

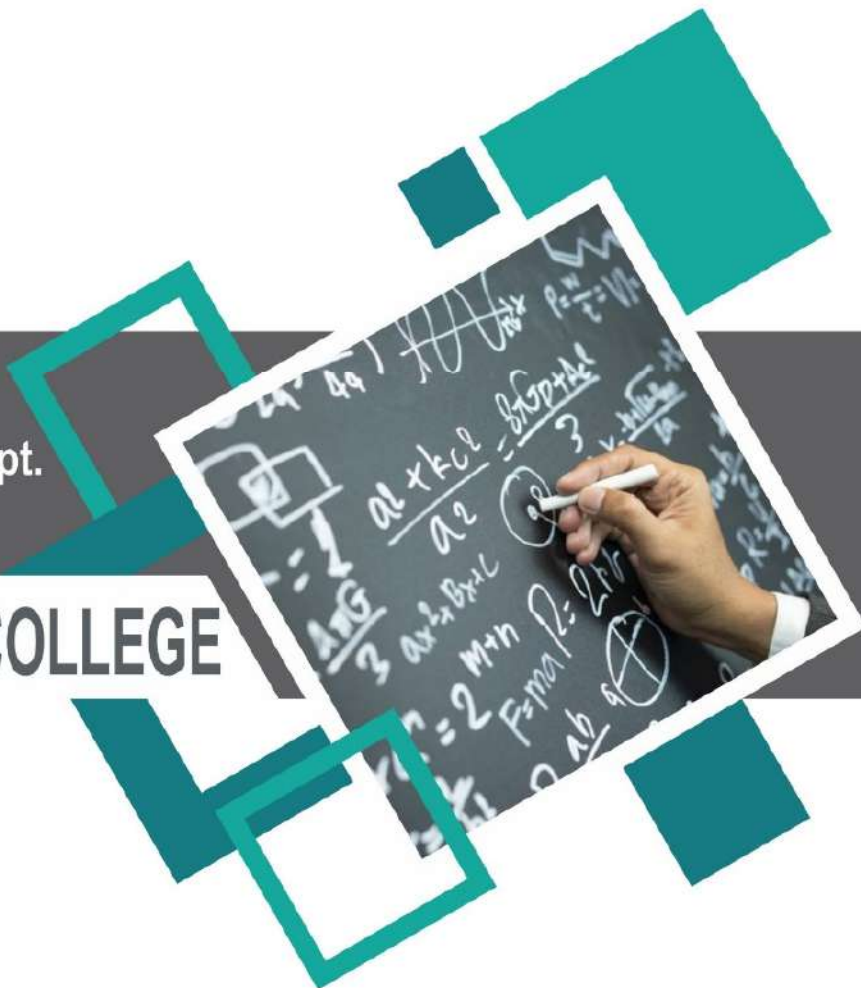
M.Sc. MATHEMATICS LAB MANUAL

1st Semester



Prepared By
Pure & Applied Science Dept.
Mathematics

MIDNAPORE CITY COLLEGE



INSTRUCTIONS TO STUDENTS

- Before entering the lab, the student should carry the following things (MANDATORY)
 1. Identity card issued by the college.
 2. Class notes
 3. Lab observation book
 4. Lab Manual
 5. Lab Record
- Student must sign in and sign out in the register provided when attending the lab session without fail.
- Come to the laboratory in time. Students, who are late more than 10 min., will not be allowed to attend the lab.
- Students need to maintain 80% attendance in lab if not a strict action will be taken.
- All students must follow a Dress Code while in the laboratory.
- Foods, drinks are NOT allowed.
- All bags must be left at the indicated place.
- Refer to the lab staff if you need any help in using the lab.
- Respect the laboratory and its other users.
- Workspace must be kept clean and tidy after experiment is completed.
- Read the Manual carefully before coming to the laboratory and be sure about what you are supposed to do.
- Do the experiments as per the instructions given in the manual.
- Copy all the programs to observation which are taught in class before attending the lab session.
- Students are not supposed to use floppy disks, pen drives without permission of lab- in charge.
- Lab records need to be submitted on or before the date of submission.

OVERVIEW

MATLAB (**MAT**rix **LAB**oratory) is a fourth-generation high-level programming language and interactive environment for numerical computation, visualization and programming.

MATLAB is developed by MathWorks.

It allows matrix manipulations; plotting of functions and data; implementation of algorithms; creation of user interfaces; interfacing with programs written in other languages, including C, C++, Java, and FORTRAN; analyze data; develop algorithms; and create models and applications.

It has numerous built-in commands and math functions that help you in mathematical calculations, generating plots, and performing numerical methods.

MATLAB'S POWER OF COMPUTATIONAL MATHEMATICS

MATLAB is used in every facet of computational mathematics. Following are some commonly used mathematical calculations where it is used most commonly:

- Dealing with Matrices and Arrays
- 2-D and 3-D Plotting and graphics
- Linear Algebra
- Algebraic Equations
- Non-linear Functions
- Statistics
- Data Analysis
- Calculus and Differential Equations
- Numerical Calculations
- Integration
- Transforms
- Curve Fitting
- Various other special functions

FEATURES OF MATLAB

Following are the basic features of MATLAB:

- It is a high-level language for numerical computation, visualization and application development.
- It also provides an interactive environment for iterative exploration, design and problem solving.
- It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.
- It provides built-in graphics for visualizing data and tools for creating custom plots.
- MATLAB's programming interface gives development tools for improving code quality, maintainability, and maximizing performance.
- It provides tools for building applications with custom graphical interfaces.
- It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

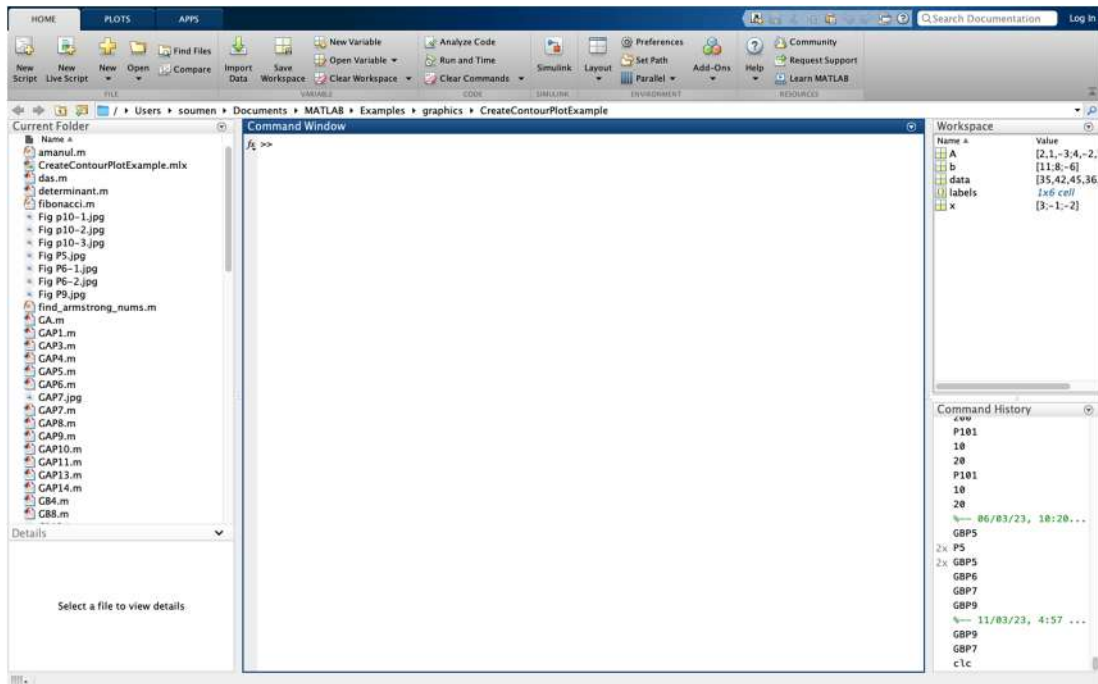
USES OF MATLAB

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including:

- Signal processing and Communications
- Image and video Processing
- Control systems
- Test and measurement
- Computational finance
- Computational biology

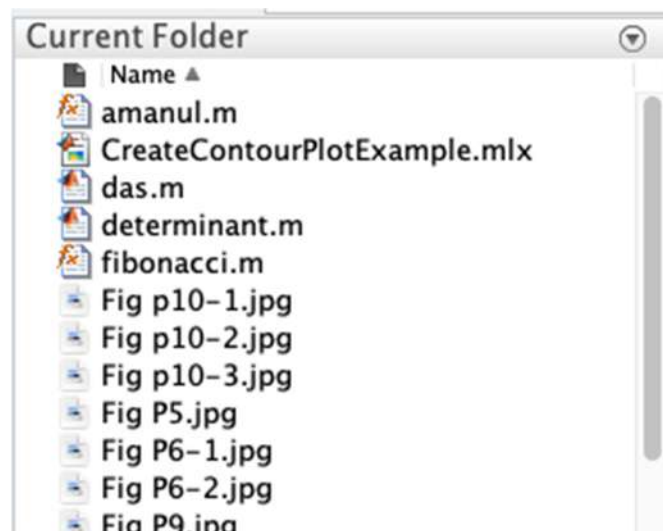
UNDERSTANDING THE MATLAB ENVIRONMENT

MATLAB development IDE can be launched from the icon created on the desktop. The main working window in MATLAB is called the desktop. When MATLAB is started, the desktop appears in its default layout:

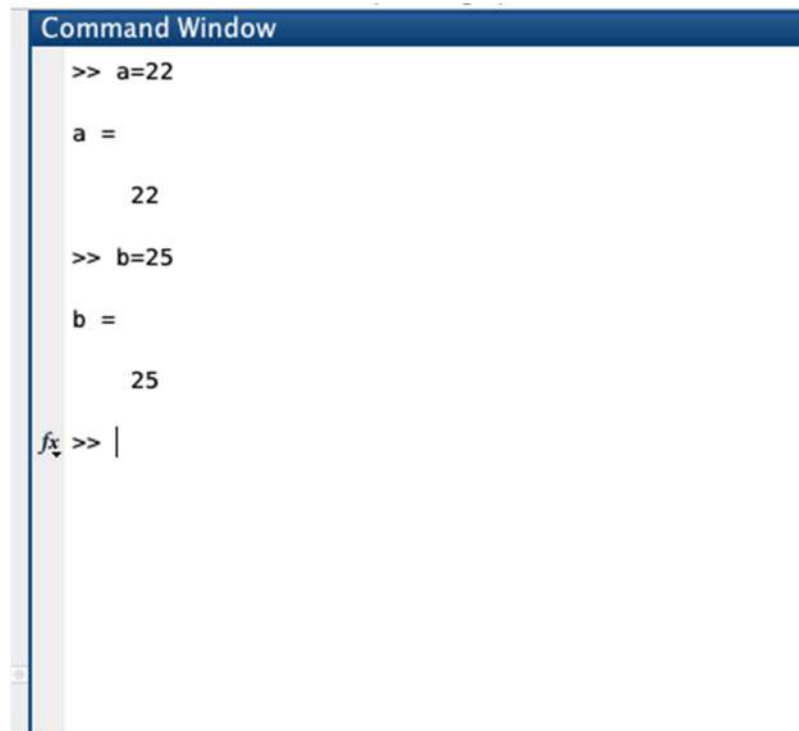


The desktop has the following panels:

Current Folder - This panel allows you to access the project folders and files.

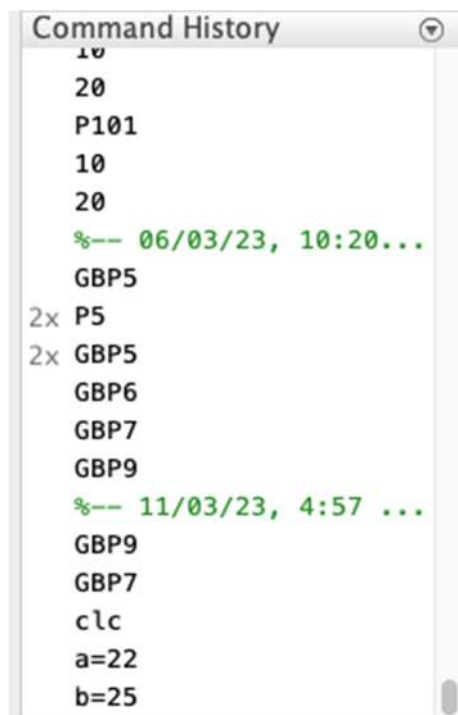


Command Window - This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



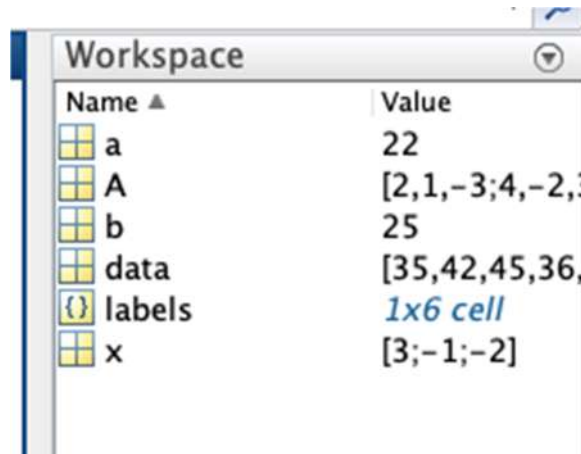
```
>> a=22
a =
    22
>> b=25
b =
    25
fx >> |
```

Command History - This panel shows or rerun commands that are entered at the command line.



```
10
20
P101
10
20
%-- 06/03/23, 10:20...
GBP5
2x P5
2x GBP5
GBP6
GBP7
GBP9
%-- 11/03/23, 4:57 ...
GBP9
GBP7
clc
a=22
b=25
```

Workspace - The workspace shows all the variables created and/or imported from files.



Name ▲	Value
a	22
A	[2,1,-3;4,-2,3]
b	25
data	[35,42,45,36,39,42]
labels	1x6 cell
x	[3;-1;-2]

INDEX

Sl. No.	Program Description	Page No.
1.	Write a script in MATLAB to find the sum of natural numbers between 1 and 100	01
2.	Write a script in MATLAB to find the sum of natural numbers between two specified numbers	02
3.	Write a script in MATLAB to check whether a given number is prime or not	03
4.	Write a script in MATLAB to check whether a given number is divisible or not.	04
5.	Write a script in MATLAB to check whether a given number is palindrome or not.	05
6.	Write a script in MATLAB to generate Fibonacci sequence.	06
7.	Write a script in MATLAB to find the pascal triangle	07
8.	Write a script in MATLAB to find the prime factors of a positive integer	08
9.	Write a script in MATLAB to find the roots of a quadratic equation	09
10.	Write a script in MATLAB to find the addition of two matrices	10
11.	Write a script in MATLAB to find the mean, median, variance and standard deviation for a set of discrete data	11
12.	Write a script in MATLAB to draw $\sin t$ and $\cos t$ curves in the same figure with different line specification	12
13.	Write a script in MATLAB to display three-dimensional surface plot	13
14.	Write a script in MATLAB to draw a pie chart for a set of data number	14
15.	Write a script in MATLAB to draw histogram for a set of data	15
16.	Write a script in MATLAB to plot $5x^2 + 15$ in a given interval	16

17.	Write a script in MATLAB to plot $y = x $ and $y = \log x$ in a same figure.	17
18.	Write a script file to compute the sum of the first n terms in the series $5k^2 - k, k = 1, 2, 3, \dots, n$.	18
19.	Write a script program in MATLAB to display all primes numbers between two specified numbers.	19
20.	Write a script program to find all palindrome numbers between two specified numbers.	20
21.	Write a script program to find the value of $\int_a^b f(x) dx$ by Trapezoidal Rule.	21
22.	Write a script in MATLAB to represent the graphs of the functions $x(t) = e^{-0.2t} \cos 2t$ and $y(t) = e^{-0.2t} \sin 2t$. the text of each equation is properly positioned within the graphs.	22
23.	Write a script in MATLAB to draw the surface and contour of the equation $z(x, y) = xe^{-x^2-y^2}$ in the range $-2 \leq x \leq 2$ and $-3 \leq y \leq 3$ in steps of 0.1.	23-24
24.	Write a script in MATLAB to find the solution of the following linear equations $\begin{aligned} 2x + y - 3z &= 11 \\ 4x - 2y + 3z &= 8 \\ -2x + 2y - z &= -6 \end{aligned}$	25
25.	Write a program in MATLAB to convert among decimal, binary, octal, Hexadecimal based on your inputs.	26
26.	Write a script in MATLAB to draw the Pie diagram of a M.Sc. 1st Semester student of the following marks 35, 42, 45, 36, 38, 15	27
27.	Write a script program to create a mesh, surface and contour plots of the function $z = e^{x+iy}$ in the interval $-1 < x < 1$ and $-2\pi < y < 2\pi$. In each case plot the real part of z versus x and y .	28-30
28.	Write a script program to find either minimum or maximum or sum according to your response of the function $y = x \sin x$ in the range $-\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$ with spacing 0.2 using switch statement.	31

29. Write a script in MATLAB to draw the surface and contour of the equation $z(x, y) = xe^{-x^2-y^2}$ in the range $-3 \leq x \leq 3$ and $-3 \leq y \leq 3$. 32-33
30. Write a script in MATLAB to draw the following function in the interval $[-1, 4]$.
- $$f(x) = \begin{cases} x^2 + 1, & -1 \leq x < 0 \\ 0, & x = 0 \\ x^3 + 2x + 5, & x > 0 \end{cases}$$
31. Write a script in MATLAB to find the solution of the following linear equations
- $$\begin{aligned} 3x + 5y - 6z &= 6 \\ 8x - y + 2z &= 1 \\ 5x - 6y - 4z &= -5 \end{aligned}$$
- Using rref, pinv, and left division methods. 35
32. Write a script in MATLAB to create two lists of numbers and perform the following arithmetic operations on it (i) addition (ii) subtraction (iii) element-wise multiplication (iv) element-wise division. 36
33. Write a function program in MATLAB to find all Armstrong numbers between two specified numbers. 37
34. Write a script program in MATLAB to solve the following ODE and find the value of $f(1)$ using Runge-Kutta method $\frac{dy}{dx} = y - x^2 + 1$, $y(0)=0.5$. 38
35. Write a MATLAB function program to find a real root of the equation $x^2 - \sin 2x - 1 = 0$ by Newton-Raphson's method. 39
36. Write a script in MATLAB to find the mean, median, variance and standard deviation for a set of discrete data. 40
37. Write a script in MATLAB to find an invertible matrix P and a diagonal D such that $PDP^{-1} = A$, then compare A^5 and PA^5P^{-1} . 41
38. Write a user defined function in MATLAB to generate Fibonacci sequence. Use this function; write a script to find the Fibonacci numbers between two specified numbers. 42
39. Write a script in MATLAB to represent the graphs in the same window of the functions $\sin x$, $\sin 2x$ and $\sin 3x$ in the range $(0, 2\pi)$ with mentions different line specification, title, axes and axes limits. 43-44

40. Write a user defined function in MATLAB to determine the roots of a quadratic equation. Use this function; write a script find the roots of the equation $x^2 + 5x + 6 = 0$. 45
41. Write a user define function to find the value of $\int_a^b f(x)dx$ by Simpson 1/3's rule. Use this function; write a script program to find the value of the following integration $\int_0^1 x^2 + x dx$. 46
42. Write a script program in MATLAB to solve the following ODE and find the values of $f(0.1)$ and $f(0.2)$ using Euler method $\frac{dy}{dx} = x^2 + y^2$, $y(0) = 1$. 47
43. Write a user defined function in MATLAB to calculate the sum of a list of numbers. Using it, find the sum of all-natural numbers between two specified numbers. 48
44. For a diagonalizable A, write a function program that returns true if A is positive definite and false otherwise. Also, write a script program to illustrate it. 49
45. For a given square matrix of order 5, write a script program to carry out of the following:
 (i) sort each column and store the result in an array B.
 (ii) sort each row and store the result in an array C.
 (iii) add each column and store the result in an array D.
 add each row and store the result in an array E. 50-51

Programming 1: Write a script in MATLAB to find the sum of natural numbers between 1 and 100.

Code:

```
Sum=0;
for i=1:100
    Sum=Sum+i;
end
fprintf('The sum of natural numbers between 1 and 100 is %d\n', Sum);
```

Input/output:

The sum of natural numbers between 1 and 100 is 5050.

Programming 2: Write a script in MATLAB to find the sum of natural numbers between two specified numbers.

Code:

```
m=input('Enter the lower value\n');
n=input('Enter the upper value\n');
Sum=0;
for i=m:n
    Sum=Sum+i;
end
fprintf('The sum of the natural numbers between %d and %d is
%d\n',m,n,Sum);
```

Input/output:

```
Enter the lower value
1
Enter the upper value
50
The sum of the natural numbers between 1 and 50 is 1275.
```

Programming 3: Write a script in MATLAB to check whether a given number is prime or not.

Code:

```
n=input('Enter the number\n');  
if isprime(n)==1  
    fprintf('The number %d is prime\n',n);  
else  
    fprintf('The number %d is not prime\n',n);  
end
```

Input/output:

```
Enter the number  
25  
The number 25 is not prime.
```

Programming 4: Write a script in MATLAB to check whether a given number is divisible or not.

Code:

```
m=input('Enter the dividend\n');
n=input('Enter the divisor\n');
if rem(m,n)==0
    fprintf('The number %d is divisible by %d \n',m,n);
else
    fprintf('The number %d is not divisible by %d \n',m,n);
end
```

Input/output:

```
Enter the dividend
25
Enter the divisor
5
The number 25 is divisible by 5.
```

Programming 5: Write a script in MATLAB to check whether a given number is palindrome or not.

Code:

```
n=input('Enter the number \n');
P=n;
rev=0;
while(n>0)
    dig=rem(n,10);
    rev=(rev*10)+dig;
    n=floor(n/10);
end

if rev==P
    fprintf('The number %d is palindrome\n',P);
else
    fprintf('The number %d is not palindrome\n',P);
end
```

Input/output:

```
Enter the number
111
The number 111 is palindrome.
```


Programming 6: Write a script in MATLAB to generate Fibonacci sequence**Code:**

```
n=input('How many numbers are required\n');
if(n==1)
    fprintf('The first Fibonacci number is 0\n');
else
    f1=0;
    f2=1;
    fprintf('The first %d Fibonacci numbers are
\t%d\t%d',n,f1,f2);
    for i=1:n-2
        f=f1+f2;
        f1=f2;
        f2=f;
        fprintf('\t%d',f);
    end
end
fprintf('\n');
```

Input/output:

How many numbers are required

10

The first 10 Fibonacci numbers are 0 1 1 2 3 5
 8 13 21 34

Programming 7: Write a script in MATLAB to find the Pascal triangle**Code:**

```
row=input('Enter the number of rows\n');
b=1
fprintf('The Pascal triangle with %d rows is \n',row);
for n=0:(row-1)
    for s=(25-3*n):-1:1
        fprintf(' ');
    end
    for r=0:n
        if n==0||r==0
            b=1;
        else
            b=b*(n-r+1);
        end
        fprintf('% 6d',b);
    end
    fprintf('\n');
end
```

Input/output:

Enter the number of rows

5

b =

1

The Pascal triangle with 5 rows is

```

          1
         1 1
        1 2 1
       1 3 3 1
      1 4 6 4 1
```

Programming 8: Write a script in MATLAB to find the prime factors of a positive integer**Code:**

```
n=input('Enter a positive integer:\n');
fprintf('The prime factors of %d are:\n',n);
for j=2:n
    if rem(n,j)==0
        if isprime(j)==1
            fprintf('\t %d',j);
        end
    end
end
fprintf('\n');
```

Input/output:

```
Enter a positive integer:
45
The prime factors of 45 are:
        3        5
```

Programming 9: Write a script in MATLAB to find the roots of a quadratic equation**Code:**

```
fprintf('Enter the values of the coefficient a, b, c of x^2, x^1
and x^0 respectively:\n');
a=input('');
b=input('');
c=input('');
if a==0
    fprintf('Not a quadratic equation');
else
    d=(b*b-4*a*c);
    p=-b/(2*a);
    q=sqrt(abs(d))/(2*a);
    if d >=0
        fprintf('The real roots are %f and %f \n',p+q,p-q);
    else
        fprintf('The complex roots are %f+%fi and %f-%fi
\n',p,q,p,q);
    end
end
end
```

Input/output:

Enter the values of the coefficient a, b, c of x², x¹ and x⁰ respectively:

2

6

1

The real roots are -0.177124 and -2.822876

Programming 10: Write a script in MATLAB to find the addition of two matrices**Code:**

```
x=input('Enter the first matrix:\n');
y=input('Enter the second matrix:\n');
a=size(x);
b=size(y);
d=a-b;
if any(d)
    fprintf('Addition is not possible. Matrix order must be same');
else
    z=x+y;
    fprintf('The required sum of the matrices is:\n');
    disp(z);
end
```

Input/output:

Enter the first matrix:

[2 3; 1 2]

Enter the second matrix:

[4 5; 6 7]

The required sum of the matrices is:

6 8

7 9

Programming 11: Write a script in MATLAB to find the mean, median, variance and standard deviation for a set of discrete data.

Code:

```
n=input('Enter the size of the sample:\n');
fprintf('Enter all the sample values:\n');
sum=0;
sumsq=0;
for i=1:n
    x(i)=input(' ');
    sum=sum+x(i);
    sumsq=sumsq+(x(i)*x(i));
end
xbar=sum/n;
var=(sumsq/n)-(xbar*xbar);
fprintf('Mean=%f\n Variance=%f\n Standard deviation
=%f\n',xbar,var,sqrt(var));
for i=1:n-1
    for j=i:n
        if x(i)>x(j)
            t=x(i);
            x(i)=x(j);
        end
    end
end
if rem(n,2)==0
    m=(x(n/2)+x(n/2+1))/2;
else
    m=x((n+1)/2);
end
fprintf('Median=%f\n',m);
```

Input/output:

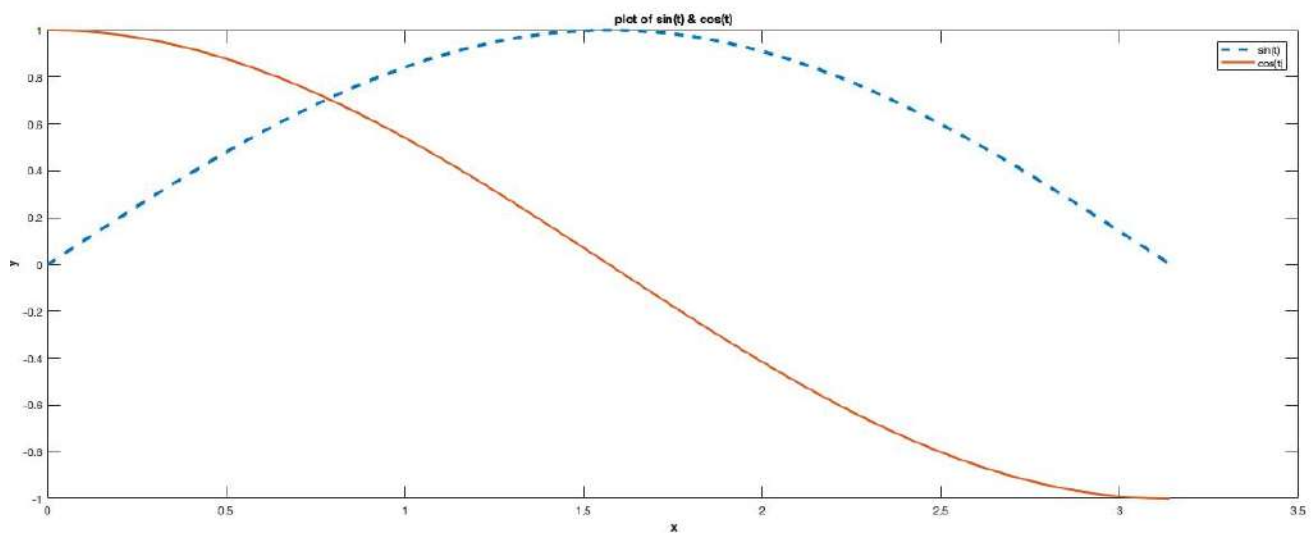
```
Enter the size of the sample:
3
Enter all the sample values:
1
2
3
Mean=2.000000
Variance=0.666667
Standard deviation =0.816497
Median=2.000000
```

Programming 12: Write a script in MATLAB to draw $\sin t$ and $\cos t$ curves in the same figure with different line specification

Code:

```
t=0:0.001:2*pi;  
y=sin(t);  
plot(t,y);  
hold on  
z=cos(t);  
plot(t,z);  
legend ('sin(t)', 'cos(t)');
```

Input/output:



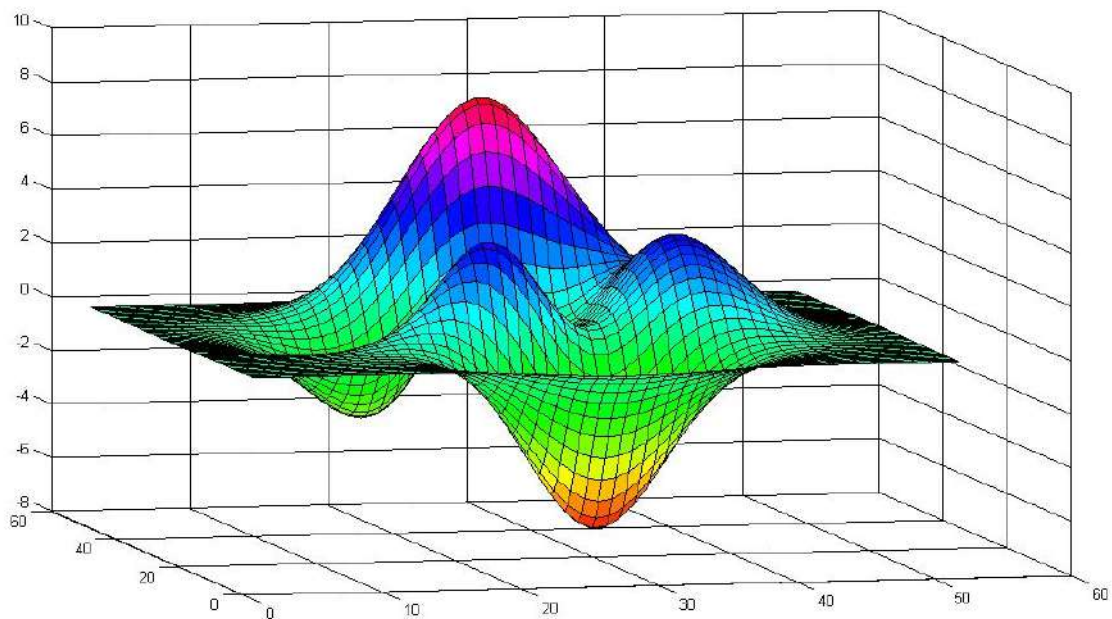
Programming 13: Write a script in MATLAB to display three-dimensional surface plot**Code:**

```
a=input('Enter the value of a:\n');  
z=peaks(a);  
surface(z);  
colormap(hsv);
```

Input/output:

Enter the value of a:

45



Programming 14: Write a script in MATLAB to draw a pie chart for a set of data number

Code:

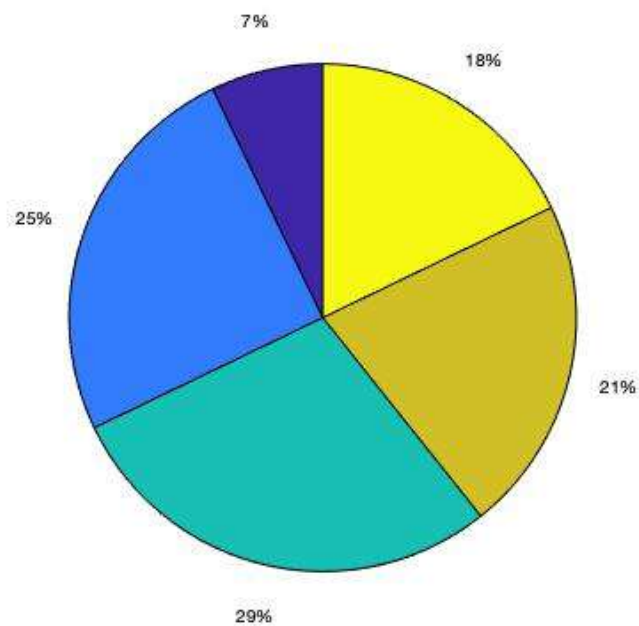
```
X=input('Enter the data:\n');  
fprintf('The required pie chart is:\n')  
pie(X);
```

Input/output:

Enter the data:

[2 7 8 6 5]

The required pie chart is:



Programming 15: Write a script in MATLAB to draw histogram for a set of data

Code:

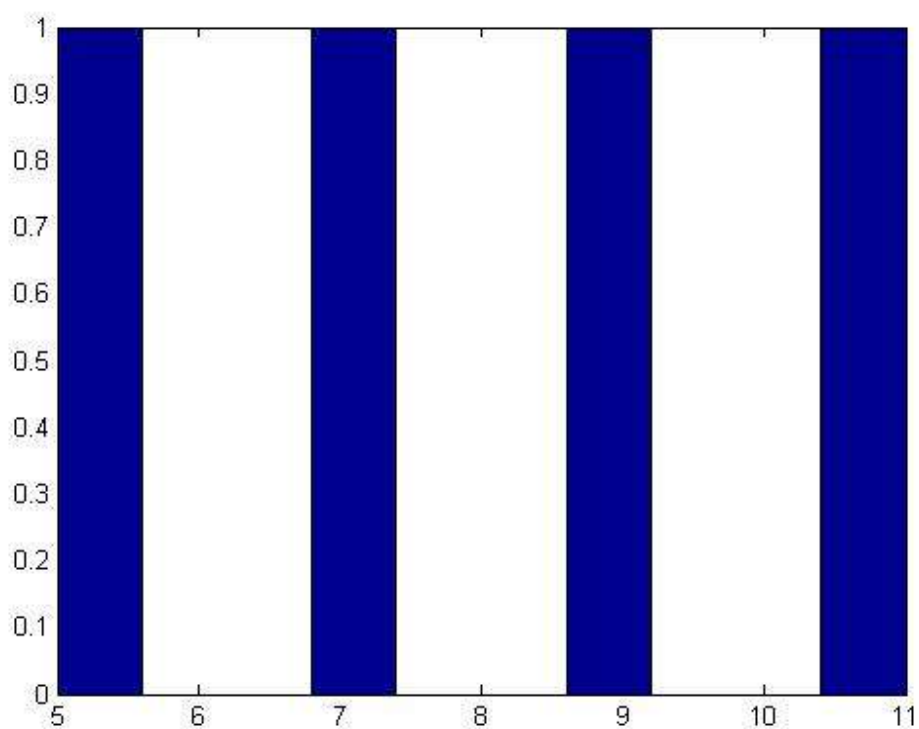
```
X=input('Enter the data:\n')
fprintf('The required histogram is:\n');
hist(X);
```

Input/output:

Enter the data:

[2 7 8 6 5]

The required histogram is:



Programming 16: Write a script in MATLAB to plot $5x^2 + 15$ in a given interval

Code:

```
a=input('Enter the lower limit:\n');
b=input('Enter the upper limit:\n');
x=a:0.001:b;
y=2*x.^2+16;
plot(x,y,'r');
title('Graph of y=2x^2+16');
xlabel('\bf X');
ylabel('\bf Y');
```

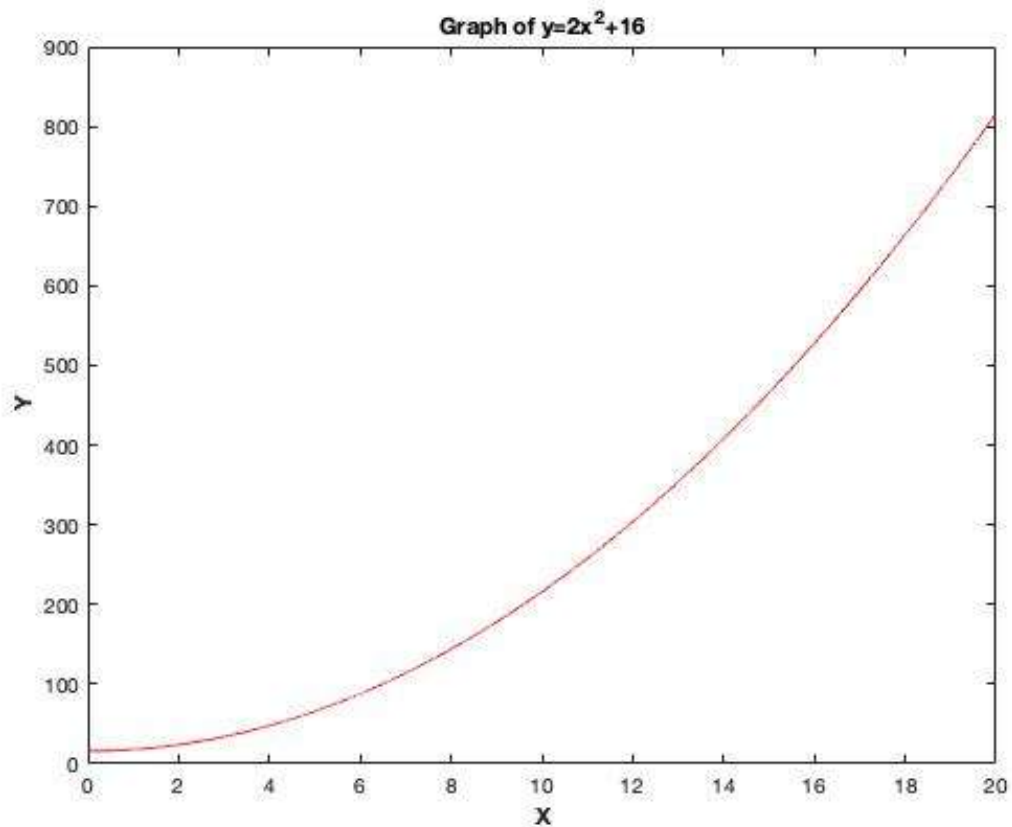
Input/output:

Enter the lower limit:

0

Enter the upper limit:

20



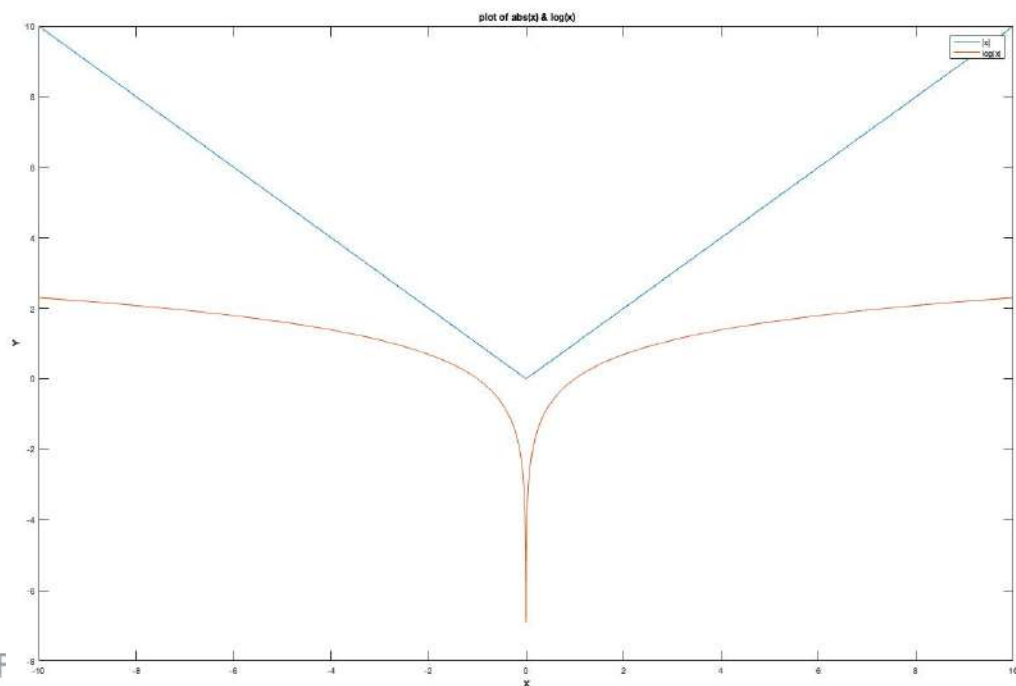
Programming 17: Write a script in MATLAB to plot $y = |x|$ and $y = \log x$ in a same figure

Code:

```
a=input('Enter the lower limit:\n');
b=input('Enter the upper limit:\n');
x=a:0.001:b;
y=abs(x);
plot(x,y);
hold on
z=log(x);
plot(x,z);
title('\bf plot of abs(x) & log(x)')
% adds title to the plot
xlabel('\bf X');
ylabel('\bf Y');
legend ('|x|', 'log(x)');
```

Input/output:

```
Enter the lower limit:
-10
Enter the upper limit:
10
```



Programming 18: Write a script file to compute the sum of the first n terms in the series $5k^2 - k, k = 1, 2, 3, \dots, n$.

Code:

```
% Get the value of n from the user
n = input('Enter the value of n: ');
% Initialize the running total to 0
total = 0;
% Loop over the values of k from 1 to n
for k = 1:n
    % Compute the value of the kth term in the series
    term = 5*k^2 - k;
    % Add the term to the running total
    total = total + term;
end
% Display the total
fprintf('The sum of the first %d terms in the series is %d\n',
n,total);
```

Input/output:

Enter the value of n: 10

The sum of the first 10 terms in the series is 1870.

Programming 19: Write a script program in MATLAB to display all primes numbers between two specified numbers.

Code:

```
clc;
a =input('Enter num1 value: ');
b =input('Enter num2 value: ');
n = a:b;
p = isprime(n);
n(p) %displays the primes.
```

Input/output:

Enter num1 value: 10

Enter num2 value: 20

ans =

11 13 17 19

Programming 20: Write a script program to find all palindrome numbers between two specified numbers.

Code:

```
% Prompt user to enter the two numbers
lower_bound = input('Enter the lower bound: ');
upper_bound = input('Enter the upper bound: ');

% Loop through all numbers between the bounds
for i = lower_bound:upper_bound
    % Convert the number to a string and reverse it
    num_str = num2str(i);
    num_str_reverse = flip(num_str);

    % Check if the reversed string is the same as the original
    string
    if strcmp(num_str, num_str_reverse)
        % Print the palindrome number
        fprintf('%d\n', i);
    end
end
```

Note: For each number, the program converts it to a string and reverses the string using the *flip* function. The program checks if the reversed string is the same as the original string using the *strcmp* function.

Input/output:

```
Enter the lower bound: 10
Enter the upper bound: 100
11
22
33
44
55
66
77
88
99
```

Programming 21: Write a script program in MATLAB to find the value of $\int_a^b f(x) dx$ by Trapezoidal Rule.

Code:

```
% Define the function to integrate
f = @(x) x.^2;

% Define the integration interval and number of trapezoids
a = 0;
b = 1;
n = 100;

% Calculate the width of each trapezoid
h = (b - a) / n;

% Initialize the sum of the areas of the trapezoids
area = 0;

% Loop through all the trapezoids and add their areas to the sum
for i = 1:n
    x1 = a + (i - 1) * h;
    x2 = a + i * h;
    area = area + (f(x1) + f(x2)) / 2 * h;
end

% Print the approximation of the definite integral
fprintf('The approximation of the definite integral of f(x) from
%f to %f is %f\n', a, b, area);
```

Input/output: The approximation of the definite integral of $f(x)$ from 0.000000 to 1.000000 is 0.333350.

Programming 22: Write a script in MATLAB to represent the graphs of the functions $x(t) = e^{-0.2t} \cos 2t$ and $y(t) = e^{-0.2t} \sin 2t$. The text of each equation is properly positioned within the graphs.

Code:

```
% Define the time vector
t = 0:0.01:50;

% Define the functions
x = exp(-0.2*t).*cos(2*t);
y = exp(-0.2*t).*sin(2*t);

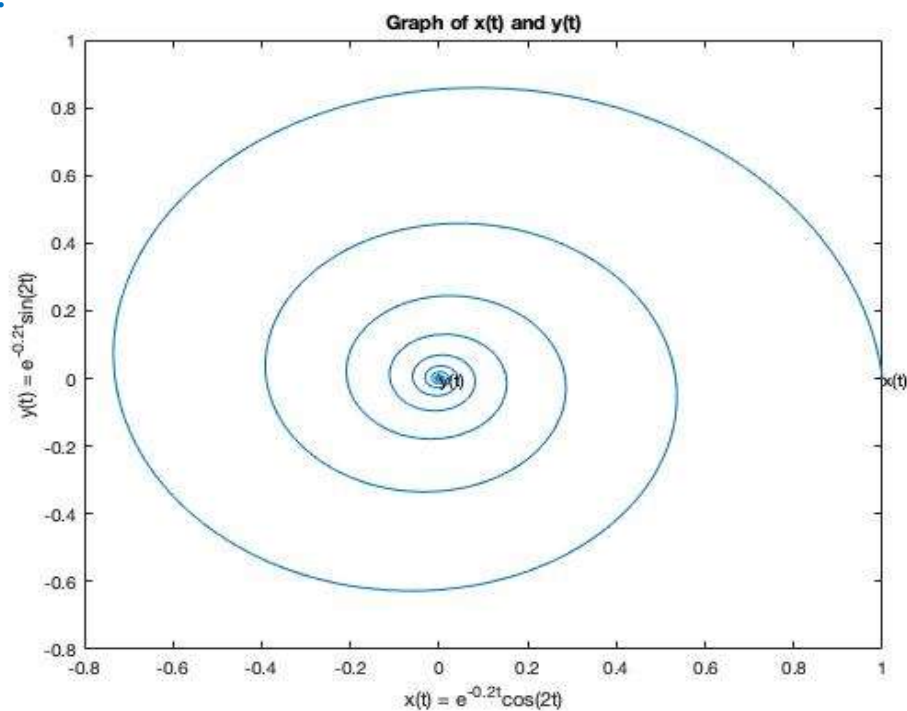
% Create the plot
plot(x, y);

% Label the axes and add a title
xlabel('x(t) = e^{-0.2t}cos(2t)');
ylabel('y(t) = e^{-0.2t}sin(2t)');
title('Graph of x(t) and y(t)');

% Add text labels for the equations
text(x(1), y(1), 'x(t)');
text(x(end), y(end), 'y(t)');
```

Note: Note that we use element-wise multiplication (.*) to ensure that the exponential function is applied to each element of t. Plot the functions using the **plot** function and add text annotations using the **text** function.

Input/output:



Programming 23: Write a script in MATLAB to draw the surface and contour of the equation $z(x,y) = xe^{-x^2-y^2}$ in the range $-2 \leq x \leq 2$ and $-3 \leq y \leq 3$ in steps of 0.1.

Code:

```
% Define the range and step size of x and y
x = -2:0.1:2;
y = -3:0.1:3;

% Create a meshgrid from x and y
[X,Y] = meshgrid(x,y);

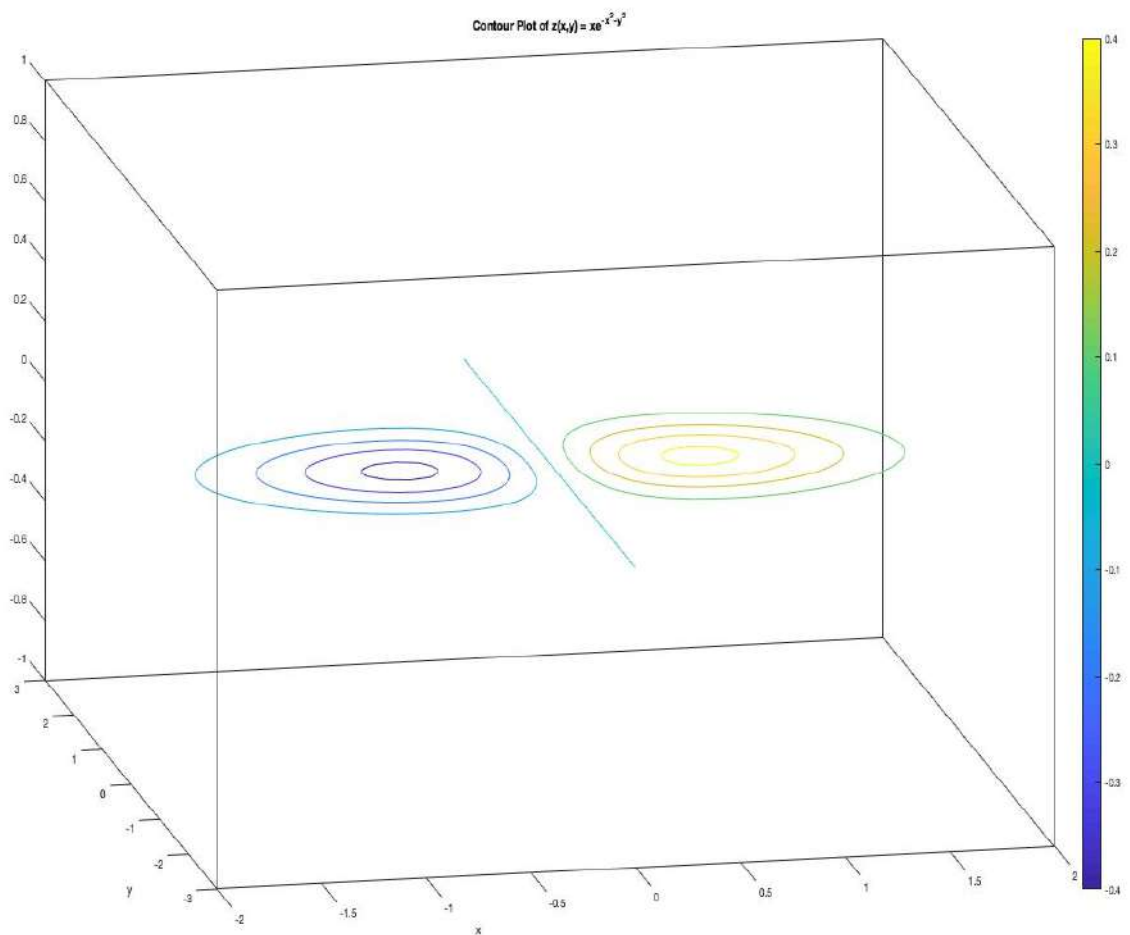
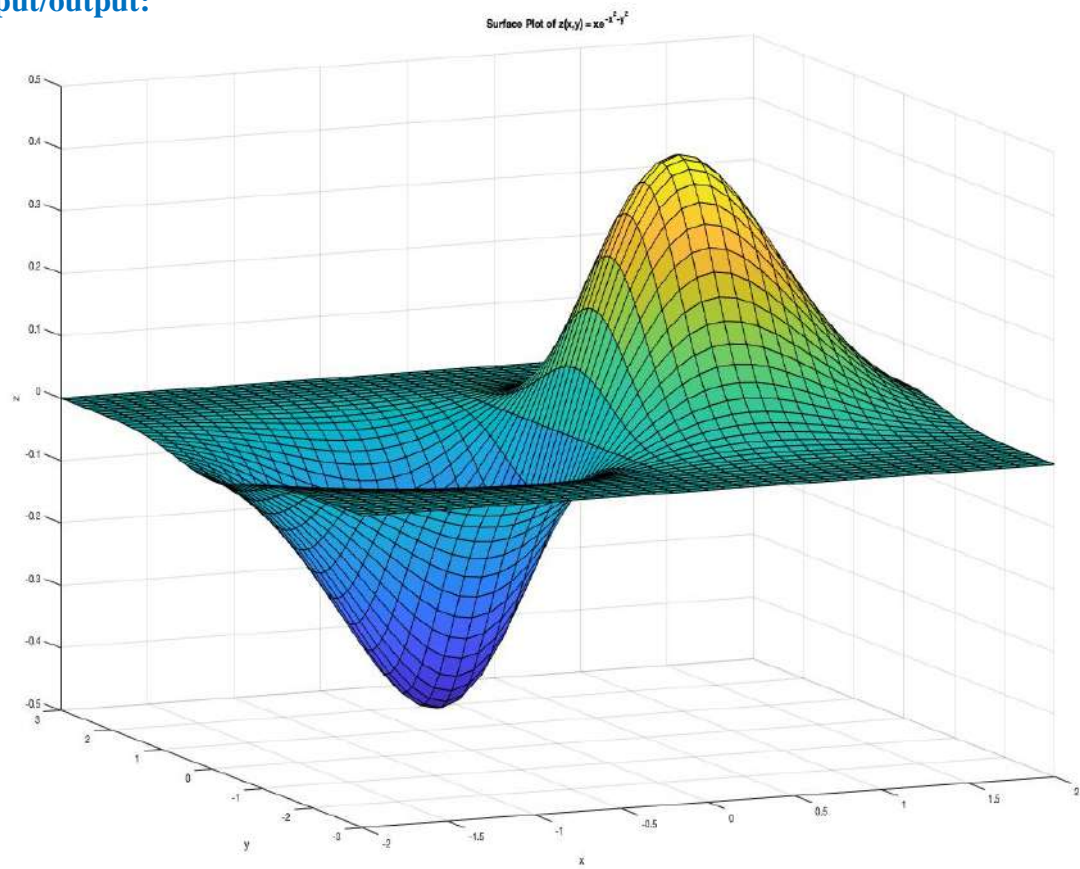
% Compute the values of z(x,y)
Z = X .* exp(-X.^2 - Y.^2);

% Generate the surface plot
figure (1);
surf(X,Y,Z);
title('Surface Plot of z(x,y) = xe^{-x^2-y^2}');
xlabel('x');
ylabel('y');
zlabel('z');

% Generate the contour plot
figure (2);
contour(X,Y,Z);
title('Contour Plot of z(x,y) = xe^{-x^2-y^2}');
xlabel('x');
ylabel('y');
colorbar;
```

Note: This script first defines the range and step size of x and y. It then creates a meshgrid from x and y, which is a 2D grid of points that can be used to evaluate the equation $z(x,y) = xe^{-x^2-y^2}$. The script then computes the values of z(x,y) for each point on the meshgrid, and generates a surface plot and a contour plot using the surf and contour functions, respectively. The surface plot shows the function as a 3D surface, while the contour plot shows the function as a series of contour lines at different heights.

Input/output:



Programming 24: Write a script in MATLAB to find the solution of the following linear equations

$$2x + y - 3z = 11$$

$$4x - 2y + 3z = 8$$

$$-2x + 2y - z = -6$$

Code:

```
% define the coefficients matrix A and the constant vector b
A = [2 1 -3; 4 -2 3; -2 2 -1];
b = [11; 8; -6];

% solve the system of linear equations
x = linsolve(A, b);

% print the solution
fprintf('The solution to the system of linear equations is:\n');
fprintf('x = %g\n', x(1));
fprintf('y = %g\n', x(2));
fprintf('z = %g\n', x(3));
```

Note: To solve this system of linear equations in MATLAB, we can use the built-in function "linsolve".

Input/output:

The solution to the system of linear equations is:

$$x = 3$$

$$y = -1$$

$$z = -2$$

Programming 25: Write a program in MATLAB to convert among decimal, binary, octal, Hexadecimal based on your inputs.

Code:

```
% Prompt the user to enter a number and its base
num = input('Enter a number: ');
base = input('Enter the base (2 for binary, 8 for octal, 10 for
decimal, or 16 for hexadecimal): ');

% Convert the number to decimal
if base == 2 % Binary to decimal conversion
    decimal_num = bin2dec(num);
elseif base == 8 % Octal to decimal conversion
    decimal_num = oct2dec(num);
elseif base == 16 % Hexadecimal to decimal conversion
    decimal_num = hex2dec(num);
else % Decimal input
    decimal_num = num;
end

% Convert the decimal number to other bases
binary_num = dec2bin(decimal_num);
octal_num = dec2base(decimal_num, 8);
hexadecimal_num = dec2hex(decimal_num);

% Print the results
fprintf('Decimal: %d\n', decimal_num);
fprintf('Binary: %s\n', binary_num);
fprintf('Octal: %s\n', octal_num);
fprintf('Hexadecimal: %s\n', hexadecimal_num);
```

Input/output:

```
Enter a number: 44
Enter the base (2 for binary, 8 for octal, 10 for decimal, or 16 for
hexadecimal): 10
Decimal: 44
Binary: 101100
Octal: 54
Hexadecimal: 2C
```

Programming 26: Write a script in MATLAB to draw the Pie diagram of a M.Sc. 1st Semester student of the following marks 35, 42, 45, 36, 38, 15.

Code:

```
% define the data as a vector
data = [35, 42, 45, 36, 38, 15];

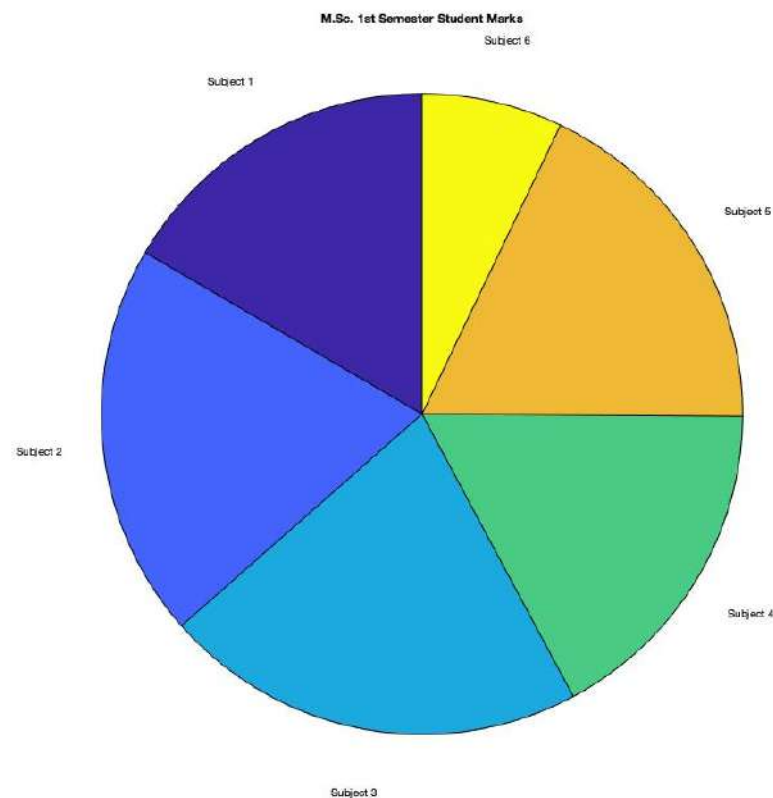
% define the labels as a cell array
labels = {'Subject 1', 'Subject 2', 'Subject 3', 'Subject 4',
         'Subject 5', 'Subject 6'};

% create the pie chart
pie(data, labels);

% add a title to the chart
title('M.Sc. 1st Semester Student Marks');
```

Note: To draw a pie chart in MATLAB, we use the built-in function "pie". This creates a pie chart with the student marks as wedges and labels on each wedge indicating the subject.

Input/output:



Programming 27: Write a script program to create a mesh, surface and contour plots of the function $z = e^{x+iy}$ in the interval $-1 < x < 1$ and $-2\pi < y < 2\pi$. In each case plot the real part of z versus x and y .

Code:

```
x=-1:0.001:1;
y=-2*pi:0.001:2*pi;
[x,y]=meshgrid(x,y); % initializes the variables for x and y
z=exp(x+i*y); % calculates the function at different values of x
and y

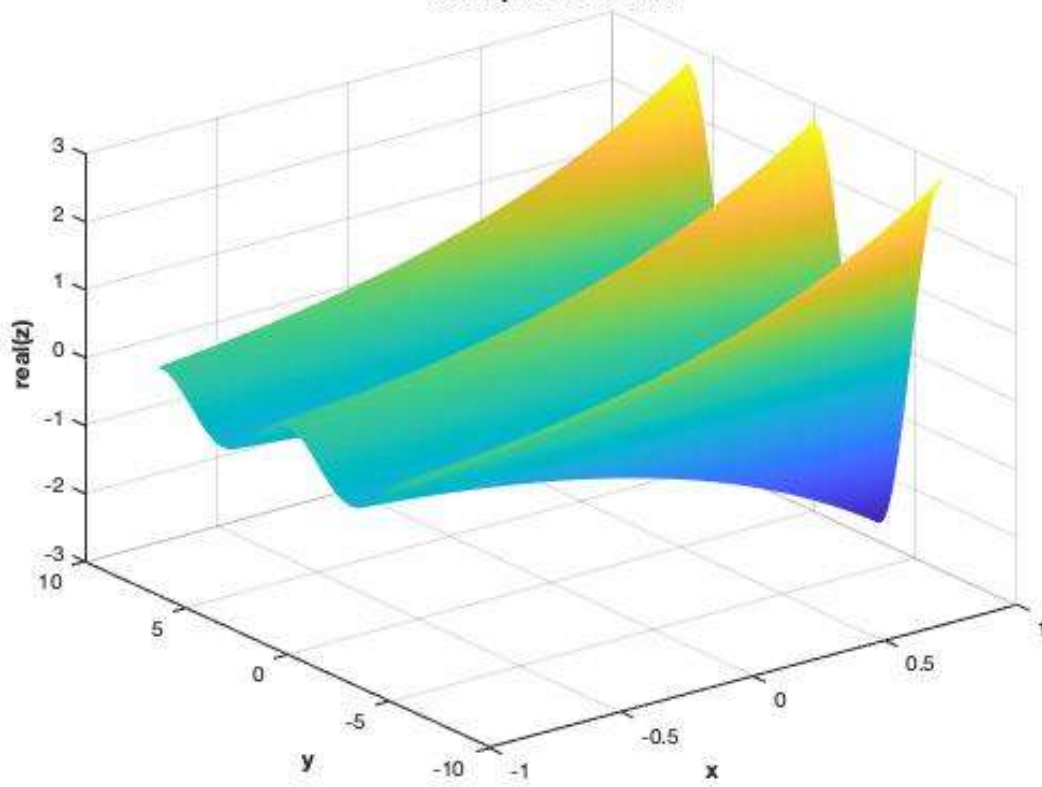
figure(1);
mesh(x,y,real(z)); % plots the mesh plot with real z versus x and
y
title('\bf Mesh plot of function')
% adds title to the plot
xlabel('\bf x');
ylabel('\bf y');
zlabel('\bf real(z)');

figure(2);
surf(x,y,real(z)); % plots the surface plot with real z versus x
and y
title('\bf Surface plot of function')
% adds title to the plot
xlabel('\bf x');
ylabel('\bf y');
zlabel('\bf real(z)');

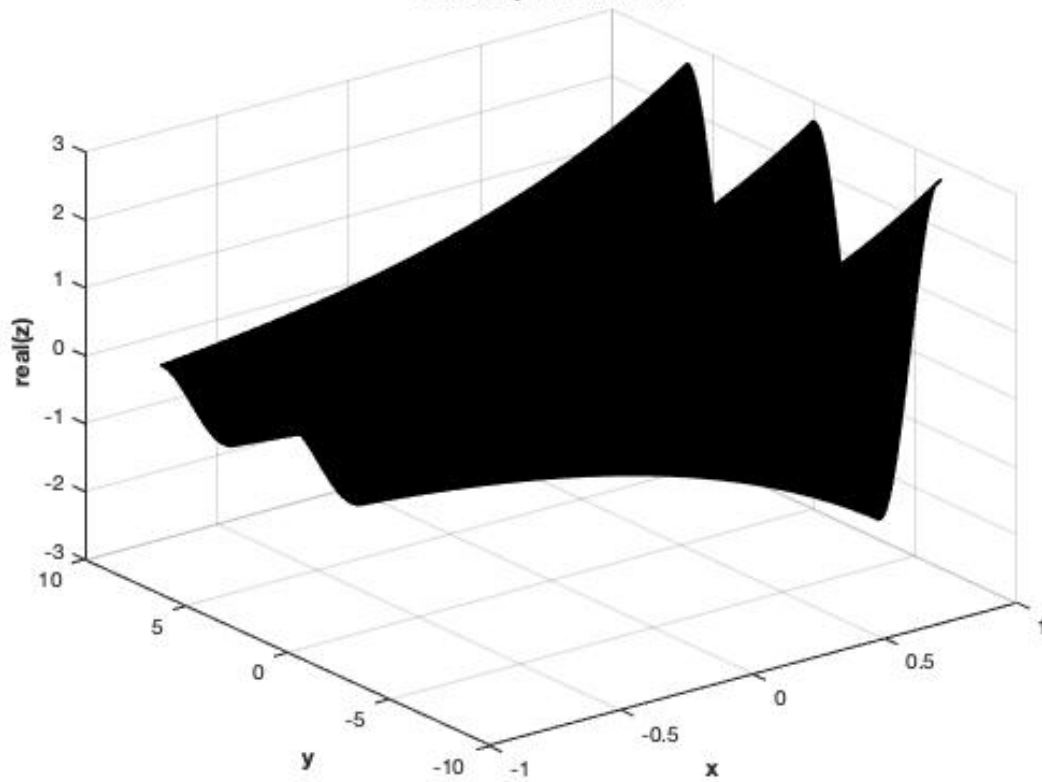
figure(3);
contour(x,y,real(z)); % plots the contour plot with real z versus
x and y
title('\bf Contour plot of function')
% adds title to the plot
xlabel('\bf x');
ylabel('\bf y');
zlabel('\bf real(z)');
```

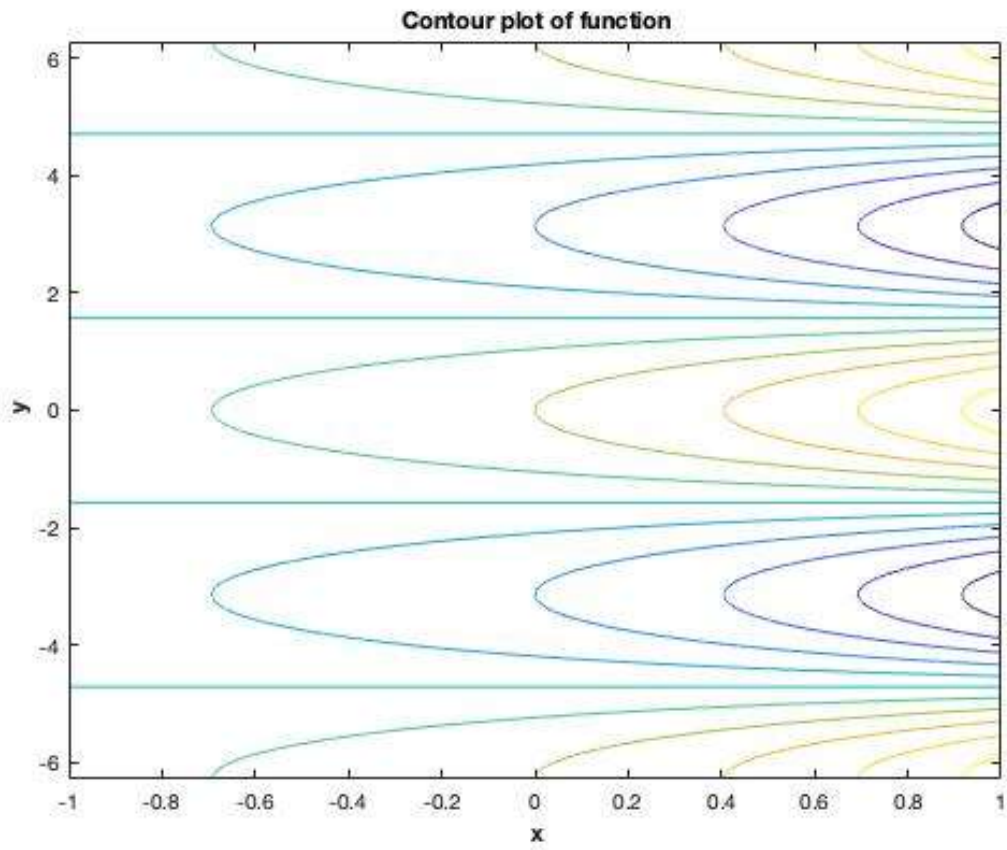
Input/output:

Mesh plot of function



Surface plot of function





Programming 28: Write a script program to find either minimum or maximum or sum according to your response of the function $y = x \sin x$ in the range $-\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$ with spacing 0.2 using switch statement.

Code:

```
% Define the function
f = @(x) x.*sin(x);

% Define the range and spacing
x = -pi/2:0.2:pi/2;

% Prompt the user for input
choice = input('Enter "min", "max", or "sum": ', 's');

% Evaluate the selected choice using a switch statement
switch choice
    case 'min'
        [y, idx] = min(f(x));
        fprintf('Minimum value of %f occurs at x = %f\n', y,
x(idx));
    case 'max'
        [y, idx] = max(f(x));
        fprintf('Maximum value of %f occurs at x = %f\n', y,
x(idx));
    case 'sum'
        y = sum(f(x));
        fprintf('Sum of values is %f\n', y);
    otherwise
        fprintf('Invalid input\n');
end
```

Input/output:

```
Enter "min", "max", or "sum": min
```

```
Minimum value of 0.000853 occurs at x = 0.029204
```

Programming 29: Write a script in MATLAB to draw the surface and contour of the equation $z(x,y) = xe^{-x^2-y^2}$ in the range $-3 \leq x \leq 3$ and $-3 \leq y \leq 3$.

Code:

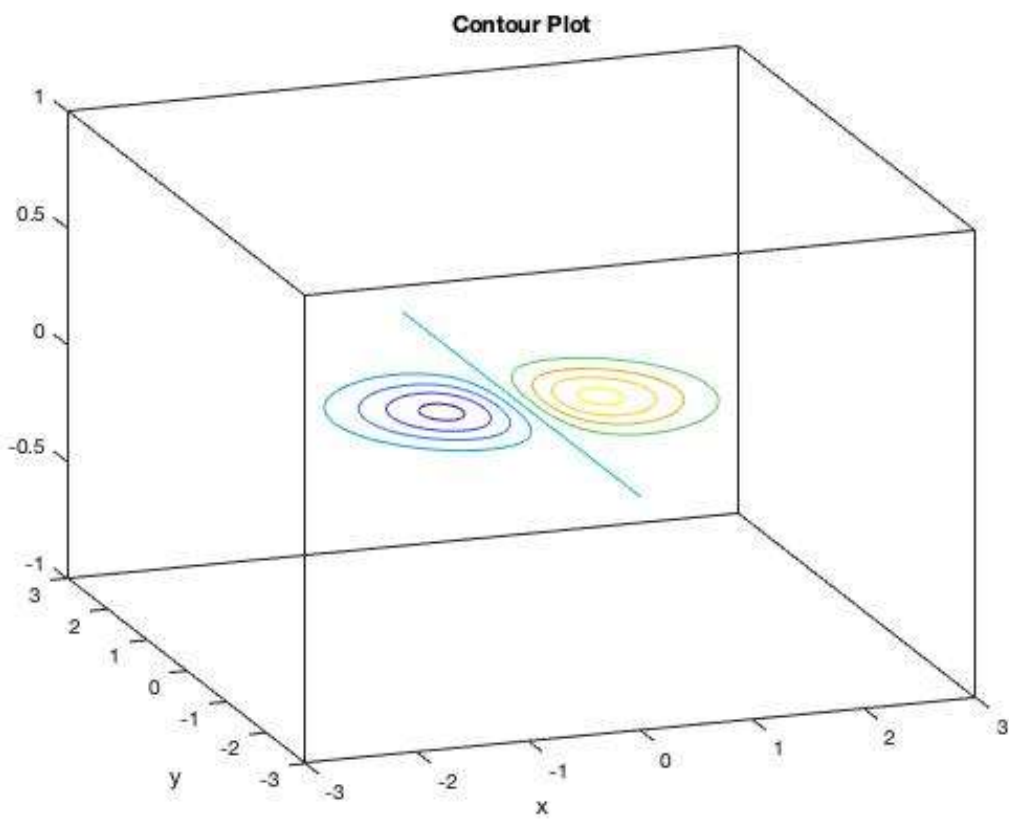
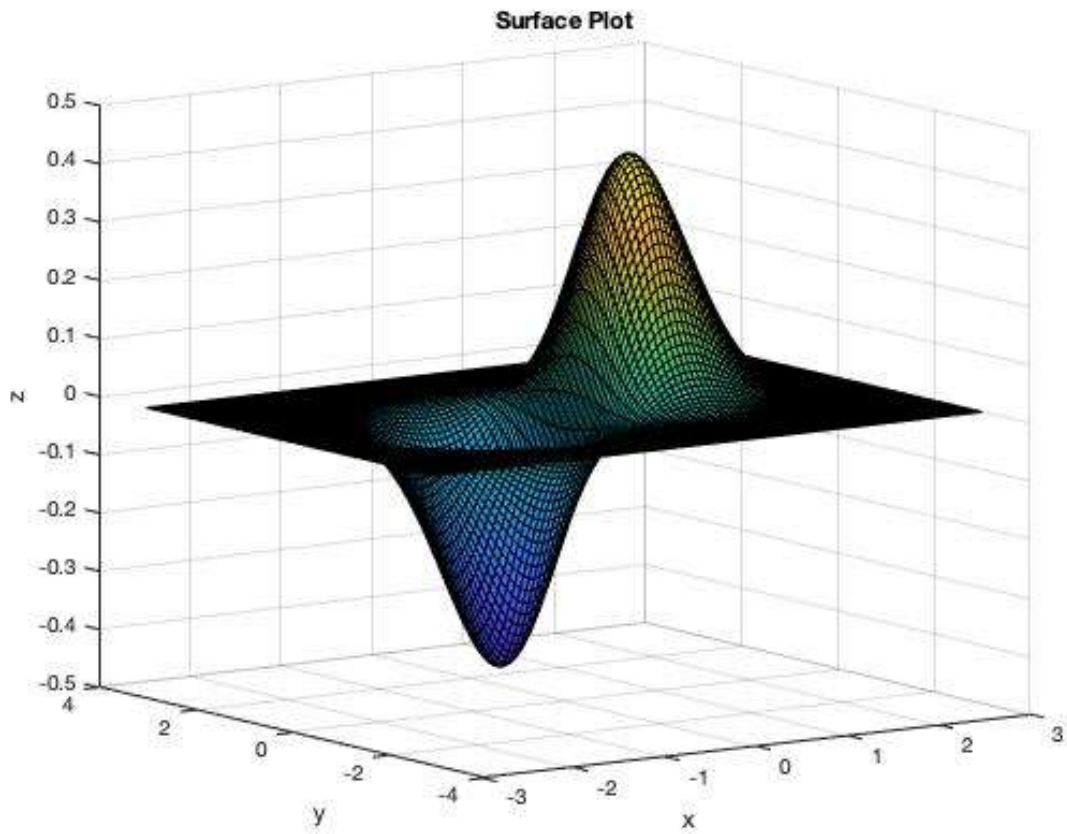
```
% Define the function
f = @(x,y) x.*exp(-x.^2-y.^2);

% Define the range and spacing
x = linspace(-3,3,100);
y = linspace(-3,3,100);
[X,Y] = meshgrid(x,y);
Z = f(X,Y);

% Draw the surface plot
figure;
surf(X,Y,Z);
title('Surface Plot');
xlabel('x');
ylabel('y');
zlabel('z');

% Draw the contour plot
figure;
contour(X,Y,Z);
title('Contour Plot');
xlabel('x');
ylabel('y');
```

Input/output:



Programming 30: Write a script in MATLAB to draw the following function in the interval $[-1, 4]$.

$$f(x) = \begin{cases} x^2 + 1, & -1 \leq x < 0 \\ 0, & x = 0 \\ x^3 + 2x + 5, & x > 0 \end{cases}$$

Code:

```
f = @(x) (x < 0) .* (x.^2 + 1) + (x == 0) .* 0 + (x > 0) .* (x.^3
+ 2*x + 5);

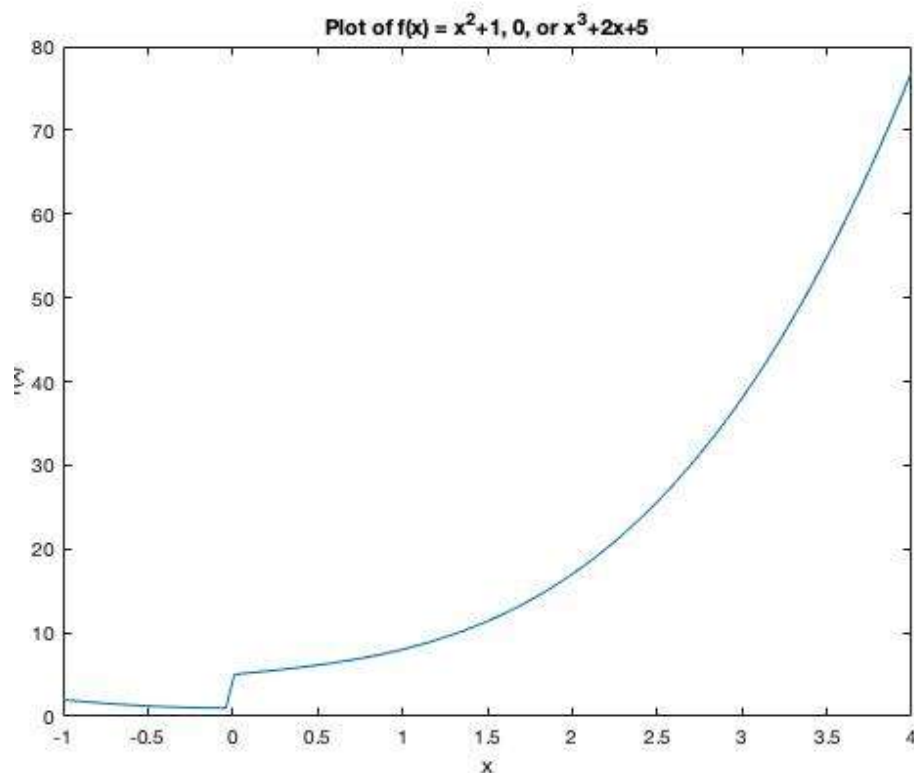
% Define the interval
x = linspace(-1, 4);

% Evaluate the function for each point in the interval
y = f(x);

% Plot the function
plot(x, y)

% Add labels and title
xlabel('x')
ylabel('f(x)')
title('Plot of f(x) = x^2+1, 0, or x^3+2x+5')
```

Input/output:



Programming 31: Write a script in MATLAB to find the solution of the following linear equations

$$3x + 5y - 6z = 6$$

$$8x - y + 2z = 1$$

$$5x - 6y - 4z = -5$$

Using *rref*, *pinv*, and left division methods.

Code:

```
% Define the coefficients matrix and the constants vector
A = [3, 5, -6; 8, -1, 2; 5, -6, -4];
b = [6; 1; -5];

% Find the solutions using rref
x_rref = rref([A, b]);
x_rref = x_rref(:,end);

% Find the solutions using pinv
x_pinv = pinv(A)*b;

% Find the solutions using left division
x_ldiv = A\b;

% Display the solutions
fprintf('Solutions using rref: x = %f, y = %f, z = %f\n', x_rref(1),
x_rref(2), x_rref(3));
fprintf('Solutions using pinv: x = %f, y = %f, z = %f\n', x_pinv(1),
x_pinv(2), x_pinv(3));
fprintf('Solutions using left division: x = %f, y = %f, z = %f\n',
x_ldiv(1), x_ldiv(2), x_ldiv(3));
```

Input/output:

Solutions using rref: x = 0.255814, y = 1.046512, z = 0.000000

Solutions using pinv: x = 0.255814, y = 1.046512, z = 0.000000

Solutions using left division: x = 0.255814, y = 1.046512, z = -
0.000000

Programming 32: Write a script in MATLAB to create two lists of numbers and perform the following arithmetic operations on it (i) addition (ii) subtraction (iii) element-wise multiplication (iv) element-wise division.

Code:

```
% Define the two arrays
a = input('Enter a row vector: ');
b = input('Enter a row vector: ');

% Perform addition
c = a + b;
fprintf('Addition: %s\n', mat2str(c));

% Perform subtraction
c = a - b;
fprintf('Subtraction: %s\n', mat2str(c));

% Perform element-wise multiplication
c = a .* b;
fprintf('Element-wise multiplication: %s\n', mat2str(c));

% Perform element-wise division
c = a ./ b;
fprintf('Element-wise division: %s\n', mat2str(c));
```

Note: 'mat2str' is a MATLAB function that converts a matrix or array to a string representation with square brackets and commas separating the elements. The purpose of mat2str is to provide a way to display an array as a string that can be easily read and understood by a human.

Input/output:

```
Enter a row vector: [1 2 3 4]
Enter a row vector: [5 6 7 8]
Addition: [6 8 10 12]
Subtraction: [-4 -4 -4 -4]
Element-wise multiplication: [5 12 21 32]
Element-wise division: [0.2 0.333333333333333 0.428571428571429 0.5]
```

Programming 33: Write a function program in MATLAB to find all Armstrong numbers between two specified numbers.

Note: An Armstrong number is a number that is equal to the sum of its digits raised to the power of the number of digits. For example, 153 is an Armstrong number because $1^3 + 3^3 + 5^3 = 153$.

Code:

Here is the MATLAB function that checks Armstrong numbers:

```
function armstrong_nums = find_armstrong_nums(start_num,
end_num)
    armstrong_nums = [];
    for num = start_num:end_num
        % Find the number of digits
        n = numel(num2str(num));
        % Compute the sum of digits raised to the power of n
        digit_sum = sum((num2str(num) - '0').^n);
        % Check if the number is an Armstrong number
        if num == digit_sum
            % Add the number to the output array
            armstrong_nums(end+1) = num;
        end
    end
end
```

end

Here is a MATLAB script that illustrates the usage of the function:

```
% Find Armstrong numbers between two specified numbers
start_num = input('Enter the first number: ');
end_num = input('Enter the last number: ');
armstrong_nums = find_armstrong_nums(start_num, end_num);
```

Input/output:

```
Enter the last number: 100
Enter the last number: 1000
The Armstrong are:
    153    370    371    407
```


Programming 34: Write a script program in MATLAB to solve the following ODE and find the value of $f(1)$ using Runge-Kutta method $\frac{dy}{dx} = y - x^2 + 1, y(0)=0.5$.

Code:

```
Here's we first define the function odefun that represents the
given ODE:
% Step 1: Define the ODE function
function dydx = odefun(x, y)
    dydx = y - x^2 + 1;
end

                                     ****

% Step 2: Set initial condition and range of independent variable
y0 = 0.5;
xspan = [0 1];

% Step 3: Solve the ODE using ode45
[x, y] = ode45(@odefun, xspan, y0);

% Step 4: Evaluate f(1)
f1 = interp1(x, y, 1);

% Display the result
fprintf('The value of f(1) is %f\n', f1);
fprintf('%d ', fib_range);
fprintf('\n');
```

Remark: In this code, the `ode45` function is used to solve the ODE numerically. The `interp1` function is used to interpolate the value of y at $x=1$. The Runge-Kutta method is implemented by default in `ode45` function in MATLAB, so you do not need to specify it explicitly.

Input/output: The value of $f(1)$ is 2.640859.

Programming 35: Write a MATLAB function program to find a real root of the equation $x^2 - \sin 2x - 1 = 0$ by Newton-Raphson's method.

Code:

```
function root = newton_raphson()
% set the initial guess
x0 = 1;

% set the maximum number of iterations
max_iter = 100;

% set the tolerance
tol = 1e-6;

% loop until the maximum number of iterations is reached
for i = 1:max_iter
    % calculate the function value and derivative at x0
    f = x0^2 - sin(2*x0) - 1;
    df = 2*x0 - 2*cos(2*x0);

    % update x0 using the Newton-Raphson formula
    x0 = x0 - f/df;

    % check if the tolerance is reached
    if abs(f) < tol
        root = x0;
        return
    end
end

% if the maximum number of iterations is reached, print an error
message
error('The maximum number of iterations is reached.');
```

Input/output:

newton_raphson

ans =

1.2587

Programming 36: Write a script in MATLAB to find the mean, median, variance and standard deviation for a set of discrete data.

Code:

```
% input the data as a row vector
data = input('Enter the data: ');

% calculate the mean
mean_data = mean(data);

% calculate the median
median_data = median(data);

% calculate the variance
var_data = var(data);

% calculate the standard deviation
std_data = std(data);

% print the results
fprintf('Mean: %f\n', mean_data);
fprintf('Median: %f\n', median_data);
fprintf('Variance: %f\n', var_data);
fprintf('Standard Deviation: %f\n', std_data);
```

Input/output:

```
Enter the data: [10 5 6 7 9 20 30]
Mean: 12.428571
Median: 9.000000
Variance: 84.952381
Standard Deviation: 9.216962
```

Programming 37: Write a script in MATLAB to find an invertible matrix P and a diagonal D such that $PDP^{-1} = A$, then compare A^5 and PA^5P^{-1} .

Code:

```
% prompt the user to input a matrix
A = input('Enter a matrix: ');

% find the eigenvalues and eigenvectors of A
[V, D] = eig(A);

% construct the diagonal matrix D
D = diag(diag(D));

% construct the invertible matrix P
P = V * inv(sqrt(D));

% check that P*D*inv(P) = A
if norm(P*D*inv(P) - A) < 1e-10
    fprintf('P*D*inv(P) equals A.\n');
else
    error('P*D*inv(P) does not equal A.');
```

```
end

% compare A^5 and P*A^5*inv(P)
if norm(A^5 - P*A^5*inv(P)) < 1e-10
    fprintf('A^5 equals P*A^5*inv(P).\n');
else
    fprintf('A^5 does not equal P*A^5*inv(P).\n');
```

```
end
```

Input/output:

Enter a matrix: [1 1 -2;-1 2 1;0 1 -1]

P*D*inv(P) equals A.

A^5 does not equal P*A^5*inv(P).

Programming 38: Write a user defined function in MATLAB to generate Fibonacci sequence. Use this function; write a script to find the Fibonacci numbers between two specified numbers.

Code:

Here's a user-defined function in MATLAB to generate the Fibonacci sequence:

```
function fib = fibonacci(n)
% Inputs:
% n: the number of Fibonacci numbers to generate
% Outputs:
% fib: a vector of the first n Fibonacci numbers

% initialize the first two numbers in the sequence
fib(1) = 1;
fib(2) = 1;

% generate the remaining numbers in the sequence
for i = 3:n
    fib(i) = fib(i-1) + fib(i-2);
end
end
```

To use this function to find the Fibonacci numbers between two specified numbers, we can write a script like this:

```
% set the lower and upper bounds
lower = input('Enter the lower range:');
upper = input('Enter the upper range:');

% generate the Fibonacci numbers up to the upper bound
n = 2;
while fibonacci(n) < upper
    n = n + 1;
end
fib = fibonacci(n);

% find the Fibonacci numbers between the lower and upper bounds
fib_range = fib(fib >= lower & fib <= upper);

% display the results
fprintf('The Fibonacci numbers between %d and %d are: ', lower,
upper);
fprintf('%d ', fib_range);
```

Input/output: The Fibonacci numbers between 1 and 10 are: 1 1 2 3 5 8.

Programming 39: Write a script in MATLAB to represent the graphs in the same window of the functions $\sin x$, $\sin 2x$ and $\sin 3x$ in the range $(0, 2\pi)$ with mentions different line specification, title, axes and axes limits.

Code:

```
% define the range of x
x = linspace(0, 2*pi);

% calculate the values of the functions
y1 = sin(x);
y2 = sin(2*x);
y3 = sin(3*x);

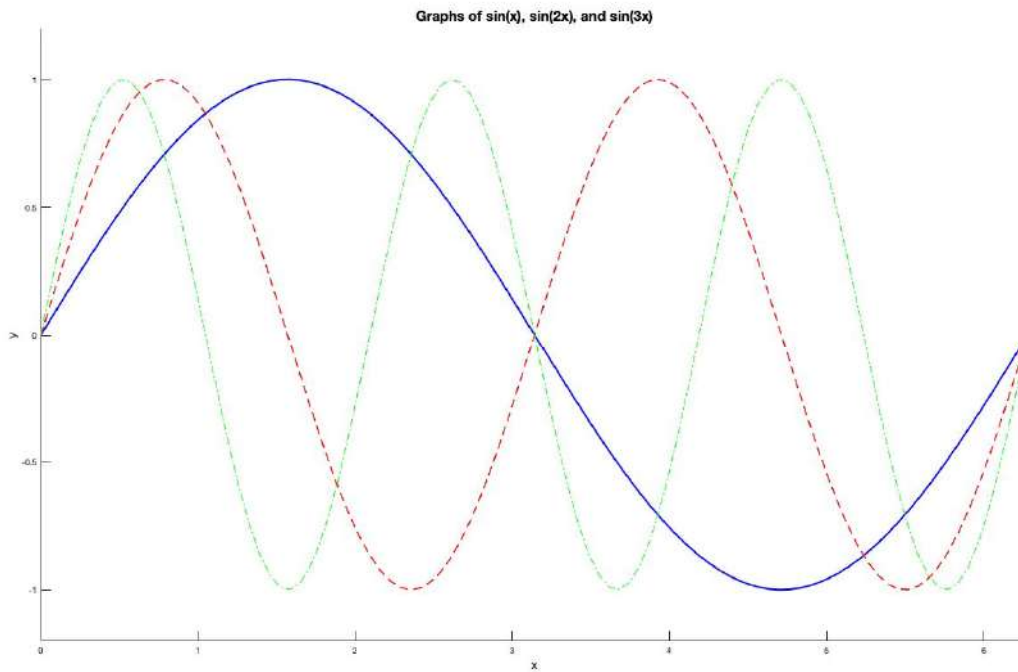
% plot the functions in the same window
hold on;
plot(x, y1, 'b-', 'LineWidth', 2);
plot(x, y2, 'r--', 'LineWidth', 1.5);
plot(x, y3, 'g-.', 'LineWidth', 1);
hold off;

% set the title and axis labels
title('Graphs of sin(x), sin(2x), and sin(3x)', 'FontSize', 16);
xlabel('x', 'FontSize', 14);
ylabel('y', 'FontSize', 14);

% set the axis limits
xlim([0, 2*pi]);
ylim([-1.2, 1.2]);

% add a legend
legend('sin(x)', 'sin(2x)', 'sin(3x)', 'FontSize', 14);
```

Input/output:



Programming 40: Write a user defined function in MATLAB to determine the roots of a quadratic equation. Use this function; write a script find the roots of the equation $x^2 + 5x + 6 = 0$.

Code:

Here's a user-defined function in MATLAB to determine the roots of a quadratic equation:

```
function [root1, root2] = quadratic(a, b, c)
% Inputs:
% a, b, c: coefficients of the quadratic equation ax^2 + bx + c = 0
% Outputs:
% root1, root2: the roots of the quadratic equation

% calculate the discriminant
D = b^2 - 4*a*c;

% check if there are real roots
if D < 0
    error('The quadratic equation has no real roots.');
```

To use this function to find the roots of the equation $x^2 + 5x + 6 = 0$, we can write a script like this:

```
% set the coefficients of the quadratic equation
a = 1;
b = 5;
c = 6;

% call the quadratic function to find the roots
[root1, root2] = quadratic(a, b, c);

% display the results
fprintf('The roots of the quadratic equation are: %.2f, %.2f\n',
root1, root2);
```

Input/output: The roots of the quadratic equation are: -2.00, -3.00.

Programming 41: Write a user define function to find the value of $\int_a^b f(x)dx$ by Simpson 1/3's rule. Use this function; write a script program to find the value of the following integration $\int_0^1 x^2 + x dx$.

Code:

Here's a user-defined function in MATLAB to find the value of an integral $\int_a^b f(x)dx$ using Simpson's 1/3 rule:

```
function I = simpson13(f, a, b, n)
% Inputs:
% f: a function handle for the integrand
% a: the lower limit of integration
% b: the upper limit of integration
% n: the number of subintervals (must be even)
% Output:
% I: the value of the integral

h = (b - a) / n; % width of each subinterval
x = a:h:b; % generate the nodes for integration

% evaluate the integrand at each node
y = f(x);

% calculate the integral using Simpson's 1/3 rule
I = h/3 * (y(1) + 4*sum(y(2:2:end-1)) + 2*sum(y(3:2:end-2)) +
y(end));
end
```

To use this function to find the value of the integral $\int_0^1 x^2 + x dx$, we can write a script like this:

```
% define the integrand as a function handle
f = @(x) x.^2 + x;

% set the limits of integration and the number of subintervals
a = 0;
b = 1;
n = 10; % must be even

% call the Simpson 1/3 rule function to find the integral
I = simpson13(f, a, b, n);

% display the result
fprintf('The value of the integral is % 6f\n', I);
```

Input/output: The value of the integral is 0.833333.

Programming 42: Write a script program in MATLAB to solve the following ODE and find the values of $f(0.1)$ and $f(0.2)$ using Euler method $\frac{dy}{dx} = x^2 + y^2$, $y(0) = 1$.

Code:

```
% Define the ODE and initial conditions
dydx = @(x, y) x^2 + y^2;
x0 = 0;
y0 = 1;

% Define the step size and endpoint
h = 0.01;
x_end = 0.2;

% Use the Euler method to solve the ODE
x = x0:h:x_end;
y = zeros(size(x));
y(1) = y0;
for i = 1:length(x)-1
    y(i+1) = y(i) + h * dydx(x(i), y(i));
end

% Find the values of f(0.1) and f(0.2)
f_01 = y(x==0.1);
f_02 = y(end);

% Display the results
fprintf('f(0.1) = %f\n', f_01);
fprintf('f(0.2) = %f\n', f_02);
```

Input/output:

$f(0.1) = 1.110130$

$f(0.2) = 1.249332$

Programming 43: Write a user defined function in MATLAB to calculate the sum of a list of numbers. Using it, find the sum of all-natural numbers between two specified numbers.

Code:

Here's an example user-defined MATLAB function that calculates the sum of a list of numbers:

```
function [total] = sum_list(numbers)
% Calculate the sum of a list of numbers
% Inputs:
% - numbers: a vector or matrix of numbers to be summed
% Outputs:
% - total: the sum of the input numbers
```

```
total = sum(numbers(:));
```

```
end
```

To find the sum of all-natural numbers between two specified numbers, we can use the sum_list function in the following script:

```
% Define the two numbers between which to sum the natural numbers
```

```
start_num=input('Enter the first number\n');
```

```
end_num = input('Enter the end number\n');
```

```
% Generate the sequence of natural numbers using the colon
operator
```

```
nat_nums = start_num:end_num;
```

```
% Calculate the sum of the natural numbers using the sum_list
function
```

```
nat_num_sum = sum_list(nat_nums);
```

```
% Display the result
```

```
fprintf('The sum of all natural numbers between %d and %d is
%d.\n', start_num, end_num, nat_num_sum);
```

Input/output:

```
Enter the first number
```

```
2
```

```
Enter the end number
```

```
100
```

```
The sum of all natural numbers between 2 and 100 is 5049.
```

Programming 44: For a diagonalizable A , write a function program that returns true if A is positive definite and false otherwise. Also, write a script program to illustrate it.

Code:

Here is the MATLAB function that checks if a diagonalizable matrix A is positive definite:

```
function [is_pos_def] = isPositiveDefinite(A)
% Check if a diagonalizable matrix A is positive definite
% Inputs:
% - A: a diagonalizable matrix
% Outputs:
% - is_pos_def: a boolean value indicating whether A is positive
definite

% Calculate the eigenvalues of A
eig_vals = eig(A);

% Check if all eigenvalues are positive
if all(eig_vals > 0)
    is_pos_def = true;
else
    is_pos_def = false;
end

end
```

Here is a MATLAB script that illustrates the usage of the function:

```
% Define a matrix A
A=input('Enter the matrix\n');

% Check if A is positive definite
if isPositiveDefinite(A)
    disp('A is positive definite');
else
    disp('A is not positive definite');
```

Input/output:

```
Enter the matrix
[1 0 0;0 1 0;0 0 1]
A is positive definite
```

Programming 45: For a given square matrix of order 5, write a script program to carry out of the following:

- (i) sort each column and store the result in an array B.
- (ii) sort each row and store the result in an array C.
- (iii) add each column and store the result in an array D.
- (iv) add each row and store the result in an array E.

Code:

```
% Define the matrix
A = randi([1,10],5);

% Sort each column and store the result in an array B
B = sort(A);

% Sort each row and store the result in an array C
C = sort(A, 2);

% Add each column and store the result in an array D
D = sum(A);

% Add each row and store the result in an array E
E = sum(A, 2);

% Display the original matrix and the results
disp('Original matrix:');
disp(A);
disp('Sorted columns:');
disp(B);
disp('Sorted rows:');
disp(C);
disp('Column sums:');
disp(D);
disp('Row sums:');
disp(E);
```

Input/output:

Original matrix:

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

Sorted columns:

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

Sorted rows:

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

Column sums:

38	30	36	35	34
----	----	----	----	----

Row sums:

21
29
37
43
43