Using MATLAB as a Calculator

For example, let's suppose you want to calculate the expression,
1 + 2 * 3. *You type it at the prompt command (>>) as follows:*

```
>> 1+2*3
ans =
        7
```

MATLAB uses a default variable ans, short for answer, to store the results of the current calculation.

To avoid this, you may assign a value to a variable or output argument name. For example,

```
>> x = 1+2*3
x =
7
```

**This variable name can always be used to refer to the results of the previous computations. Therefore, computing 4*x will*
result in

```
>> 4*x
ans =
28.0000
```

Table 1.1 gives the partial list of arithmetic operators.

Table 1.1: Basic arithmetic operators

SYMBOL	OPERATION	EXAMPLE
+	Addition	2 + 3
−	Subtraction	2 − 3
*	Multiplication	2 * 3
/	Division	2/3
\	left division	(2\4 = 2, 2\5 = 2.5)

To end your MATLAB session, type quit in the Command Window, or select File *Exit* ⟶ MATLAB in the desktop main menu.

Command Window and Basic Arithmetic

EXAMPLE 1-1

Use MATLAB to evaluate $5\left(\frac{3}{4}\right)+\frac{9}{5}$ and $4^3\left[\frac{3}{4}+\frac{9}{(2)3}\right]$

SOLUTION 1-1

The command required to find the value of the first expression is

```
>> 5*(3/4) + 9/5
ans =
5.5500
```

2nd expression:

```
>> r = 4^3
r =
64
>> s = 3/4 + 9/(2*3)
s =
2.2500
>> t=r*s
t =
144
```

```
>> 2+3
MATLAB returns the answer:
ans =
     5
Similary:
>> 5-3
ans =
     2
>> 3.59*2.34
ans =
    8.4006
>> 5/2
ans =
    2.5000
>> 2\5
ans =
    2.5000
>> 4^23.6
ans =
  1.6166e+014
```

$$2.3 * 10^3 \xrightarrow{in\ MATLAB} 2.3e3$$

$$4.6 * 10^{-7} \xrightarrow{in\ MATLAB} 4.6e-7$$

Adding the semicolon (;) to the end of the operation to suppress the output

```
>> 25*3;
>>
```

Also type:

```
>> 1/0
ans =
   Inf
>> -1/0
ans =
  -Inf
>> 0/0
ans =
   NaN
```

```
>> 2+3,4*6,6^8
ans =
     5
ans =
    24
ans =
     1679616
```

Table 1.1: Hierarchy of arithmetic operations

Precedence	Mathematical operations
First	The contents of all parentheses are evaluated ¯rst, starting from the innermost parentheses and working outward.
Second	All exponentials are evaluated, working from left to right
Third	All multiplications and divisions are evaluated, working from left to right
Fourth	All additions and subtractions are evaluated, starting

Example: $\frac{1}{2+3^2}+\frac{4}{5}\times\frac{6}{7}$

In MATLAB, it becomes

```
>> 1/(2+3^2)+4/5*6/7
ans  =
     0.7766
```

or, if parentheses are missing,

```
>> 1/2+3^2+4/5*6/7
ans  =
     10.1857
```

Controlling the appearance of floating point number

```
>> format short
>> x=-163.6667
```

If we want to see all 15 digits, we use the command `format long`

```
>> format long
>> x= -1.636666666666667e+002
```

Note - Up to now, we have let MATLAB repeat everything that we enter at the
prompt .To prevent MATLAB from echoing what we type, simply enter a semicolon (;) at
the end of the command. For example,

>> x=-163.6667; **enter**

and then ask about the value of x by typing,

```
>> x
x =
-163.6667
1
```

Entering multiple statements per line:

It is possible to enter multiple statements per line. Use commas (,) or semicolons (;) to enter more than one statement at once.

```
>> a=7; b=cos(a), c=cosh(a)
b    =
     0.6570
c    =
     548.3170
```

Miscellaneous commands

- To clear the Command Window, type `clc`
- To abort a MATLAB computation, type `ctrl-c`
- To continue a line, type ...

Getting help

On the other hand, information about any command is available by typing

>> help Command

Ex:

```
>> help sqrt enter
 SQRT   Square root.
   SQRT(X) is the square root of the elements of X. Complex
   results are produced if X is not positive.

   See also sqrtm, realsqrt, hypot.

   Overloaded methods:
     distributed/sqrt
     sym/sqrt

   Reference page in Help browser
     doc sqrt
```

Mathematical functions

MATLAB offers many predefined mathematical functions for technical computing which contains a large set of mathematical functions.

Table 2.1: Elementary functions

`cos(x)`	Cosine	`abs(x)`	Absolute value
`sin(x)`	Sine	`sign(x)`	Signum function
`tan(x)`	Tangent	`max(x)`	Maximum value
`acos(x)`	Arc cosine	`min(x)`	Minimum value
`asin(x)`	Arc sine	`ceil(x)`	Round towards $+\infty$
`atan(x)`	Arc tangent	`floor(x)`	Round towards $-\infty$
`exp(x)`	Exponential	`round(x)`	Round to nearest integer
`sqrt(x)`	Square root	`rem(x)`	Remainder after division
`log(x)`	Natural logarithm	`angle(x)`	Phase angle
`log10(x)`	Common logarithm	`conj(x)`	Complex conjugate

Table 2.2: Predefined constant values

`pi`	The π number, $\pi = 3.14159\ldots$
`i,j`	The imaginary unit i, $\sqrt{-1}$
`Inf`	The infinity, ∞
`NaN`	Not a number

Examples

We illustrate here some typical examples which related to the elementary functions previously defined.

As a first example, the value of the expression

$$y = e^{-a}\sin(x) + 10\sqrt{y},$$

for a = 5, x = 2, and y = 8 is computed by:

```
>> a = 5; x = 2; y = 8;
>> y = exp(-a)*sin(x)+10*sqrt(y)
y =
28.2904
```

The subsequent examples are

```
>> log(142)
ans =
4.9558
>> log10(142)
ans =
2.1523
```

Note the difference between the natural logarithm log(x) and the decimal logarithm (base 10) log10(x).

To calculate sin(¼=4) *and exp(10), we enter the following commands in MATLAB,*

```
>> sin(pi/4)
ans =
0.7071
>> exp(10)
ans =
2.2026e+004
```

Notes:

- ***Only use built-in functions on the right hand side of an expression. Reassigning the*** **value to a built-in function can create proble** $\sqrt{-1}$
- ***There are some exceptions. For example, i and j are pre-assigned to .However,*** **one or both of i or j are often used as loop indices.**
- ***To avoid any possible confusion, it is suggested to use instead ii or jj as loop indices.***