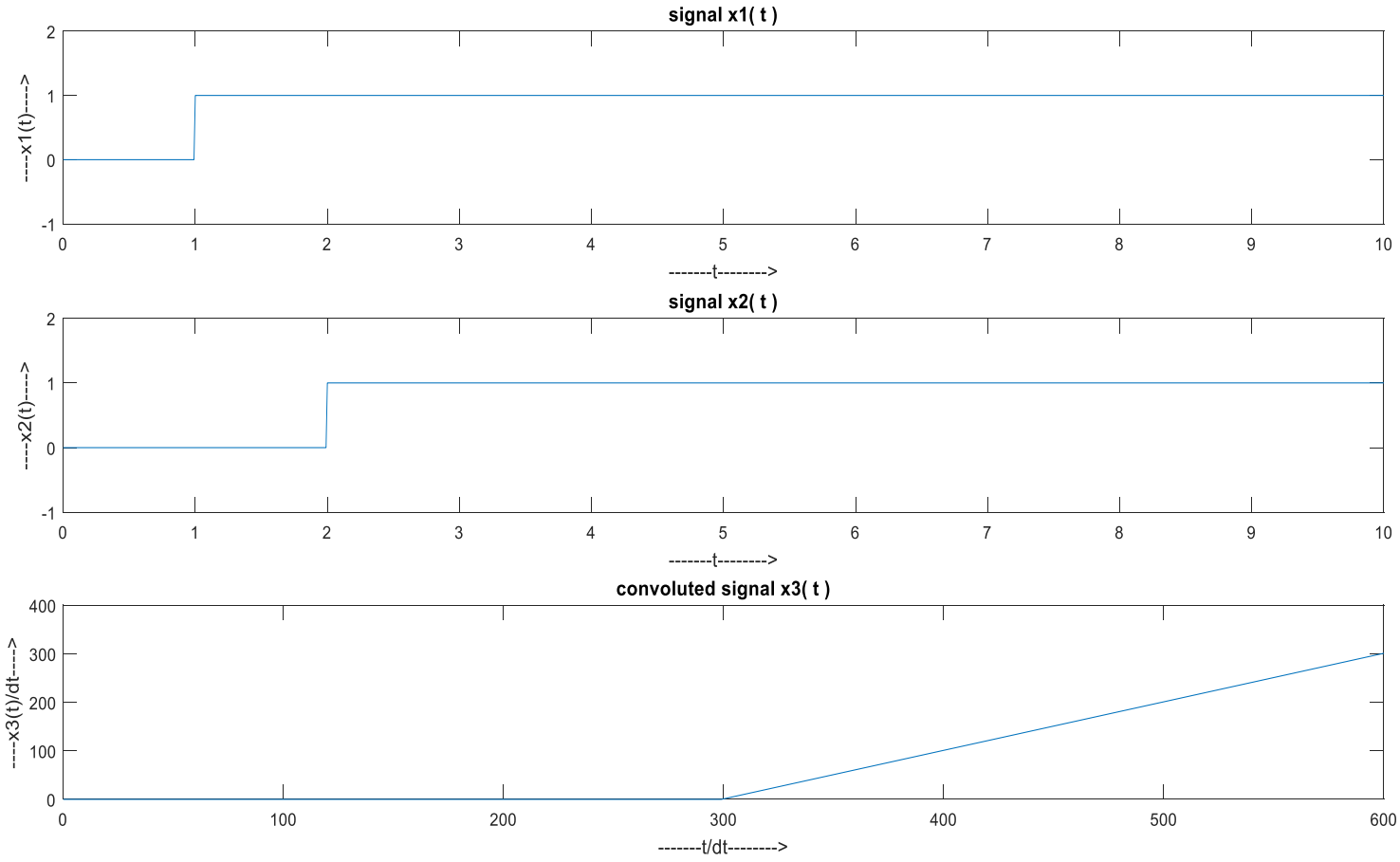


- To perform convolution of two continuous signals

```
% signals---> x1(t)=1 ; 1<t<0
%             x2(t)=1 ; 2<t<10
clc;
clear all;
close all;
t=0:0.01:10;
x1=1.*(t>=1 & t<=10);
x2=1.*(t>=2 & t<=10);
x3= conv(x1,x2);
n3=length(x3);
t1=0:1:n3-1;
```

```
subplot(3,1,1);
plot(t,x1);
xlabel('-------t-------->');
ylabel('----x1(t)---->');
title('signal x1(t)');
axis([0 10 -1 2])

subplot(3,1,2);
plot(t,x2);
xlabel('-------t-------->');
ylabel('----x2(t)---->');
title('signal x2(t)');
axis([0 10 -1 2])

subplot(3,1,3);
plot(t1,x3); xlim([0 600]);
xlabel('-------t/dt-------->');
ylabel('----x3(t)/dt---->');
title('convoluted signal x3(t)');
```

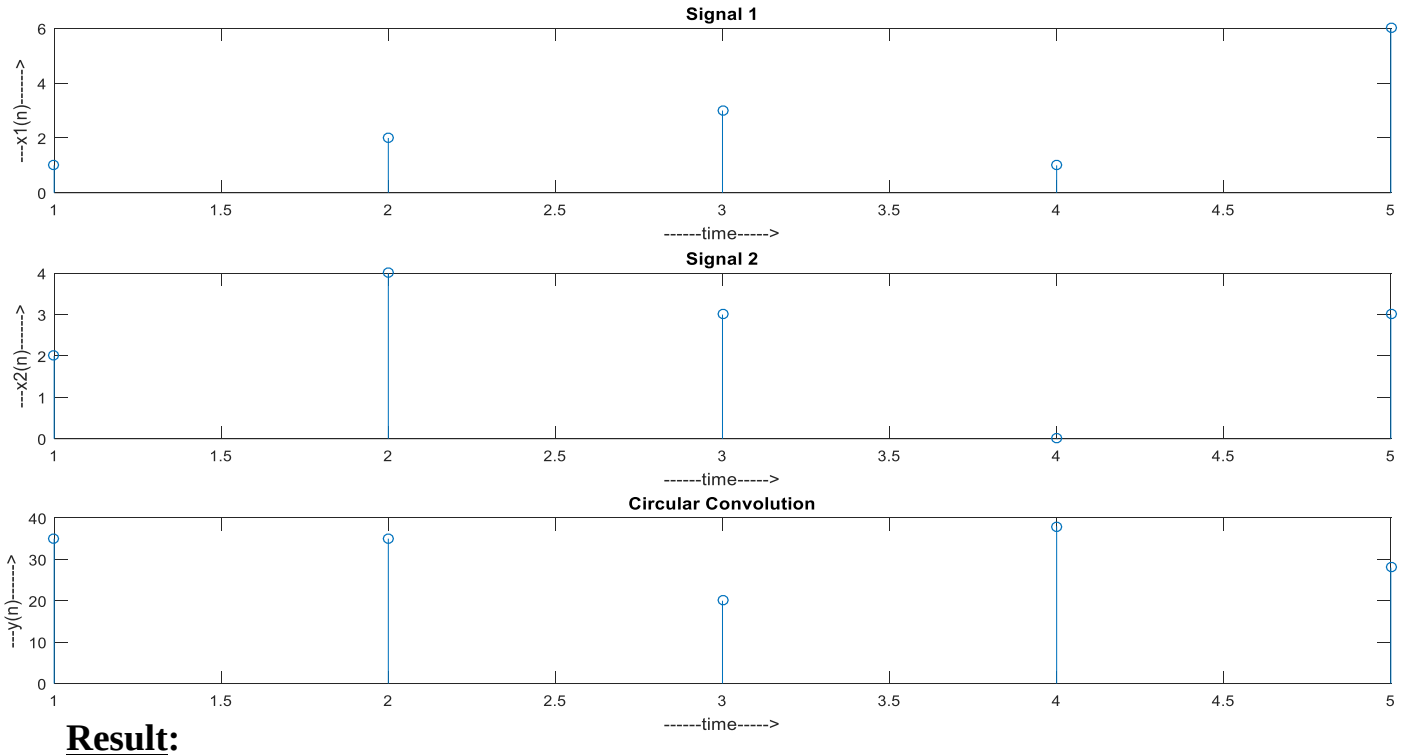
Output:



➢ To perform circular convolution of two discrete signals

```
clc;
close all;
clear all;
x1=[1 2 3 1 6];
x2=[2 4 3 0 3];
z=cconv(x1,x2,length(x1));
subplot(3,1,1)
stem(x1)
title('Signal 1')
xlabel('------time----->');
ylabel('---x1(n)------>');
subplot(3,1,2)
stem(x2)
title('Signal 2')
xlabel('------time----->');
ylabel('---x2(n)------>');
subplot(3,1,3)
stem(z)
title('Circular Convolution')
xlabel('------time----->');
ylabel('---y(n)------>');
```

Output:



Result:

With the help of MATLAB program the desired output is obtained and the result is verified.