K. J. Somaiya College of Engineering, Mumbai-77

**Batch:A-3** **Roll No.: 16010122104**

**Experiment No. 1**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

**Title:** Introduction to Matlab

**Objective:** To familiarize the beginner to MATLAB by introducing the basic features and commands of the program.

**Expected Outcome of Experiment:**

| CO | Outcome |
|---|---|
| CO1 | Identify various discrete time signals and systems and perform signal manipulation |

**Books/ Journals/ Websites referred:**


1. http://www.mathworks.com/support/
2. www.math.mtu.edu/~msgocken/intro/intro.html
3. www.mccormick.northwestern.edu/docs/efirst/matlab.pdf

**Pre Lab/ Prior Concepts:**

**INTRODUCTION TO MATLAB**

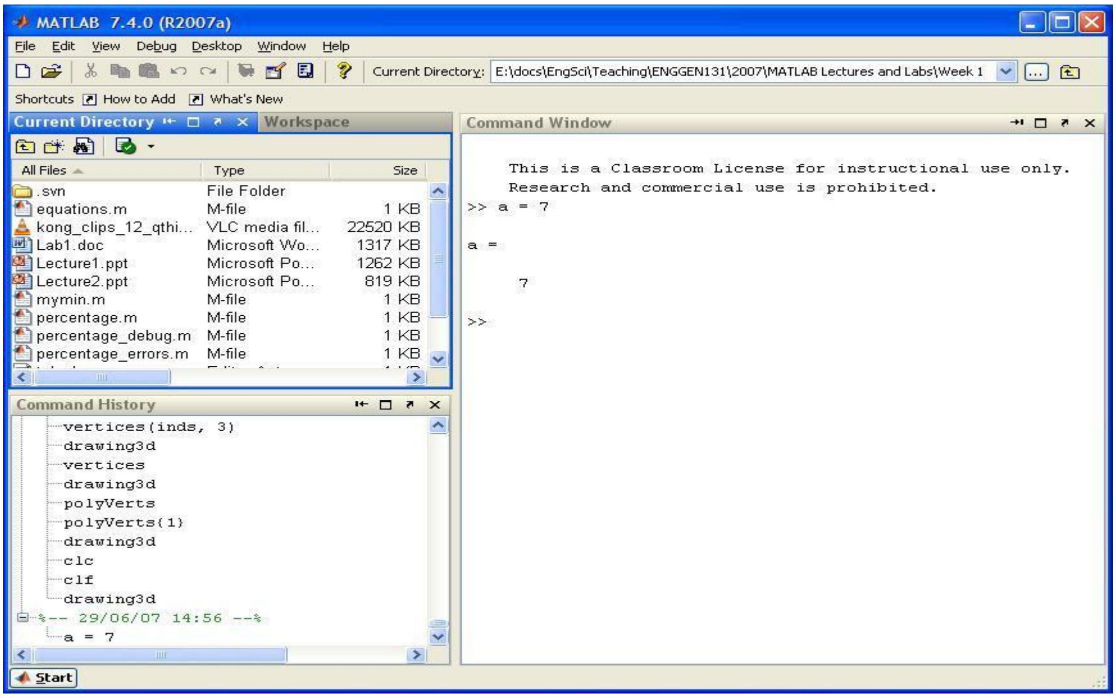
MATLAB (MATrix LABoratory) is an interactive software system for numerical computations and graphics. As the name suggests, MATLAB is especially designed for linear equations, performing matrix forth. In addition, it has a variety of through programs written in its own matrix computations: solving systems of transformations, factoring matrices, and so graphical capabilities, and can be extended programming language.

**KEY FEATURES**

- High-level language for numerical computation, visualization, and application development.
- Interactive environment for iterative exploration, design, and problem solving.
- Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration, and solving ordinary differential equations.
- Built-in graphics for visualizing data and tools for creating custom plots.
- Development tools for improving code quality and maintainability and maximizing performance.
- Tools for building applications with custom graphical interfaces.
- Functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET.

**GETTING STARTED**

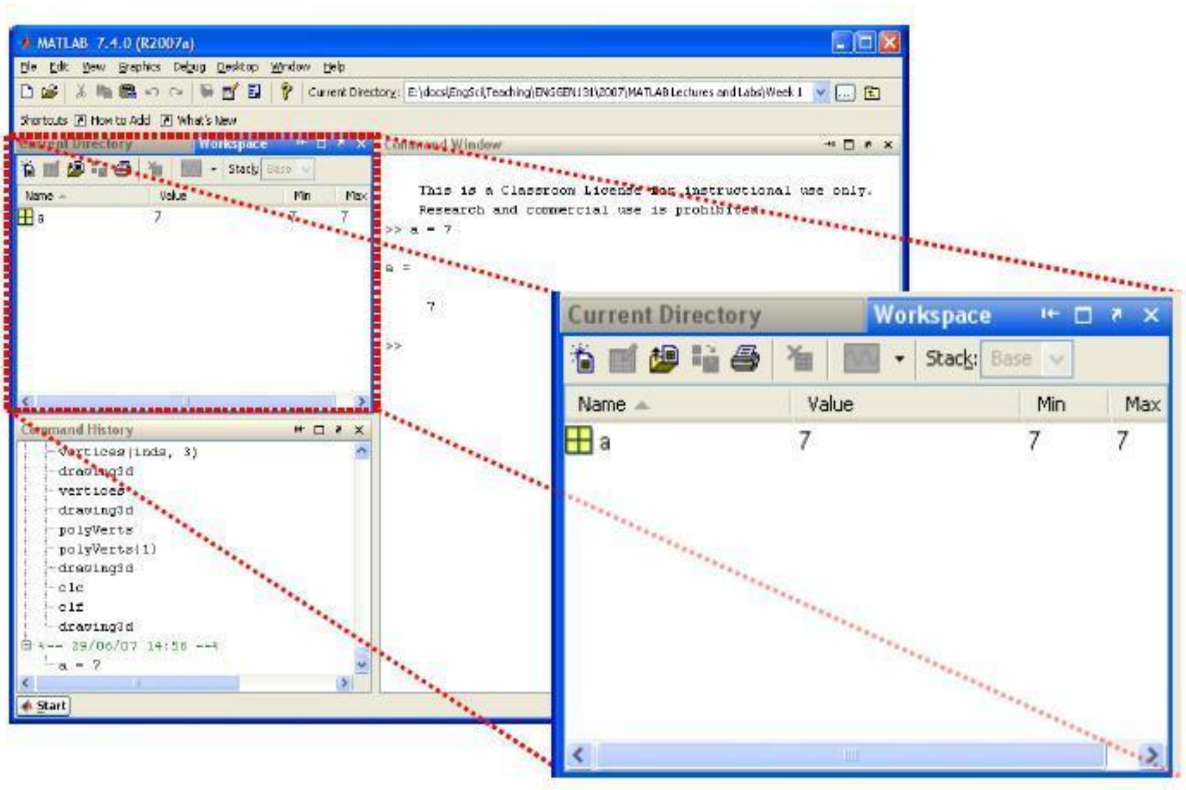
Double click on the MATLAB icon. The MATLAB window should come up on your screen. It looks like this:




The main window on the right is called the *Command* Window. This is the window in which you interact with MATLAB. Once the MATLAB prompt >> is displayed, all MATLAB commands are executed from this window. In the figure, you can see that we execute the command:

**>> a = 7**

In the top left corner you can view the *Workspace* window and the *Current Directory* window. Swap from one to the other by clicking on the appropriate tab. Note that when you first start up MATLAB, the workspace window is empty. However, as you execute commands in the *Command* window, the *Workspace* window will show all variables that you create in your current MATLAB session. In this example, the workspace contains the variable **a**.




In the bottom left corner you can see the *Command History* window, which simply gives a chronological list of all MATLAB commands that you used.

During the MATLAB sessions you will create files to store programs or workspaces. Therefore create an appropriate folder to store the lab files.

**THE MATLAB HELP SYSTEM**

MATLAB’s help system provides information to assist you while using MATLAB. Select *Help for MATLAB Help* to open the help system.

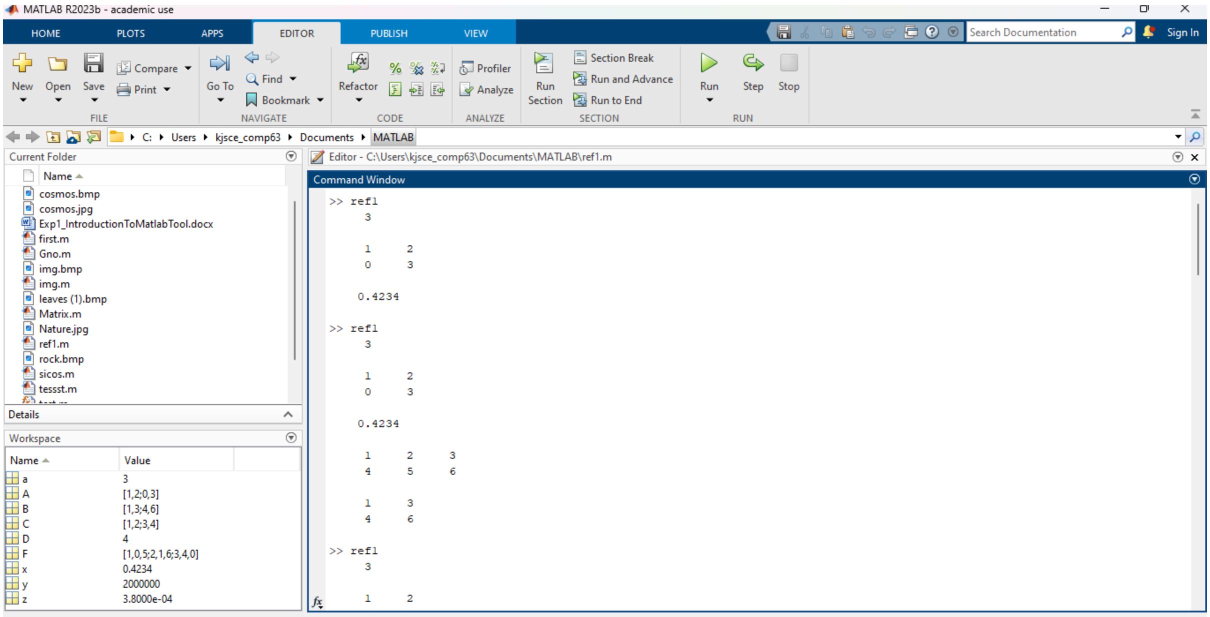
We can then browse the commands via the *Contents* window, look through the *Index* of commands or *Search* for keywords or phrases within the documentation. This is the most comprehensive documentation for MATLAB and is the best place to find out what you need. You can also start the MATLAB *Help* using the **helpwin** command.

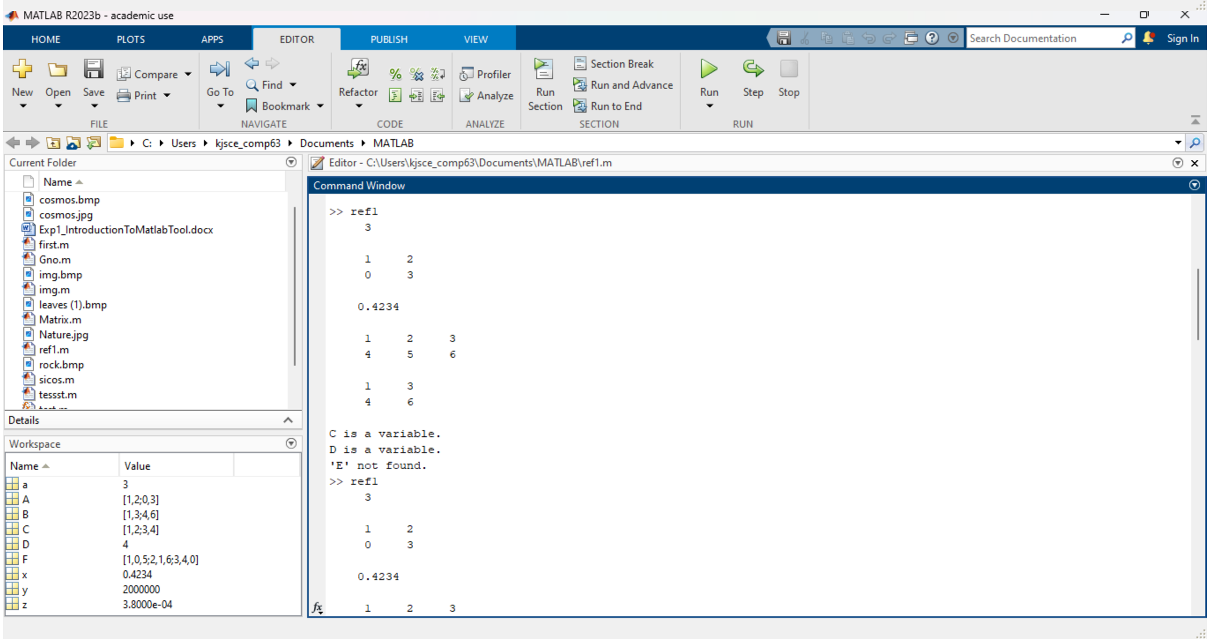
A faster way to find out information about commands is via the **help** command. If you just type **help** and hit return, you will be presented with a menu of help options. If you type **help <command>**, MATLAB will provide help for that particular command. For example, **help hist** will call up information on the **hist** command, which produces histograms. Note that the **help** command only displays a short summary of how to use the command you are interested in. For more detailed help documentation, use the **doc** command. For example, **doc hist** will display the full documentation of the **hist** command, including examples of how to use it and sample output.

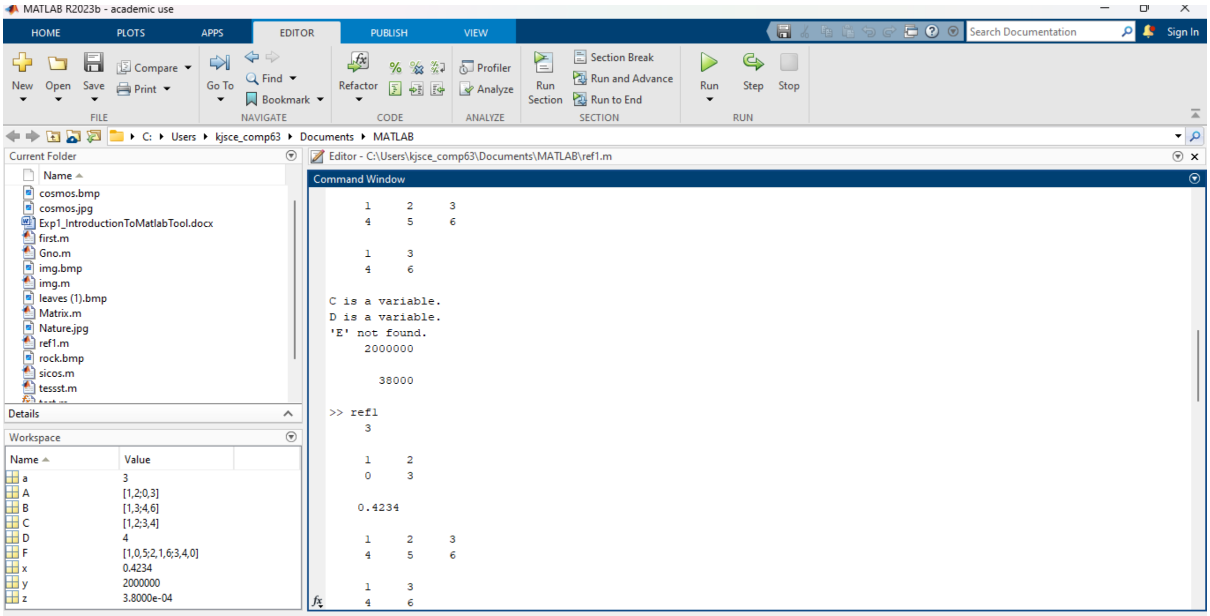
**Matlab Commands used for Signal and Image processing:**

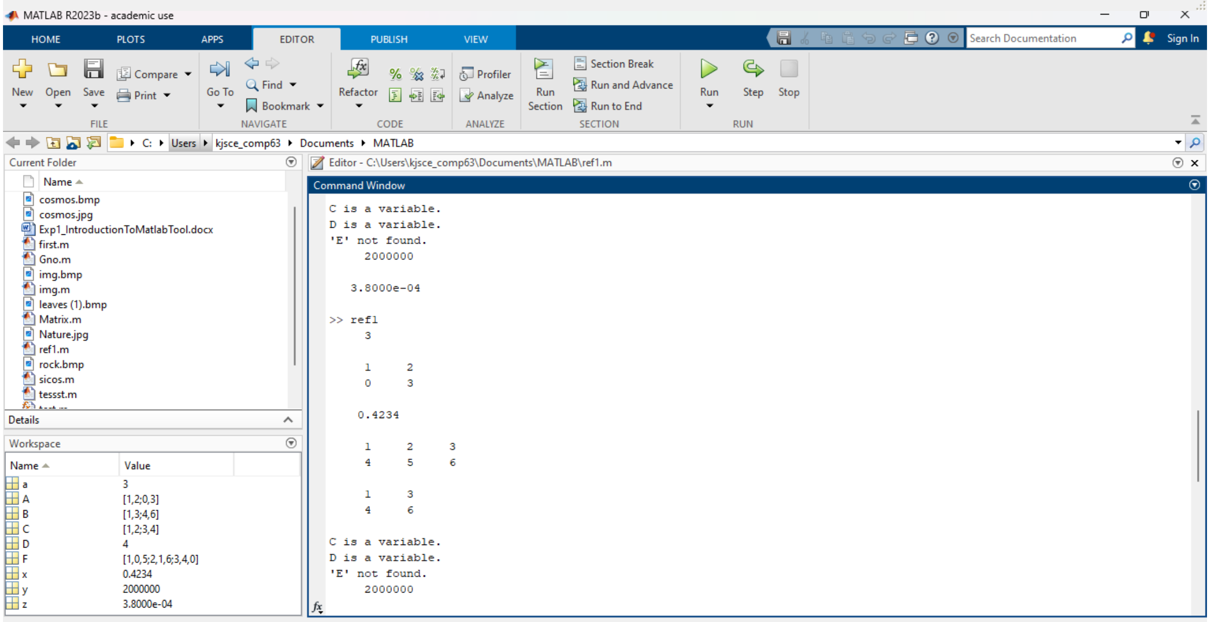
```
a=3;
disp(a);
A=[1 2;0 3];
disp(A);
x=3*sin(a);
disp(x);
B=[1 2 3; 4 5 6];
disp(B);
B(:,2)=[];
disp(B);
C=[1 2;3 4];
D=4;
which -all C
which -all D
which -all E
y=2e6;
disp(y);
z=3.8e-4;
disp(z);
F=[1 0 5;2 1 6;3 4 0];
disp(rank(F));
disp(det(F));
disp(inv(F));
disp(diag(F));
disp(ones(3));
disp(eye(3));

figure(1);
t=-2*pi:0.1*pi:2*pi;
z=t.*sin(t);
plot(t,z);
title('Plot of tsin(t) vs t');
xlabel('t');
ylabel('z');
grid on

figure(2);
x=0:pi/10:2*pi;
[X,Y,Z]=cylinder(4*cos(x));
subplot(2,2,1); mesh(X)
subplot(2,2,2); mesh(Y)
subplot(2,2,3); mesh(Z)
subplot(2,2,4); mesh(X,Y,Z)
```

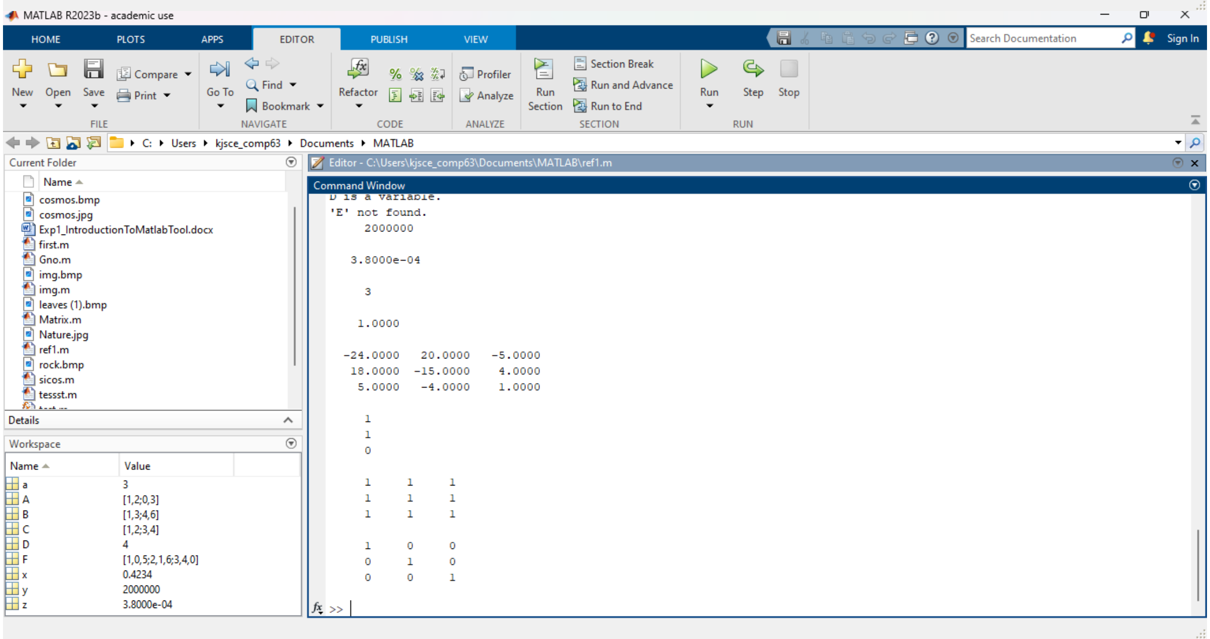
```
a=6;
b=5;
if(a>b)
    disp('a is greater than b');
elseif(b>a)
    disp('b is greater tha a');
else
    disp('a is equal to b');
end
c=5;
for C=1:c
disp(C);
end
d=0;
while d<3
    d=d+1;
    disp(d);
end
```

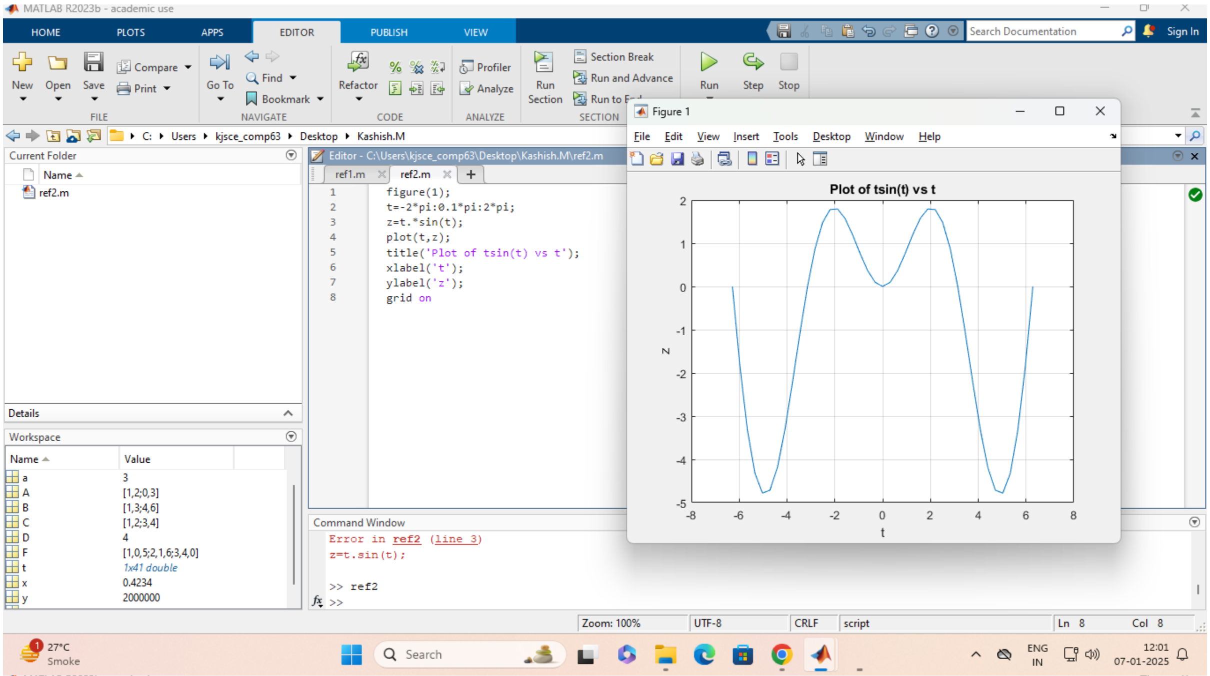


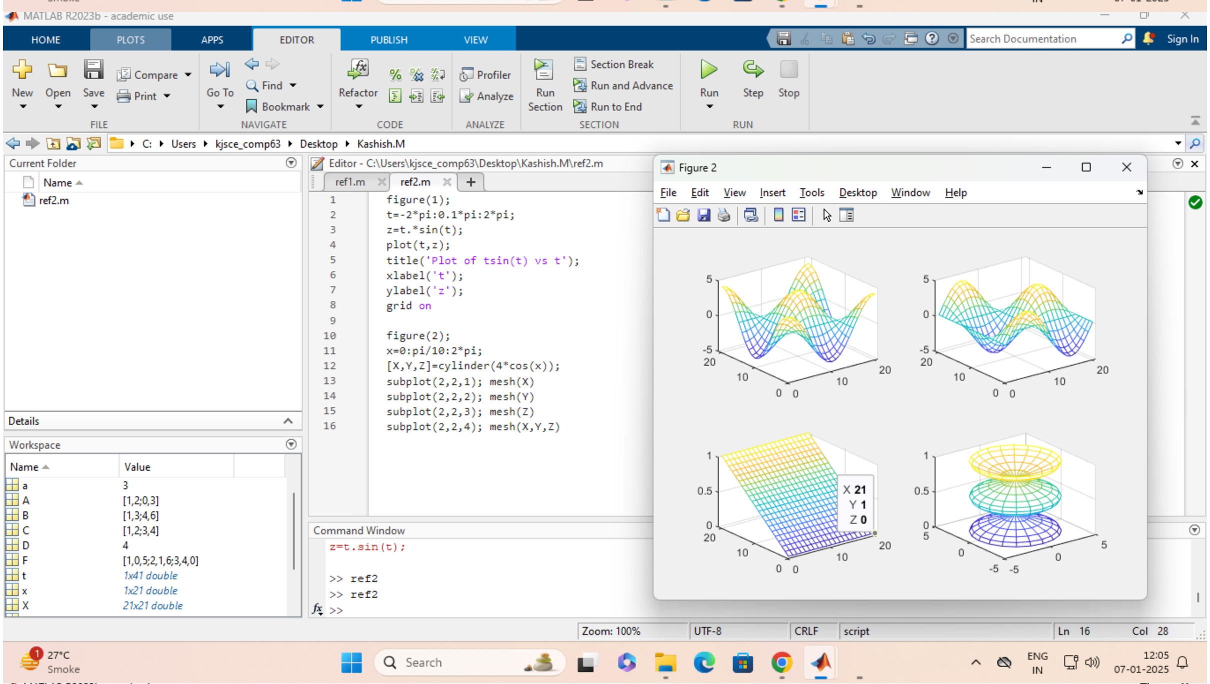




MATLAB R2023b - academic use
Editor - C:\Users\kjsce_comp63\Documents\MATLAB\ref1.m
Command Window
1 2 3
4 5 6
1 3
4 6
C is a variable.
D is a variable.
'E' not found.
2000000
38000
>> ref1
3
1 2
0 3
0.4234
1 2 3
4 5 6
1 3
4 6
Workspace
a 3
A [1,2;0,3]
B [1,3;4,6]
C [1,2;3,4]
D 4
F [1,0,5;2,1,6;3,4,0]
x 0.4234
y 2000000
z 3.8000e-04

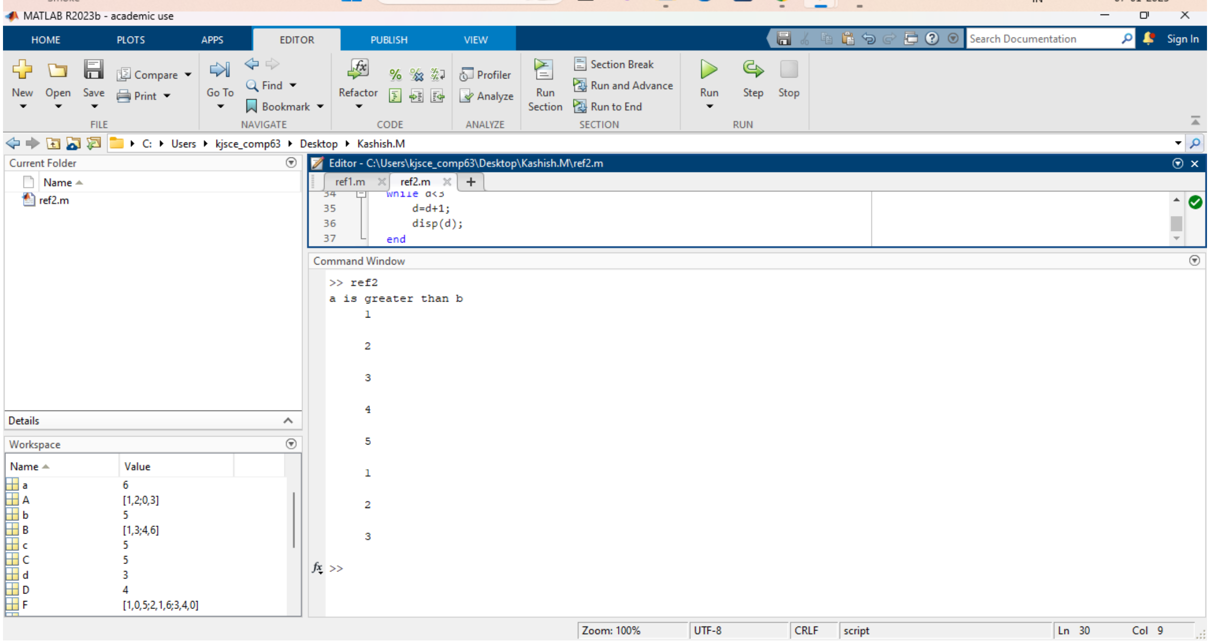

MATLAB R2023b - academic use
Editor - C:\Users\kjsce_comp63\Documents\MATLAB\ref1.m
Command Window
C is a variable.
D is a variable.
'E' not found.
2000000
3.8000e-04
>> ref1
3
1 2
0 3
0.4234
1 2 3
4 5 6
1 3
4 6
C is a variable.
D is a variable.
'E' not found.
2000000


MATLAB R2023b - academic use
Editor - C:\Users\kjsce_comp63\Documents\MATLAB\ref1.m
Command Window
'E' not found.
2000000
3.8000e-04
3
1.0000
-24.0000 20.0000 -5.0000
18.0000 -15.0000 4.0000
5.0000 -4.0000 1.0000
1
1
0
1 1 1
1 1 1
1 1 1
1 0 0
0 1 0
0 0 1
>>


MATLAB R2023b - academic use
Editor - C:\Users\kjsce_comp63\Desktop\Kashish.M\ref2.m
figure(1);
t=-2*pi:0.1*pi:2*pi;
z=t.*sin(t);
plot(t,z);
title('Plot of tsin(t) vs t');
xlabel('t');
ylabel('z');
grid on
Figure 1
Plot of tsin(t) vs t
Command Window
Error in ref2 (line 3)
z=t.sin(t);
>> ref2
>>
27°C Smoke
12:01 07-01-2025


MATLAB R2023b - academic use
Editor - C:\Users\kjsce_comp63\Desktop\Kashish.M\ref2.m
figure(1);
t=-2*pi:0.1*pi:2*pi;
z=t.*sin(t);
plot(t,z);
title('Plot of tsin(t) vs t');
xlabel('t');
ylabel('z');
grid on

figure(2);
x=0:pi/10:2*pi;
[X,Y,Z]=cylinder(4*cos(x));
subplot(2,2,1); mesh(X)
subplot(2,2,2); mesh(Y)
subplot(2,2,3); mesh(Z)
subplot(2,2,4); mesh(X,Y,Z)
Figure 2
Command Window
z=t.sin(t);
>> ref2
>> ref2
>>
27°C Smoke
12:05 07-01-2025


MATLAB R2023b - academic use
Editor - C:\Users\kjsce_comp63\Desktop\Kashish.M\ref2.m
while d<3
d=d+1;
disp(d);
end
Command Window
>> ref2
a is greater than b
1
2
3
4
5
1
2
3
>>

**Conclusion:-**

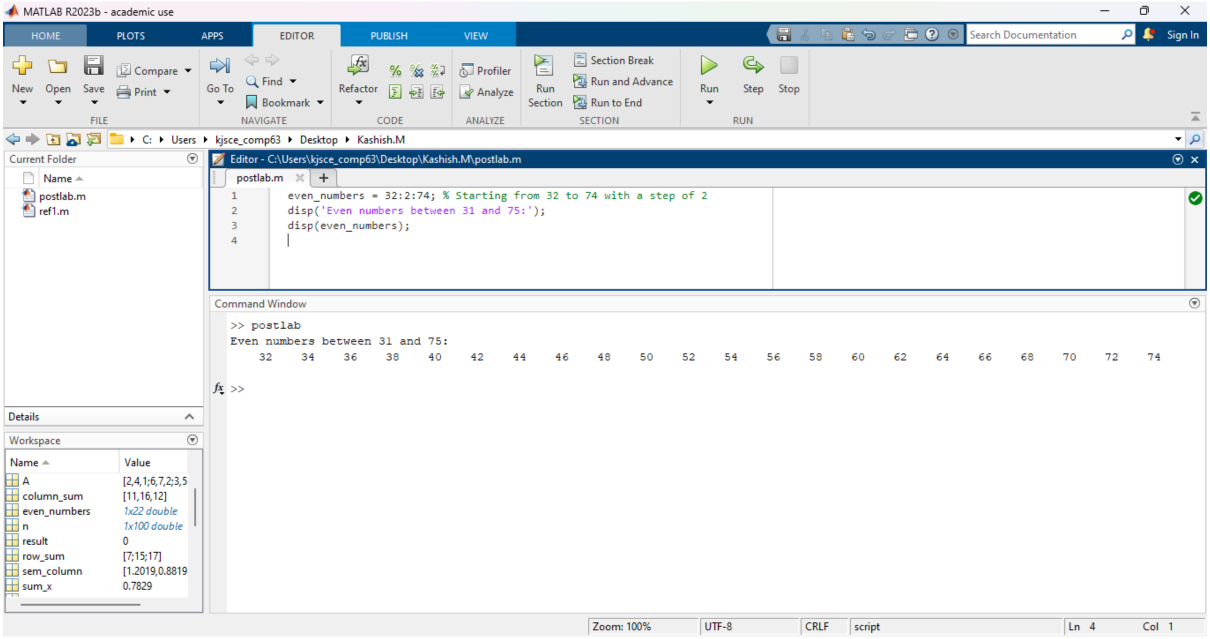
We learnt the basic syntax and functions used in Matlab.

**Date: 14/01/2024** **Signature of faculty in-charge**

**<u>Post Lab Questions</u>**

1. Create a vector of the even whole numbers between **31 and 75.**

```
even_numbers = 32:2:74; % Starting from 32 to 74 with a step of 2
disp('Even numbers between 31 and 75:');
disp(even_numbers);
```



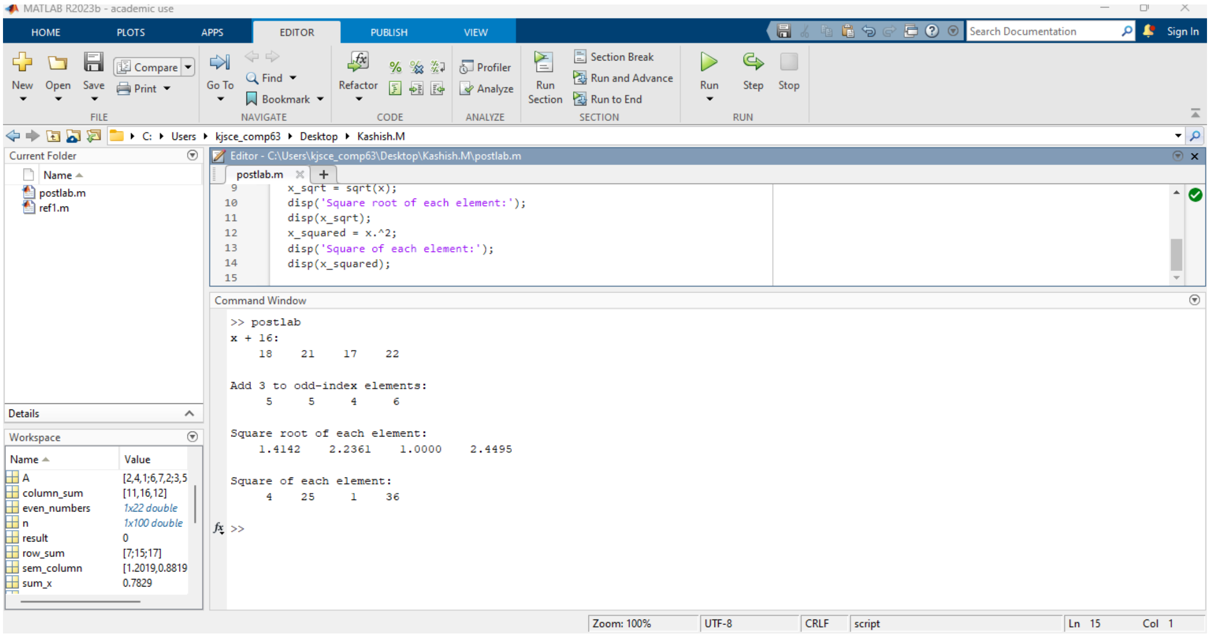

2. Let x = **[2 5 1 6].**

   a. Add 16 to each element

   b. Add 3 to just the odd-index elements
   c. Compute the square root of each element

   d. Compute the square of each element

```
x = [2, 5, 1, 6];
x_plus_16 = x + 16;
disp('x + 16:');
disp(x_plus_16);
x_odd_index_plus_3 = x;
x_odd_index_plus_3(1:2:end) = x_odd_index_plus_3(1:2:end) + 3;
disp('Add 3 to odd-index elements:');
disp(x_odd_index_plus_3);
x_sqrt = sqrt(x);
disp('Square root of each element:');
disp(x_sqrt);
x_squared = x.^2;
disp('Square of each element:');
disp(x_squared);
```



3. Let x = **[3 2 6 8]'** and y = **[4 1 3 5]'** (NB. x and y should be column vectors).

a. Add the sum of the elements in x to y

b. Raise each element of x to the power specified by the corresponding element in y.

c. Divide each element of y by the corresponding element in x

d. Multiply each element in x by the corresponding element in y, calling the result "z".

e. Add up the elements in z and assign the result to a variable called "w".

f. Compute x' *y - w and interpret the result

```
x = [3; 2; 6; 8];
y = [4; 1; 3; 5];
y_sum_added = y + sum(x);
disp('y + sum(x):');
disp(y_sum_added);
x_powered = x .^ y;
disp('x raised to the power of corresponding elements in y:');
disp(x_powered);
y_divided_by_x = y ./ x;
disp('y divided by corresponding elements in x:');
disp(y_divided_by_x);
z = x .* y;
disp('Element-wise multiplication of x and y (z):');
disp(z);
w = sum(z);
disp('Sum of elements in z (w):');
disp(w);
```

```
result = (x' * y) - w;
disp('Result of (x'' * y) - w:');
disp(result);
fprintf('Interpretation: This value indicates how much extra value is contributed by direct multiplication compared to their combined linear interaction.\n');
```

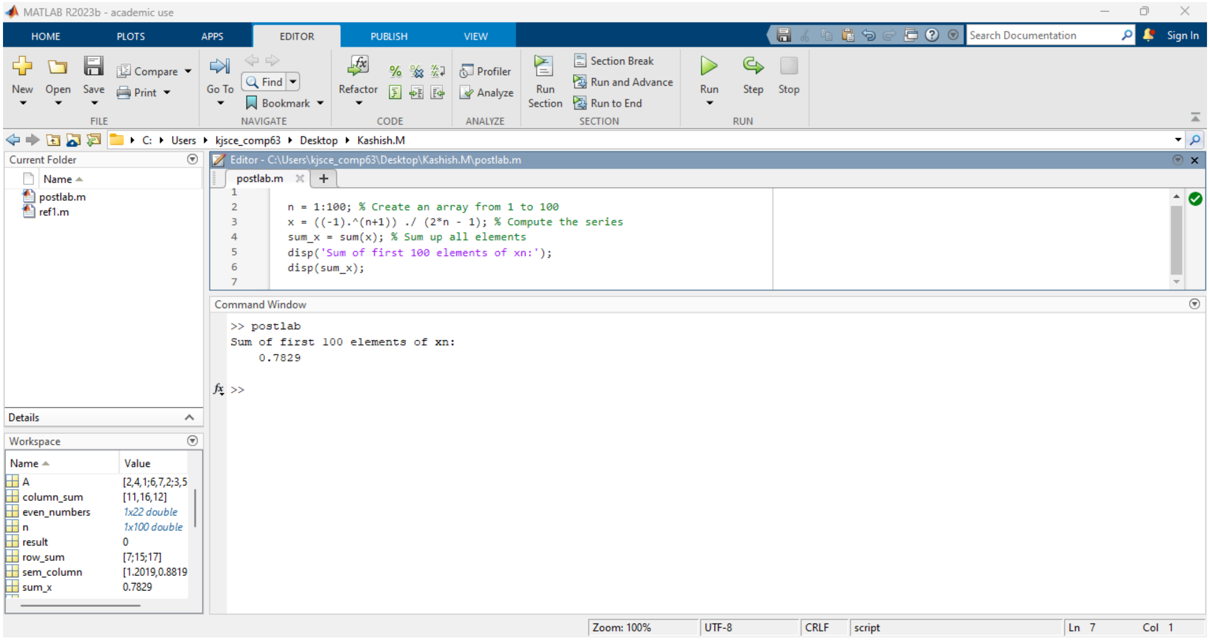
```
Command Window
>> postlab
y + sum(x):
    23
    20
    22
    24

x raised to the power of corresponding elements in y:
          81
           2
         216
       32768

y divided by corresponding elements in x:
    1.3333
    0.5000
    0.5000
    0.6250
```

```
Command Window
    0.5000
    0.5000
    0.6250

Element-wise multiplication of x and y (z):
    12
     2
    18
    40

Sum of elements in z (w):
    72

Result of (x' * y) - w:
     0

Interpretation: This value indicates how much extra value is contributed by direct multiplication compared to their combined linear inte
>>
```

4 Create a vector x with the elements, $\mathbf{x_n = (-1)^{n+1}/(2n-1)}$

Add up the elements of the version of this vector that has 100 elements.

```
n = 1:100; % Create an array from 1 to 100
x = ((-1).^(n+1)) ./ (2*n - 1); % Compute the series
sum_x = sum(x); % Sum up all elements
disp('Sum of first 100 elements of xn:');
disp(sum_x);
```




5. Given the array **A = [2 4 1; 6 7 2; 3 5 9],** provide the commands needed to

a. Assign the first row of A to a vector called x1

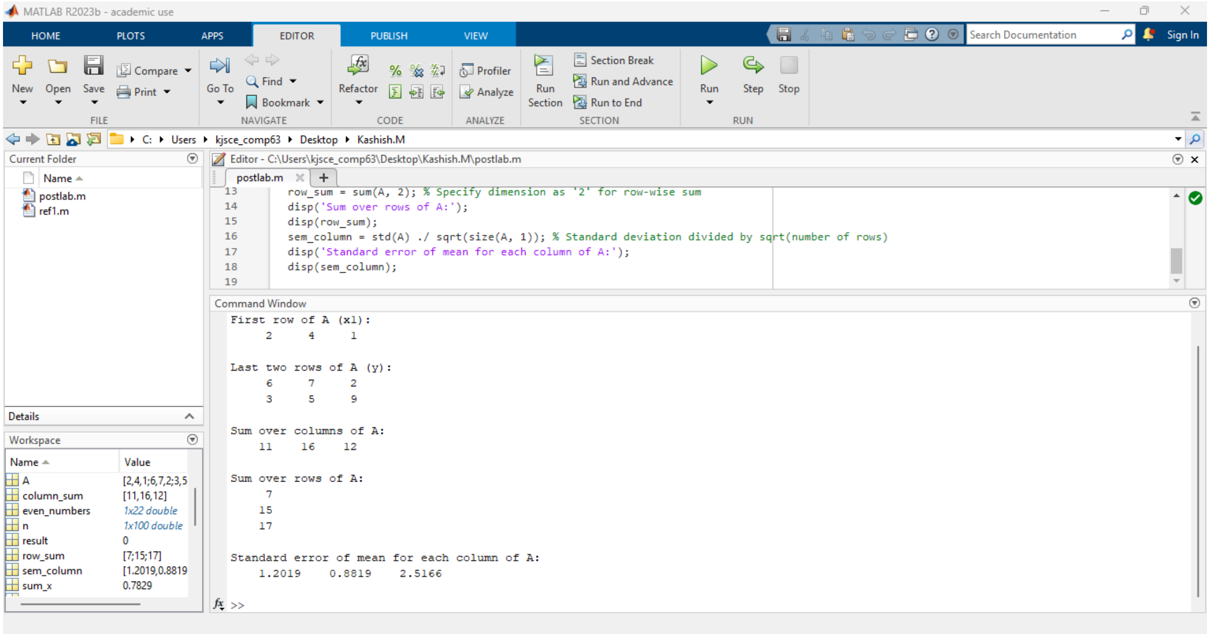
b. Assign the last 2 rows of A to an array called y

c. Compute the sum over the columns of A

d. Compute the sum over the rows of A

e. Compute the standard error of the mean of each column of A (NB. the standard error of the mean is defined as the standard deviation divided by the square root of the number of elements used to compute the mean.)

```
A = [2, 4, 1;
    6, 7, 2;
    3, 5, 9];
x1 = A(1, :);
disp('First row of A (x1):');
disp(x1);
y = A(2:end, :);
disp('Last two rows of A (y):');
disp(y);
column_sum = sum(A);
disp('Sum over columns of A:');
disp(column_sum);
row_sum = sum(A, 2); % Specify dimension as '2' for row-wise sum
disp('Sum over rows of A:');
disp(row_sum);
sem_column = std(A) ./ sqrt(size(A, 1)); % Standard deviation divided by
sqrt(number of rows)
disp('Standard error of mean for each column of A:');
disp(sem_column);
```


MATLAB R2023b - academic use
Editor - C:\Users\kjsce_comp63\Desktop\Kashish.M\postlab.m
row_sum = sum(A, 2); % Specify dimension as '2' for row-wise sum
disp('Sum over rows of A:');
disp(row_sum);
sem_column = std(A) ./ sqrt(size(A, 1)); % Standard deviation divided by sqrt(number of rows)
disp('Standard error of mean for each column of A:');
disp(sem_column);
Command Window
First row of A (x1):
2 4 1
Last two rows of A (y):
6 7 2
3 5 9
Sum over columns of A:
11 16 12
Sum over rows of A:
7
15
17
Standard error of mean for each column of A:
1.2019 0.8819 2.5166
Workspace
A [2,4,1;6,7,2;3,5
column_sum [11,16,12]
even_numbers 1x22 double
n 1x100 double
result 0
row_sum [7;15;17]
sem_column [1.2019,0.8819
sum_x 0.7829