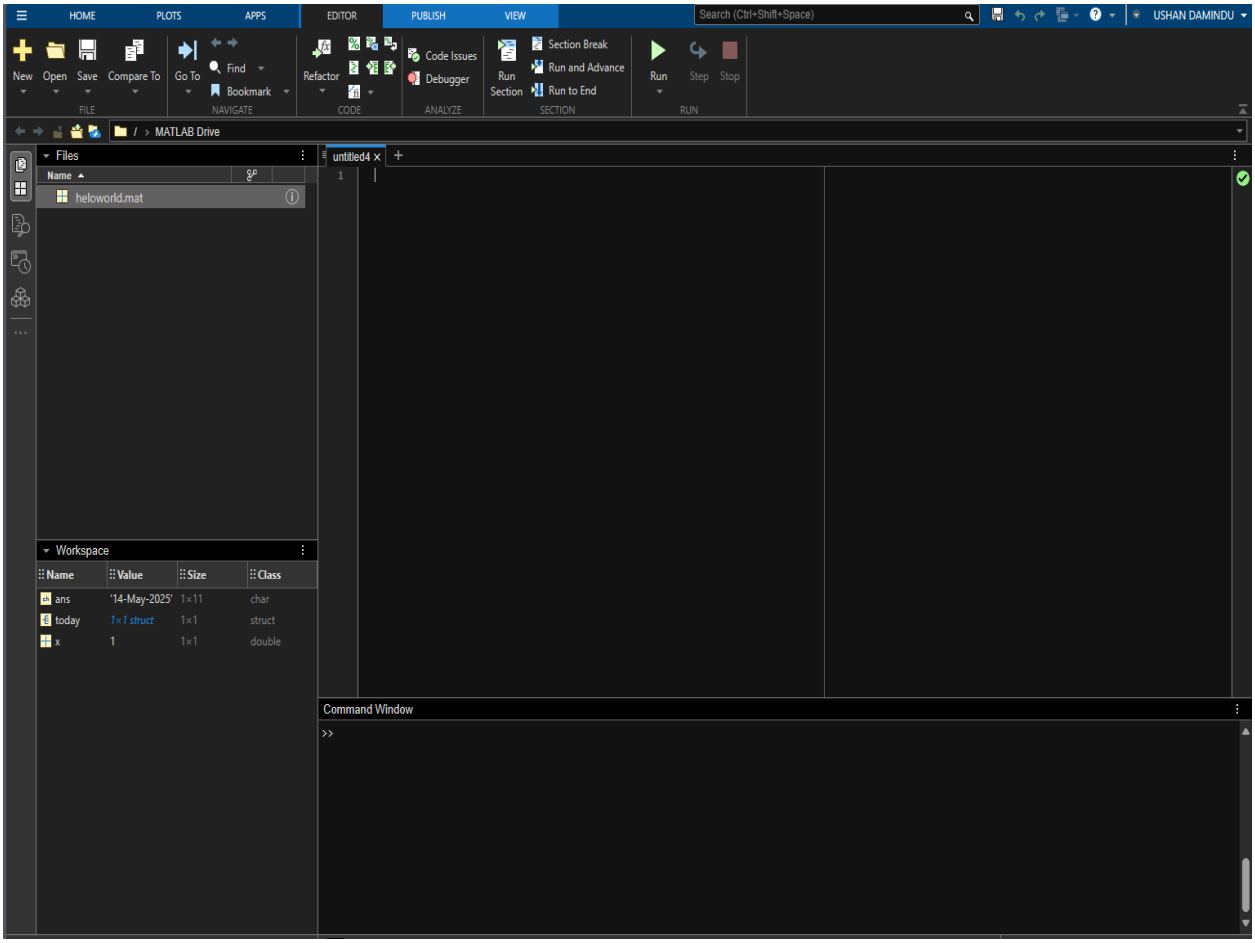
Introduction to MATLAB

Objective:

To provide hands-on practice in MATLAB, covering basic commands, matrix operations, image processing, and graph plotting, giving a strong foundation for further image processing work.

MATLAB Windows Overview: Command Window, Workspace and Editor.

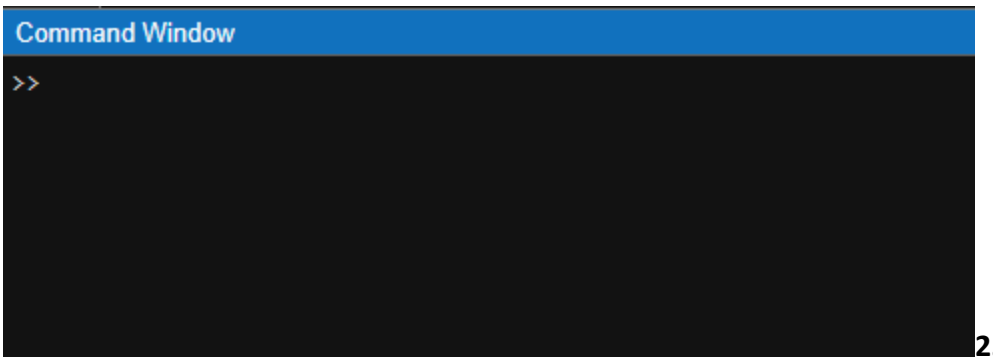


MATLAB Home:- using this link you can work online without install MATLAB (only for 20 hours per month)

Command Window

This is where you type commands and see results immediately. For example:

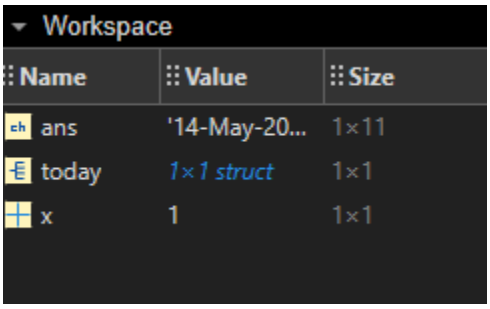
This will display **Hello World!** in the Command Window.



2. Workspace

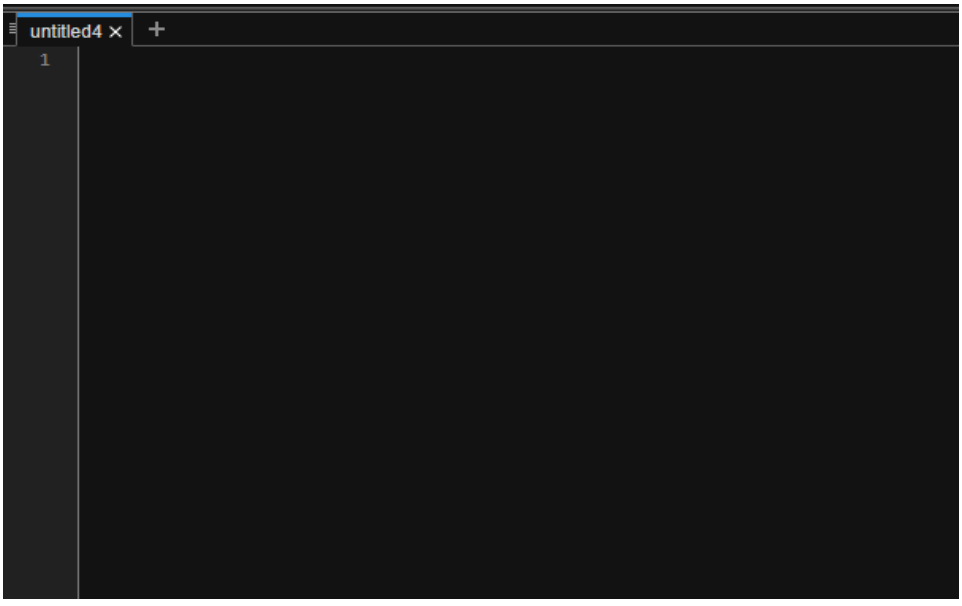
The **Workspace** stores all the variables you've created during the session. You can see them by typing:

Think of it as a **storage area** where MATLAB keeps track of your work.



Editor

The **Editor** allows you to write and save scripts (MATLAB programs). Instead of typing commands one by one,



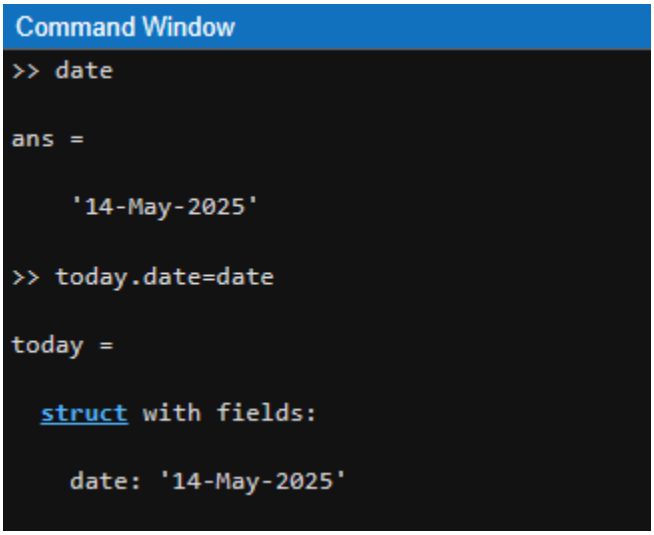
you can write a full script and run it all at once.

1.Display

- Run a simple command like **disp('Hello World!')** in the Command Window.

2. Date Command

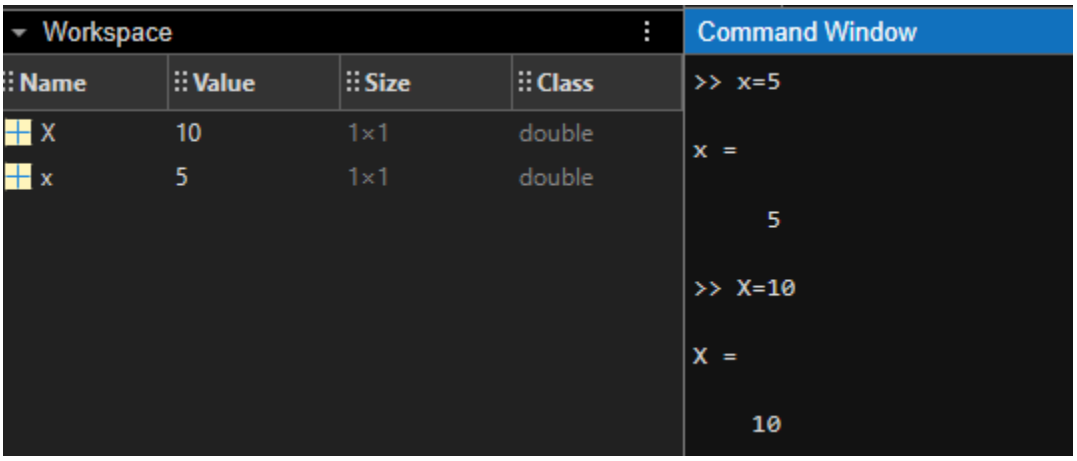
- Use the **"date"** command to display today's date.



This command creates a **structure** named Today, with a **field** named date, and assigns it the value '14-May-2025'.

3. Defining Variable and Case Sensitivity.

- Define variables x = 5, Redefine a variable using uppercase (X = 10), then display both **x** and **X** to observe case sensitivity.



4. Usage of Semicolon.

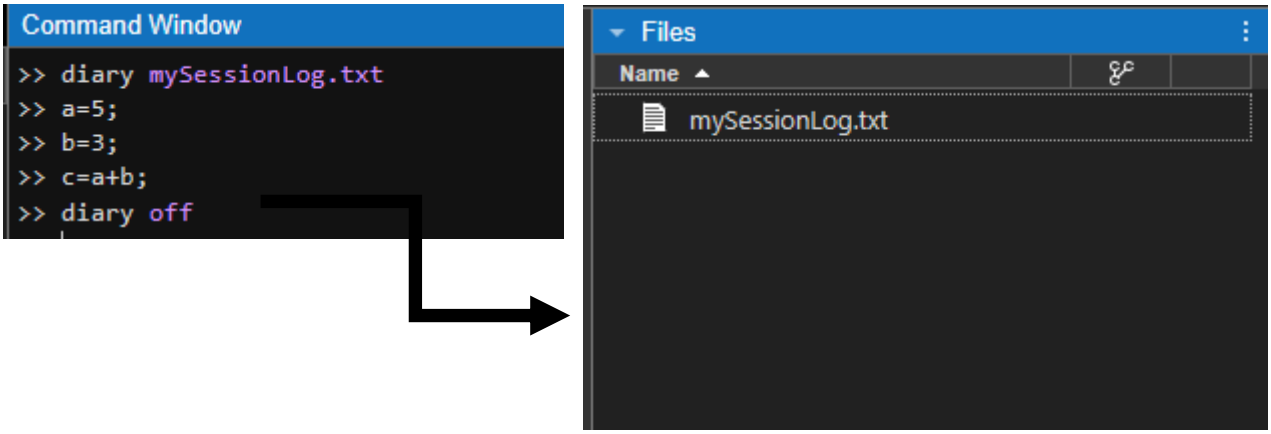
- Define z = 25; and observe that no output is displayed.

- Remove the semicolon from the end of the statement and re-run to see the difference.

```
Command Window
>> z=25;
>> z=25

z =

    25

>>
```

5. Diary Command: It is used to record command window session into a text file.

- Run diary command and save the text file as **mySessionLog.txt**



6. Usage of who, whos, clc and clear Commands.

- Create multiple variables (a = 1, b = 2, c = 3).

- Use who and whos to inspect the workspace and note the differences.

```
Command Window
>> a=1; b=2; c=3;
>> who

Your variables are:

a  b  c

>> whos
  Name      Size            Bytes  Class     Attributes

  a         1x1                 8  double
  b         1x1                 8  double
  c         1x1                 8  double
```

- (**clc)** Clears the command window**, (clear)** Clears all variables

7. Arithmetic Operations.

- Perform arithmetic operations a = 15.5 and b = 7.2

```
Command Window
>> a=15.5;b=7.2;
>> addResult=a+b

addResult =

   22.7000

>> subResult=a-b

subResult =

    8.3000

>> mulResult=a*b

mulResult =

  111.6000

>> divResult=a/b

divResult =

    2.1528
```

-Compute power and exponential

```
>> a=15.5;b=7.2;
>> powerResult=a^b

powerResult =

   3.7187e+08

>> expResult=exp(a)

expResult =

   5.3897e+06
```

8. Creating and Accessing Vectors.

- Create a row vector and a column vector called V1 and V2 respectively. Call the 2nd element of each vector.

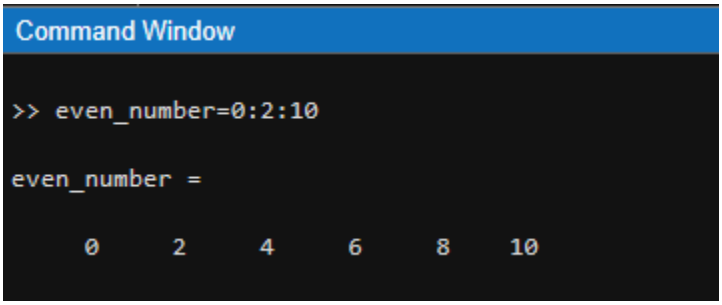
```
Command Window
>> v1=[1,2,3,4]

v1 =

     1     2     3     4

>> v2=[5;6;7;8]


v2 =

     5
     6
     7
     8

>> v1_vectorElement=v1(2)

v1_vectorElement =

     2

>> v2_vectorElement=v2(2)

v2_vectorElement =

     6
```

- Create row vector and column vector using zeros command.

```
Command Window
>> row_vector=zeros(1,5)

row_vector =

     0     0     0     0     0

>> col_vector=zeros(5,1)

col_vector =

     0
     0
     0
     0
     0
```

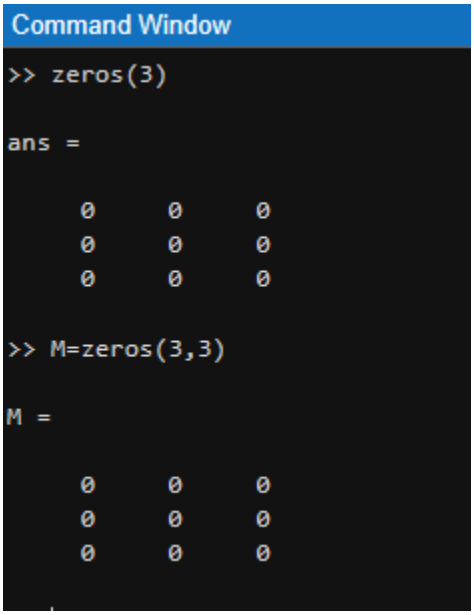
- Print 0 to 10 numbers.

- Print [0 2 4 6 8 10]



9. Creating Matrices.

- Create 3 X 3 matrix.



- Create matrix S = $\begin{matrix} 1 & 2 & 3 & 4 \\ -5 & 0 & 5 & 10 \\ 10 & 9 & 8 & 7 \end{matrix}$

```
Command Window
>> s=[1,2,3,4;-5,0,5,10;10,9,8,7]

s =

     1     2     3     4
    -5     0     5    10
    10     9     8     7

>>
```

-Create a random matrix M = (3 X 3) and access the element M(3,2) and display.

```
Command Window
>> M=rand(3)

M =

    0.8147    0.9134    0.2785
    0.9058    0.6324    0.5469
    0.1270    0.0975    0.9575

>> M(2,3)

ans =

    0.5469
```

- Add value 2 with each element in the matrix M.

```
>> M+2

ans =

    2.8147    2.9134    2.2785
    2.9058    2.6324    2.5469
    2.1270    2.0975    2.9575
```

10. Reading, Displaying and Writing Images.

Fist download image using this command or you can get any image and paste in to current directory

```
url =
"https://th.bing.com/th/id/R.2209785b9f654b33964f56213fccf222?rik=JmMpfQtnjhOadw&pid=ImgRaw
&r=0";

filename = "image.jpg";

websave(filename, url);websave(filename, url);
```

- Read an image file (image.jpg) and display it.

- Save the displayed image to a new file.

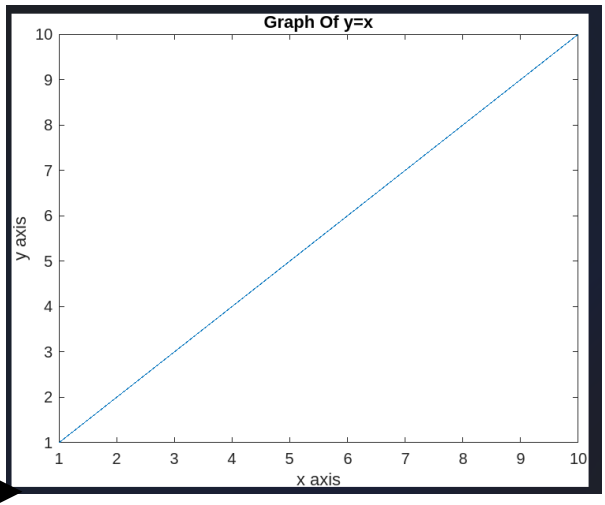
```
Command Window
>> img=imread("image.jpg");
>> imshow(img)
>> imwrite(img,"new_image.jpg");
>>
```

In this window you can see image

11. Plotting Graphs.

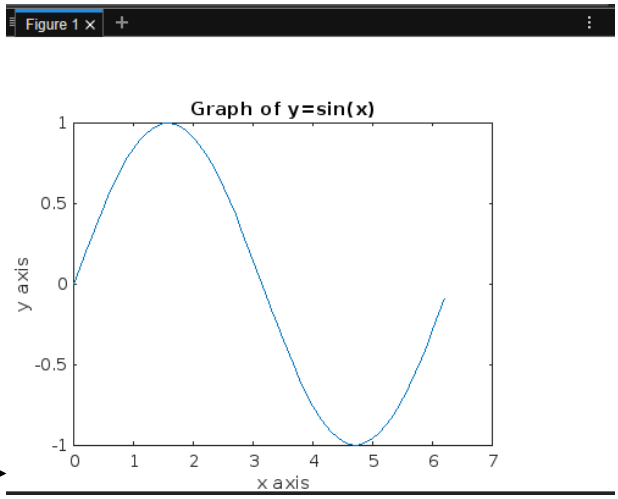
- Plot graph y = x and x = (1,2,3…….10).

```
Command Window
>> x = 1:10;
>> y = x;
>> plot(x, y); % Plot first
>> xlabel("x axis"); % Label x-axis
>> ylabel("y axis"); % Label y-axis
>> title("Graph Of y=x"); % Set title
>>
```



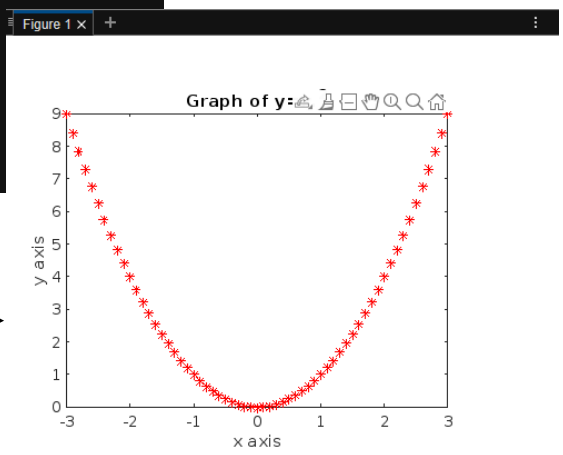
- Create an x vector (x = 0:0.1:2*pi) and a y vector (y = sin(x)), then plot the graph.

```
Command Window
>> x=0:0.1:2*pi;
>> y=sin(x);
>> plot(x,y)
>> xlabel("x axis");
>> ylabel("y axis");
>> title("Graph of y=sin(x)");
>>
```



- Plot the graph y = x^2 and x in the range of (-3,3) increment 0.1

```
Command Window
>> x=-3:0.1:3;
>> y = x.^2; % Use .^ for element-wise exponentiation
>> plot(x,y,"r*")
>> xlabel("x axis");
>> ylabel("y axis");
>> title("Graph of y=x.^2");
>>
```



12. Loops and Conditional Statements.

- Write a loop to print numbers from 1 to 10.

```
for i=i:10
    disp(i);
end
```

```
0
1
2
3
4
5
6
7
8
9
10
```

- Write an if-else condition to check if a number is positive or negative.

```
num=-5;
if num>0
    disp('positive');
else
    disp("negative");
end
```

```
Command Window
negative
>>
```

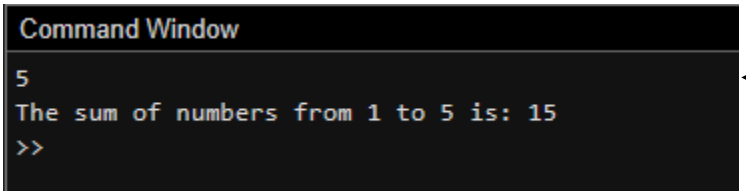
- Add all the even numbers from 0 to 100 using for loop.

```
sum = 0;  % Initialize sum

% Loop through even numbers from 0 to 100, incrementing by 2
for i = 0:2:100
    sum = sum + i;  % Add each even number to sum
end

% Display the result with corrected message
disp(["The sum of the even numbers: ", num2str(sum)])
```

```
Command Window
   "The sum of the even numbers: "    "2550"
```

- Create a script that to find sum of “n” numbers using for loop.

```
% Clear workspace and command window
clear; clc;

% Get user input for n
n = input('Enter the value of n: ');

% Initialize sum variable
sum = 0;

% Loop from 1 to n and add values to sum
for i = 1:n
    sum = sum + i;
end

% Display the final sum with a formatted output
disp(['The sum of numbers from 1 to ', num2str(n), ' is: ', num2str(sum)]);
```



```
Command Window
5
The sum of numbers from 1 to 5 is: 15
>>
```

Summary

Command	Use / Description
clc	Clears the **Command Window**
clear	Clears all **variables** from the workspace
disp('Hello World!')	Displays text in the **Command Window**
date	Displays the **current date**
x = 5; X = 10	Defines variables and shows **case sensitivity** (x ≠ X)
; (semicolon)	**Suppresses output** in Command Window
diary filename.txt	Records session in a **text file**
who	Lists **variable names** currently in the workspace
whos	Lists variables with **details** (size, bytes, class, etc.)
a + b, a * b	Performs **arithmetic operations**
a^b, exp(a)	**Power** and **exponential** operations
V(2)	Accesses the **2nd element** of a vector
zeros(1, 5)	Creates a **row vector** with five zeros
zeros(5, 1)	Creates a **column vector** with five zeros
0:10	Creates a vector from 0 to 10 in **steps of 1**
0:2:10	Creates vector: [0 2 4 6 8 10]
M = rand(3,3)	Creates a **3x3 random matrix**
M(2,3)	Accesses the element at **2nd row, 3rd column** of matrix M
M + 2	Adds 2 to **each element** of matrix M
websave(filename, url)	**Downloads** image from the internet and saves it
imread('image.jpg')	**Reads** an image file
imshow(img)	**Displays** the image in a figure window
imwrite(img, 'out.jpg')	**Writes** the image to a file
plot(x, y)	Plots a **2D graph**
x = 0:0.1:2*pi	Creates a **vector from 0 to 2π** with increment of 0.1
y = sin(x)	Calculates **sine** of each element in x
for i = 1:10	Starts a **for loop** from 1 to 10
if x > 0 ... else ...	**Conditional statement**: checks if value is positive or negative
sum = 0; for i=1:n ...	Sums first n numbers using a **for loop**