LAKSHMI NARAIN COLLEGE OF TECHNOLOGY EXCELLENCE

Simulation Lab
[EC-406]
Laboratory Manual

Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal

Submitted By: **Submitted To:**

Enrollment No:

Department of Electronics and Communication Engineering

Lakshmi Narain College of Technology Excellence, Bhopal

Department of Electronics & Communication Engineering

VISION OF THE DEPARTMENT

To become reputed in providing technical education in the field of electronics and communication engineering and produce technocrats working as leaders.

MISSION OF THE DEPARTMENT

1. *To provide congenial academic environment and adopting innovative learning process.*

2. *To keep valuing human values and transparency while nurturing the young engineers.*

3. *To strengthen the department by collaborating with industry and research organization of repute.*

4. *To facilitate the students to work in interdisciplinary environment and enhance their skills for employability and entrepreneurship.*

Lakshmi Narain College of Technology Excellence, Bhopal

Department of Electronics & Communication Engineering

PROGRAM SPECIFIC OUTCOME (PSO)

PSO1: Analyze specific engineering problems relevant to Electronics & Communication Engineering by applying the knowledge of basic sciences, engineering mathematics and engineering fundamentals.

PSO2: Apply and transfer interdisciplinary systems and engineering approaches to the various areas, like Communications, Signal processing, VLSI and Embedded system, PCB Designing.

PSO3: Inculcate the knowledge of Engineering and Management principles to meet demands of industry and provide solutions to the current real time problems.

PSO4: Demonstrate the leadership qualities and strive for the betterment of organization, environment and society.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Recognize and apply appropriate experimental and scientific skills to solve real world problems to create innovative products and systems in the field of electronics and communication engineering.

PEO2: To evolve graduates with ability to apply, analyze, design in Electronics & Communication Systems.

PEO3: Motivate graduates to become responsible citizens with moral & ethical values for the welfare of Society.

PEO4: Inculcate the habit of team work with professional quality of leadership to become successful contributors in industry and/ or entrepreneurship in view of Global & National status of technology.

Lakshmi Narain College of Technology Excellence, Bhopal

Department of Electronics & Communication Engineering

General Lab Policies

- All labs must be attended and completed. In the event that you cannot attend a lab, prior approval must be obtained. In such case, it is required that you make up the lab at another time.

- Satisfactory completion of all the labs is required for passing the course. If you are unable to complete the lab in the assigned lab time, it is expected that you would complete the lab on your own.

- At the beginning of a lab session, the instructor will collect report for the previous week's lab. Any late lab report will have its grade reduced. If, for any valid reason, you are not able to meet the deadline, let the instructor know well in advance of start of the lab.

- You are responsible for keeping track of your e-mail. In case of difficulty with any specific lab or any other problem concerning the lab, please come and talk to the lab instructor or use e-mail.

Student Name   **Instructor Name**
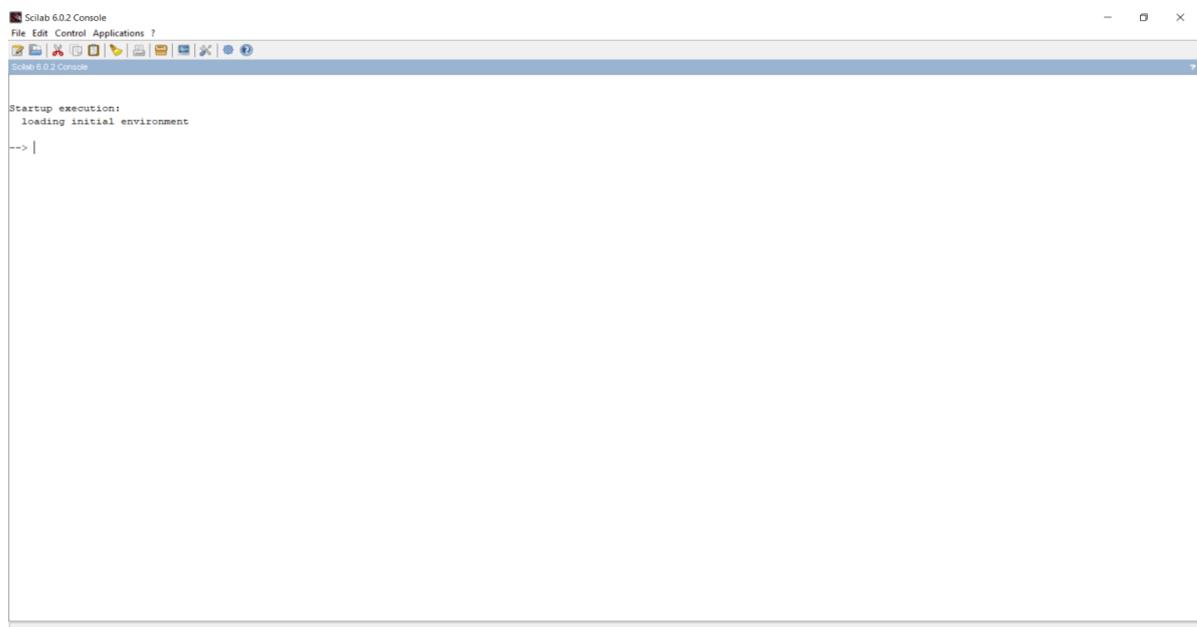
Signature _____   **Signature** _____

Lakshmi Narain College of Technology Excellence, Bhopal

Department of Electronics & Communication Engineering

EC-406 SIMULATION LAB INDEX

S.No.	List of Experiments	Date of Performing	Date of Submission	Remark
1.	Introduction to Scilab			
2.	Write a program to verify various vector operations			
3.	Define vector A and B and perform given calculations			
4.	Write a Scilab code to show the marks of ten students in bar graph			
5.	Perform basic mathematical operations on Scilab			
6.	Write programs to verify variousmatrix operations			
7.	Write a Scilab code to plot unit step discrete and unit step continuous signal			
8.	Perform radix conversion using Scilab			
9.	Study on matrix relational operations, logical operations and bitwise operations			
10.	Write a Scilab code to show 2D plot			
11.	Write for and while loop codes to perform given functions using Scilab edit window			
12.	Study the function command of Scilab			

13.	Write a code on edit window to check the sign of a number using if else conditions			
14.	Plot sine and cosine graphs using Scilab edit window			
15.	To study the application of linespace command			
16.	Study different types of graphs			
17.	Solve linear equations using matrix on Scilab			
18.	To plot graph of two functions			
19.	Generate discrete signal, exponential signal and ramp signal			
20.	To solve ordinary differential equations using Scilab			

Lakshmi Narain College of Technology Excellence, Bhopal

Department of Electronics & Communication Engineering

Practical-1

Aim: Introduction to Scilab.

Software required: Scilab version 6.0.2

Theory:

Scilab is an open-source, cross-platform numerical computation package and high level, numerically oriented programming language. It can be used for signal processing, statistical analysis, image enhancement, fluid dynamic simulations, numerical optimisation and modelling and many more.

Developer(s)	ESI Group
Stable release	6.1.0 / 25 February 2020; 6 months ago[1]
Repository	git.scilab.org/scilab
Written in	Scilab, C, C++, Java, Fortran
Operating system	BSDs (e.g., FreeBSD), Linux, macOS, Windows
Available in	English, German, Spanish, French, Italian, Japanese, Portuguese (Brazil), Russian, Ukrainian, Chinese, Czech, Polish
Type	Technical computing
License	GPLv2, previously CeCILL
Website	www.scilab.org

Scilab was created in 1990 by researchers from INRIA and ENPC. It was initially named Ψlab. Scilab Enterprises was created in June 2010. It develops and markets a comprehensive set of services for Scilab users. It also develops and maintains Scilab software. In early 2017, Scilab Enterprises was acquired by virtual prototyping pioneer, ESI group.

Scilab provides an interpreted programming environment with matrices as the main data type. This allows users to rapidly construct models for a range of mathematical problems. Scilab is available free of cost under an open source license.

Scilab syntax is largely based on MATLAB language. The simplest way to execute Scilab code is to type it in at the prompt, —>, in the graphical command window. In this way, Scilab can be used as an interactive mathematical shell.

The useful workspace in Scilab consists of several windows:

- Console for making calculations

- Editor for writing programs

- Graphics windows for displaying graphics

- Embedded help

Scilab environment by default consists of following docked windows- console, files and variables browsers, command history.

The menu bar:
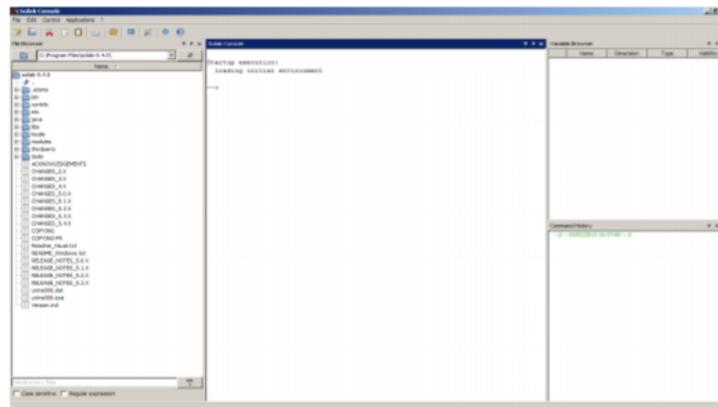
Applications

- The command history allows you to find all the commands from previous sessions to the current sessions.

- The variable browser allows you to find all the variables previously used during the current session.

Edit

- Clicking on <u>Clear Console</u> clears the entire content of the console but the command history is still available and calculations done during the session are still in memory.

Control

- Type pause in program or click on <u>Control>Interrupt</u> in the menu bar if the program is already running. In all cases, the prompt, —>, will turn into –1–>, then into –2–>, if the operation is repeated.

- To return to the time prior to the program interruption, type <u>resume </u>in the console or click on <u>Control>Resume</u>.

- To quit without any possibility of return, type <u>abort</u> in the console or click on <u>Control>Abort</u>.

The graphics windows:

A graphics window opens automatically when any graph is plotted.

It is possible to plot curves, surfaces, sequence of points.

To obtain an example of plot, type in the console:

—>plot
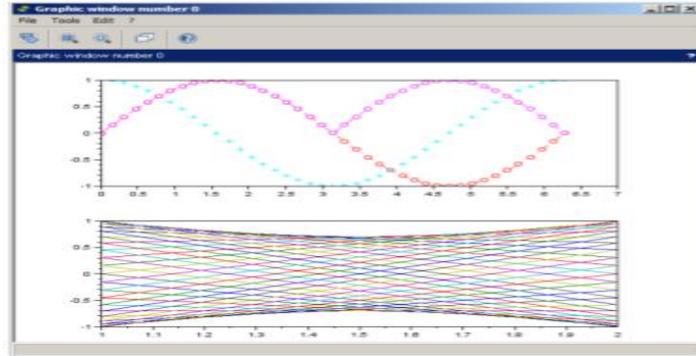
To access online help, click on <u>?>Scilab Help</u>in the menu bar or type in the console:

—>help
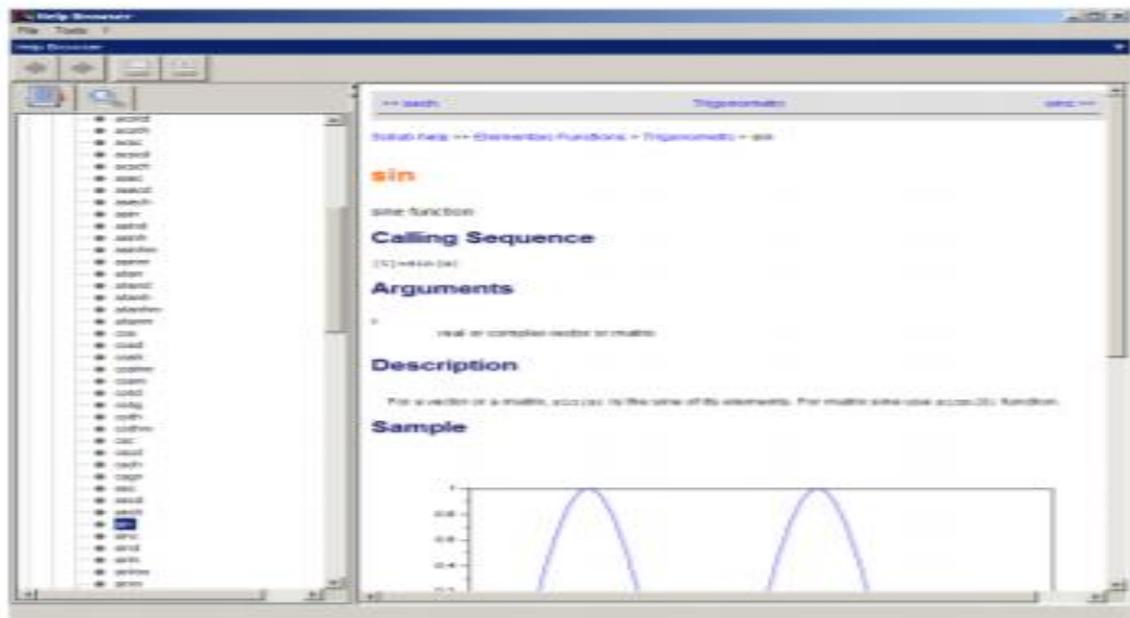
To get help with any function, type help in the console followed by the appropriate function. For example:

—>help sin

displays sine function.

PRACTICAL-2

Aim: Write a program to perform various vector operations.

Software required: Scilab version 6.0.2

Define two vectors A and B where

A= [15819]

B= [19851] and perform calculations.

```
Startup execution:
  loading initial environment

--> A=[1 5 8 19]
 A  =

   1.    5.    8.    19.


--> B=[19 8 5 1]
 B  =

   19.    8.    5.    1.


--> A'
 ans  =

   1.
   5.
   8.
   19.


--> B'
 ans  =

   19.
   8.
   5.
   1.
```

```
--> A'*B
 ans  =

   19.     8.      5.     1.
   95.    40.     25.     5.
  152.    64.     40.     8.
  361.   152.     95.    19.


--> B'*A
 ans  =

   19.    95.    152.    361.
    8.    40.     64.    152.
    5.    25.     40.     95.
    1.     5.      8.     19.


--> A'*B-B'*A
 ans  =

    0.    -87.   -147.   -360.
   87.     0.    -39.    -147.
  147.    39.      0.     -87.
  360.   147.     87.      0.


--> A*A'+B*B'
 ans  =

  902.
```

PRACTICAL -3

Aim: Define vector A and B and perform given calculations.

Software required: Scilab version 6.0.2

Define vectors A and B where

A= [1 0 1 0]

B= [2 3 4 5] and perform following calculations:

a) A+B

b) A-B

c) A'*B

d) A*B'

e) B'*A

f) B*A'

g) (A')^2*B^2

h) A^2*(B')^2

i) (B')^2*A^2

j) B^2*(A')^2

Also find out the length of A,B and sum of A and B

```
--> length(A)
 ans  =

    4.


--> length (B)
 ans  =

    4.


--> sum(A)+ sum(B)
 ans  =

   16.


--> sum(A+B)
 ans  =

   16.
```

PRACTICAL-4

Aim: Write a scilab code to show the marks of ten students in bar graph.

Software required: Scilab version 6.0.2

```
--> x= [1:10]
 x  =

   1.   2.   3.   4.   5.   6.   7.   8.   9.   10.


--> x= [1:10];

--> y= [50, 55, 60, 66, 70, 77, 80, 88, 90, 99];

--> xlabel('students');

--> ylabel('marks');

--> title('marks of 10 students');

--> bar(x,y);
```

The graph for the above code is plotted on the graphic window as shown below:

marks of 10 students
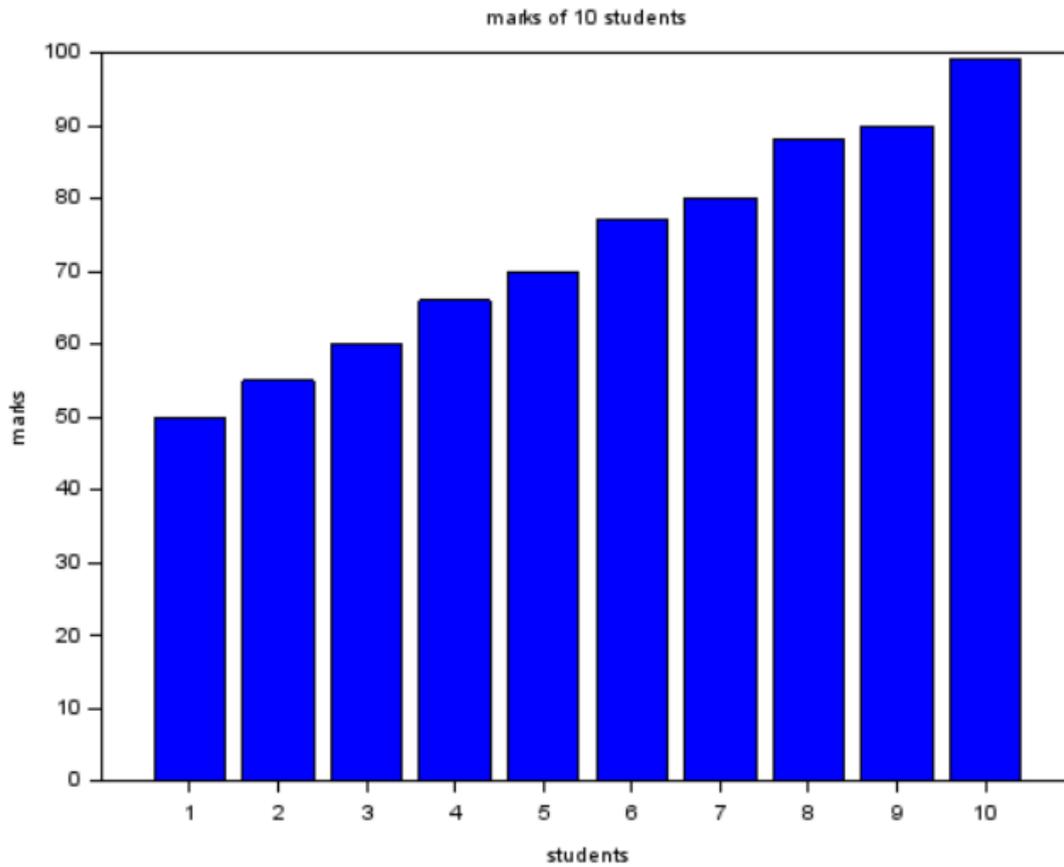
Result: Hence study of bar plot successfully performed.

PRACTICAL-5

Aim: Perform basic mathematical operations on Scilab.

Software required: Scilab version 6.0.2

```
--> 2+3
 ans  =

    5.


--> 8-9
 ans  =

   -1.


--> (2+3)^5
 ans  =

   3125.


--> %pi
 %pi  =

   3.1415927


--> %e
 %e  =

   2.7182818


--> factorial(4)
 ans  =

   24.


--> factor(26)
 ans  =

   2.    13.
```

```
--> X= [30,45,60]
 X  =

   30.    45.    60.
--> lcm(X)
 ans  =

   180.


--> gcd(X)
 ans  =

   15.
--> Y= [2,3,4,4,5,6,7,4,8,15,22,1,0,5]
 Y  =

   2.   3.   4.   4.   5.   6.   7.   4.   8.   15.   22.   1.   0.   5.


--> mean(Y)
 ans  =

   6.1428571
--> median(Y)
 ans  =

   4.5


--> C= complex(2,4)
 C  =

   2. + 4.i


--> complex([3 6 2], 7)
 ans  =

   3. + 7.i   6. + 7.i   2. + 7.i
```

```
--> x= 15.56
 x  =

    15.56


--> floor(x)
 ans  =

    15.


--> ceil(x)
 ans  =

    16.


--> floor(9.99)
 ans  =

    9.


--> ceil(58.01)
 ans  =

    59.


--> abs(58+45*%i)
 ans  =

    73.409809
```

PRACTICAL-6

Aim: Write programs to verify various matrix operations.

Software used: Scilab version 6.0.2

Perform following operations:

a) Excess the element of a row and column for a given matrix
   E=[5, 19, 15; 8, 22, 36]

b) Generate identity matrix of order 3*3

c) Determine determinant, inverse and eigen value for given matrix
   A= [1, 4, 0;2, 3, 1;4, 1, 5]

d) Extract second and third rows of the matrix X= [1, 2, 3; 4, 5, 6;
   7, 8, 9]

e) Extract second column of matrix Y= [10, 11, 12; 13, 14, 15; 16,
   17, 18]

f) Generate a zero matrix of order 3*3

g) Generate a matrix of order 3*3 whose all elements are one.

a) Excess the element of a row and column for a given matrix E= [5, 19, 15; 8, 22, 36]

```
--> E= [5 19 15; 8 22 36]
 E  =

    5.    19.    15.
    8.    22.    36.


--> E(1,2)
 ans  =

    19.


--> E(1,1)
 ans  =

    5.


--> E(2,1)
 ans  =

    8.


--> E(2,3)
 ans  =

    36.
```

b) Generate identity matrix of order 3*3

```
--> eye(3,3)
 ans  =

    1.    0.    0.
    0.    1.    0.
    0.    0.    1.
```

c) Determine determinant, inverse and eigen value for given matrix
A= [1, 4, 0;2, 3, 1;4, 1, 5]

```
--> A= [1 4 0;2 3 1;4 1 5]
 A  =

    1.    4.    0.
    2.    3.    1.
    4.    1.    5.


--> det(A)
 ans  =

  -10.


--> inv(A)
 ans  =

  -1.4    2.    -0.4
   0.6  -0.5    0.1
   1.    -1.5    0.5


--> spec(A)
 ans  =

    6.6801436
   -0.5260158
    2.8458722
```

d) Extract second and third rows of the matrix X= [1, 2, 3; 4, 5, 6; 7, 8, 9]

```
--> X= [1 2 3; 4 5 6; 7 8 9]
 X  =

    1.    2.    3.
    4.    5.    6.
    7.    8.    9.


--> X(2:3,:)
 ans  =

    4.    5.    6.
    7.    8.    9.
```

e) Extract second column of matrix Y= [10, 11, 12; 13, 14, 15; 16, 17, 18]

```
--> Y= [10 11 12; 13 14 15; 16 17 18]
 Y  =

    10.    11.    12.
    13.    14.    15.
    16.    17.    18.


--> Y(:,2,:)
 ans  =

    11.
    14.
    17.
```

f) Generate a zero matrix of order 3*3

```
--> zeros(3,3)
 ans  =

    0.    0.    0.
    0.    0.    0.
    0.    0.    0.
```

g) Generate a matrix of order 3*3 whose all elements are one

```
--> ones(3,3)
 ans  =

    1.    1.    1.
    1.    1.    1.
    1.    1.    1.
```

PRACTICAL-7

Aim: Write a Scilab code to plot unit step discrete and unit step continuous signal.

Software required: Scilab version 6.0.2

```
--> t=(0:4);

--> y= ones(1,5);

--> subplot(2,1,1);

--> plot2d3(t,y);

--> xlabel('time');

--> ylabel('amplitude');

--> title('unit step discrete signal');

--> subplot(2,1,2);

--> plot(t,y);
--> xlabel('time');

--> ylabel('amplitude');

--> title('unit step continuous signal');
```

unit step discrete signal

unit step continuous signal
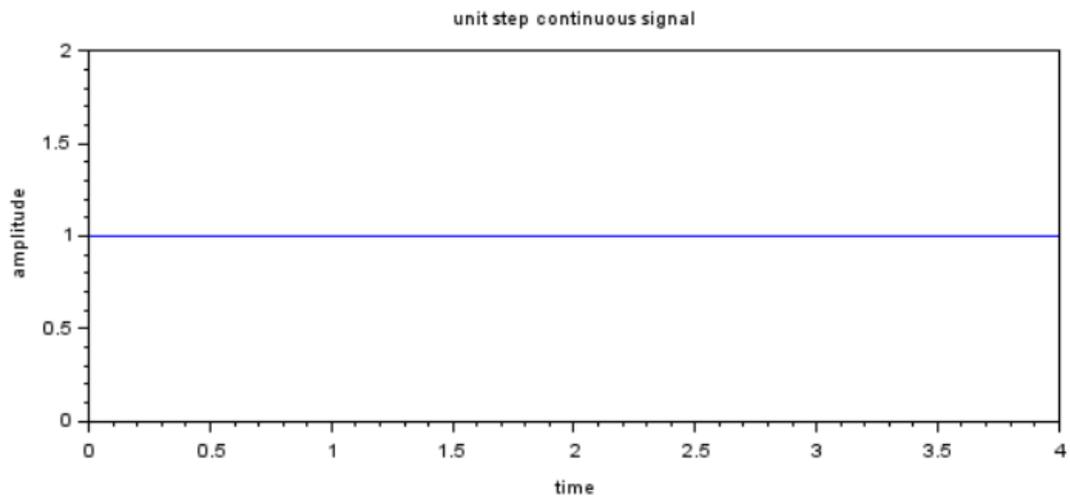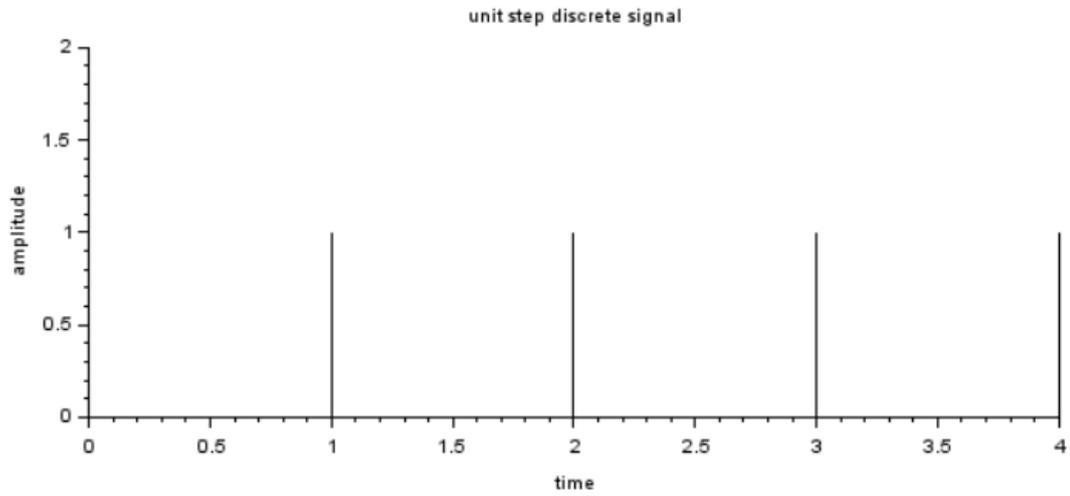
PRACTICAL-8

Aim: Perform radix conversion on Scilab.

Software required: Scilab version 6.0.2

Convert from:

a) Decimal to binary

```
--> dec2bin(27)
 ans  =

 11011
```

b) Decimal to octal

```
--> dec2oct(27)
 ans  =

 33
```

c) Decimal to hexadecimal

```
--> dec2hex(27)
 ans  =

 1B
```

d) Binary to decimal

```
--> bin2dec('11011')
 ans  =

   27.
```

e) Octal to decimal

```
--> oct2dec('33')
 ans  =

   27.
```

e) Hexadecimal to decimal

```
--> hex2dec('1B')
 ans  =

   27.
```

PRACTICAL-9

Aim: Study on matrix relational operations, logical operations and bitwise operations.

Software required: Scilab version 6.0.2

A= [5 5 5; 5 5 5; 5 5 5]

B= [1 2 3; 4 5 6; 7 8 9]

```
--> A= [5 5 5; 5 5 5; 5 5 5]
 A  =

   5.   5.   5.
   5.   5.   5.
   5.   5.   5.


--> B= [1 2 3; 4 5 6; 7 8 9]
 B  =

   1.   2.   3.
   4.   5.   6.
   7.   8.   9.
```

a) Relation operations

```
--> A>B
 ans  =

  T T T
  T F F
  F F F


--> A<B
 ans  =

  F F F
  F F T
  T T T
```

```
--> A>=B
 ans  =

   T  T  T
   T  T  F
   F  F  F


--> A<=B
 ans  =

   F  F  F
   F  T  T
   T  T  T
--> A==B
 ans  =

   F  F  F
   F  T  F
   F  F  F


--> A~=B
 ans  =

   T  T  T
   T  F  T
   T  T  T
```

b) Logical operations

```
--> A&B
 ans  =

   T  T  T
   T  T  T
   T  T  T
```

```
--> A|B
 ans  =

  T T T
  T T T
  T T T


--> ~A
 ans  =

  F F F
  F F F
  F F F
--> ~B
 ans  =

  F F F
  F F F
  F F F
```

c) Bitwise operations

```
--> bitand(A,B)
 ans  =

   1.   0.   1.
   4.   5.   4.
   5.   0.   1.


--> bitor(A,B)
 ans  =

   5.   7.   7.
   5.   5.   7.
   7.  13.  13.
```

```
--> bitxor(A,B)
 ans  =

   4.    7.    6.
   1.    0.    3.
   2.   13.   12.
```

PRACTICAL-10

Aim: Write a Scilab code to show 2D plot.

Software required: Scilab version 6.0.2

```
--> x= linspace(1,10,5)
 x  =

    1.    3.25    5.5    7.75    10.


--> y= linspace(1,20,5)
 y  =

    1.    5.75    10.5    15.25    20.


--> plot(x,y);
```
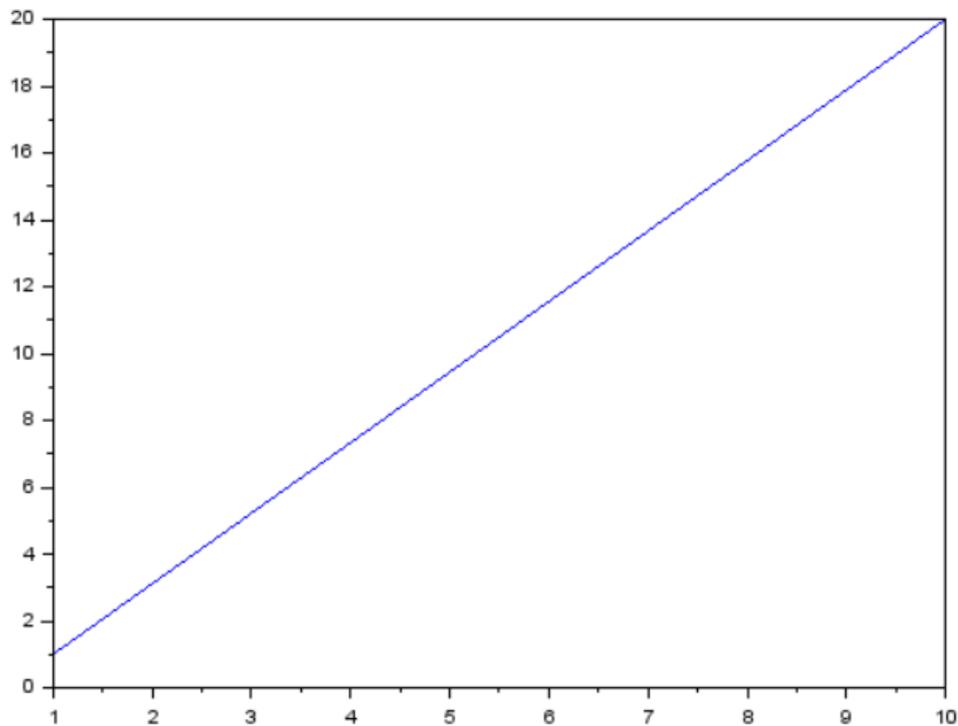
```
--> plot2d(x,y,-4);
```



```
--> plot2d(x,y,6);
```
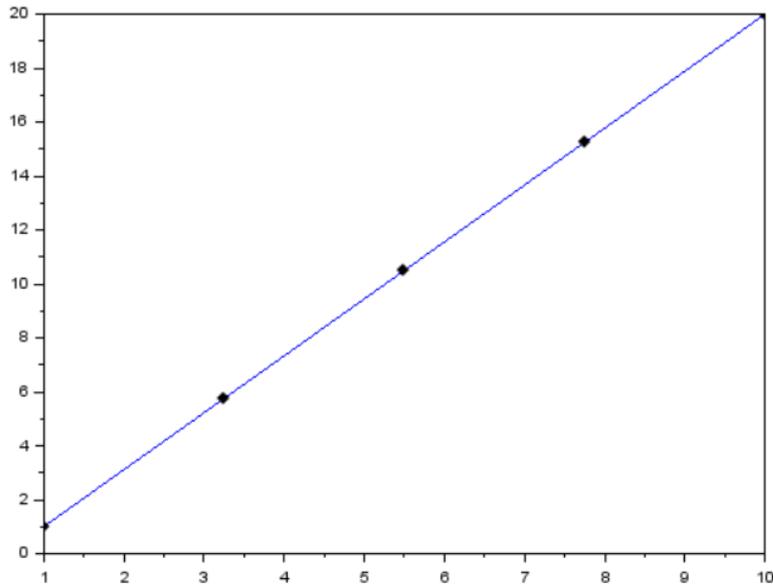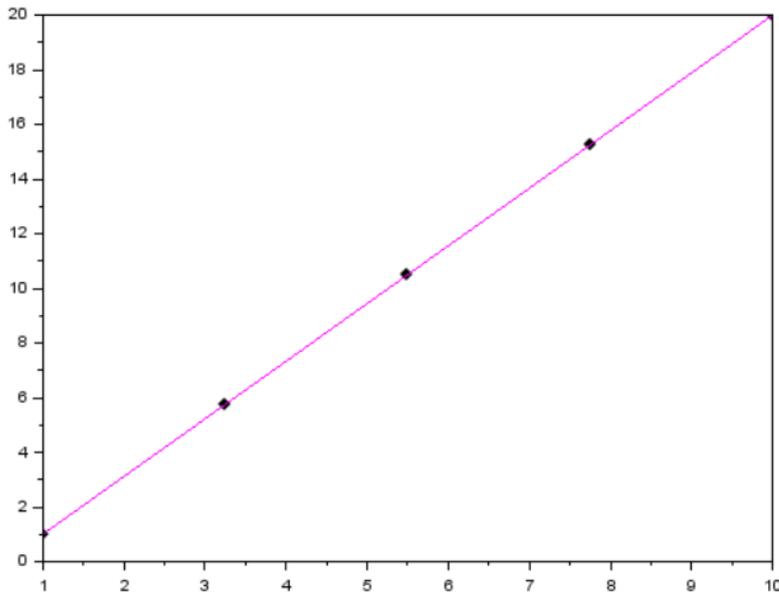
PRACTICAL-11

Aim: Write for and while loop codes to perform given functions using Scilab edit window.

Software used: scilab version 6.0.2

a) Compute sum of numbers from 1 to 10 using for loop and display the result

```
1 //finding the sum of integers from 1 to 10 using for loop
2 s=0;
3 for i=1:10;
4     s=s+i
5 end
6 disp(s,'s=');
```

Result:

```
--> exec('E:\Scilab Practicals\Edit window\Sum from 1 to 10.sce', -1)

 s=

   55.
```

b) Compute sum of numbers from 1 to 10 using while loop and display the result

```
1 //sum of numbers from 1 to 10 using while loop
2 d=0;
3 j=1;
4 while j<=10
5     d=d+j;
6     j=j+1;
7 end
8 disp(d,'d=');
```

Result:

```
--> exec('E:\Scilab Practicals\Edit window\While loop.sci', -1)

 d=

   55.
```

c) Display following series on the console

1, 2, 3, 5, 8, 13, 21, 34, 55, 89

```
1  //1-2-3-5-8-13-21-34-55
2  s=zeros(1,10)
3  s(1)=1
4  s(2)=2
5  for i=3:10
6      s(i)=s(i-2)+s(i-1)
7
8  end
9      disp(s,'s=')
```

Result:

```
--> exec('E:\Scilab Practicals\Edit window\print series.sci', -1)

 s=

   1.   2.   3.   5.   8.   13.   21.   34.   55.   89.
```

PRACTICAL-12

Aim: Study the function command of Scilab.

Software required: Scilab version 6.0.2

a) Form a function to add two numbers

```
--> function y = add(a,b)
  > y = a+b
  > endfunction

--> add(24,67)
 ans  =

   91.
```

b) Form a function to calculate square and cube of a number

```
--> function [a,b] = sqrcb(x)
  > a = x^2
  > b = x^3
  > endfunction

--> [s,q] = sqrcb(9)
 q  =

   729.

 s  =

   81.
```

PRACTICAL-13

Aim: Write a code on edit window to check the sign of a number using if else condition.

Software required: Scilab version 6.0.2

```
//if else statement
//check the sign of the number
a=input("Enter the number: ")
if a<0 then
    disp("The number is negative")
elseif a==0 then
    disp("The number is zero")
else
    disp("The number is positive")
end
```

Result:

```
--> exec('E:\Scilab Practicals\Edit window\check sign of number.sce', -1)
Enter the number: 5


 The number is positive

--> exec('E:\Scilab Practicals\Edit window\check sign of number.sce', -1)
Enter the number: -4


 The number is negative

--> exec('E:\Scilab Practicals\Edit window\check sign of number.sce', -1)
Enter the number: 0


 The number is zero
```

PRACTICAL-14

Aim: To plot sine and cosine graphs using Scilab edit window

Software required: Scilab version 6.0.2

a) Sine graph

```
//plot sine curve
x=[0:0.1:2*%pi];
plot2d(x,sin(x),style=15)
xtitle("a plot of sine curve")
xlabel("angle(radians)")
ylabel("sin(x)")
```

Result:
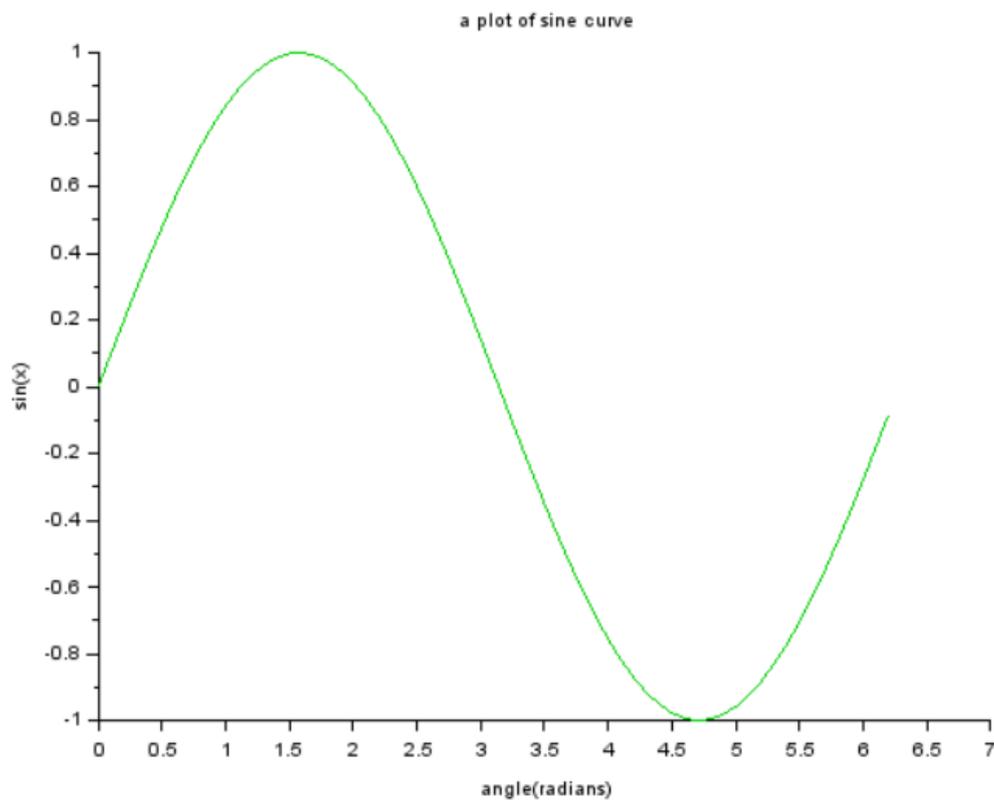
Cosine graph

```
//plot cosine curve
x=[0:0.1:3*%pi];
plot2d(x,cos(x),style=15)
xtitle("a plot of cosine curve")
xlabel("angle(radians)")
ylabel("cos(x)")
```

Result:
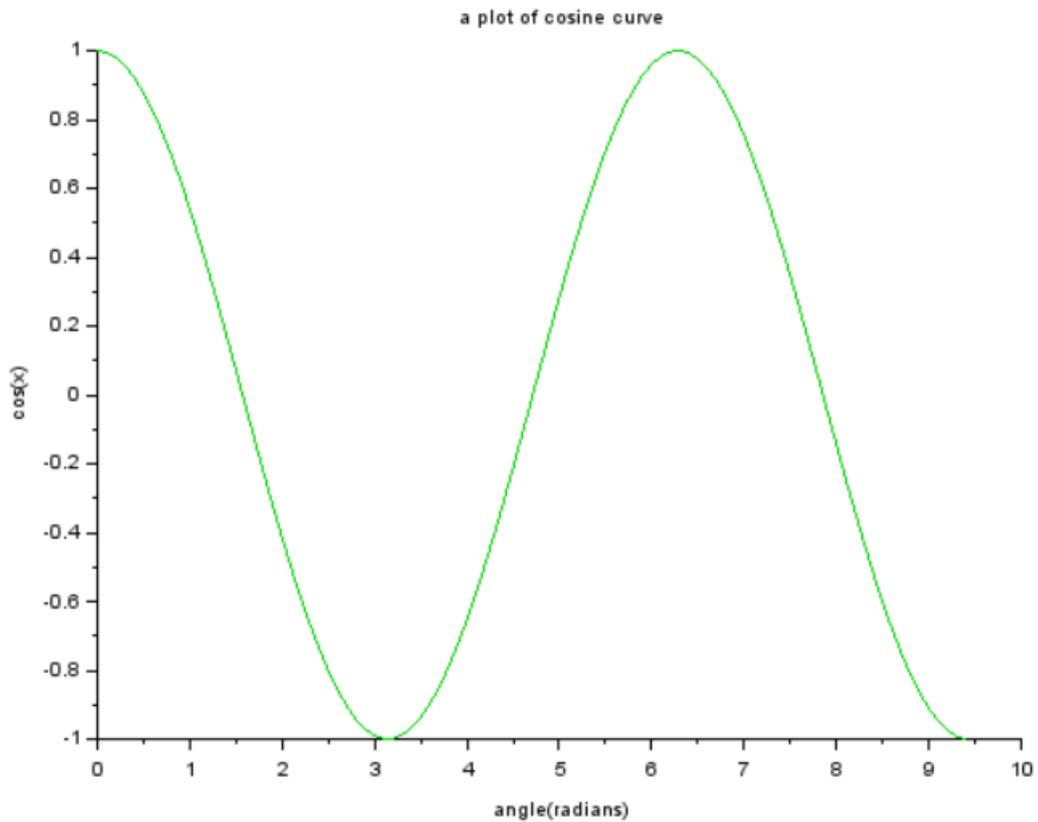
a plot of cosine curve

Sine and cosine graph with their difference

```
//sine and cosine graph
x=linspace(0,2*%pi,100);
x=x';
y=sin(x);
z=cos(x);
plot2d(x,[y,z]);
plot2d(x,[y,z,y-z]);
```

Result:
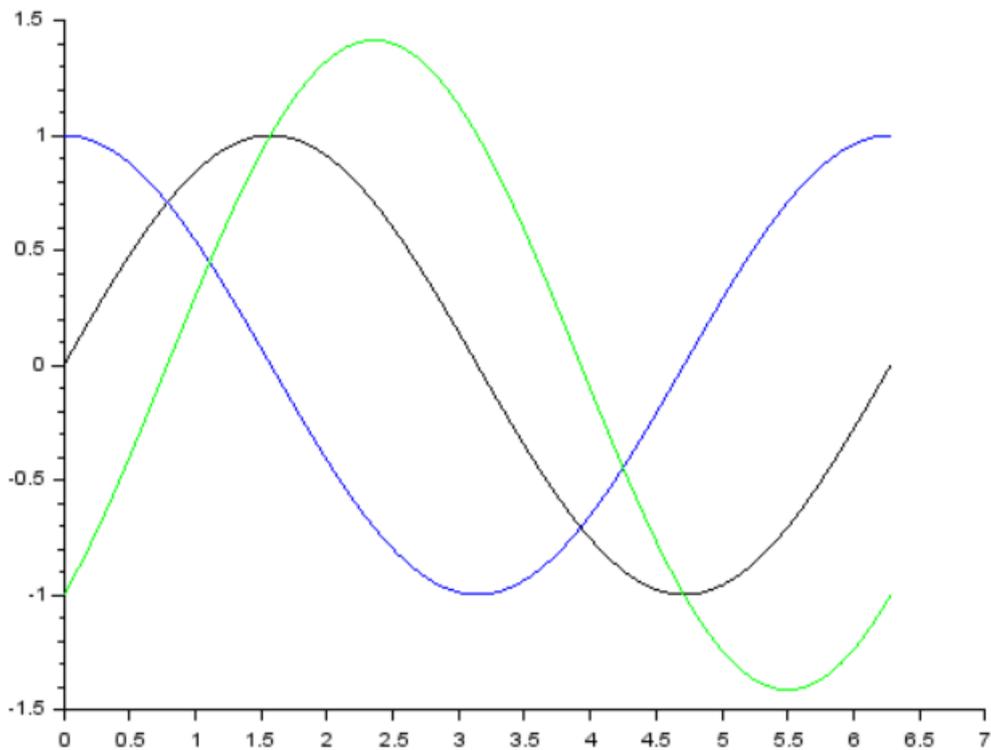
PRACTICAL-15

Aim: To study the application of 'linspace' command.

Software required: Scilab version 6.0.2

As we change the arguments of linspace, we observe following changes:

```
x=linspace(0,2*%pi,10);
x=x';
y=sin(x);
z=cos(x);
plot2d(x,[y,z,y-z]);
```
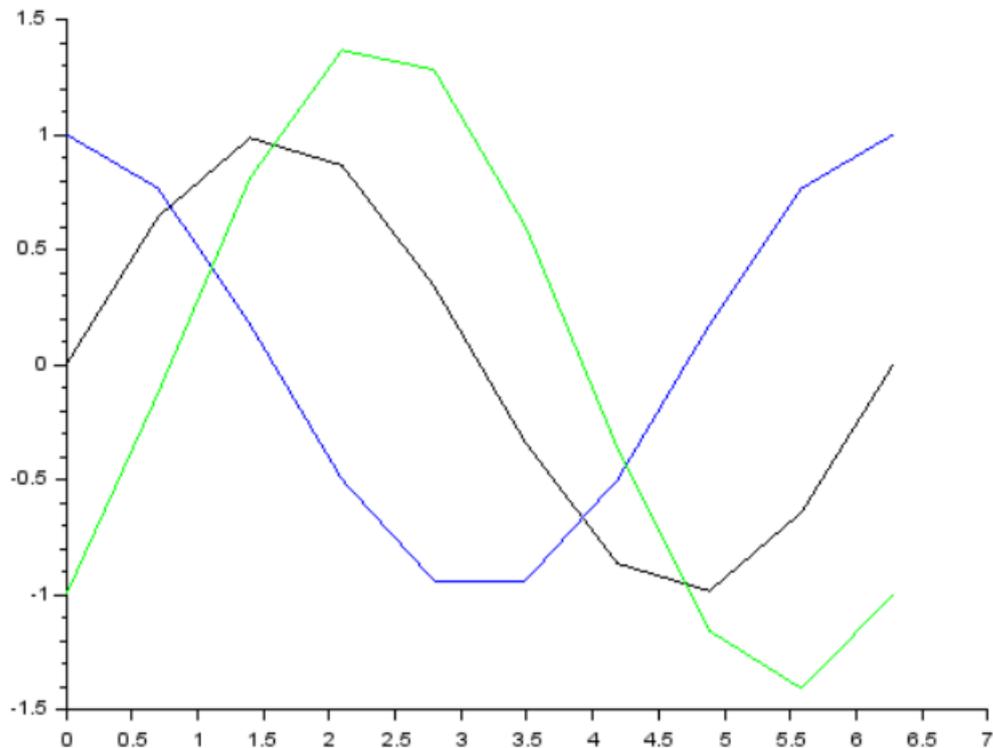
Result:

```
x=linspace(0,2*%pi,20);
x=x';
y=sin(x);
z=cos(x);
plot2d(x,[y,z,y-z]);
```
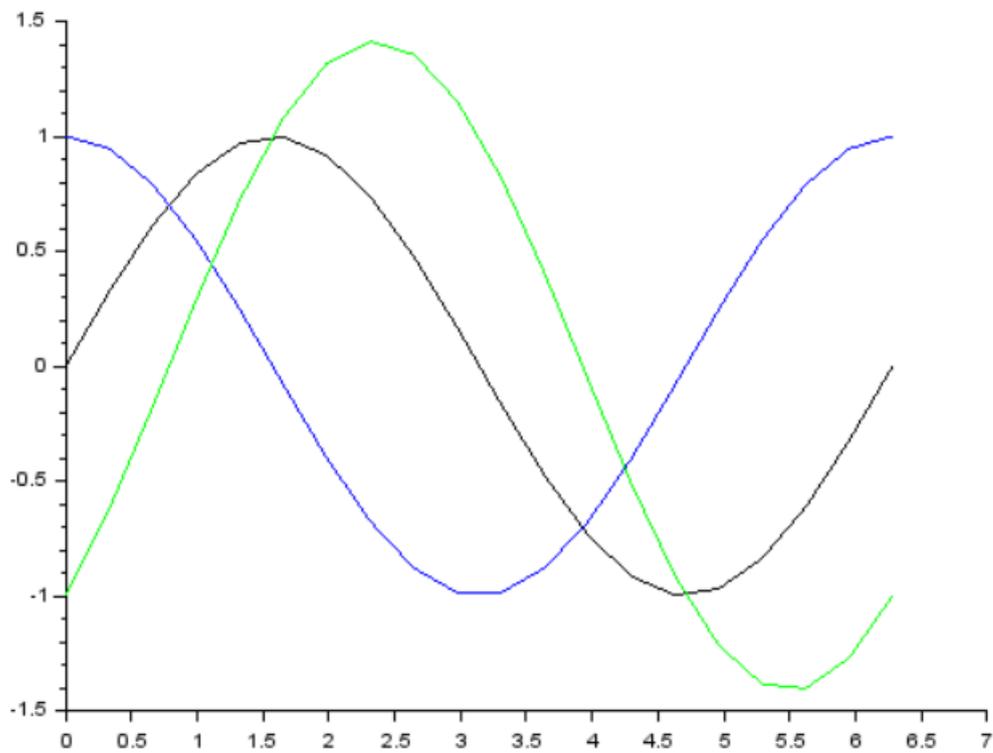
Result:

PRACTICAL-16

Aim: Study different types of graphs.

Software required: Scilab version 6.0.2

a) plot()

```
x=linspace(0,2*%pi,100);
x=x';
y=sin(x);
z=cos(x);
plot(x,[y,z,y-z]);
```

Result:
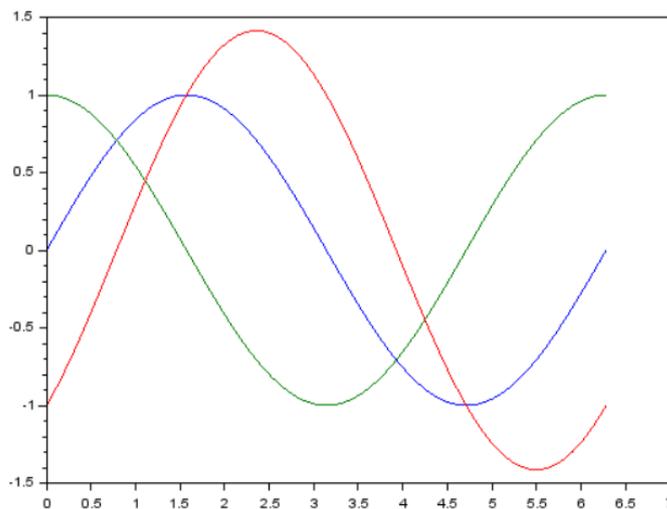
b) plo2d()

```
x=linspace(0,2*%pi,100);
x=x';
y=sin(x);
z=cos(x);
plot2d(x,[y,z,y-z]);
```

Result:

c) plot2d2()

```
x=linspace(0,2*%pi,100);
x=x';
y=sin(x);
z=cos(x);
plot2d2(x,[y,z,y-z]);
```

Result:

d) plot2d3()

```
x=linspace(0,2*%pi,100);
x=x';
y=sin(x);
z=cos(x);
plot2d3(x,[y,z,y-z]);
```

Result:

PRACTICAL-18

Aim: To plot graph of two functions.
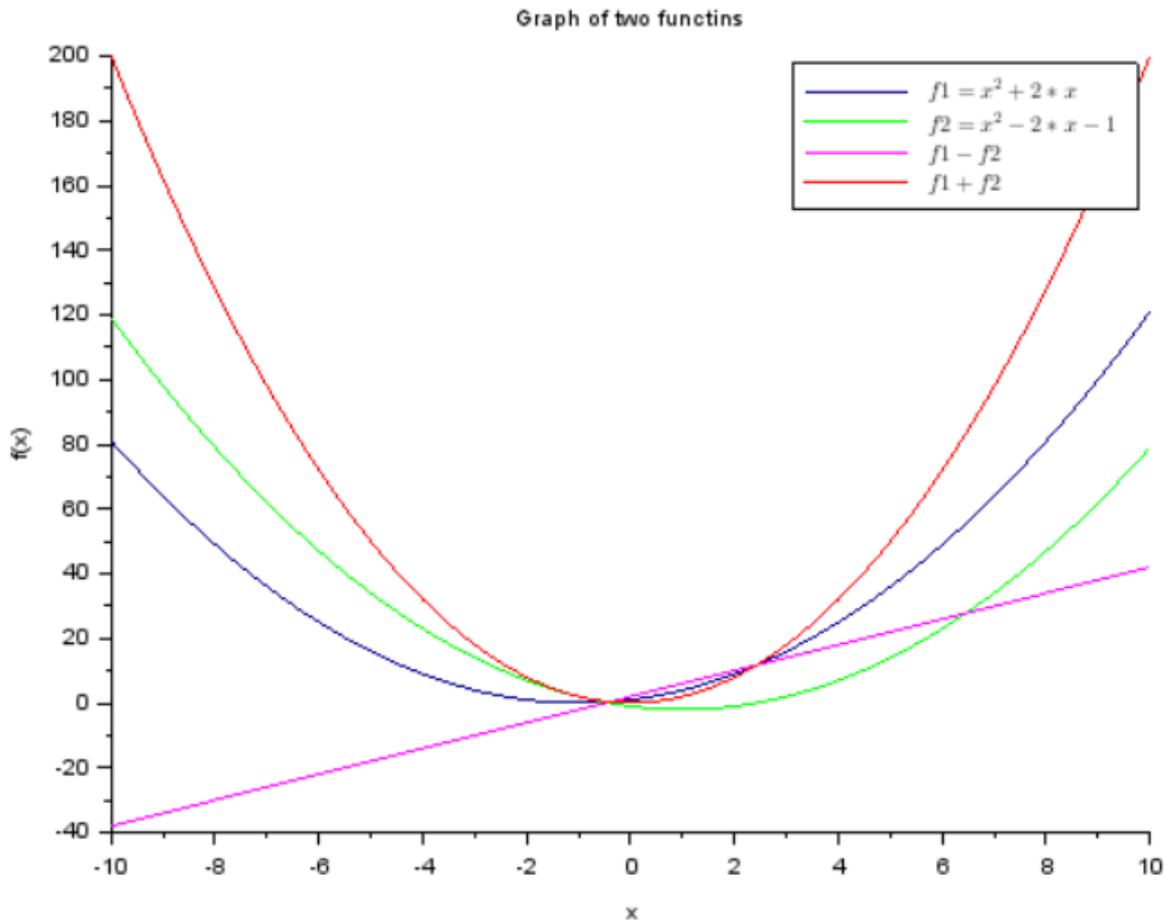
Software required: Scilab version 6.0.2

f1=x^2+2*x+1

f2=x^2-2*x-1

```
//using·scilab·code·plot·graph·of·two·functions
//f1=x^2+2x+1
//f2=x^2-2x-1
x=[-10:0.1:10];
f1=x^2+2*x+1;
f2=x^2-2*x-1;
plot2d(x,f1,9);
plot2d(x,f2,3);
plot2d(x,f1-f2,6);
plot2d(x,f1+f2,5);
xlabel("x");
ylabel("f(x)");
title("Graph·of·two·functins");
hl=legend(['$f1=x^2+2*x$','$f2=x^2-2*x-1$','$f1-f2$','$f1+f2$']);
```

Result:

PRACTICAL-19

Aim: To generate continuous unit step signal, unit step discrete signal, exponential signal, ramp signal.

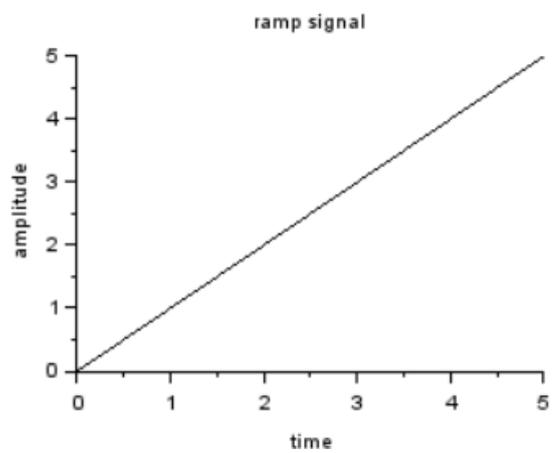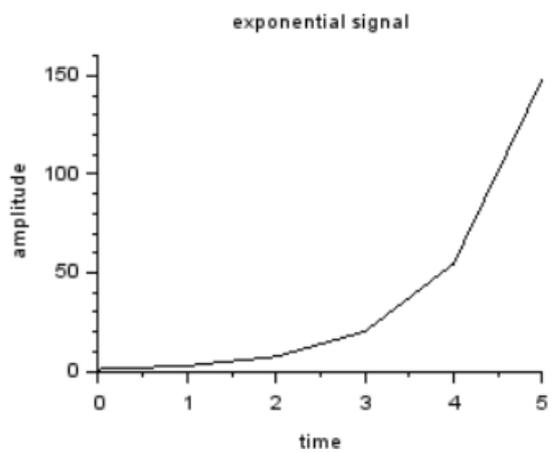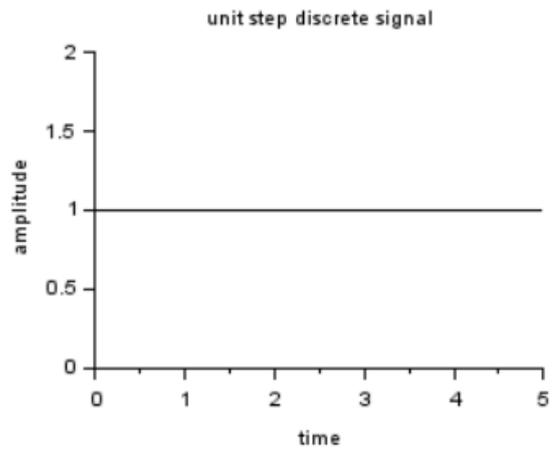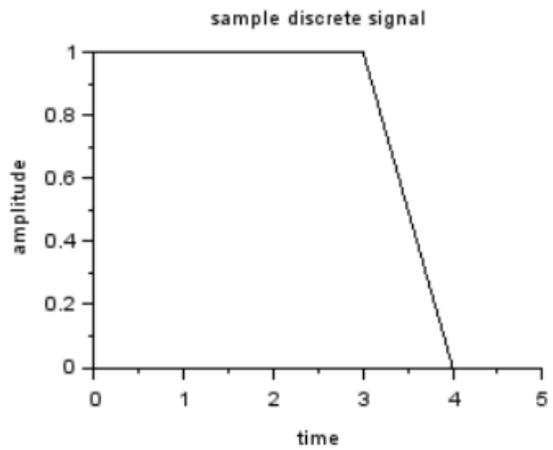Software required: Scilab version 6.0.2

a) plot2d

```
n1=(0:5);
y1=[ones(1,4),zeros(1,2)];
y2=[ones(1,6)];
y3=exp(n1);
y4=(n1);
subplot(2,2,1);
plot2d(n1,y1);
xlabel("time");
ylabel("amplitude");
title("sample discrete signal");
subplot(2,2,2);
plot2d(n1,y2);
xlabel("time");
ylabel("amplitude");
title("unit step discrete signal");
subplot(2,2,3);
plot2d(n1,y3);
xlabel("time");
ylabel("amplitude");
title("exponential signal");
subplot(2,2,4);
plot2d(n1,y4);
xlabel("time");
ylabel("amplitude");
title("ramp signal");
```

Result:

b) plot2d2

```
n1=(0:5);
y1=[ones(1,4),zeros(1,2)];
y2=[ones(1,6)];
y3=exp(n1);
y4=(n1);
subplot(2,2,1);
plot2d2(n1,y1);
xlabel("time");
ylabel("amplitude");
title("sample discrete signal");
subplot(2,2,2);
plot2d2(n1,y2);
xlabel("time");
ylabel("amplitude");
title("unit step discrete signal");
subplot(2,2,3);
plot2d2(n1,y3);
xlabel("time");
ylabel("amplitude");
title("exponential signal");
subplot(2,2,4);
plot2d2(n1,y4);
xlabel("time");
ylabel("amplitude");
title("ramp signal");
```
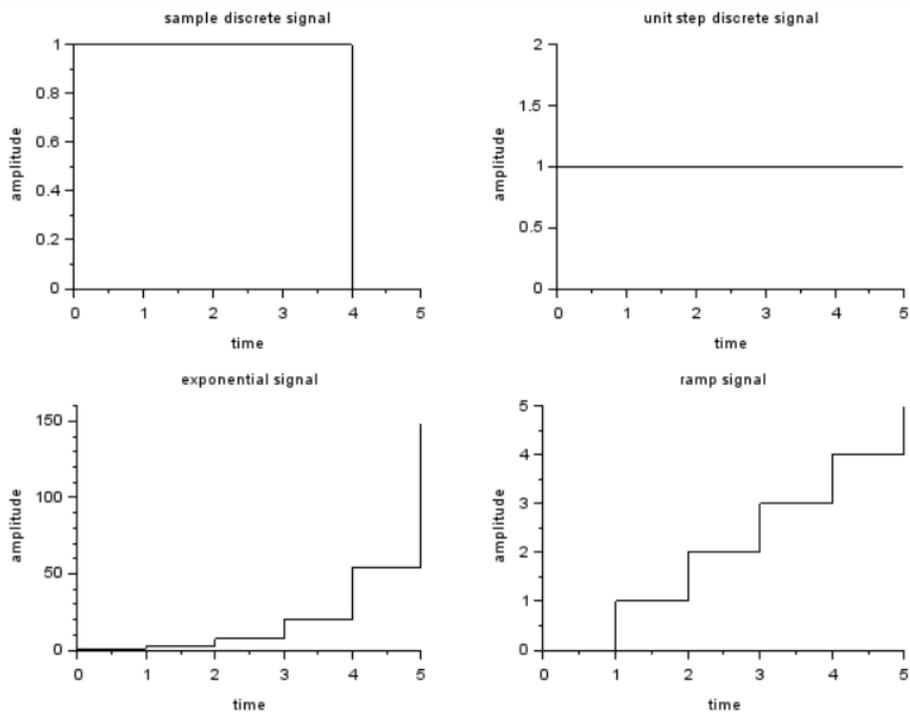
Result:

c) plot2d3

```
n1=(0:5);
y1=[ones(1,4),zeros(1,2)];
y2=[ones(1,6)];
y3=exp(n1);
y4=(n1);
subplot(2,2,2);
plot2d3(n1,y1);
subplot(2,2,1);
plot2d3(n1,y1);
xlabel("time");
ylabel("amplitude");
title("sample discrete signal");
subplot(2,2,2);
plot2d3(n1,y2);
xlabel("time");
ylabel("amplitude");
title("unit step discrete signal");
subplot(2,2,3);
plot2d3(n1,y3);
xlabel("time");
ylabel("amplitude");
title("exponential signal");
subplot(2,2,4);
plot2d3(n1,y4);
xlabel("time");
ylabel("amplitude");
title("ramp signal");
```
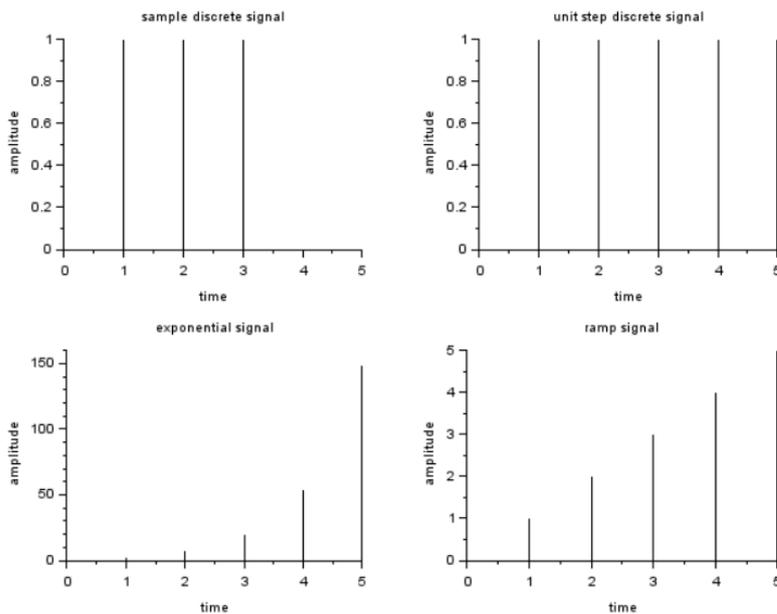
Result:

d) plot2d4

```
n1=(0:5);
y1=[ones(1,4),zeros(1,2)];
y2=[ones(1,6)];
y3=exp(n1);
y4=(n1);
subplot(2,2,1);
plot2d4(n1,y1);
xlabel("time");
ylabel("amplitude");
title("sample discrete signal");
subplot(2,2,2);
plot2d4(n1,y2);
xlabel("time");
ylabel("amplitude");
title("unit step discrete signal");
subplot(2,2,3);
plot2d4(n1,y3);
xlabel("time");
ylabel("amplitude");
title("exponential signal");
subplot(2,2,4);
plot2d4(n1,y4);
xlabel("time");
ylabel("amplitude");
title("ramp signal");
```
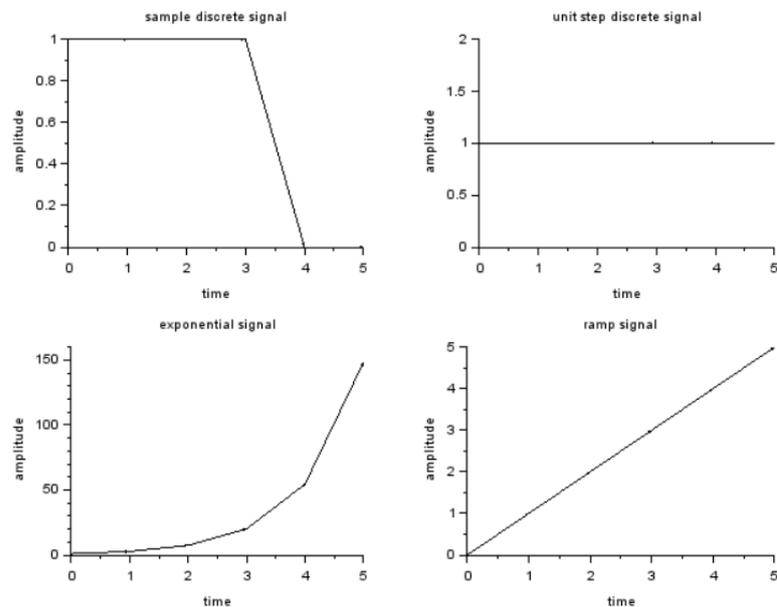
Result:

PRACTICAL-20

Aim: To solve ordinary differential equations using Scilab.

Software require: Scilab version 6.0.2

a) Example 1 – $\dfrac{dy}{dx} = -2x - y$

```
1  //ordinary differential equation
2  //dy/dx=-2x-y
3  //initial condition y(0)=-1, y(0.4)=?
4  //answer=0.81096
5  function dx=f(x,y);
6      dx=-2*x-y;
7  endfunction
8  y0=-1;
9  x0=0;
10 t=0.4;
11 sol=ode(y0,x0,t,f);
12 disp(sol,"answer");
```

Output:

```
--> exec('E:\Scilab Practicals\Edit window\differential equation.sce', -1)

 answer

  -0.8109602
```

b) Example 2 - $\dfrac{dy}{dx} = x + y$

```
1  //differencial equation
2  //dy/dx=x+y
3  //y(1)=2
4  //find y(1.2)
5  //answer=-0.4026
6  function dx=f(x,y);
7      dx=x+y;
8  endfunction
9  y1=2;
10 x1=1;
11 t=1.2;
12 sol=ode(x1,y1,t,f);
13 disp(sol,"answer");
```

Output:

```
--> exec('E:\Scilab Practicals\Edit window\differential equation2.sce', -1)

 answer

  -0.4026841
```

c) Example 3 - $x \times y$

```
1  //differencial equation
2  //dy/dx=x*y
3  //y(1)=2
4  //find y(1.2)
5  //answer=0.278
6  function dx=f(x,y);
7      dx=x*y;
8  endfunction
9  yl=2;
10 xl=1;
11 t=1.2;
12 sol=ode(xl,yl,t,f);
13 disp(sol,"answer");
```

Output:

```
--> exec('E:\Scilab Practicals\Edit window\differential equation2.sce', -1)

 answer

   0.2780373
```