**Problem 1** :

**Problem Name : Basic Operations**

**Theory :** Basic Operations Of Matrix ( Matrix Summation, Subtraction, Multiplication, Divition, Transpose, Zeros, Ones)

**Source Code :**

a=[1,2;3,4];

b=[5,6;7,8];

disp(a);

disp(b);

disp('Sum of a and b');

c=a+b;

disp(c);

disp('Subtraction of a and b');

d=a-b;

disp(d);

disp('Multiplication of a and b');

e=a\*b;

disp(e);

disp('Multiplication of each element by 2');

f=a.\*2;

disp(f);

disp('Division of each element by b');

g=a./b;

disp(g);

disp('Transpose of a matrix');

t=a';

disp(t);

disp('Zeros of 2\*3 matrix');

z=zeros(2,3);

disp(z);

disp('ones of 2\*3 matrix');

o=ones(2,3);

disp(o);

**Output**

Sum of a and b

6 8

10 12

Subtraction of a and b

-4 -4

-4 -4

Multiplication of a and b

19 22

43 50

Multiplication of each element by 2

2 4

6 8

Division of each element by b

0.2000 0.3333

0.4286 0.5000

Transpose of a matrix

1 3

2 4

Zeros of 2\*3 matrix

0 0 0

0 0 0

ones of 2\*3 matrix

1 1 1

1 1 1

**Problem 2**

**Problem Name : Sin wave generate and plot**

**Theory :** plot([X](file:///C:\Program%20Files\MATLAB\R2016a\help\matlab\ref\plot.html#inputarg_X),[Y](file:///C:\Program%20Files\MATLAB\R2016a\help\matlab\ref\plot.html#inputarg_Y)) creates a 2-D line plot of the data in Y versus the corresponding values in X . Sin wave consists of time in x axis and values of sin\*2\*pi\*t in y axis.

**Source Code :**

t = 0:0.01:2;

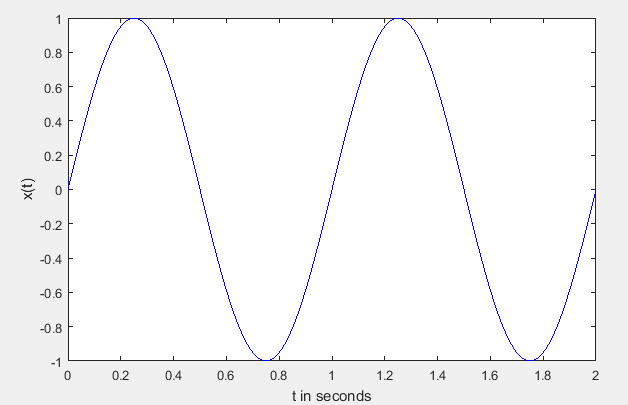
x = sin(2\*pi\*t);

plot(t,x,'b');

xlabel('t in seconds');

ylabel('x(t)');

**Output :**

****

**Problem 3**

**Problem Name : Sin wave generate with stem**

**Theory :** stem([X](file:///C:\Program%20Files\MATLAB\R2016a\help\matlab\ref\stem.html#inputarg_X),[Y](file:///C:\Program%20Files\MATLAB\R2016a\help\matlab\ref\stem.html#inputarg_Y)) plots the data sequence, Y, at values specified by X. The X and Y inputs must be vectors or matrices of the same size. Additionally, X can be a row or column vector and Y must be a matrix with length(X) rows.For plotting a set of discrete numbers or discrete time signals we willuse the setem which displays data values as a stem.

**Source Code :**

t = 0:1:40;

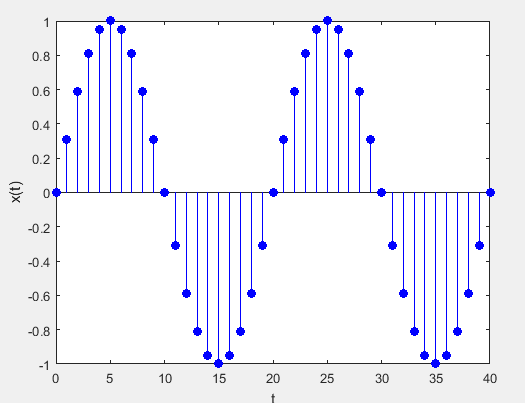
x = sin(0.1\*pi\*t);

stem(t,x,'b','filled');

xlabel('t');

ylabel('x(t)');

**Output :**



**Problem 4**

**Problem Name : Show Two sin wave using subplot**

**Theory :** subplot([m](file:///C:\Program%20Files\MATLAB\R2016a\help\matlab\ref\subplot.html#inputarg_m),[n](file:///C:\Program%20Files\MATLAB\R2016a\help\matlab\ref\subplot.html#inputarg_n),[p](file:///C:\Program%20Files\MATLAB\R2016a\help\matlab\ref\subplot.html#inputarg_p)) divides the current figure into an m-by-n grid and creates an axes for a subplot in the position specified by p. MATLAB® numbers its subplots by row, such that the first subplot is the first column of the first row, the second subplot is the second column of the first row, and so on. If the axes already exists, then the command subplot(m,n,p) makes the subplot in position p the current axes.

**Source Code :**

t = 0:0.01:2;

x = sin(2\*pi\*t);

subplot(1,2,1);

plot(t,x,'b');

xlabel('t in sec');

ylabel('value');

y=sin(2\*pi\*3\*t);

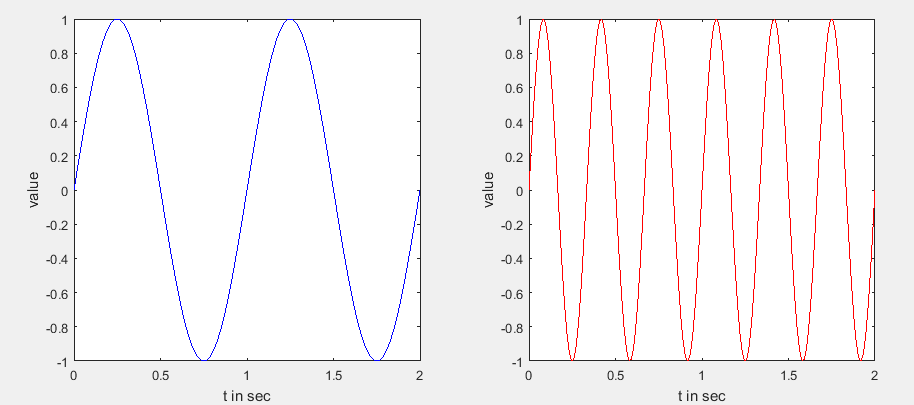
subplot(1,2,2);

plot(t,y,'r');

xlabel('t in sec');

ylabel('value');

**Output :**

****

**Problem 5**

**Problem Name : Plotting Two Curve In Different Figure**

**Theory :** figure creates a new figure window using default property values. This new figure window becomes the current figure, and it displays on top of all other figures on the screen. The title of the figure is an integer value that is not already used by an existing figure. MATLAB® saves this integer value in the figure's Number property.

**Source Code :**

t = 0:0.01:2;

x = sin(2\*pi\*t);

figure(1);

plot(t,x,'b');

xlabel('t in sec');

ylabel('value');

y=sin(2\*pi\*3\*t);

figure(2);

plot(t,y,'r');

xlabel('t in sec');

ylabel('value');

**Output :**

**Figure 1**

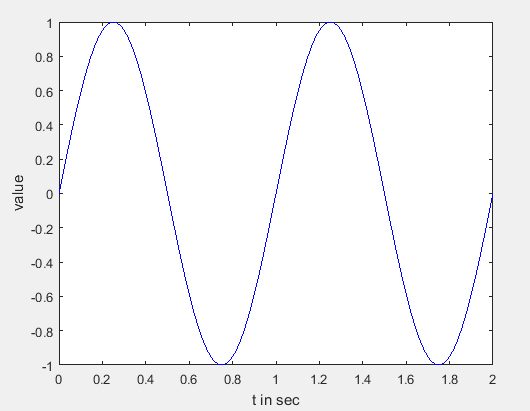
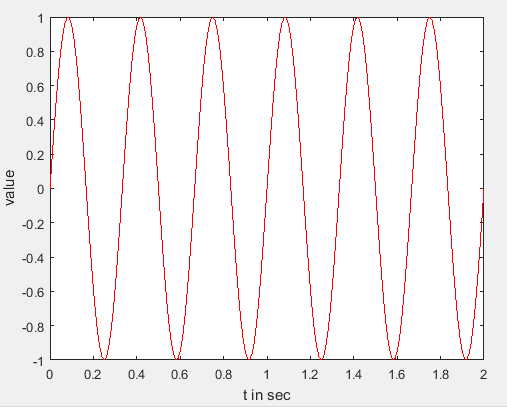


Figure 2



**Problem 6**

**Problem Name : Plotting Different Figure using hold on**

**Theory :** hold on retains plots in the current axes so that new plots added to the axes do not delete existing plots. New plots use the next colors and line styles based onthe ColorOrder and LineStyleOrder properties of the axes. MATLAB® adjusts axes limits, tick marks, and tick labels to display the full range of data.

**Source Code :**

t=0:0.01:2;

x=sin(2\*pi\*t);

y=sin(2\*pi\*2\*t);

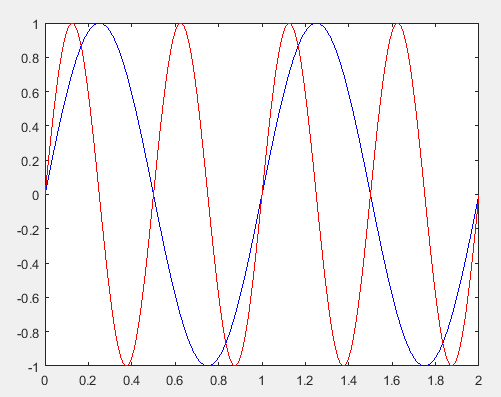
figure(1);

plot(t,x,'b');

hold on;

plot(t,y,'r');

**Output :**



**Problem 7**

**Problem Name : Representation of a sample sequence**

**Theory :** impulse calculates the unit impulse response of a dynamic system model. For continuous-time dynamic systems, the impulse response is the response to a Dirac input *δ*(*t*). if we break it down into sections: n-n0==0 will do the logical comparison of the array n-n0 to 0. This will result in a logical array with 1 where the index inside n-n0 is 0 and the rest 0. Then that result is stored in x.

**Source Code :**

a=imseq(4,-4,8);

n=[-4:8];

stem(n,a,'b');

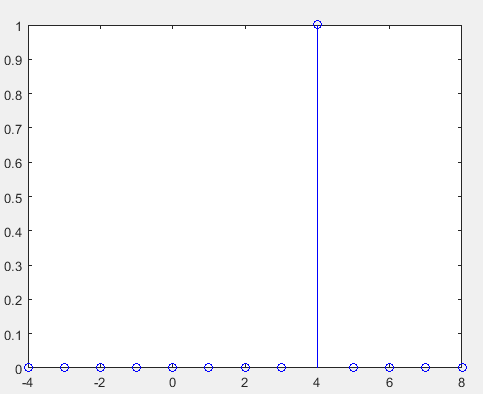
imseq function :

function [x,n]=imseq(n0,n1,n2)

n=[n1:n2];

x=[(n-n0)==0];

**Output**



**Problem 8**

**Problem Name : Impulse Sequence Generate**

**Theory :** Given x(n)=2y(4)+3y(-1)+4y(-2)-5y(3) . imseq function will generate impulse sequence for this equation . Multiplying the function by constant will be the result.

In MATLAB the function zeros(1,N) generates a row vector of *N*

zeros, which can be used to implement *δ*(*n*) over a finite interval. However,

the logical relation n==0 is an elegant way of implementing *δ*(*n*).

For example, to implement

*δ*(*n − n*0) =1*, n*= *n*0

and 0*, n!=n*0

over the *n*1 *≤n*0 *≤n*2 interval, we will use the following MATLAB

function.

**Source Code :**

a=imseq(4,-4,8);

n=[-4:8];

z=2.\*imseq(4,-4,8)+3.\*imseq(-1,-4,8)+4.\*imseq(-2,-4,8)-5.\*imseq(3,-4,8);

stem(n,z);

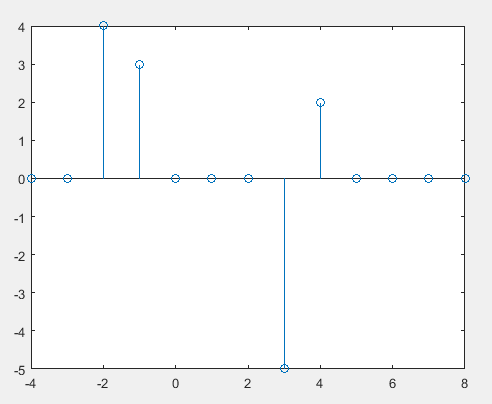
imseq function

function [x,n]=imseq(n0,n1,n2)

n=[n1:n2];

x=[(n-n0)==0];

**Output :**



**Problem 9**

**Problem Name : Unit Step Sequence Generate**

**Theory :** Given x(n)=u(3)-u(4). imstep function will generate unit step sequence for this equation .

**Source Code :**

a=imstep(4,-4,8);

n=[-4:8];

z=imstep(3,-4,8)-imstep(4,-4,8);

stem(n,z,'b');

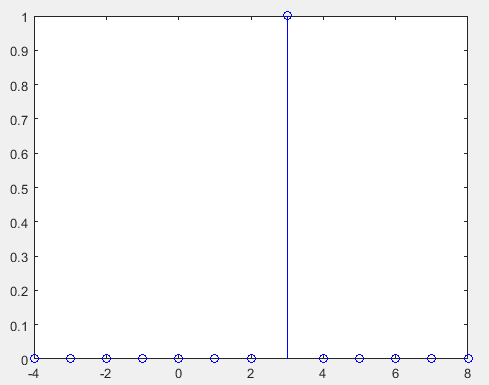
unit step function

function [x,n]=imstep(n0,n1,n2)

n=[n1:n2];

x=[(n-n0)>=0];

**Output :**



**Problem 10**

**Problem Name : Real-valued exponential sequence**:

**Theory :**

In MATLAB an array operator “.^” is required to implement a real exponential sequence.

For example, to generate

*x*(*n*) = (0*.*9)*n* 0 *≤n ≤* 10, we will need the following MATLAB script:

**Source Code :**

n = [0:10];

x = (0.9).^n;

stem(n,x);

**Output :**

