EXPERIMENT-TWO

MATLAB EXPRESSIONS

Like most other programming languages, MATLAB provides *mathematical expressions*, but unlike most programming languages, these expressions involve entire matrices. The building blocks of expressions are:

- Variables
- Numbers
- Operators
- Functions

1- Variables

MATLAB does not require any type declarations or dimension statements. When MATLAB encounters a new variable name, it automatically creates the variable and allocates the appropriate amount of storage. If the variable already exists, MATLAB changes its contents and, if necessary, allocates new storage. For example,

```
>> num_students = 25
```

creates a 1-by-1 matrix named num_students and stores the value 25 in its single element.

Variable names consist of a letter, followed by any number of letters, digits, or underscores. MATLAB uses only the first 31 characters of a variable name.
MATLAB is case sensitive; it distinguishes between uppercase and lowercase letters. A and a are *not* the same variable. To view the matrix assigned to any variable, simply enter the variable name.

2- Numbers

MATLAB uses conventional decimal notation, with an optional decimal point and leading plus or minus sign, for numbers. *Scientific notation* uses the letter e to specify a power-of-ten scale factor. *Imaginary numbers* use either i or j as a suffix. Some examples of legal numbers are

3	-99	0.0001
9.6397238	1.60210e-20	6.02252e23
1i	-3.14159j	3e5i

All numbers are stored internally using the *long* format specified by the floating-point standard. Floating-point numbers have a finite *precision* of roughly 16 significant decimal digits and a finite *range* of roughly 10^{-308} to 10^{+308}. Several special numbers provide values of useful constants.

pi	$\pi = 3.14159265\ldots$
i	Imaginary unit, $\sqrt{-1}$
j	Same as i
eps	Floating-point relative precision, 2^{-52} = 2.2204e-016
realmin	Smallest floating-point number, 2^{-1022} = 2.2251e-308
realmax	Largest floating-point number, 2^{1023} = 1.7977e+308
Inf	Infinity
NaN	Not-a-number

Infinity is generated by dividing a nonzero value by zero, or by evaluating well defined mathematical expressions that *overflow*, i.e., exceed realmax. Not-a-number is generated by trying to evaluate expressions like 0/0 or Inf-Inf that do not have well defined mathematical values.

The function names are not reserved. It is possible to overwrite any of them with a new variable, such as

```
>> eps = 1.e-6
```

and then use that value in subsequent calculations. The original function can be restored with

```
>> clear eps
```

3- <u>Operators</u>

Expressions use familiar arithmetic operators and precedence rules.

( )	Specify evaluation order
'	Complex conjugate or matrix transpose.
^	Power

\	Left division (i.e. $\frac{x}{y} = y \backslash x$)
/	Right Division (i.e. $\frac{x}{y} = x / y$)
*	Multiplication
-	Subtraction
+	Addition

4- Functions

MATLAB provides a large number of standard elementary mathematical functions, including abs, sqrt, exp, and sin. Taking the square root or logarithm of a negative number is not an error; the appropriate complex result is produced automatically. MATLAB also provides many more advanced mathematical functions, including Bessel and gamma functions. Most of these functions accept complex arguments. For a list of the elementary mathematical functions, type

```
>> help elfun
```

For a list of more advanced mathematical and matrix functions, type

```
>> help specfun
>> help elmat
```

So we can classify the elementary mathematical functions into:

- Trigonometric Functions.
- Exponential Functions.
- Complex Functions.
- Rounding and remainder Functions.
- User functions

4.1- Trigonometric Functions:

Sin(x)	Sine of x.
Sinh(x)	Hyperbolic sine (i.e. $\frac{e^{x} - e^{-x}}{2}$)

asin(x)	Inverse sine of x
asinh(x)	Inverse hyperbolic sine (i.e. $\ln(x+\sqrt{x^2+1})$)
cos(x)	Cosine of x.
cosh(x)	Hyperbolic cosine (i.e. $\frac{e^x+e^{-x}}{2}$)
acos(x)	Inverse cosine of x.
acosh	Inverse hyperbolic cosine (i.e. $\ln(x+\sqrt{x^2-1})$)
tan(x)	Tangent x.
tanh(x)	Hyperbolic tangent of x.
atan(x)	Inverse tangent of x.
atan2(y,x)	Four quadrant inverse tangent (i.e. $\tan^{-1}(\frac{y}{x})$)
atanh(x)	Inverse hyperbolic tangent (i.e. $\frac{1}{2}\ln(\frac{1+x}{1-x})$)
sec(x)	Secant x.
sech(x)	Hyperbolic secant x.
asec(x)	Inverse secant x.
asech(x)	Inverse hyperbolic secant x.
csc(x)	Cosecant x.
csch(x)	Hyperbolic cosecant x.
acsc(x)	Inverse cosecant x.
acsch(x)	Inverse hyperbolic cosecant x.
cot(x)	Cotangent x.
coth(x)	Hyperbolic cotangent x.
acot(x)	Inverse cotangent x.
acoth(x)	Inverse hyperbolic cotangent x.

4.2- Exponential Functions:

exp(x)	Exponential.
Log(x)	Natural logarithm.
log10(x)	Common (base 10) logarithm.
log2(x)	Base 2 logarithm and dissect floating point number.
pow2(x)	Base 2 power and scale floating point number.
realpow(x)	Power that will error out on complex result.
reallog(x)	Natural logarithm of real number.
realsqrt(x)	Square root of number greater than or equal to zero.

sqrt(x)	Square root.
nextpow2(x)	Next higher power of 2.

4.3- Complex Functions:

abs(x)	Absolute value.
angle(x)	Phase angle.
complex(x)	Construct complex data from real and imaginary parts.
conj(x)	Complex conjugate.
imag(x)	Complex imaginary part.
real(x)	Complex real part.
unwrap(x)	Unwrap phase angle.
isreal(x)	True for real array.
cplxpair(x)	Sort numbers into complex conjugate pairs.

4.4- Rounding and remainder Functions:

fix(x)	Round towards zero.
floor(x)	Round towards minus infinity.
ceil(x)	Round towards plus infinity.
round(x)	Round towards nearest integer.
mod(x,y)	Modulus (signed remainder after dividing x over y).
rem(x,y)	Remainder after division.
sign(x)	Signum.

Note: mod(x,y) and rem(x,y) are equal if x and y have the same sign, but differ by y if x and y have different signs.

4.5- User functions:

Some of the functions, like sqrt and sin, are *built-in*. They are part of the MATLAB core so they are very efficient, but you may want to implement your only function in M-files and even modify it if you want. The form that you use it to build your only function in M-file is:

function **output variables** = **function_name** (**input_variables**)
output variables = any relation with the **input variables**

Note that the function name will be the M-file name.

Exercises:

1. Use MATLAB to evaluate the following expressions:

(a) $2^{2\times 3} + 10^{-5}$
(b) $1.5 \times 10^{-4} + 2.5 \times 10^{-2}$
(c) $1000\,(1 + 0.15 / 12)^{60}$
(e) $(0.0000123 + 5.678 \times 10^{-3}) \times 0.4567 \times 10^{-4}$

2. Translate the following into MATLAB statement (make x=3.0024):
(a) Round x towards zero and store the result in i.
(b) Round x towards minus infinity and store the result in j.
(c) Round x towards plus infinity and store the result in k.
(d) Round x towards nearest integer and store the result in m.

3. Write an m-file program to calculate x_1 and x_2, where

$$x_1, x_2 = \frac{-b \mp \sqrt{b^2 - 4ac}}{2a}$$

Find the solution for :

(a) a = 2, b = -10, c = 12.
(b) a = 1, b = 2, c = 1.
(c) a = 6, b = 5, c = 8.

4. The steady-state current I flowing in a circuit that contains a resistance R = 5, capacitance C = 10, and inductance L = 4 in series is given by:

$$I = \frac{E}{\sqrt{R^2 + (2\pi\omega L - \frac{1}{2\pi\omega C})^2}}$$

Where E = 2 and ω = 2 are the input voltage and angular frequency respectively. Compute the value of I.

5. Find the value of the following by using a simplified form of function in one m-file program:

(a) $\ln(2\pi + \sqrt{4\pi^2 - 1})$

(b) $\frac{1}{2}\ln(\frac{1+1.5\pi}{1-1.5\pi})$

(c) $2^{15} + \frac{e^{2.5\pi} + e^{-2.5\pi}}{2}$

(d) $\tan^{-1}(\frac{4.3\pi}{5.4\pi}) + e^{2.3}$

6. Write a program to calculate sum and average of three numbers in functions called (“Summation”) and (‘Average”) respectively. Enter the numbers in the command window, call the functions and find the result.