EXPERIMENT-ONE

INTRODUCTION

1- What Is MATLAB ?

MATLAB® is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation.

The name MATLAB stands for MATrix LABoratory. MATLAB was originally written to provide easy access to matrix software. MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of application-specific solutions called *toolboxes*. Very important to most users of MATLAB, toolboxes allow you to *learn* and *apply* specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

2- The MATLAB System

The MATLAB system consists of five main parts:

2.1- Development Environment: This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are Graphical User Interfaces (GUI). It includes the MATLAB desktop and Command Window, a command history, and browsers for viewing help, the workspace, files, and the search path.

2.2- The MATLAB Mathematical Function Library: This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

2.3- The MATLAB Language: This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

2.4- Handle Graphics: This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

2.5- The MATLAB Simulink: This is a software for modeling, simulating, and analyzing dynamic systems. It supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two. Systems can also be multirate, i.e., have different parts that are sampled or updated at different rates. Simulink enables you to posea question about a system, model it, and see what happens.

3- Starting, Quitting MATLAB and Opening M-files

3.1- Starting MATLAB: On a Microsoft Windows platform, to start MATLAB, double-click the MATLAB shortcut icon on your Windows desktop. After starting MATLAB, the MATLAB desktop opens.

3.2- Quitting MATLAB: To end your MATLAB session, select **Exit MATLAB** from the **File** menu in the desktop, or type quit in the Command Window, or from the closed bottom in the upper right corner of the command window.

3.2- Opening M-files: M-files are the files that we will write our only programs on it. From the MATLAB command window select the **File** menu and choose **New/M-file**. This action opens a Notepad (Untitled) window. You can regard this as a 'scratch pad' in which to write programs. From the M-file **save** menu you can save your program after typing any name instead of Untitled. Also you can run this program by choosing **Run** from the M-file **Debug** menu (or press F5 key). The results will appear on the MATLAB Desktop.

4- MATLAB Desktop

When you start MATLAB, the MATLAB desktop appears, containing tools (graphical user interfaces) for managing files, variables, and applications associated with MATLAB. The first time MATLAB starts, the desktop appears as shown in Fig-1, your desktop looks by opening, closing, moving, and resizing the tools in it. You can also move tools outside of the desktop or return them back inside the desktop (docking). All the desktop tools provide common features such as context menus and keyboard shortcuts. You can specify certain characteristics for the desktop tools by selecting **Preferences** from the **File** menu. For example, you can specify the font characteristics for Command Window text. For more information, click the **Help** button in the **Preferences** dialog box. Although your Launch Pad may contain different entries, you can change the way you want the desktop appearance.

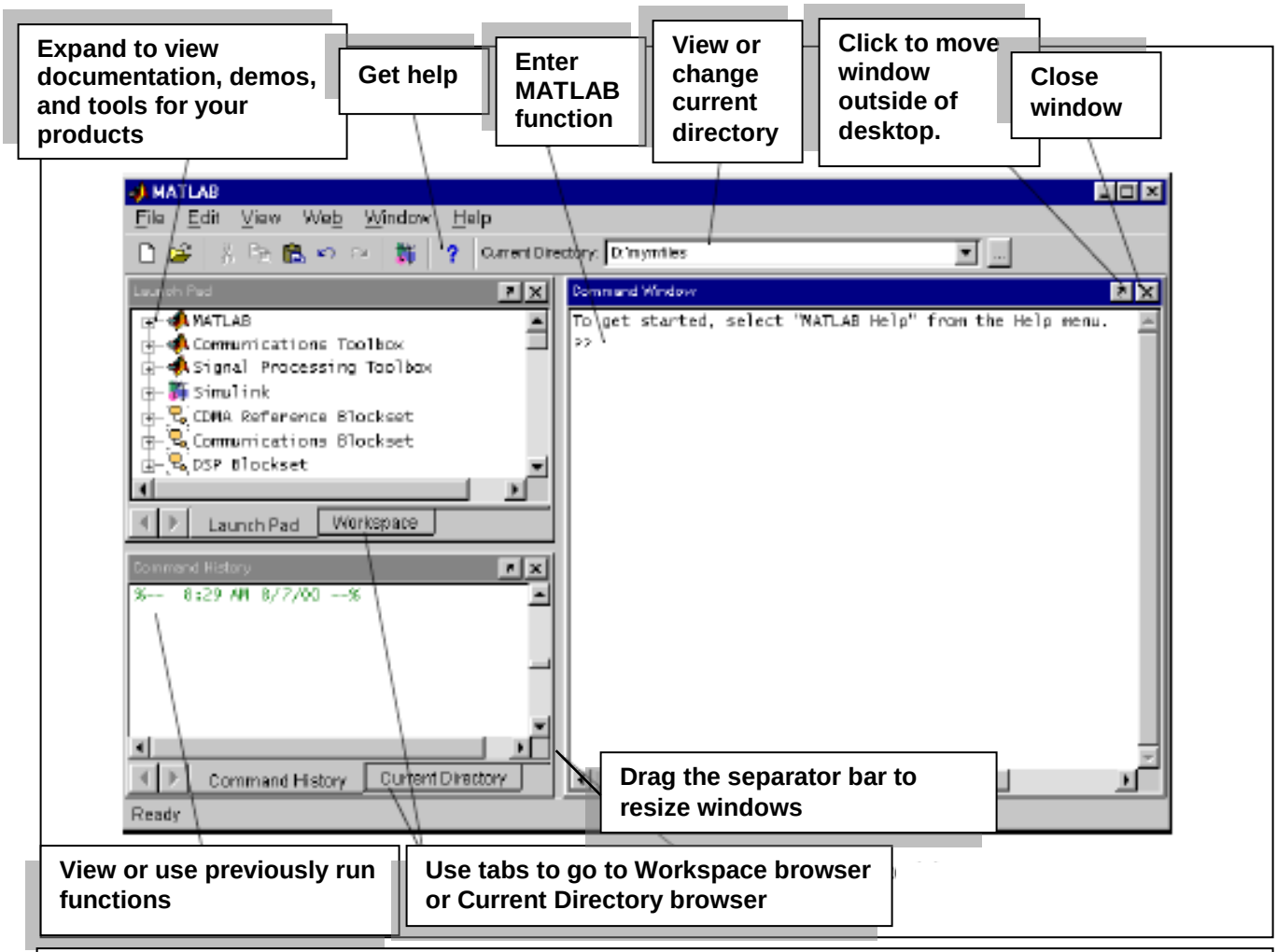

Fig-1: The MATLAB Desktop

Usually the following tools are appeared in the MATLAB's desktop:

- The Command Window.
- The Command History.
- The Launch Pad.
- Workspace Browser.
- The Array Editor.
- Editor/Debugger.

4.1- <u>Command Window:</u> Use the **Command Window** to enter variables and run functions and M-files and controlling input and output data. The command window is shown in Fig-2.

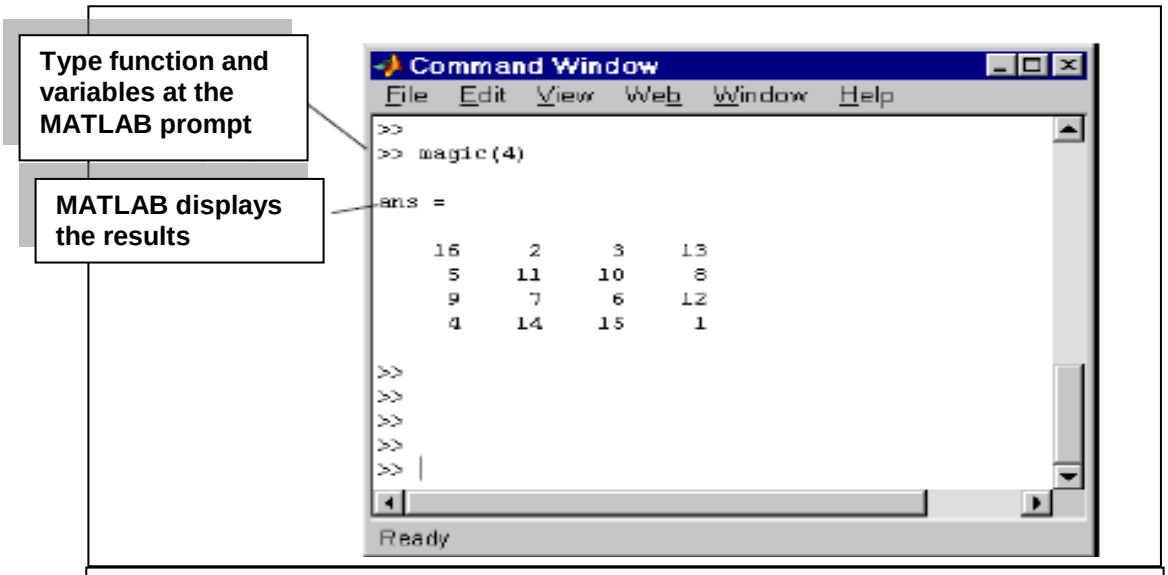

Fig-2: Command Window

4.2- <u>Command History:</u> Lines you enter in the Command Window are logged in the **Command History** window shown in Fig-3. In the Command History, you can view previously used functions, and copy and execute selected lines.

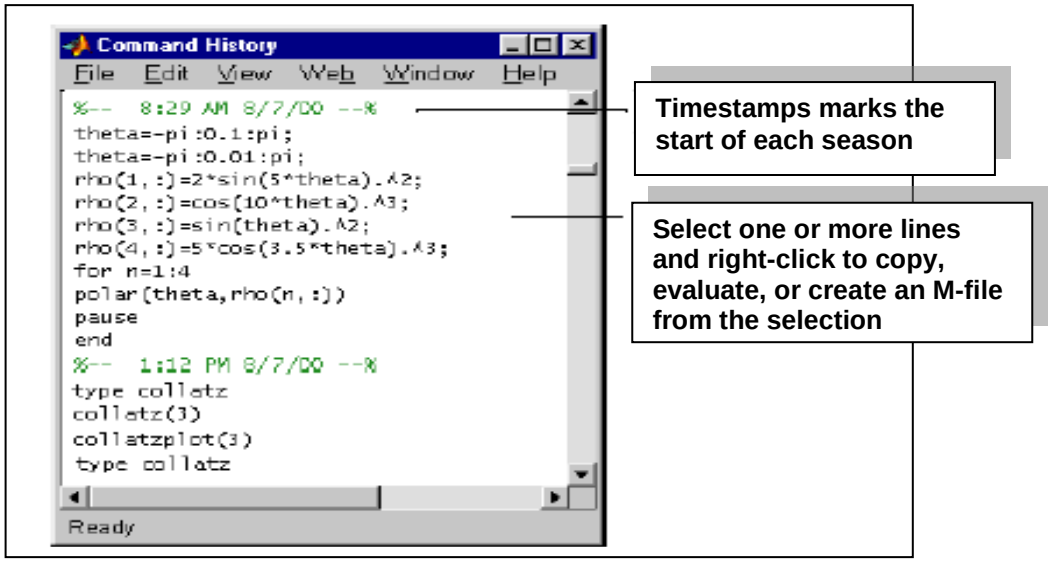

Fig-3: Command History

4.3- Launch Pad: MATLAB's **Launch Pad** (shown in Fig-4) provides easy access to tools, demos, and documentation.

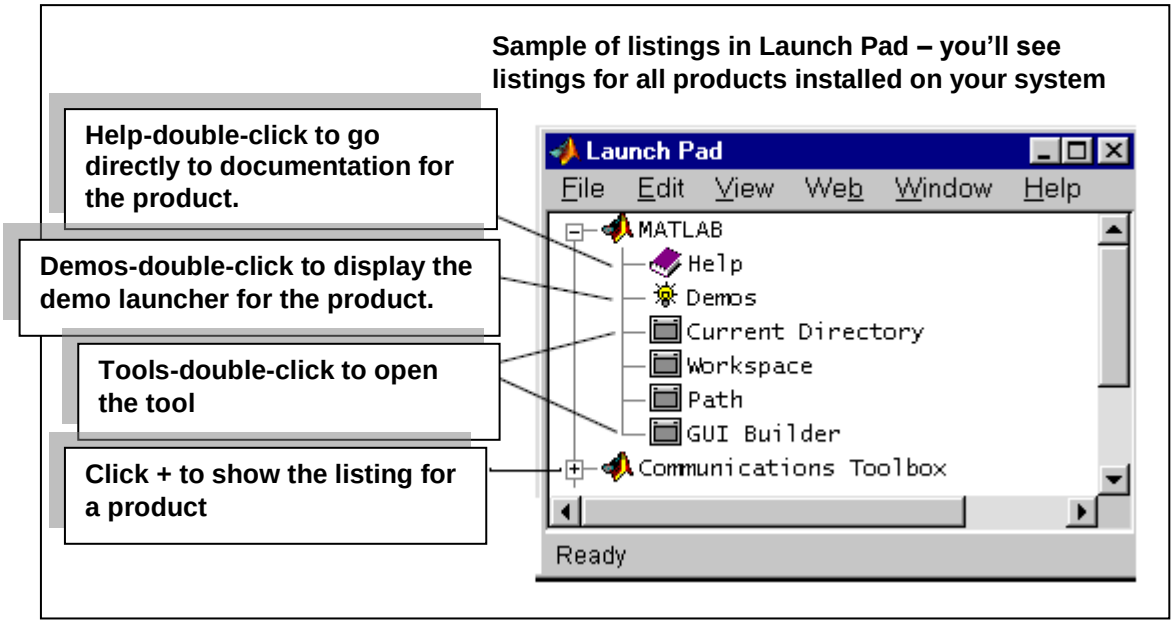

Fig-4: Launch Pad

4.4- Workspace Browser: The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. You add variables to the workspace by using functions, running M-files, and loading saved workspaces. To view the workspace and information about each variable, use the Workspace browser (shown in Fig-5), or use the functions who and whos.

To delete variables from the workspace, select the variable and select **Delete** from the **Edit** menu. Alternatively, use the clear function. The workspace is not maintained after you end the MATLAB session. To save the workspace to a file that can be read during a later MATLAB session, select **Save Workspace As** from the **File** menu, or use the save function. This saves the workspace to a binary file called a MAT-file, which has a .mat extension. There are options for saving to different formats. To read in a MAT-file, select **Import Data** from the **File** menu, or use the load function.

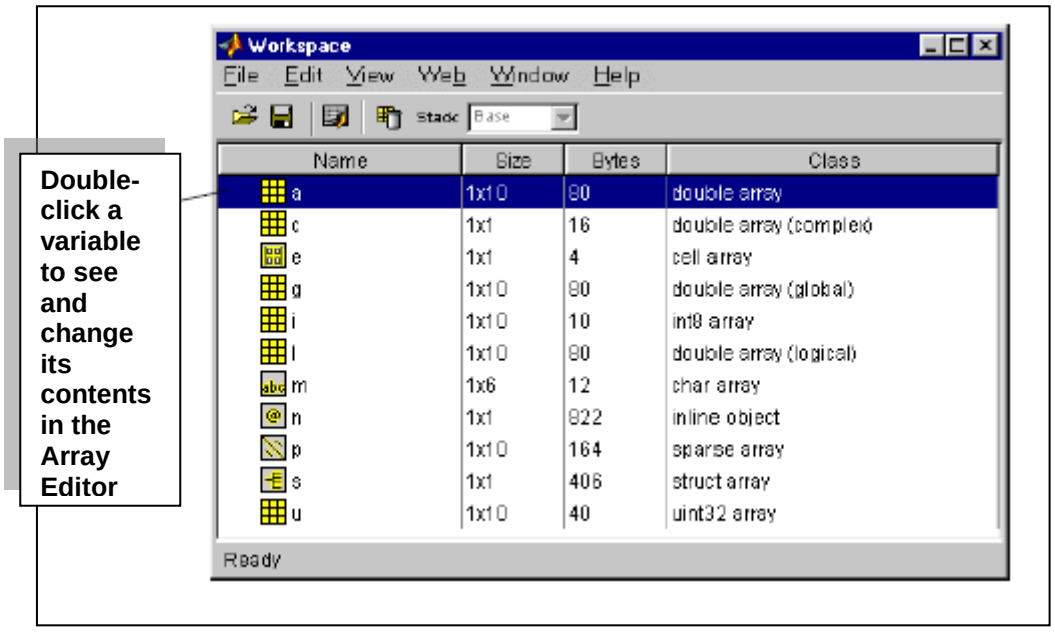

Fig-5: The Workspace Browser

4.5- Array Editor: Double-click on a variable in the Workspace browser to see it in the Array Editor shown in Fig-6. Use the Array Editor to view and edit a visual representation of one- or two-dimensional numeric arrays, strings, and cell arrays of strings that are in the workspace.

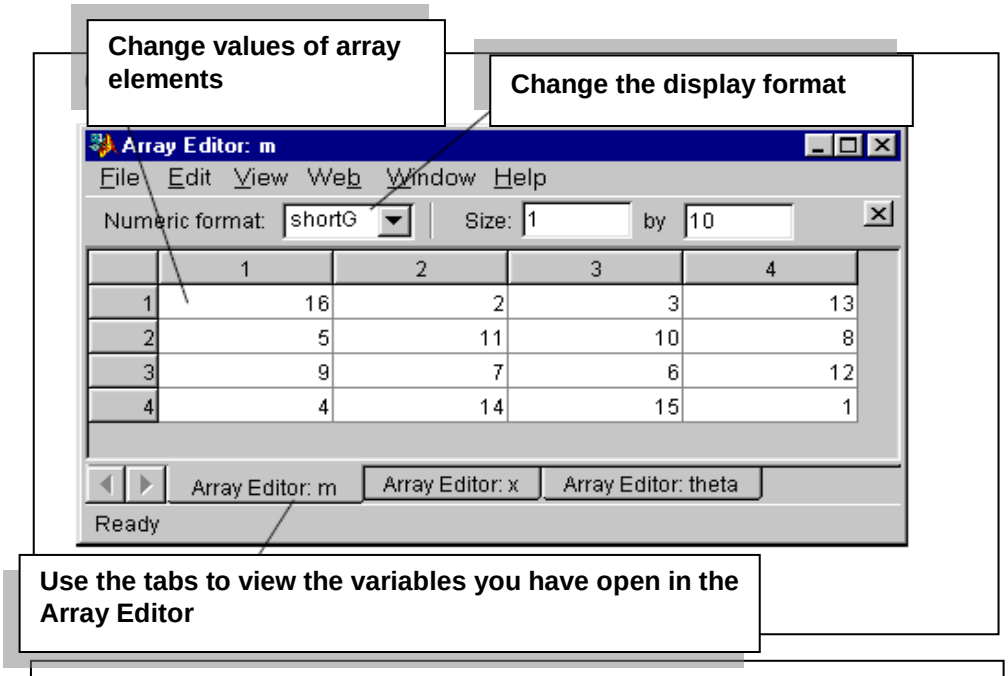

Fig-6: The Array Editor

4.6- <u>Editor/Debugger:</u> Use the Editor/Debugger to create and debug M-files, which are programs you write to run MATLAB functions. The Editor/Debugger (shown in Fig-7) provides a graphical user interface for basic text editing, as well as for M-file debugging. You can use any text editor to create M-files, such as Emacs, and can use preferences (accessible from the desktop **File** menu) to specify that editor as the default. If you use another editor, you can still use the MATLAB Editor/Debugger for debugging, or you can use debugging functions, such as dbstop, which sets a breakpoint. If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.

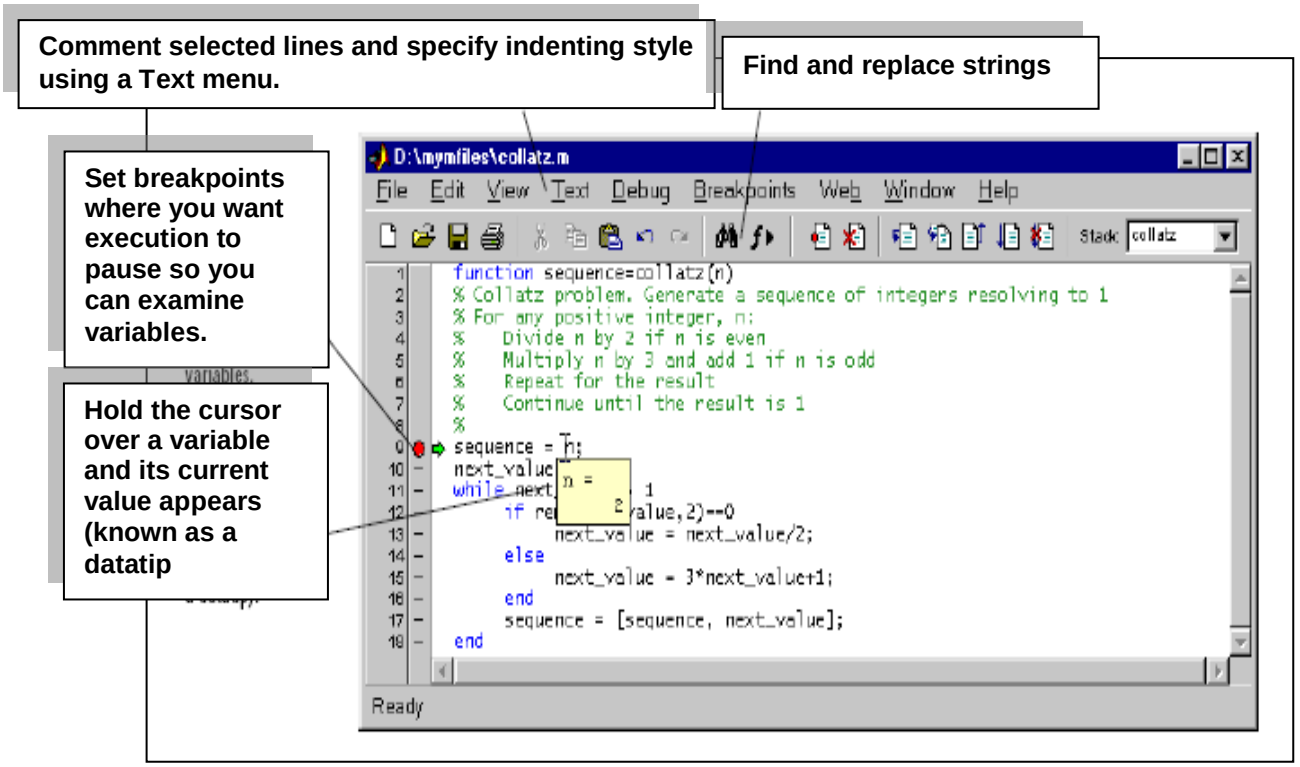

Fig-7: The Edditor/Debugger

5- Controlling Command Window Input and Output

So far, you have been using the MATLAB command line, typing commands and expressions, and seeing the results printed in the Command Window. This section describes how to:

- Control the appearance of the output values (format command).
- Suppress output from MATLAB commands.
- Enter long commands at the command line.
- Edit the command line.

5.1- The format Command: The format command controls the numeric format of the values displayed by MATLAB. The command affects only how numbers are displayed, not how MATLAB computes or saves them. Here are the different formats, together with the resulting output produced from a vector x with components of different magnitudes.

```
>> x = [4/3       1.2345e-6]
>> format short            % Scaled fixed point format with 5 digits (default display).
>> x       < Enter >
 x =
    1.3333       0.0000

>> format short e          % Floating point format with 5 digits plus exponent.
>> x       < Enter >
 x =
    1.3333e+000     1.2345e-006

>> format long             % Scaled fixed point format with 15 digits.
>> x       < Enter >
x =
   1.33333333333333        0.00000123450000

>> format long e           % Floating point format with 15 digits.
>> x       < Enter >
 x =
    1.333333333333333e+000        1.234500000000000e-006
>> format bank             % Fixed format for dollars and cents.
>> x       < Enter >
 x =
    1.33     0.00

>> format rat              % Approximation by ratio of small integers.
>> x       < Enter >
x =
    4/3     1/810045

>> format hex              % Hexadecimal format.
>> x       < Enter >
x =
    3ff5555555555555        3eb4b6231abfd271
```

If the largest element of a matrix is larger than 10^3 or smaller than 10^{-3}, MATLAB applies a common scale factor for the short and long formats. In addition to the format commands shown above, format compact suppresses many of the blank lines that appear in the output. This lets you view more information on a screen or window.

5.2- Suppressing Output: If you simply type a statement and press **Return** or **Enter**, MATLAB automatically displays the results on screen. However, if you end the line with a semicolon, MATLAB performs the computation but does not display any output. This is particularly useful when you generate large matrices. For example,

```
>> A = 100;       < Enter >
>>
```

5.3- Entering Long Command Lines: If a statement does not fit on one line, use three periods, ..., followed by **Return** or **Enter** to indicate that the statement continues on the next line. For example,

```
>> s = 1 -1/2 + 1/3 -1/4 + 1/5 - 1/6 + 1/7 ...      < Enter >
    - 1/8 + 1/9 - 1/10 + 1/11 - 1/12;                < Enter >
>>
```

Blank spaces around the = , + , and - signs are optional, but they improve readability.

5.4- Command Line Editing: Various arrow and control keys on your keyboard allow you to recall, edit, and reuse commands you have typed earlier. For example, suppose you mistakenly enter

```
>> rho = (1 + sqt(5))/2      < Enter >
>> Undefined function or variable 'sqt'.
```

You have misspelled sqrt. MATLAB responds with. So instead of retyping the entire line, simply press the ↑ □key. The misspelled command is redisplayed. Use the ← □key to move the cursor over and insert the missing r. Repeated use of the ↑ □key recalls earlier lines. Typing a few characters and then the ↑ □key finds a previous line that begins with those characters. You can also copy previously executed commands from the Command History.

The list of available command line editing keys is different on different computers. Experiment to see which of the following keys is available on your machine.

↑ □	**Ctrl+p**	Recall previous line
↓	□**Ctrl+n**	Recall next line
←	□**Ctrl+b**	Move back one character
→ □	**Ctrl+f**	Move forward one character

Ctrl+→	**Ctrl+r**	Move right one word
Ctrl+←	**Ctrl+l**	Move left one word
Home	**Ctrl+a**	Move to beginning of line
End	**Ctrl+e**	Move to end of line
Esc	**Ctrl+u**	Clear line
Del	**Ctrl+d**	Delete character at cursor
Backspace	**Ctrl+h**	Delete character before cursor
	Ctrl+k	Delete to end of line

Exercises:

1. What is MATLAB and what are its applications?
2. How many parts is the MATLAB system consists of? What are they?
3. What are the main parts of the MATLAB desktop? List the advantage of each part.
4. How can we control the appearance of the output values? And how can we suppress output from MATLAB commands.
5. Apply the format commands on a two numbers (A & B) on the command window.
6. Build the following program in M-file. Run the program and get the results;

```
% This is a test program
disp ('The variables value are')
x=3.05
y=5.6e-2
a=x/y;
b=x*y;
c=x\y;
res1=sqrt(a)+(b/c)
res2=res1+(c^2)
disp ('The program ends here')
```