**Algorithm**

An algorithm is a set of instructions designed to perform a specific task. This can be a simple process, such as multiplying two numbers, or a complex operation, such as playing a compressed video file. [Search engines](http://techterms.com/definition/searchengine) use proprietary algorithms to display the most relevant results from their search [index](http://techterms.com/definition/index) for specific [queries](http://techterms.com/definition/query).

**Computer program**

In computer programming, algorithms are often created as [functions](http://techterms.com/definition/function). These functions serve as small programs that can be referenced by a larger [program](http://techterms.com/definition/program). For example, an image viewing [application](http://techterms.com/definition/application) may include a library of functions that each use a custom algorithm to render different image [file formats](http://techterms.com/definition/file_format). An image-editing program may contain algorithms designed to process image data. Examples of image processing algorithms include cropping, resizing, sharpening, blurring, red-eye reduction, and color enhancement.

**Matlab Workspace Environment**

* **Command Window** (real life calculator)

The writing area that you will see when you start MATLAB is the command window. The command window is also useful if you just want to use MATLAB as a scientific calculator or as a graphing tool.

* **Work Space**

Store the variable and their values in a list.

* **Current Directory**

Shows the folders that the user worked on them currently.

* **Command History**:

The commands you type in the command window are stored by MATLAB and can be viewed in the *Command History* window. To repeat a command you have already used, you can simply double-click on the command in the history window.

**Operation in Matlab**

|  |  |
| --- | --- |
| **Operation, function or constant** | **MATLAB command** |
| + (addition) | + |
| - (subtraction) | - |
| × (multiplication) | \* |
| / (division) | / |
| **|**x**|** (absolute value of x) | abs(x) |
| square root of x | sqrt(x) |
| ex | exp(x) |
| ln x (natural log) | log(x) |
| log10 x (base 10 log) | log10(x) |
| sin x | sin(x) |
| cos x | cos(x) |
| tan x | tan(x) |
| cot x | cot(x) |
| arcsin x | asin(x) |
| arccos x | acos(x) |
| arctan x | atan(x) |
| arccot x | acot(x) |
| n! (n factorial) | gamma(n+1) |
| e (2.71828...) | exp(1) |
| **** (3.14159265...) | pi |
| ***i***   (imaginary unit, sqrt(-1)) | i |

**Exercises**

Compute the following expressions using MATLAB:

* 3cos(pi)
* 1+1+1/2+1/6+1/24-e
* ln (1000+2pi-2)
* e i pi
* The number of combinations in which 12 persons can stand in line. (Hint: Use factorials.)

**Obtaining Help on MATLAB commands**

To obtain help on any of the MATLAB commands, you simply need to type

**help <command>**

at the command prompt. For example, to obtain help on the gamma function, we type at the command prompt:

**help gamma**

Try this now. You may also get help about commands using the "Help Desk", which can be accessed by selecting the MATLAB Help option under the Help menu.

**Variables in MATLAB**

We can easily define our own variables in MATLAB. Let's say we need to use the value of 3.5sin(2.9) repeatedly. Instead of typing 3.5\*sin(2.9) over and over again, we can denote this variable as x by typing the following:

**x=3.5\*sin(2.9)**

(Please try this in MATLAB.) Now type

**x+1**

and observe what happens. Note that we did not need to declare x as a variable that is supposed to hold a floating point number as we would need to do in most programming languages.

Often, we may not want to have the result of a calculation printed-out to the command window. **To supress this output, we put a semi-colon at the end of the command**; MATLAB still performs the command in "the background". If you defined x as above, now type

**y=2\*x;**

**y**

and observe what happened.

In many cases we want to know what variables we have declared. We can do this by typing **whos**. Alternatively, we can view the values by opening the "Workspace" window. This is done by selecting the Workspace option from the View menu. If you want to erase all variables from the MATLAB memory, type **clear**. To erase a specific variable, say x, type **clear x**. To clear two specific variables, say x and y, type **clear x y**, that is separate the different variables with a space. Variables can also be cleared by selecting them in the Workspace window and selecting the delete option.

**Creating M-Files**

**1) Script M-File**

Scripts are the simplest kind of program file because they have no input or output arguments. They are useful for automating series of MATLAB® commands, such as computations that you have to perform repeatedly from the command line or series of commands you have to reference.

**You can open a new script in the following ways:**

* Highlight commands from the Command History, right-click, and select **Create Script**.
* Click the **New Script** http://www.mathworks.com/help/releases/R2015a/matlab/matlab_prog/new_script_ts_16.png button on the **Home** tab.

**You can run the code in your file using either of these methods:**

* Type the script name your file on the command line and press **Enter**
* Click the **Run** http://www.mathworks.com/help/releases/R2015a/matlab/matlab_prog/run_ts_16.png button on the **Editor** tab.

**2) Function M-File**

function [y1,...,yN] = myfun(x1,...,xM) declares a function named myfun that accepts inputs x1,...,xM and returns outputsy1,...,yN. This declaration statement must be the first executable line of the function.

Save the function code in a text file with a .m extension. The name of the file should match the name of the first function in the file. Valid function names begin with an alphabetic character, and can contain letters, numbers, or underscores.

Example:

* Dude function that takes x and return y=x^2+cos(x).