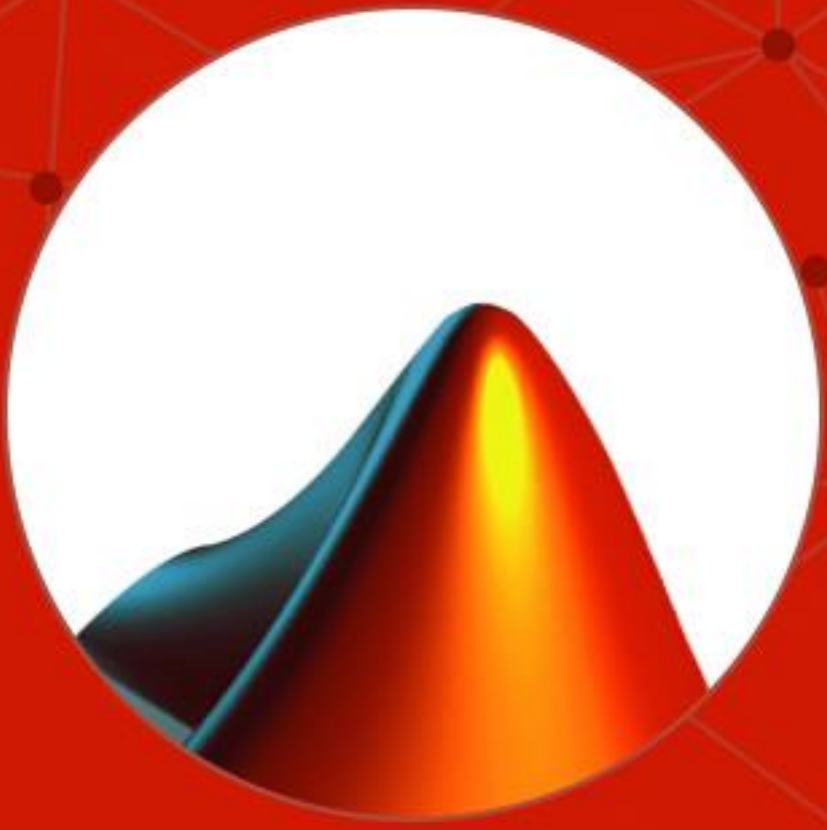


MATLAB

numerical computing

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

MATLAB is a programming language developed by MathWorks. It started out as a matrix programming language where linear algebra programming was simple. It can be run both under interactive sessions and as a batch job.

This tutorial gives you aggressively a gentle introduction of MATLAB programming language. It is designed to give students fluency in MATLAB programming language. Problem-based MATLAB examples have been given in simple and easy way to make your learning fast and effective.

Audience

This tutorial has been prepared for the beginners to help them understand basic to advanced functionality of MATLAB. After completing this tutorial you will find yourself at a moderate level of expertise in using MATLAB from where you can take yourself to next levels.

Prerequisites

We assume you have a little knowledge of any computer programming and understand concepts like variables, constants, expressions, statements, etc. If you have done programming in any other high-level language like C, C++ or Java, then it will be very much beneficial and learning MATLAB will be like a fun for you.

Copyright & Disclaimer Notice

© Copyright 2014 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com

Table of Contents

About the Tutorial i

Audience i

Prerequisites i

Copyright & Disclaimer Notice i

Table of Contents ii

1. OVERVIEW 1

MATLAB's Power of Computational Mathematics 1

Features of MATLAB 1

Uses of MATLAB 2

2. ENVIRONMENT 3

Local Environment Setup 3

Understanding the MATLAB Environment 4

3. BASIC SYNTAX 7

Hands on Practice 7

Use of Semicolon (;) in MATLAB 8

Adding Comments 8

Commonly used Operators and Special Characters 9

Special Variables and Constants 10

Naming Variables 11

Saving Your Work 11

4. VARIABLES 12

Multiple Assignments 13

I have forgotten the Variables! 13

Long Assignments 14

tutorialspoint
SIMPLYEASYLEARNING

The format Command 15

Creating Vectors 17

Creating Matrices 18

5. COMMANDS 20

Commands for Managing a Session 20

Commands for Working with the System 20

Input and Output Commands 22

Vector, Matrix, and Array Commands 23

Plotting Commands 25

6. M-FILES 27

The M Files 27

Creating and Running Script File 27

7. DATA TYPES 30

Data Types Available in MATLAB 30

Data Type Conversion 32

Determination of Data Types 34

8. OPERATORS 39

Arithmetic Operators 39

Functions for Arithmetic Operations 42

Relational Operators 46

Logical Operators 49

Functions for Logical Operations 50

Bitwise Operations 55

Set Operations 57

9. DECISION MAKING 60

tutorialspoint
SIMPLYEASYLEARNING

if... end Statement ····· 61

if...else...end Statement ····· 63

if...elseif...elseif...else...end Statements ····· 64

The Nested if Statements ····· 66

The switch Statement ····· 67

The Nested Switch Statements ····· 69

10. LOOP TYPES ····· 71

The while Loop ····· 72

The for Loop ····· 73

The Nested Loops ····· 76

Loop Control Statements ····· 78

The break Statement ····· 79

The continue Statement ····· 80

11. VECTORS ····· 83

Row Vectors ····· 83

Column Vectors ····· 83

Referencing the Elements of a Vector ····· 84

Vector Operations ····· 85

Addition and Subtraction of Vectors ····· 85

Scalar Multiplication of Vectors ····· 86

Transpose of a Vector ····· 86

Appending Vectors ····· 87

Magnitude of a Vector ····· 89

Vector Dot Product ····· 90

Vectors with Uniformly Spaced Elements ····· 90

tutorialspoint
SIMPLYEASYLEARNING

12. MATRIX 92

Referencing the Elements of a Matrix 92

Deleting a Row or a Column in a Matrix 94

Matrix Operations 96

Addition and Subtraction of Matrices 96

Division (Left, Right) of Matrix 97

Scalar Operations of Matrices 98

Transpose of a Matrix 99

Concatenating Matrices 99

Matrix Multiplication 101

Determinant of a Matrix 102

Inverse of a Matrix 102

13. ARRAYS 104

Special Arrays in MATLAB 104

A Magic Square 106

Multidimensional Arrays 106

Array Functions 109

Sorting Arrays 112

Cell Array 113

Accessing Data in Cell Arrays 114

14. COLON NOTATION 116

15. NUMBERS 119

Conversion to Various Numeric Data Types 119

Smallest and Largest Integers 121

Smallest and Largest Floating Point Numbers 123

tutorialspoint
SIMPLYEASYLEARNING

16. STRINGS 125

Rectangular Character Array 126

Combining Strings into a Cell Array 128

String Functions in MATLAB 129

17. FUNCTIONS 134

Anonymous Functions 135

Nested Functions 138

Private Functions 139

Global Variables 140

18. DATA IMPORT 142

Low-Level File I/O 146

Import Text Data Files with Low-Level I/O 147

19. DATA OUTPUT 152

Writing to Diary Files 154

Exporting Data to Text Data Files with Low-Level I/O 154

20. PLOTTING 156

Adding Title, Labels, Grid Lines, and Scaling on the Graph 158

Drawing Multiple Functions on the Same Graph 159

Setting Colors on Graph 160

Setting Axis Scales 161

Generating Sub-Plots 162

21. GRAPHICS 164

Drawing Bar Charts 164

Drawing Contours 165

Three-Dimensional Plots 167

tutorialspoint
SIMPLYEASYLEARNING

22. ALGEBRA 169

Solving Basic Algebraic Equations in MATLAB 169

Solving Quadratic Equations in MATLAB 171

Expanding and Collecting Equations in MATLAB 176

Expanding and Collecting Equations in Octave 177

Factorization and Simplification of Algebraic Expressions 179

23. CALCULUS 181

Calculating Limits 181

Verification of Basic Properties of Limits using Octave 184

Left and Right Sided Limits 185

24. DIFFERENTIAL 188

Verification of Elementary Rules of Differentiation 189

Derivatives of Exponential, Logarithmic, and Trigonometric Functions 193

Computing Higher Order Derivatives 198

Finding the Maxima and Minima of a Curve 200

Solving Differential Equations 204

25. INTEGRATION 206

Finding Indefinite Integral Using MATLAB 206

Finding Definite Integral Using MATLAB 210

26. POLYNOMIALS 216

Evaluating Polynomials 216

Polynomial Curve Fitting 217

27. TRANSFORMS 219

The Laplace Transform 219

The Inverse Laplace Transform 220

tutorialspoint
SIMPLYEASYLEARNING

The Fourier Transforms **222**

Inverse Fourier Transforms **224**

28. GNU OCTAVE TUTORIAL 225

MATLAB vs Octave **225**

29. SIMULINK 229

Using Simulink **230**

tutorialspoint
SIMPLYEASYLEARNING

1. OVERVIEW

MATLAB (matrix laboratory) 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

tutorialspoint
SIMPLYEASYLEARNING

2. ENVIRONMENT

Local Environment Setup

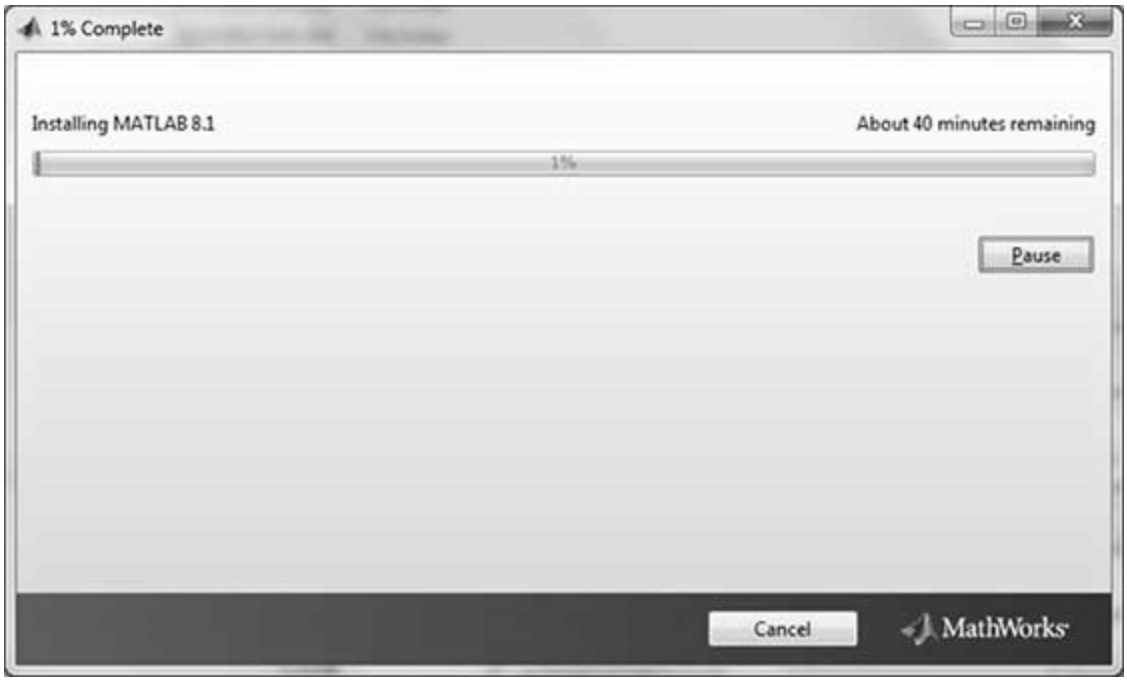
Setting up MATLAB environment is a matter of few clicks. The installer can be downloaded from http://in.mathworks.com/downloads/web_downloads:

MathWorks provides the licensed product, a trial version and a student version as well. You need to log into the site and wait a little for their approval.

After downloading the installer the software can be installed through few clicks.

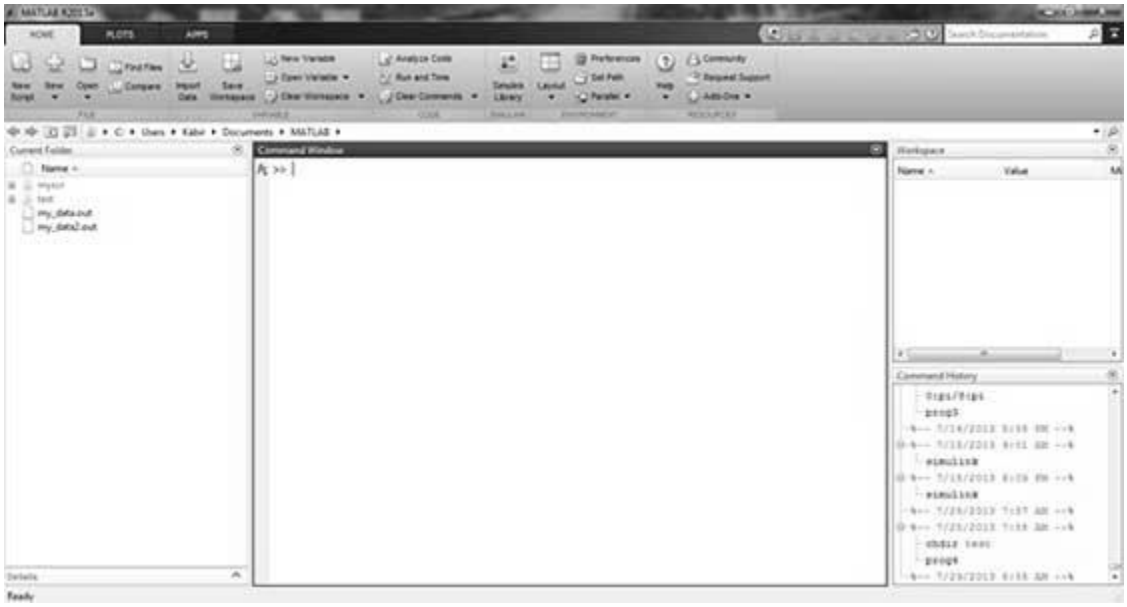


tutorialspoint
SIMPLYEASYLEARNING



Understanding the MATLAB Environment

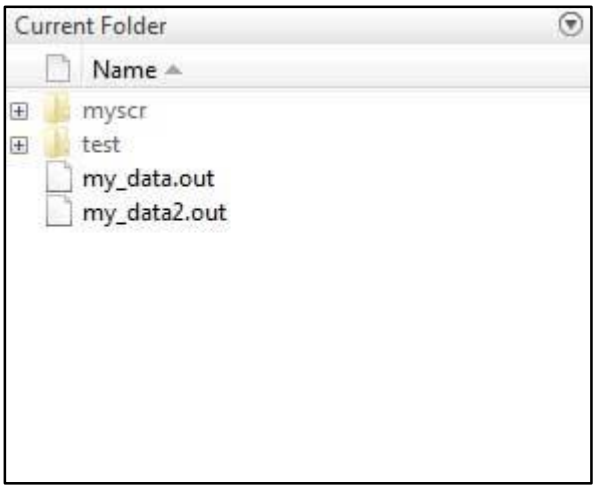
MATLAB development IDE can be launched from the icon created on the desktop. The main working window in MATLAB is called the desktop. When MATLAB is started, the desktop appears in its default layout:



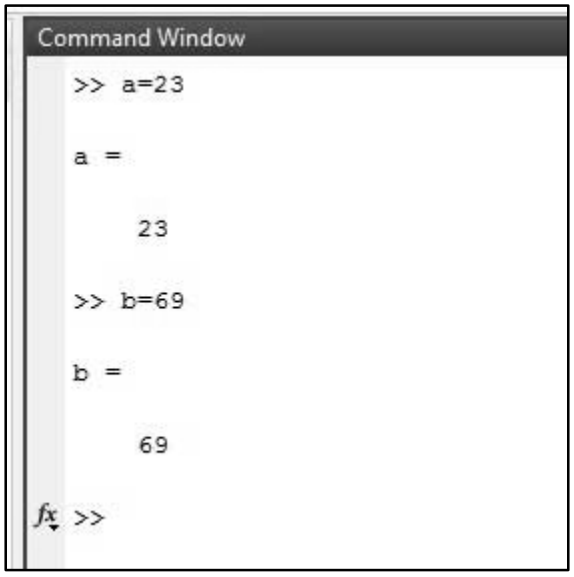
The desktop has the following panels:

tutorialspoint
SIMPLYEASYLEARNING

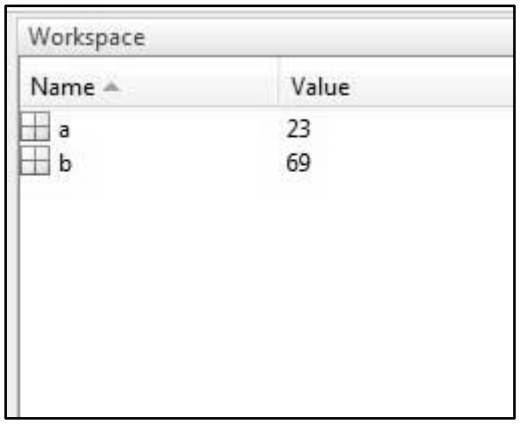
Current Folder - This panel allows you to access the project folders and files.



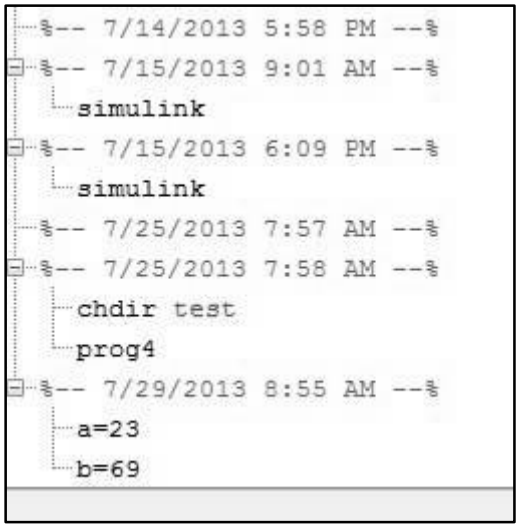
Command Window - This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



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



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



Set up GNU Octave

If you are willing to use Octave on your machine (Linux, BSD, OS X or Windows), then kindly download latest version from http://www.gnu.org/software/octave/download.html. You can check the given installation instructions for your machine

tutorialspoint
SIMPLYEASYLEARNING

3. BASIC SYNTAX

MATLAB environment behaves like a super-complex calculator. You can enter commands at the >> command prompt.

MATLAB is an interpreted environment. In other words, you give a command and MATLAB executes it right away.

Hands on Practice

Type a valid expression, for example,

```
5 + 5
```

And press ENTER

When you click the Execute button, or type Ctrl+E, MATLAB executes it immediately and the result returned is:

```
ans = 10
```

Let us take up few more examples:

```
3 ^ 2       % 3 raised to the power of 2
```

When you click the Execute button, or type Ctrl+E, MATLAB executes it immediately and the result returned is:

```
ans = 9
```

Another example,

```
sin(pi /2)    % sine of angle 90o
```

When you click the Execute button, or type Ctrl+E, MATLAB executes it immediately and the result returned is:

tutorialspoint
SIMPLYEASYLEARNING

```
ans = 1
```

Another example,

```
7/0             % Divide by zero
```

When you click the Execute button, or type Ctrl+E, MATLAB executes it immediately and the result returned is:

```
ans = Inf

warning: division by zero
```

Another example,

```
732 * 20.3
```

When you click the Execute button, or type Ctrl+E, MATLAB executes it immediately and the result returned is:

```
ans =  1.4860e+04
```

MATLAB provides some special expressions for some mathematical symbols, like pi for π, Inf for ∞, i (and j) for √-1 etc. **Nan** stands for 'not a number'.

Use of Semicolon (;) in MATLAB

Semicolon (;) indicates end of statement. However, if you want to suppress and hide the MATLAB output for an expression, add a semicolon after the expression.

For example,

```
x = 3;

y = x + 5
```

When you click the Execute button, or type Ctrl+E, MATLAB executes it immediately and the result returned is:

tutorialspoint
SIMPLYEASYLEARNING

```
y =  8
```

Adding Comments

The percent symbol (%) is used for indicating a comment line. For example,

```
x = 9      % assign the value 9 to x
```

You can also write a block of comments using the block comment operators % { and % }.

The MATLAB editor includes tools and context menu items to help you add, remove, or change the format of comments.

Commonly used Operators and Special Characters

MATLAB supports the following commonly used operators and special characters:

Operator	Purpose
+	Plus; addition operator.
-	Minus; subtraction operator.
*****	Scalar and matrix multiplication operator.
.*	Array multiplication operator.
^	Scalar and matrix exponentiation operator.
.^	Array exponentiation operator.
****	Left-division operator.
/	Right-division operator.

.	Array left-division operator.
./	Array right-division operator.
:	Colon; generates regularly spaced elements and represents an entire row or column.
( )	Parentheses; encloses function arguments and array indices; overrides precedence.
[ ]	Brackets; enclosures array elements.
.	Decimal point.
...	Ellipsis; line-continuation operator
,	Comma; separates statements and elements in a row
;	Semicolon; separates columns and suppresses display.
%	Percent sign; designates a comment and specifies formatting.
_	Quote sign and transpose operator.
._	Non-conjugated transpose operator.
=	Assignment operator.

Special Variables and Constants

MATLAB supports the following special variables and constants:

tutorialspoint
SIMPLYEASYLEARNING

Name	Meaning
ans	Most recent answer.
eps	Accuracy of floating-point precision.
i,j	The imaginary unit √-1.
Inf	Infinity.
NaN	Undefined numerical result (not a number).
pi	The number π

Naming Variables

Variable names consist of a letter followed by any number of letters, digits or underscore.

MATLAB is **case-sensitive**.

Variable names can be of any length, however, MATLAB uses only first N characters, where N is given by the function **namelengthmax**.

Saving Your Work

The **save** command is used for saving all the variables in the workspace, as a file with .mat extension, in the current directory.

For example,

```
save myfile
```

You can reload the file anytime later using the **load** command.

```
load myfile
```


tutorialspoint
SIMPLYEASYLEARNING

4. VARIABLES

In MATLAB environment, every variable is an array or matrix.

You can assign variables in a simple way. For example,

```
x = 3        % defining x and initializing it with a value
```

MATLAB will execute the above statement and return the following result:

```
x =

     3
```

It creates a 1-by-1 matrix named *x* and stores the value 3 in its element. Let us check another example,

```
x = sqrt(16)       % defining x and initializing it with an expression
```

MATLAB will execute the above statement and return the following result:

```
x =

     4
```

Please note that:

Once a variable is entered into the system, you can refer to it later.

Variables must have values before they are used.

When an expression returns a result that is not assigned to any variable, the system assigns it to a variable named ans, which can be used later.

For example,

```
sqrt(78)
```

MATLAB will execute the above statement and return the following result:

tutorialspoint
SIMPLYEASYLEARNING

```
ans =

    8.8318
```

You can use this variable ***ans***:

```
9876/ans
```

MATLAB will execute the above statement and return the following result:

```
ans =

   1.1182e+03
```

Let's look at another example:

```
x = 7 * 8;

y = x * 7.89
```

MATLAB will execute the above statement and return the following result:

```
y =

  441.8400
```

Multiple Assignments

You can have multiple assignments on the same line. For example,

```
a = 2; b = 7; c = a * b;
```

MATLAB will execute the above statement and return the following result:

```
c =

    14
```


tutorialspoint
SIMPLYEASYLEARNING

I have forgotten the Variables!

The **who** command displays all the variable names you have used.

```
who
```

MATLAB will execute the above statement and return the following result:

```
Your variables are:

a    ans  b    c    x    y
```

The **whos** command displays little more about the variables:

Variables currently in memory

Type of each variables

Memory allocated to each variable

Whether they are complex variables or not

```
whos
```

MATLAB will execute the above statement and return the following result:

```
  Name      Size            Bytes  Class     Attributes


  a         1x1                 8  double

  ans       1x1                 8  double

  b         1x1                 8  double

  c         1x1                 8  double

  x         1x1                 8  double

  y         1x1                 8  double
```

The **clear** command deletes all (or the specified) variable(s) from the memory.

```
clear x     % it will delete x, won't display anything

clear      % it will delete all variables in the workspace
```


tutorialspoint
SIMPLYEASYLEARNING

```
            %  peacefully and unobtrusively
```

Long Assignments

Long assignments can be extended to another line by using an ellipses (...). For example,

```
initial_velocity = 0;

acceleration = 9.8;

time = 20;

final_velocity = initial_velocity ...

    + acceleration * time
```

MATLAB will execute the above statement and return the following result:

```
final_velocity =

   196
```

The format Command

By default, MATLAB displays numbers with four decimal place values. This is known as ***short format***.

However, if you want more precision, you need to use the **format** command.

The **format long** command displays 16 digits after decimal.

For example:

```
format long

x = 7 + 10/3 + 5 ^ 1.2
```

MATLAB will execute the above statement and return the following result:

```
x =

  17.231981640639408
```


tutorialspoint
SIMPLYEASYLEARNING

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：https://d.book118.com/518012005013006054