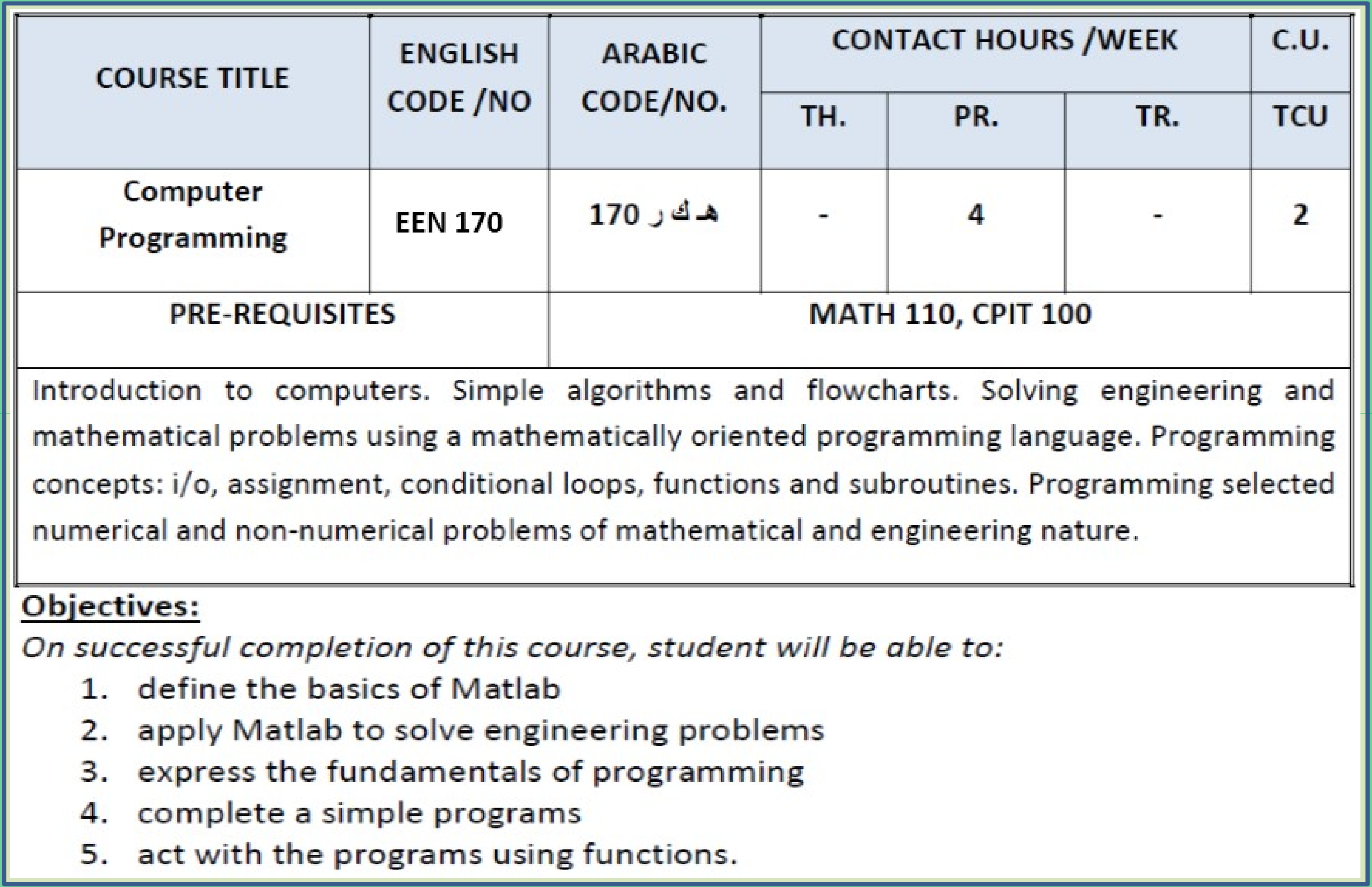
Computer Programming

Course code : EEN 170

Comm. Hours: 2hr theory & 2hr practical

Computer Programming EEN170

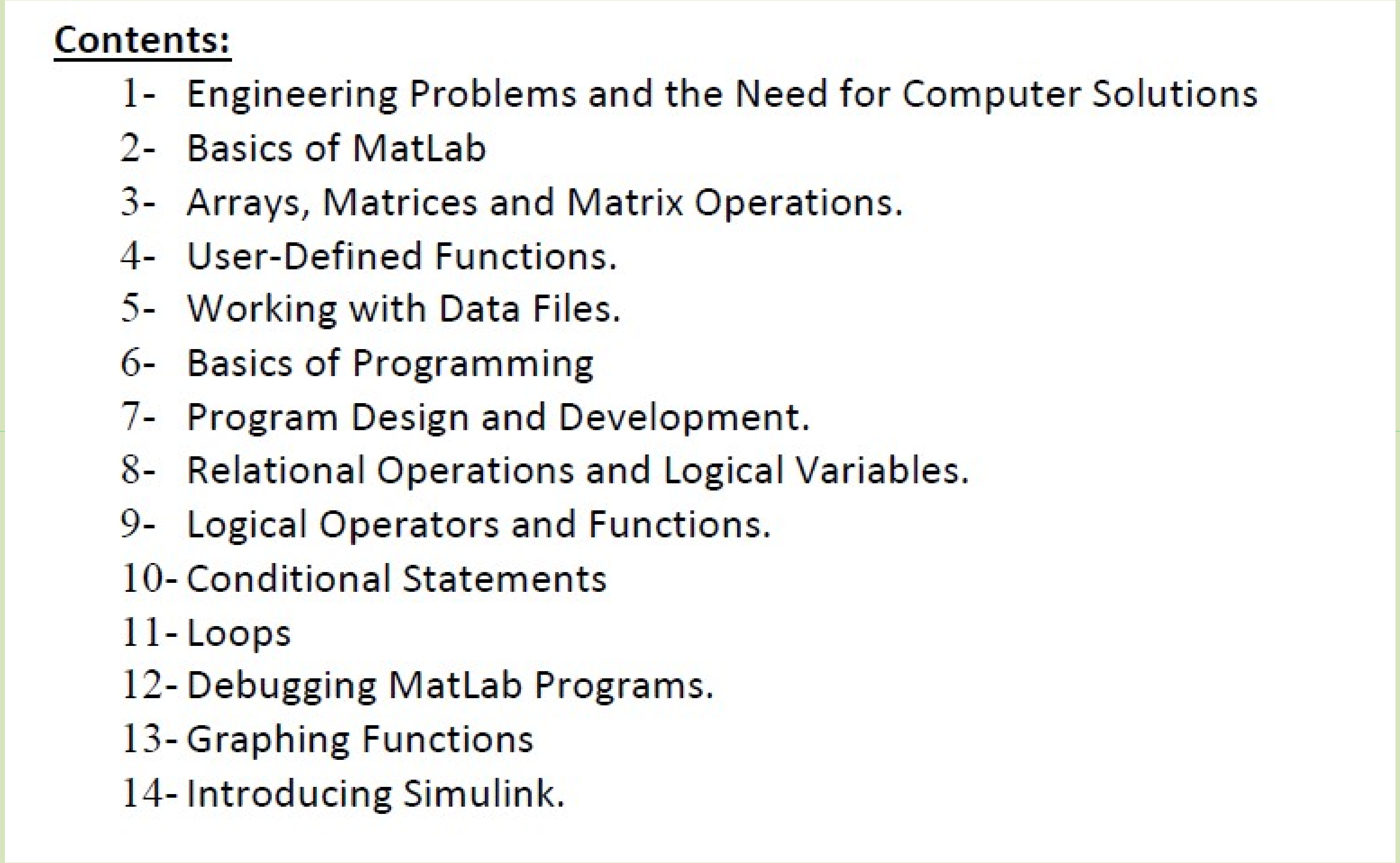


COURSE TITLE	ENGLISH CODE /NO	ARABIC CODE/NO.	CONTACT HOURS /WEEK			C.U.
			TH.	PR.	TR.	TCU
Computer Programming	EEN 170	هـك ر 170	-	4	-	2
PRE-REQUISITES		MATH 110, CPIT 100				
Introduction to computers. Simple algorithms and flowcharts. Solving engineering and mathematical problems using a mathematically oriented programming language. Programming concepts: i/o, assignment, conditional loops, functions and subroutines. Programming selected numerical and non-numerical problems of mathematical and engineering nature.						

Objectives:

On successful completion of this course, student will be able to:

1. define the basics of Matlab
2. apply Matlab to solve engineering problems
3. express the fundamentals of programming
4. complete a simple programs
5. act with the programs using functions.



Contents:

1- Engineering Problems and the Need for Computer Solutions
2- Basics of MatLab
3- Arrays, Matrices and Matrix Operations.
4- User-Defined Functions.
5- Working with Data Files.
6- Basics of Programming
7- Program Design and Development.
8- Relational Operations and Logical Variables.
9- Logical Operators and Functions.
10- Conditional Statements
11- Loops
12- Debugging MatLab Programs.
13- Graphing Functions
14- Introducing Simulink.

WHAT IS MATLAB?

MATLAB
is a powerful computing system for handling the calculations involved in scientific and engineering problems.

WHAT MEANS BY **MATLAB**?

The name MATLAB stands for **MAT**rix **LAB**oratory, because the system was designed to make matrix computations particularly easy.

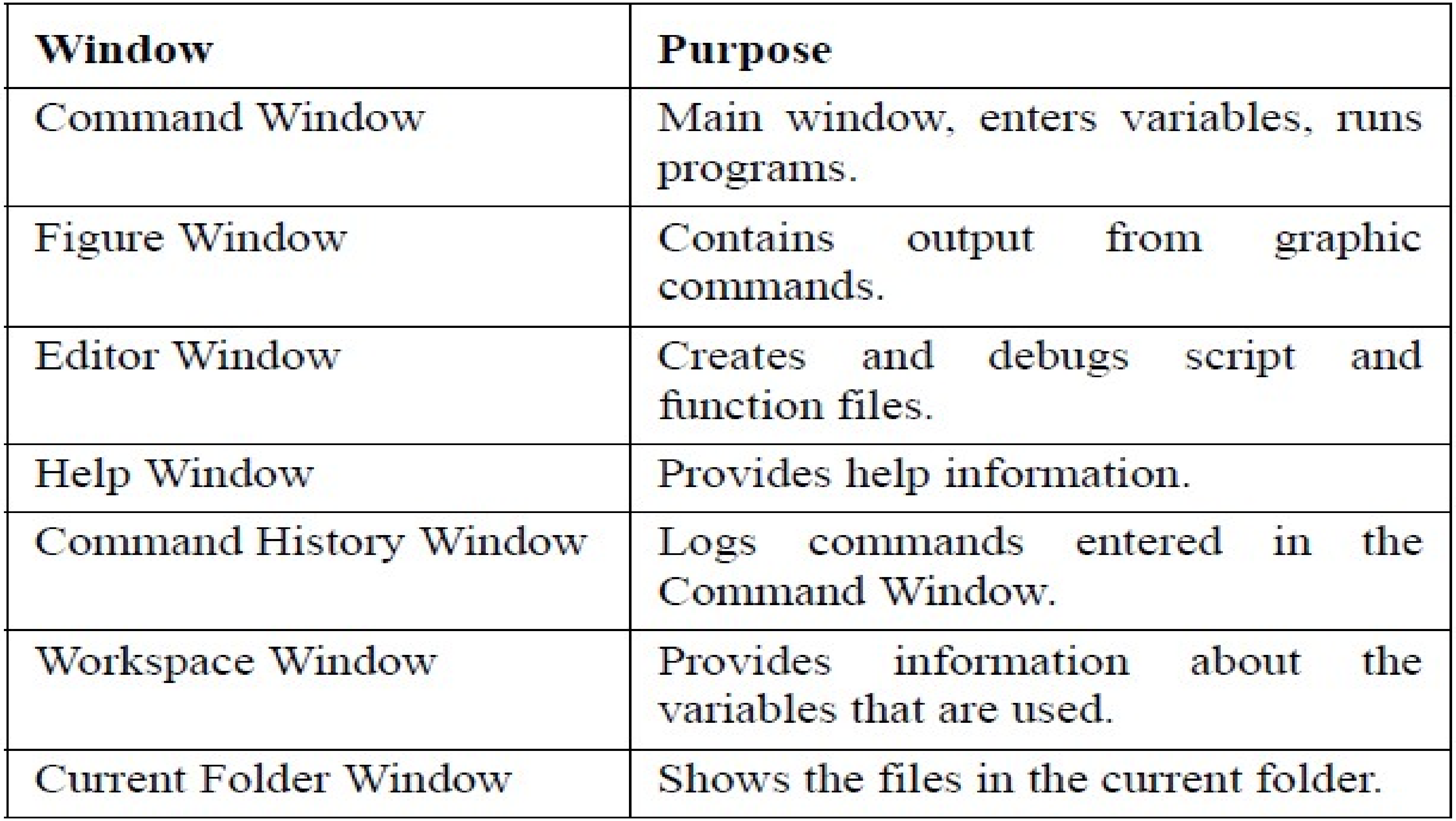
for successful MATLAB, There are two essential requirements programming:

- ➤ You need to learn the *exact rules for writing MATLAB statements.*

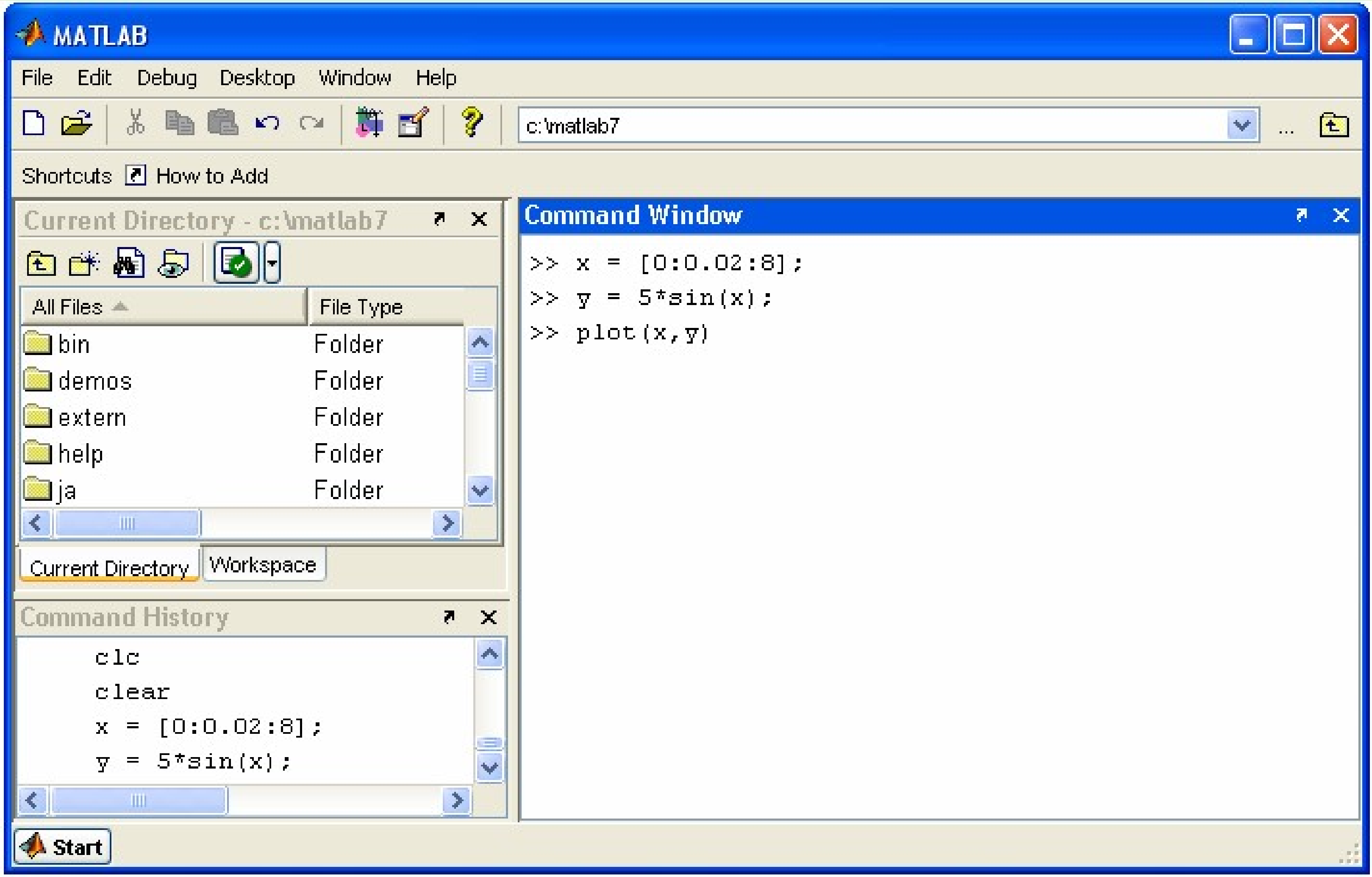
- ➤ You need to develop a logical plan of attack for solving particular problems.

- ➢If you give MATLAB a WEAK instruction, you will get an **UNACCURATE** result.

THE MATLAB DESKTOP



Window	Purpose
Command Window	Main window, enters variables, runs programs.
Figure Window	Contains output from graphic commands.
Editor Window	Creates and debugs script and function files.
Help Window	Provides help information.
Command History Window	Logs commands entered in the Command Window.
Workspace Window	Provides information about the variables that are used.
Current Folder Window	Shows the files in the current folder.



Command History:

```
clc
clear
x = [0:0.02:8];
y = 5*sin(x);
```

Command Window:

```
>> x = [0:0.02:8];
>> y = 5*sin(x);
>> plot(x,y)
```

MATLAB has the following basic window components:

Work-space

Displays all the defined variables.

Command Window

To execute commands in the MATLAB environment

Command History

Displays record of the commands used

- **To start MATLAB from Windows,**
  double-click the MATLAB icon on your Windows desktop.

- When MATLAB starts, the MATLAB desktop opens as shown in the above Figure .

- The window in the desktop that concerns us is the Command Window, where the special prompt (>>)appears.

- This prompt means that MATLAB is waiting for a command.

You can quit MATLAB at any time with one of the following:

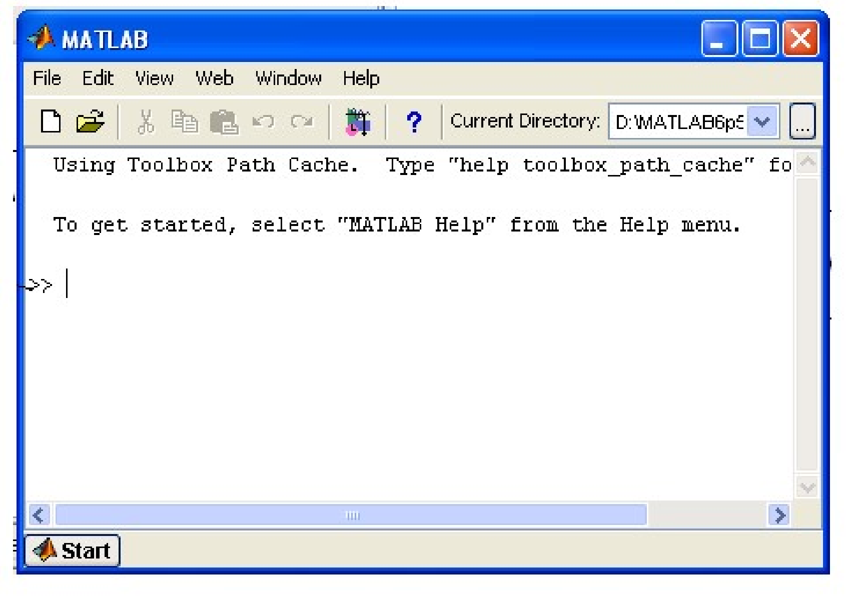
- Select Exit MATLAB from the desktop File menu.
- Enter quit or exit at the Command Window prompt.

WORKING IN THE COMMAND WINDOW

- ❖The command window allows you to interact with MATLAB just as if you type things in a calculator

- ❖Cut and paste operations ease the repetition of tasks

- ❖Use ‘up-arrow’ key to repeat commands (command history)

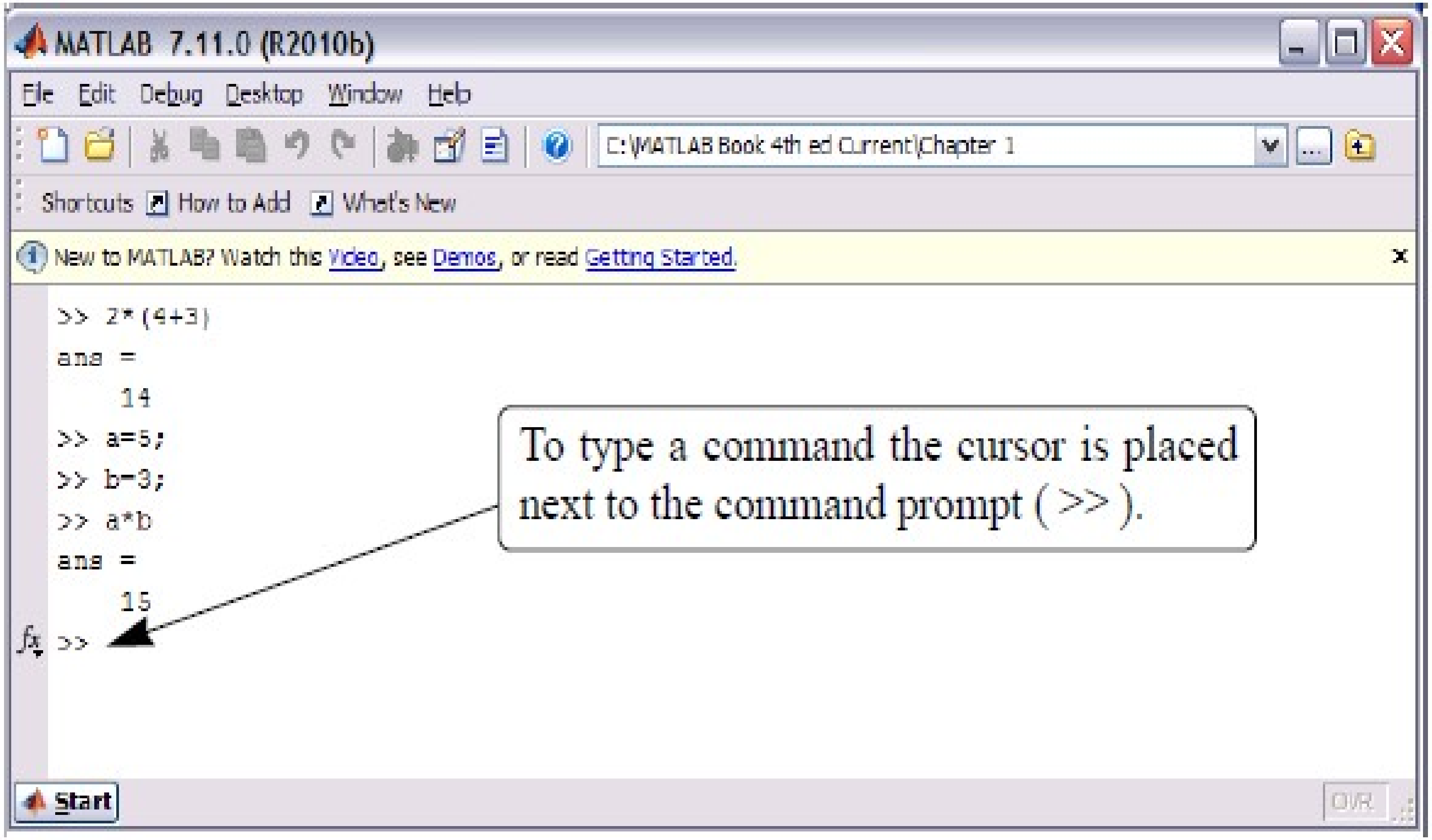

MATLAB
File Edit View Web Window Help
Current Directory: D:\MATLAB6p5
Start

```
Using Toolbox Path Cache.  Type "help toolbox_path_cache" fo

To get started, select "MATLAB Help" from the Help menu.

>> |
```

Command Window


MATLAB 7.11.0 (R2010b)
File Edit Debug Desktop Window Help
C:\MATLAB Book 4th ed Current\Chapter 1
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Start
OVR

```
>> 2*(4+3)
ans =
    14
>> a=5;
>> b=3;
>> a*b
ans =
    15
>>
```

To type a command the cursor is placed next to the command prompt (>>).

Notes for working in the Command Window:

- **To type a command the cursor must be placed next to the command prompt (>>).**

- **Once a command is typed and the Enter key is pressed, the command is executed.** However, only the last command is executed.

- Everything executed previously (that might be still displayed) is unchanged.

Notes for working in the Command Window (cont.)

- **Several commands can be typed in the same line. This is done by typing a comma** between the commands.

- When the **Enter key is pressed the commands are Executed** in order from left to right.

- **It is not possible to go back to a previous line that is displayed in the Command** Window, make a correction, and then re-execute the command. of the command is then typed in the new line. The command can continue

Notes for working in the Command Window (cont.)

- **A previously typed command can be recalled to the command prompt with the up-arrow** key (↑). When the command is displayed at the command prompt, it can be modified if needed and then executed. The down-arrow key (↓) can be used to move down the list of previously typed commands.
- **If a command is too long to fit in one line, it can be continued to the next line by** typing three periods … (called an ellipsis) and pressing the **Enter key.**
- **The continuation** of the command is then typed in the new line. The command can continue line after line up to a total of 4,096 characters

The semicolon (;):

When a command is typed and the **Enter key is** pressed, the command is executed. Any output is displayed If a semicolon (;) is typed at the end of a command the output of the command is not displayed.

Typing a semicolon is useful when there is no need to display the result

Typing %:

When the symbol % (percent) is typed at the beginning of a line, the line is designated as a comment.

This means that when the **Enter key is pressed the line is not** executed.

The clc command:

The clc command (type clc and press **Enter) clears the Command Window**.

The command does not change anything that was done before.

For example, if some variables were defined previously , they still exist and can be used.

The up-arrow key can also be used to recall commands that were typed before

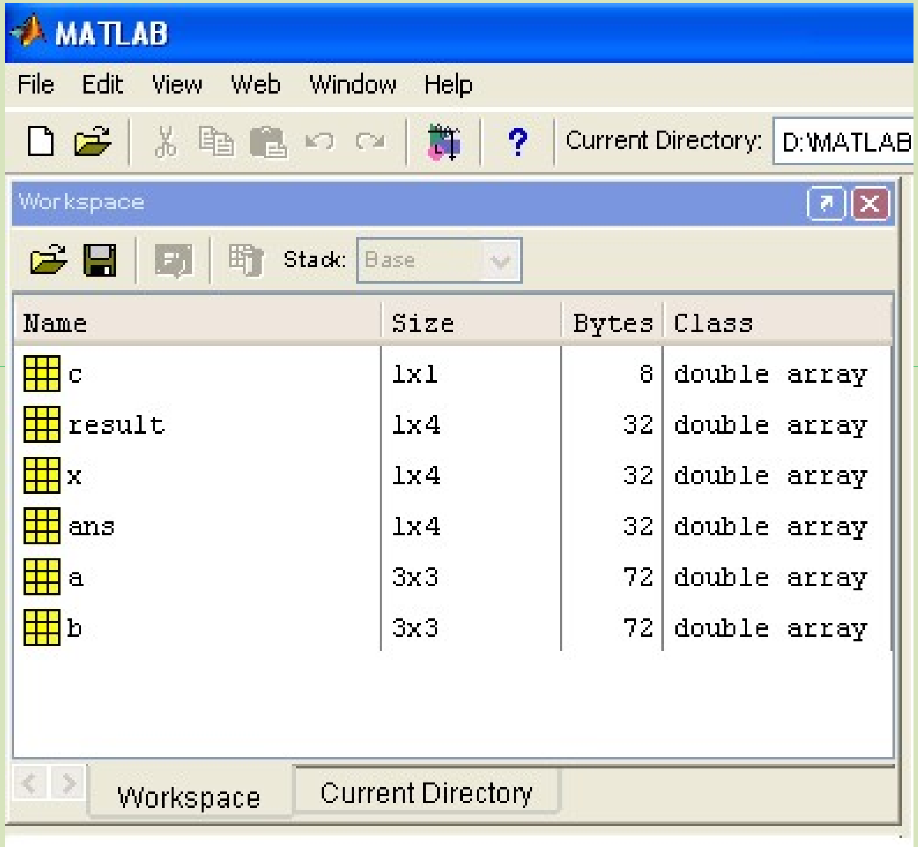
The Command History Window:

- The Command History lists the commands that have been entered in the Command Window. This includes commands from previous sessions.
- A command in the Command History can be used again in the Command Window . By double-clicking on the command, the command is reentered in the Command Window and executed.
- It is also possible to drag the command to the Command Window, make changes if needed, and then execute it.
- The list in Command History Window can be cleared by selecting the lines to be deleted and then selecting **Delete Selection from the Edit menu (or right-click the mouse** when the lines are selected and then choose **Delete Selection in the menu that** opens).

Matlab Workspace

- Workspace of Matlab shows all the defined variables along with their “order”,
- memory space they occupy and
- the class to which they belong

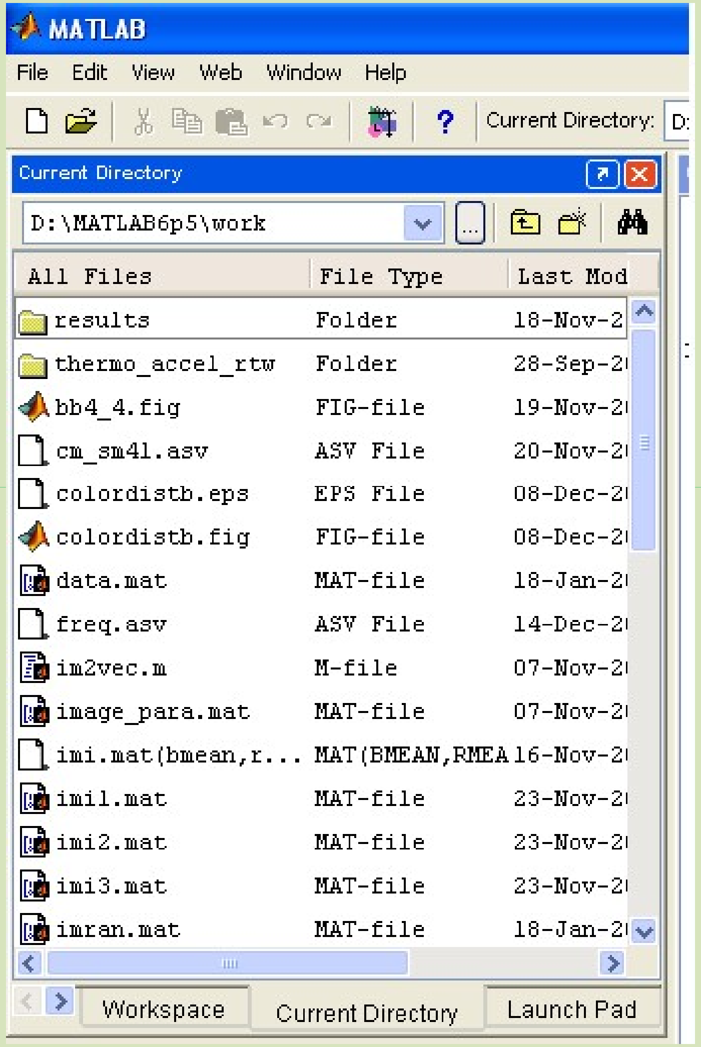
Beside fig shows a few such variables



Name	Size	Bytes	Class
c	1x1	8	double array
result	1x4	32	double array
x	1x4	32	double array
ans	1x4	32	double array
a	3x3	72	double array
b	3x3	72	double array

Current Directory

- ➢Provides quick access to all files available in your Path
- ➢Provides a brief description (when files are commented out) of each M-file



D:\MATLAB6p5\work

All Files	File Type	Last Mod
results	Folder	18-Nov-2
thermo_accel_rtw	Folder	28-Sep-2
bb4_4.fig	FIG-file	19-Nov-2
cm_sm41.asv	ASV File	20-Nov-2
colordistb.eps	EPS File	08-Dec-2
colordistb.fig	FIG-file	08-Dec-2
data.mat	MAT-file	18-Jan-2
freq.asv	ASV File	14-Dec-2
im2vec.m	M-file	07-Nov-2
image_para.mat	MAT-file	07-Nov-2
imi.mat(bmean,r...	MAT(BMEAN,RMEA	16-Nov-2
imi1.mat	MAT-file	23-Nov-2
imi2.mat	MAT-file	23-Nov-2
imi3.mat	MAT-file	23-Nov-2
imran.mat	MAT-file	18-Jan-2

ARITHMETIC OPERATIONS WITH SCALARS

The symbols of arithmetic operations are:

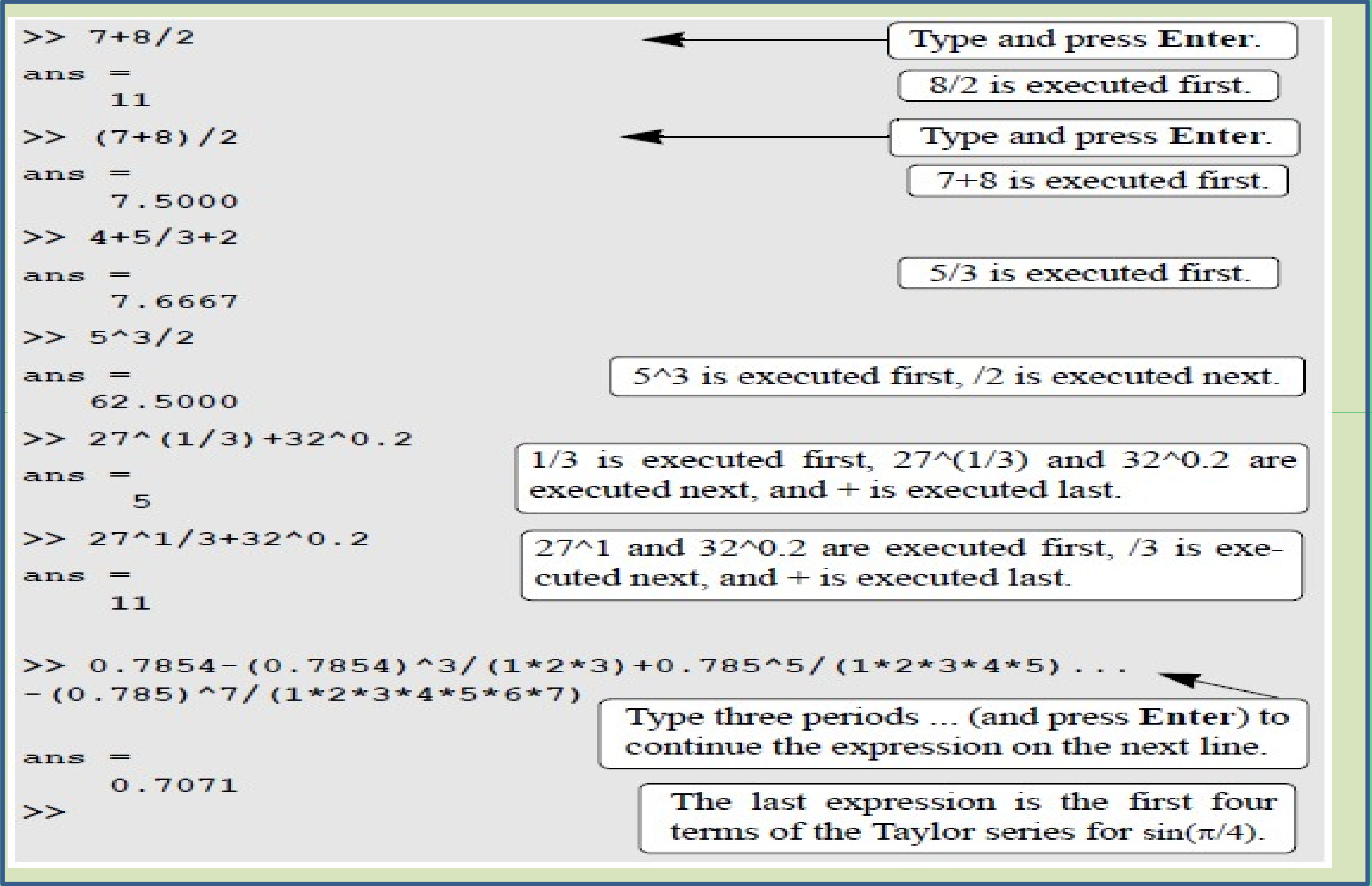
Operation	**Symbol**	**Example**
Addition	+	5 + 3
Subtraction	–	5 – 3
Multiplication	*	5 * 3
Right division	/	5 / 3
Left division	\	5 \ 3 = 3 / 5
Exponentiation	^	5 ^ 3 (means 53 = 125)

Order of Precedence

MATLAB executes the calculations according to the order of precedence displayed below.

Precedence	**Mathematical Operation**
First	Parentheses. For nested parentheses, the innermost are executed first.
Second	Exponentiation.
Third	Multiplication, division (equal precedence).
Fourth	Addition and subtraction.

If two or more operations have the same precedence, the expression is executed from left to right.



```
>> 7+8/2
ans =
    11
>> (7+8)/2
ans =
    7.5000
>> 4+5/3+2
ans =
    7.6667
>> 5^3/2
ans =
   62.5000
>> 27^(1/3)+32^0.2
ans =
     5
>> 27^1/3+32^0.2
ans =
    11

>> 0.7854-(0.7854)^3/(1*2*3)+0.785^5/(1*2*3*4*5)...
-(0.785)^7/(1*2*3*4*5*6*7)

ans =
    0.7071
>>
```

Type and press **Enter**.

8/2 is executed first.

Type and press **Enter**.

7+8 is executed first.

5/3 is executed first.

5^3 is executed first, /2 is executed next.

1/3 is executed first, 27^(1/3) and 32^0.2 are executed next, and + is executed last.

27^1 and 32^0.2 are executed first, /3 is executed next, and + is executed last.

Type three periods ... (and press **Enter**) to continue the expression on the next line.

The last expression is the first four terms of the Taylor series for sin(π/4).

MATH BUILT-IN FUNCTIONS

- In addition to basic arithmetic operations, expressions in MATLAB can include functions.

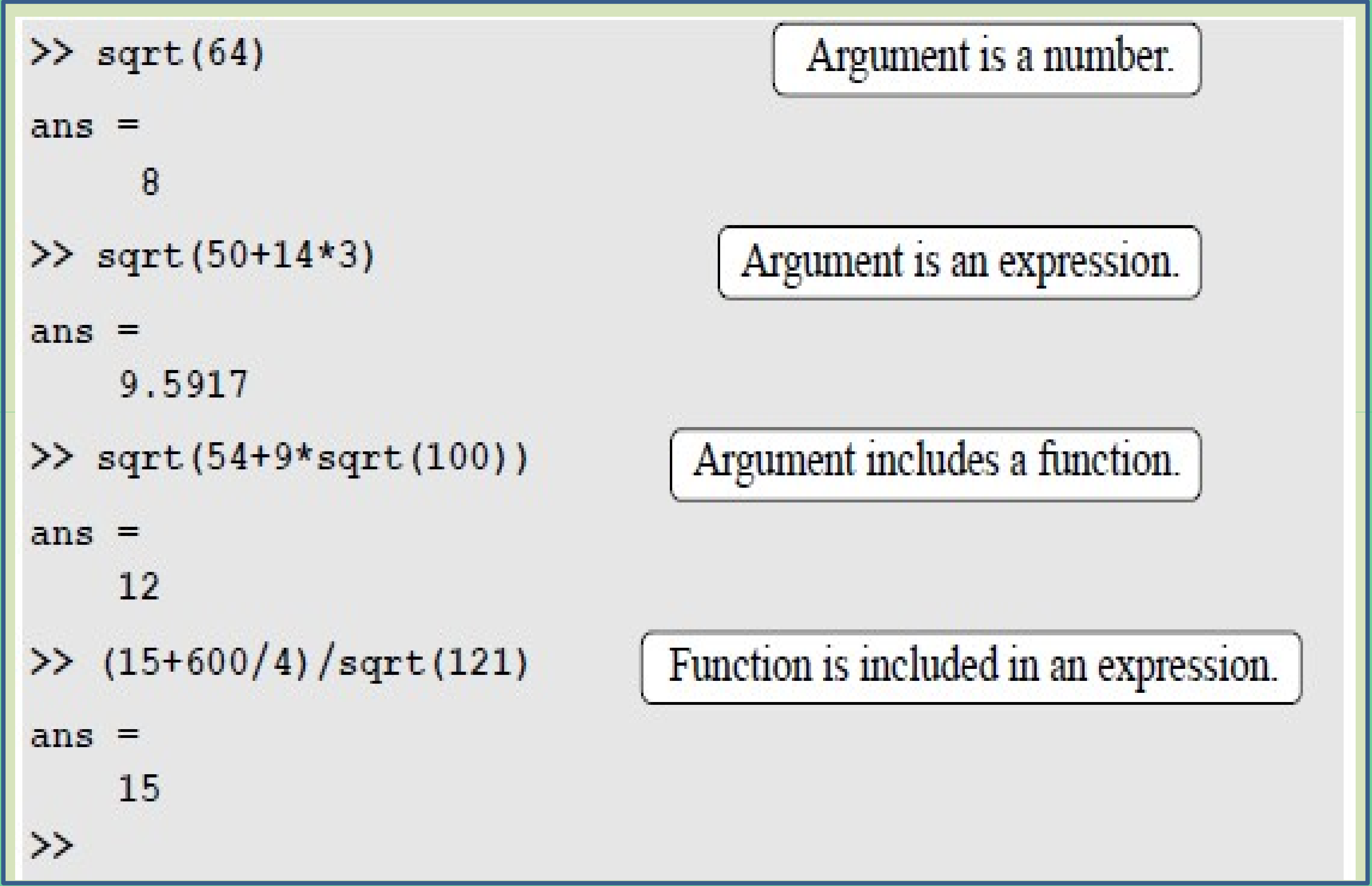
- MATLAB has a very large library of built-in functions.

- A function has a name and an argument in parentheses.

For example, the function that calculates the square root of a number is sqrt(x). Its name is sqrt, its argument is x.

- When the function is used, the argument can be a number, a variable that has been assigned a numerical value , or a computable expression that can be made up of numbers and/or variables.

- Functions can also be included in arguments, show examples below:



```
>> sqrt(64)

ans =

     8

>> sqrt(50+14*3)

ans =

    9.5917

>> sqrt(54+9*sqrt(100))

ans =

    12

>> (15+600/4)/sqrt(121)

ans =

    15

>>
```

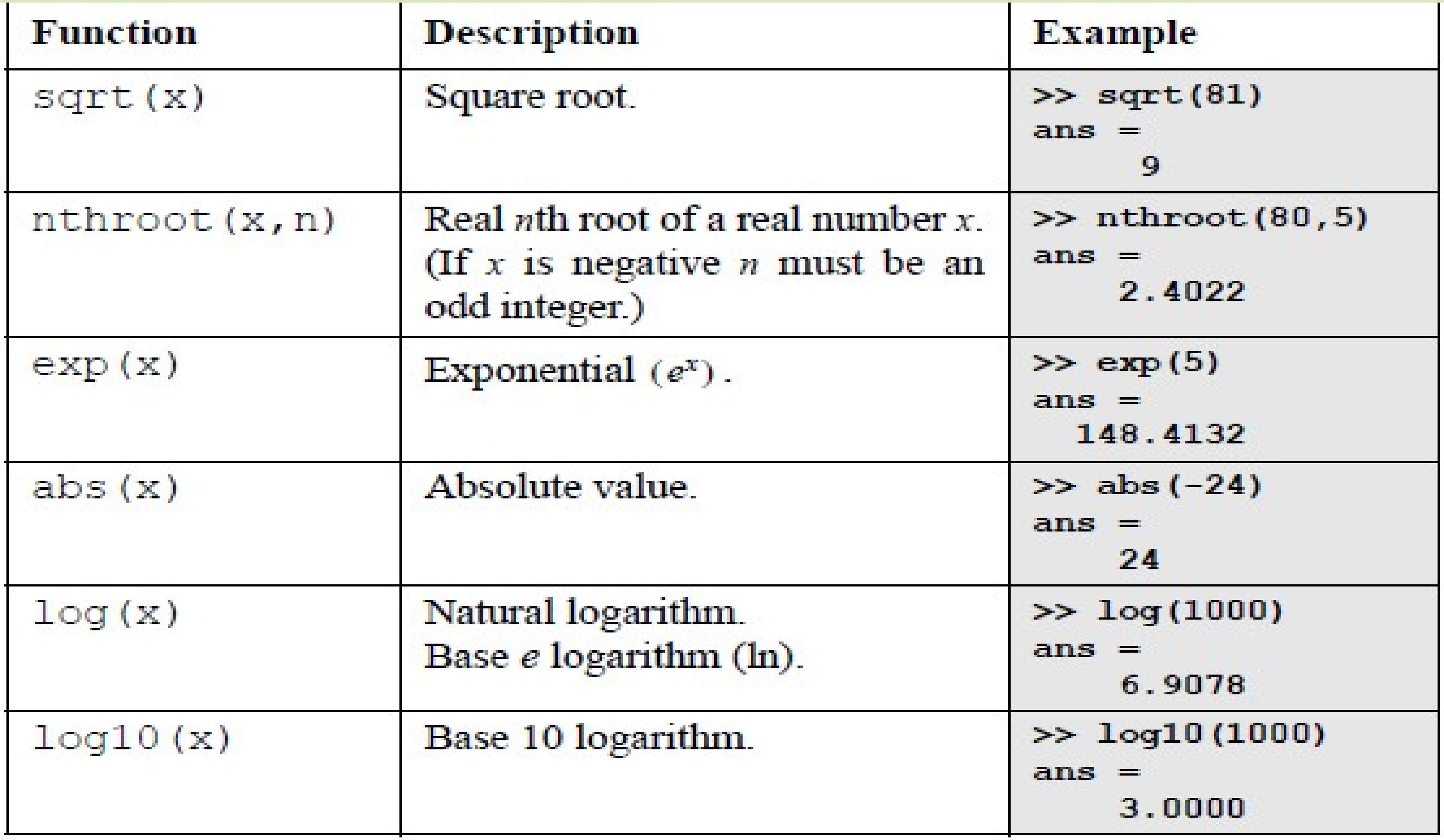
Argument is a number.

Argument is an expression.

Argument includes a function.

Function is included in an expression.

Elementary MATH functions



Function	Description	Example

`sqrt(x)`	Square root.	`>> sqrt(81)`
		`ans =`
		`     9`

`nthroot(x,n)`	Real *n*th root of a real number *x*. (If *x* is negative *n* must be an odd integer.)	`>> nthroot(80,5)`
		`ans =`
		`    2.4022`

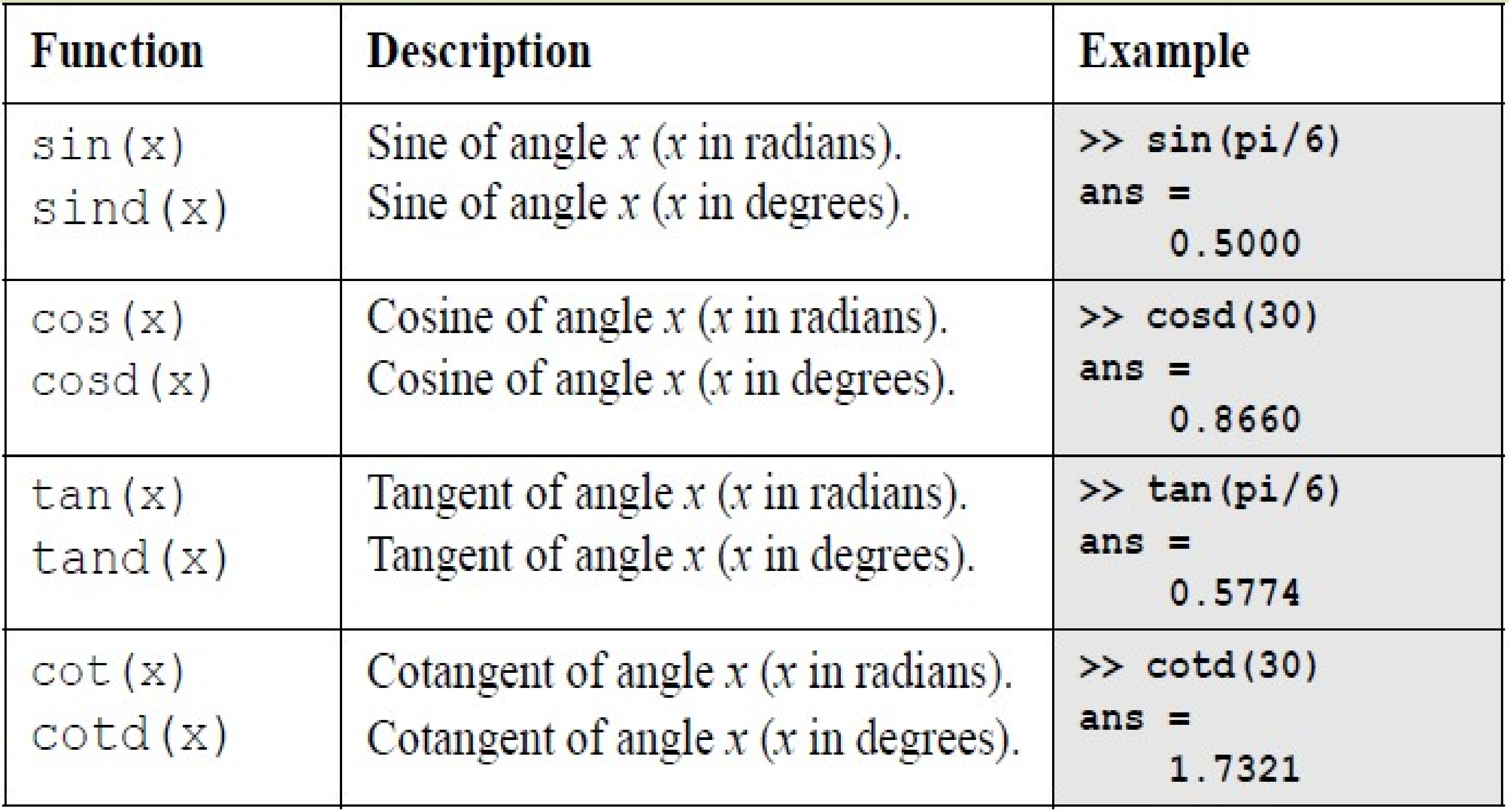
`exp(x)`	Exponential (e^x).	`>> exp(5)`
		`ans =`
		`  148.4132`

`abs(x)`	Absolute value.	`>> abs(-24)`
		`ans =`
		`    24`

`log(x)`	Natural logarithm.
	Base *e* logarithm (ln).
		`>> log(1000)`
		`ans =`
		`    6.9078`

`log10(x)`	Base 10 logarithm.	`>> log10(1000)`
		`ans =`
		`    3.0000`

Trigonometric math functions



Function	Description	Example

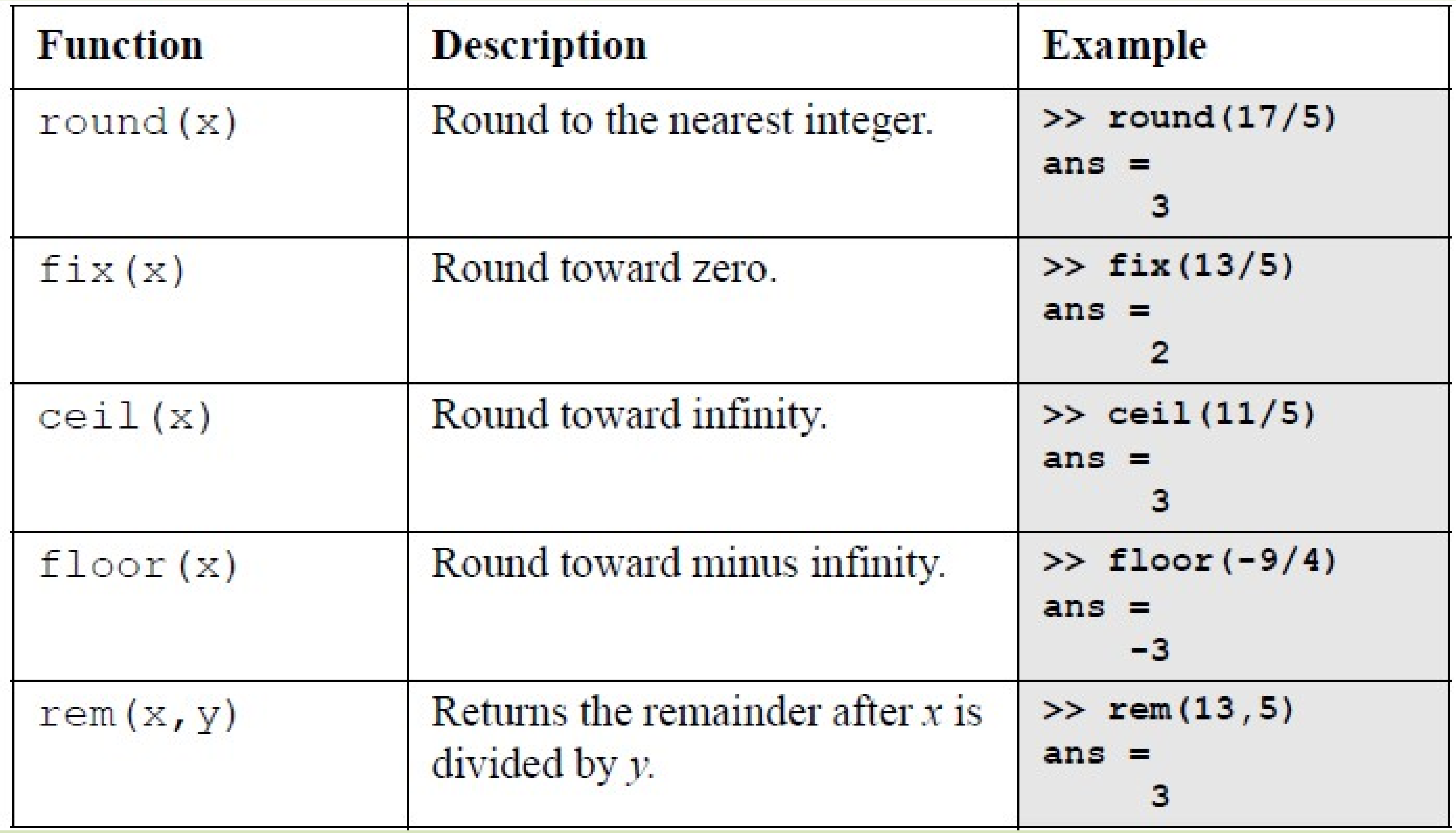
`sin(x)`
`sind(x)`
	Sine of angle *x* (*x* in radians).
	Sine of angle *x* (*x* in degrees).
		`>> sin(pi/6)`
		`ans =`
		`      0.5000`

`cos(x)`
`cosd(x)`
	Cosine of angle *x* (*x* in radians).
	Cosine of angle *x* (*x* in degrees).
		`>> cosd(30)`
		`ans =`
		`      0.8660`

`tan(x)`
`tand(x)`
	Tangent of angle *x* (*x* in radians).
	Tangent of angle *x* (*x* in degrees).
		`>> tan(pi/6)`
		`ans =`
		`      0.5774`

`cot(x)`
`cotd(x)`
	Cotangent of angle *x* (*x* in radians).
	Cotangent of angle *x* (*x* in degrees).
		`>> cotd(30)`
		`ans =`
		`      1.7321`

Rounding functions



Function	Description	Example

`round(x)`	Round to the nearest integer.	`>> round(17/5)`
		`ans =`
		`3`

`fix(x)`	Round toward zero.	`>> fix(13/5)`
		`ans =`
		`2`

`ceil(x)`	Round toward infinity.	`>> ceil(11/5)`
		`ans =`
		`3`

`floor(x)`	Round toward minus infinity.	`>> floor(-9/4)`
		`ans =`
		`-3`

`rem(x,y)`	Returns the remainder after *x* is divided by *y*.	`>> rem(13,5)`
		`ans =`
		`3`

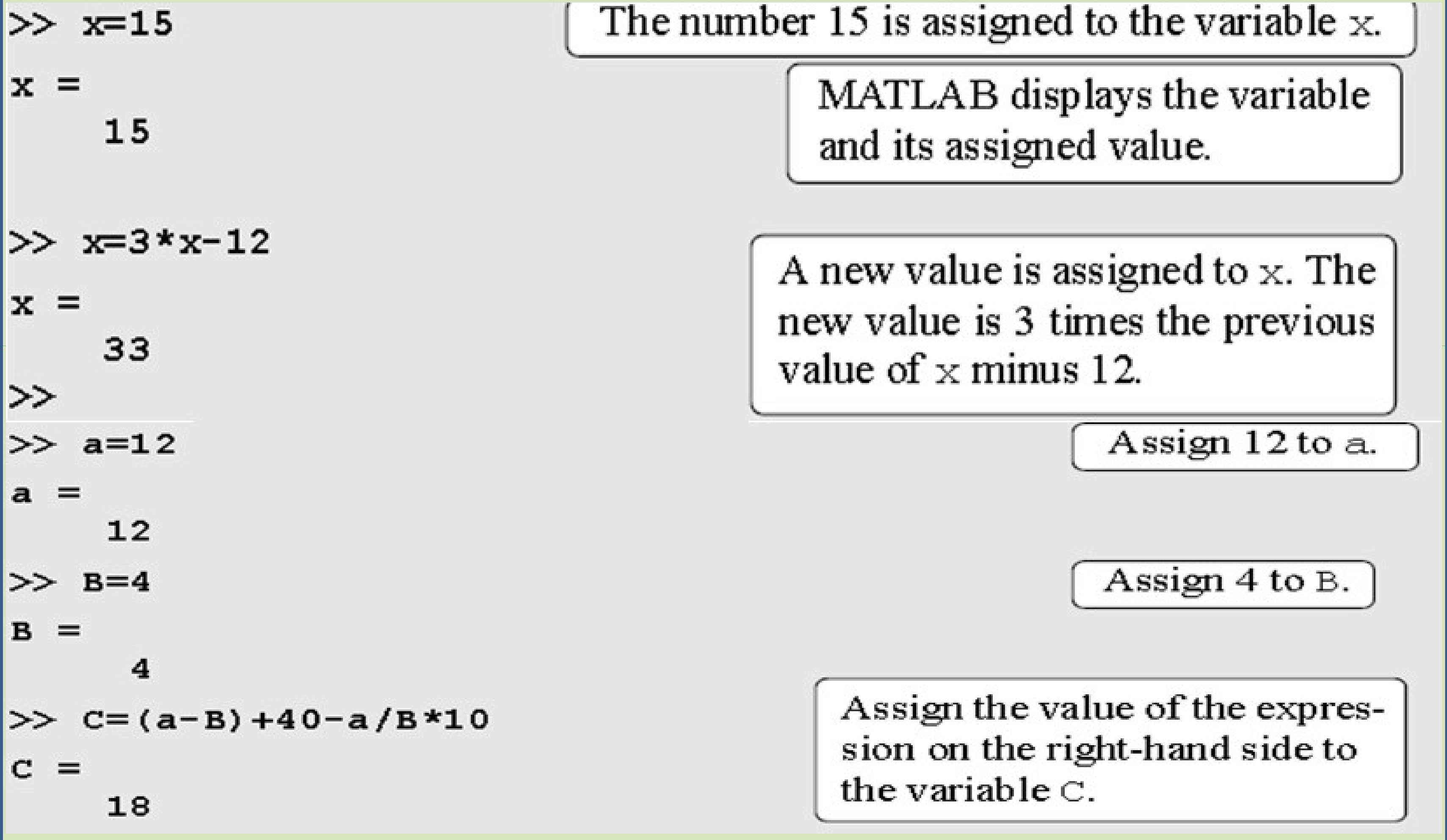
The Assignment Operator

- In MATLAB the = sign is called the assignment operator.
- The assignment operator assigns a value to a variable.

  Variable_name = A numerical value, or a computable expression

- The left-hand side of the assignment operator can include only one variable name
- The right-hand side can be a number, or a computable expression that can include numbers and/or variables that were previously assigned numerical values.
- When **Enter key is pressed the numerical value of the right-hand side is assigned** , and MATLAB displays the variable and its assigned value in the next two lines.

Example for the assignment operator



```
>> x=15
x =
     15

>> x=3*x-12
x =
     33
>>
>> a=12
a =
     12
>> B=4
B =
      4
>> C=(a-B)+40-a/B*10
C =
     18
```

The number 15 is assigned to the variable x.

MATLAB displays the variable and its assigned value.

A new value is assigned to x. The new value is 3 times the previous value of x minus 12.

Assign 12 to a.

Assign 4 to B.

Assign the value of the expression on the right-hand side to the variable C.

Notes

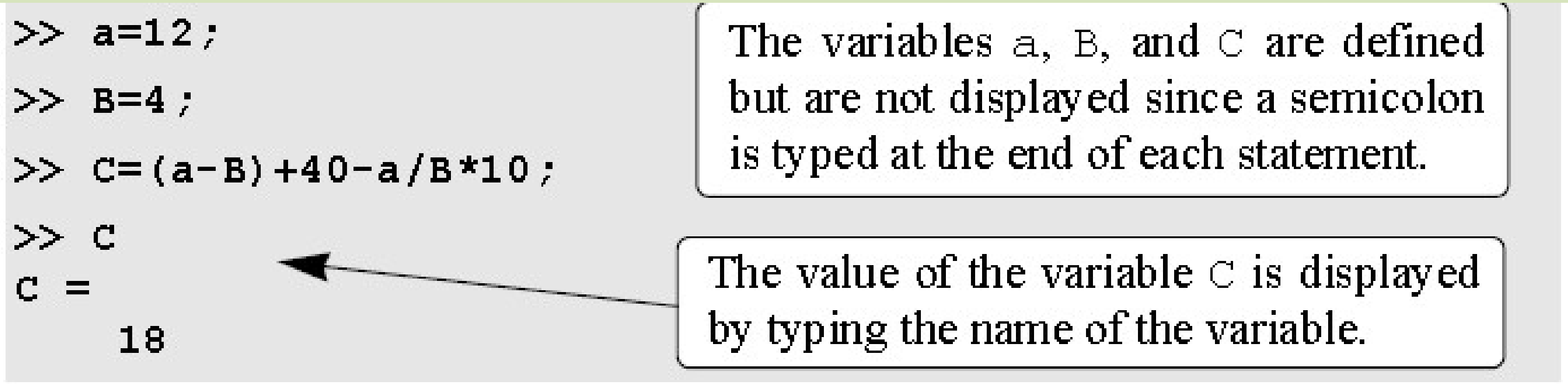
❖ *The statement ($x = 3x - 12$) illustrates the difference between the assignment operator and the equal sign.*

❖ *If in this statement the = sign meant equal, the value of x would be 6 (solving the equation for x).*

❖ *If a semicolon is typed at the end of the command, then when Enter key is pressed, MATLAB does not display the variable with its assigned value (the variable still exists and is stored in memory).*

❖ *If a variable already exists, typing the variable's name and pressing Enter key will display the variable and its value in the next two lines.*

As an example,



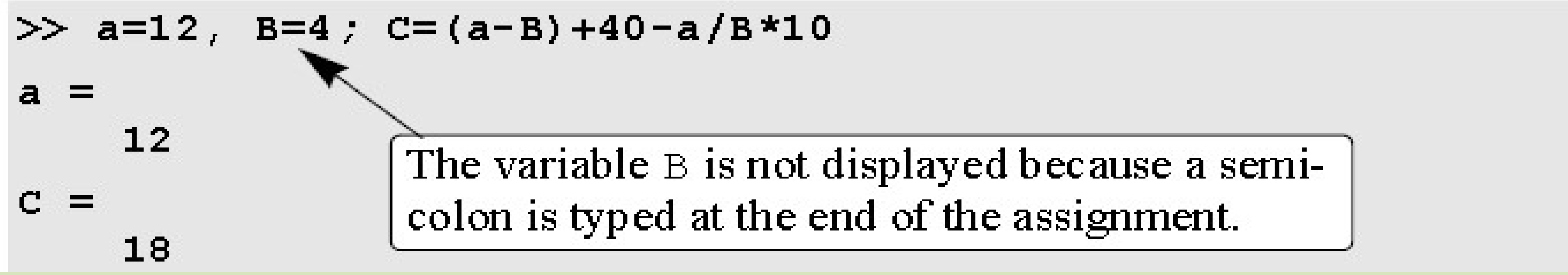
```
>> a=12;
>> B=4;
>> C=(a-B)+40-a/B*10;
>> C
C =
    18
```

The variables a, B, and C are defined but are not displayed since a semicolon is typed at the end of each statement.

The value of the variable C is displayed by typing the name of the variable.

❖Several assignments can be typed in the same line. Assignments must be separated with a comma (spaces can be added after the comma).

❖Assignments are executed and displayed from left to right



```
>> a=12, B=4; C=(a-B)+40-a/B*10
a =
    12
C =
    18
```

The variable B is not displayed because a semicolon is typed at the end of the assignment.

Rules About Variable Names

A variable can be named according to the following rules:

- **Must begin with a letter.**
- **Can be up to 63 characters long.**
- **Can contain letters, digits, and the underscore character.**
- **Cannot contain punctuation characters (e.g., period, comma, semicolon).**
- ***MATLAB is case sensitive: it distinguishes between uppercase and lowercase letters*.**

For example, AA, Aa, aA, and aa are the names of four different variables.

- **No spaces are allowed between characters (use the underscore where a space is** desired).

- **Avoid using the name of a built-in function for a variable (i.e., avoid using cos, sin, exp, sqrt,** etc.). Once a function name is used to define a variable, the function cannot be used.

Keywords and Predefined Variables

There are 20 words, called keywords, that are reserved by MATLAB for various purposes and cannot be used as variable names. These words are:

break case catch classdef continue else
Elseif end for function global if
otherwise parfor persistent return spmd switch
try while

Keywords

pi The number π.
Eps The smallest difference between two numbers. Equal to 2^(–52),
Inf Used for infinity.
i ,j Defined as $\sqrt{-1}$ which is: 0 + 1.0000i.
NaN Stands for Not-a-Number. Used when MATLAB cannot determine a valid numeric value. Example: 0/0.

USEFUL COMMANDS FOR MANAGING VARIABLES

are commands that can be used to eliminate variables or to obtain information about variables:

Command	**Outcome**
Clear	Removes all variables from the memory.
clear x y z	Removes only variables x, y, and z from the memory.
who	Displays a list of the variables currently in the memory.
Whos	Displays a list of the variables currently in the memory and their sizes together with information about their bytes and class

SCRIPT FILES

Is an executable file that contains a list of commands instead of Using command window to execute commands individually

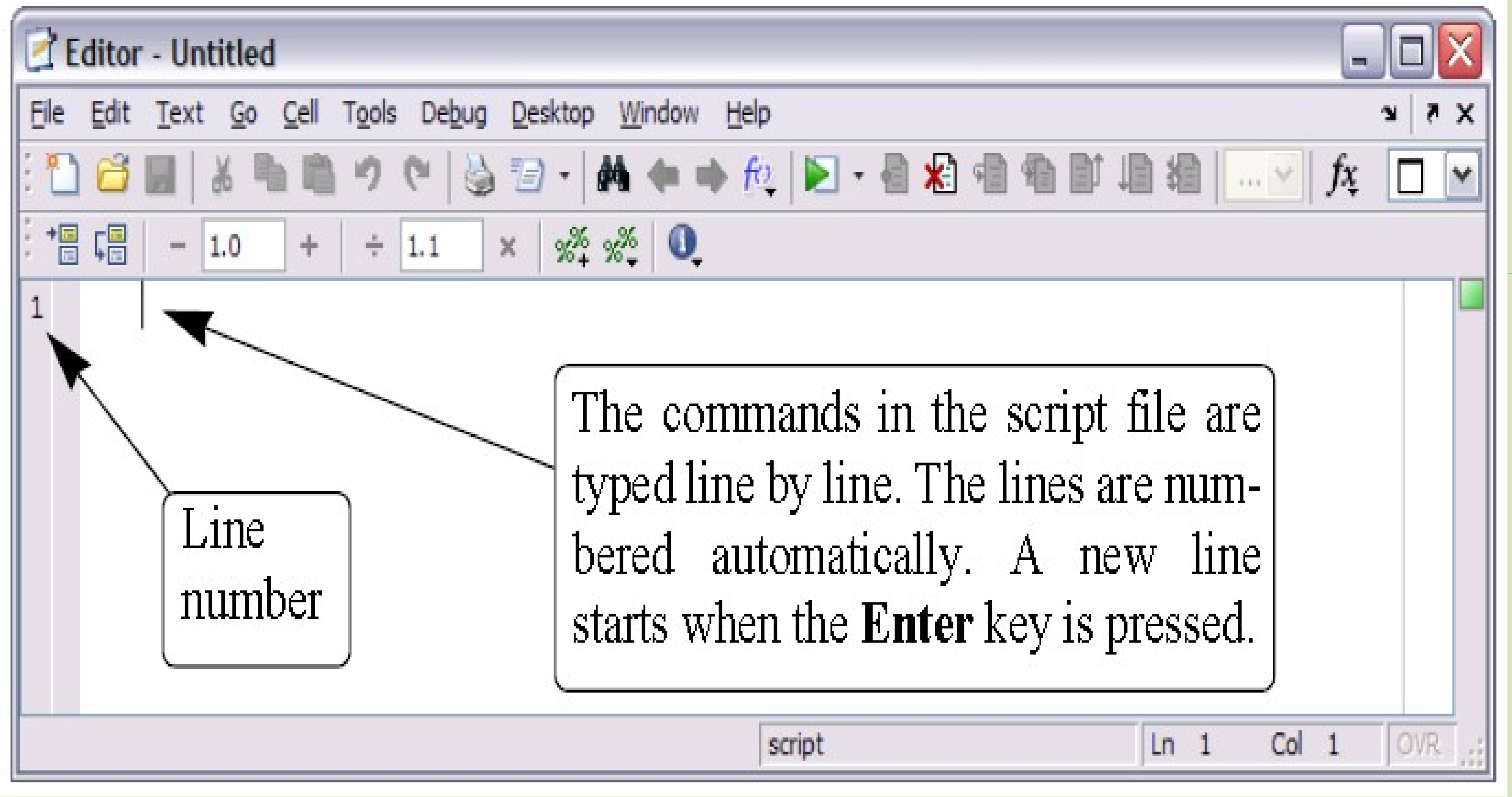
Notes About Script Files

- **script file is called a program.**
- **When it is executed, MATLAB executes the commands in theorder they are written .**
- **a command that generates an output (e.g., assignment of a value to a variable without a semicolon at the end), the output is displayed in the Command Window.**

- **script file is convenient because it can be edited (corrected or otherwise changed) and executed many times.**

Creating and Saving a Script File

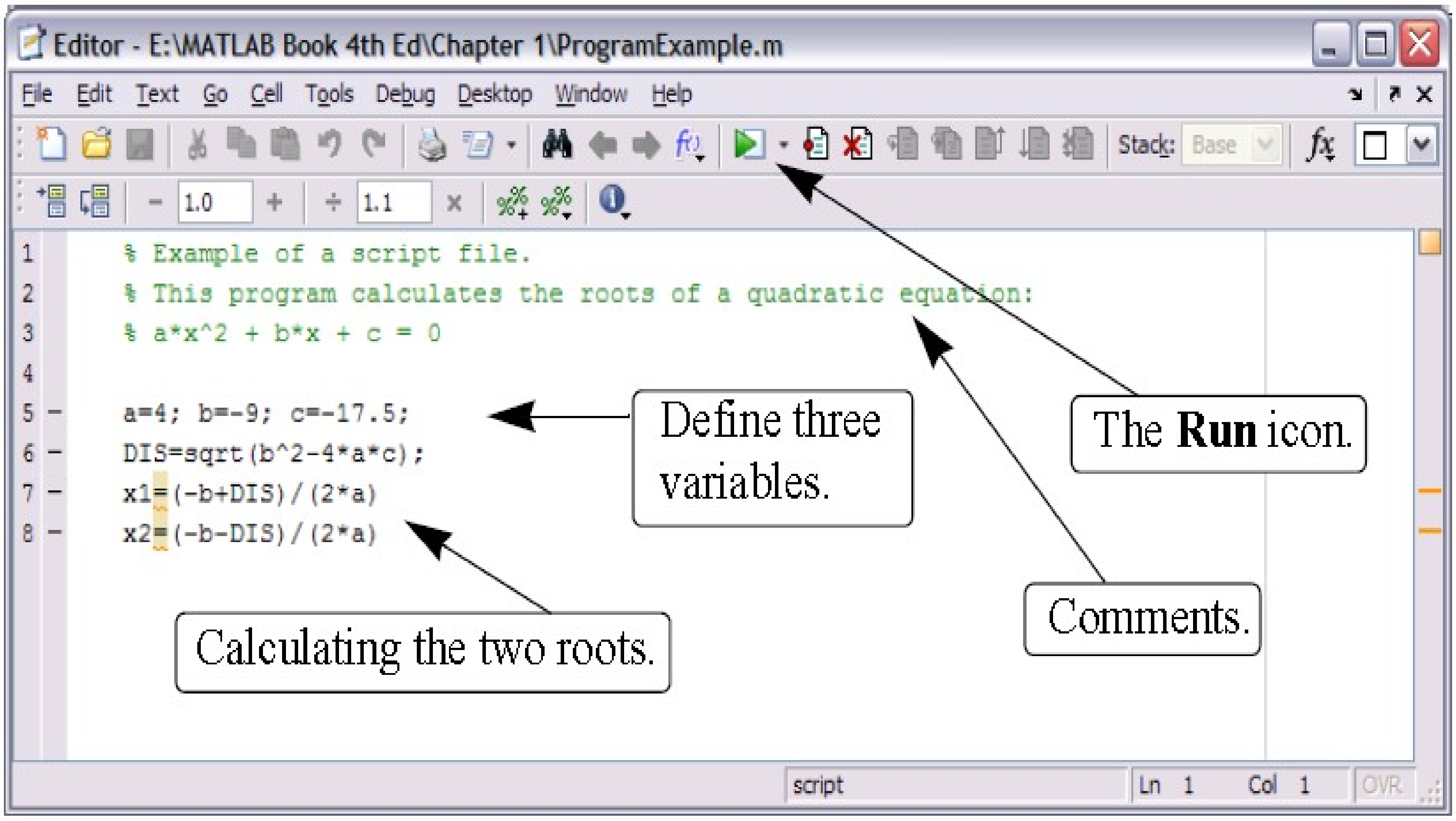
script files are created and edited in the Editor/Debugger Window



Line number

The commands in the script file are typed line by line. The lines are numbered automatically. A new line starts when the **Enter** key is pressed.

A program typed in the Editor/Debugger Window.



Editor - E:\MATLAB Book 4th Ed\Chapter 1\ProgramExample.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
fx
1.0
1.1
script
Ln 1
Col 1
OVR

```
% Example of a script file.
% This program calculates the roots of a quadratic equation:
% a*x^2 + b*x + c = 0

a=4; b=-9; c=-17.5;
DIS=sqrt(b^2-4*a*c);
x1=(-b+DIS)/(2*a)
x2=(-b-DIS)/(2*a)
```

Define three variables.

The **Run** icon.

Comments.

Calculating the two roots.