Al-Mustaqbal University College
Department of Medical Instrumentation Techniques Engineering
Class: Third
Subject: Arguments and return values,
M-file, input-output statement
Lecturer: Dr. Ali Kareem Abba
Lecture: (7)

Arguments and return values

Arguments: The values provided as inputs to a command. And it's accepting a variable number of inputs or outputs, check for valid values. Additionally, you can check how many inputs or outputs your function receives, the names of input variables, and more.

Returns: The results provided by the computer in response to a command.

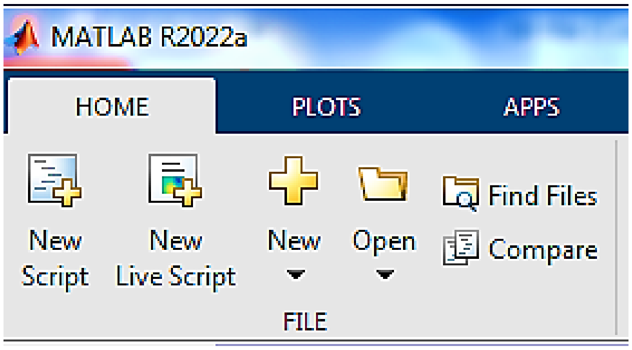
M-file

***Script File*:** Group of MATLAB commands placed in a text file with a text editor. MATLAB can open and execute the commands exactly as if they were entered at the MATLAB prompt. The term "script" indicates that MATLAB reads from the "script" found in the file. Also called "M-files," as the filenames must end with the extension '.m', e.g. example1.m.

M-files are text files and may be created and modified with any text editor. The steps to create a script are:

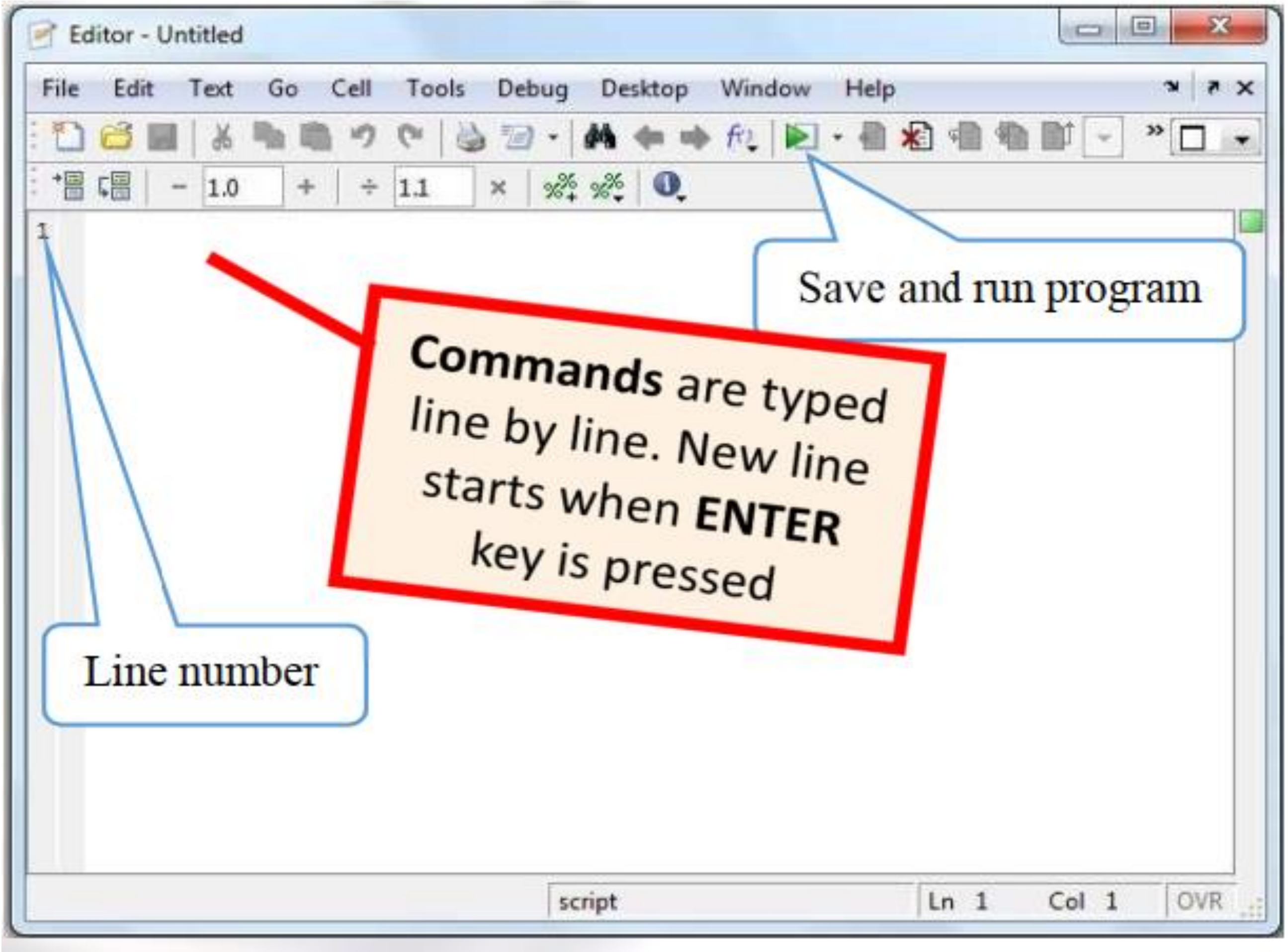
1) Click on **New Script** icon on the MATLAB toolbar

2) Press keys (Ctrl + N)



Script File

M-Files have a filename extension of " **.m** " and can be created or modified by using **Editor/Debugger Window**.

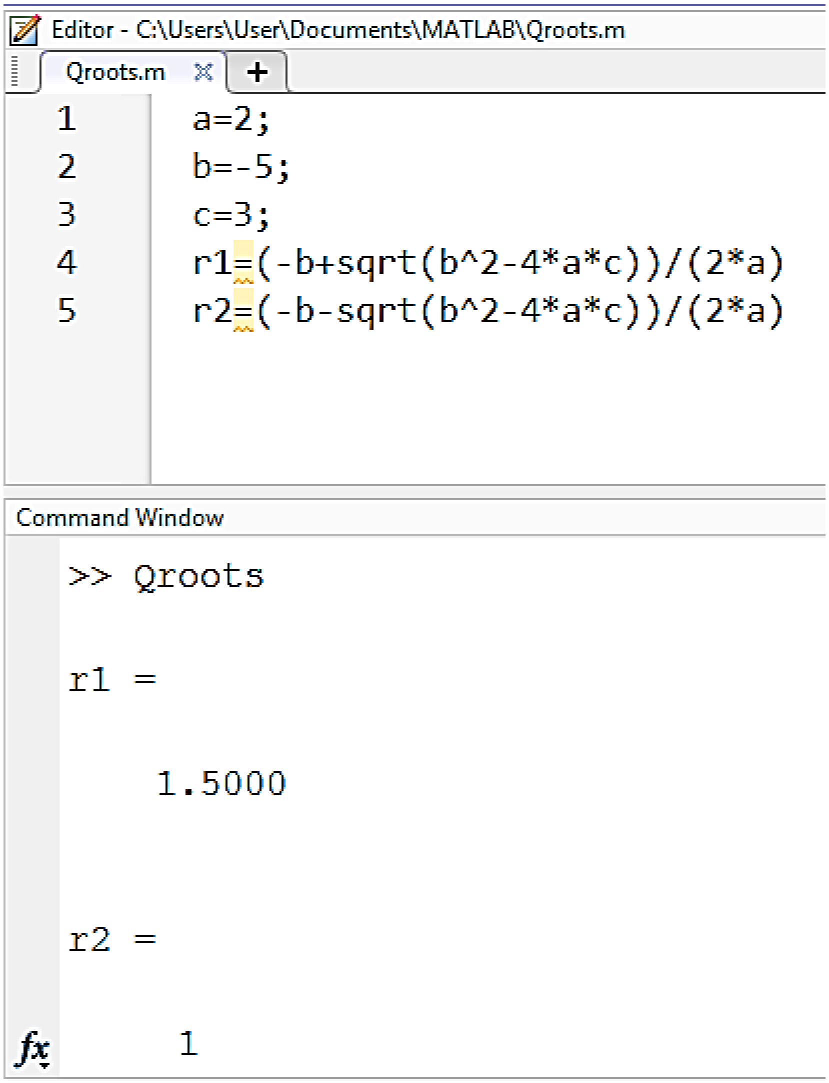


Commands are typed line by line. New line starts when **ENTER** key is pressed

Example

Write a program (m-file) and named it "**Qroots**.m" to find the quadratic equation roots:

$$2x^2 - 5x + 3 = 0$$



```
a=2;
b=-5;
c=3;
r1=(-b+sqrt(b^2-4*a*c))/(2*a)
r2=(-b-sqrt(b^2-4*a*c))/(2*a)
```

```
>> Qroots

r1 =

    1.5000


r2 =

     1
```

Notes :

- When file is executed, All its variables are displayed in workspace window.
- It is useful to use functions such as (clc , clear , format ,…) in script file to improve the results.

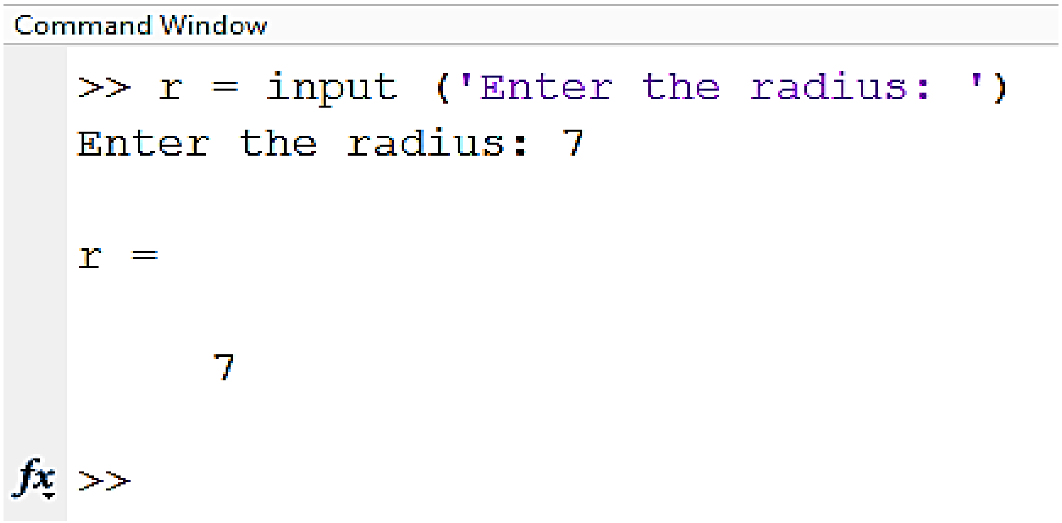
Input-output statement

The script would be much more useful if it were more general; for example, if the value of the radius could be read from an external source rather than being assigned in the script. Also, it would be better to have the script print the output in a nice, informative way. Statements that accomplish these tasks are called input/output statements, or I/O for short. With examples of input and output statements will be shown here from the Command Window, these statements will make the most sense in scripts.

Input function

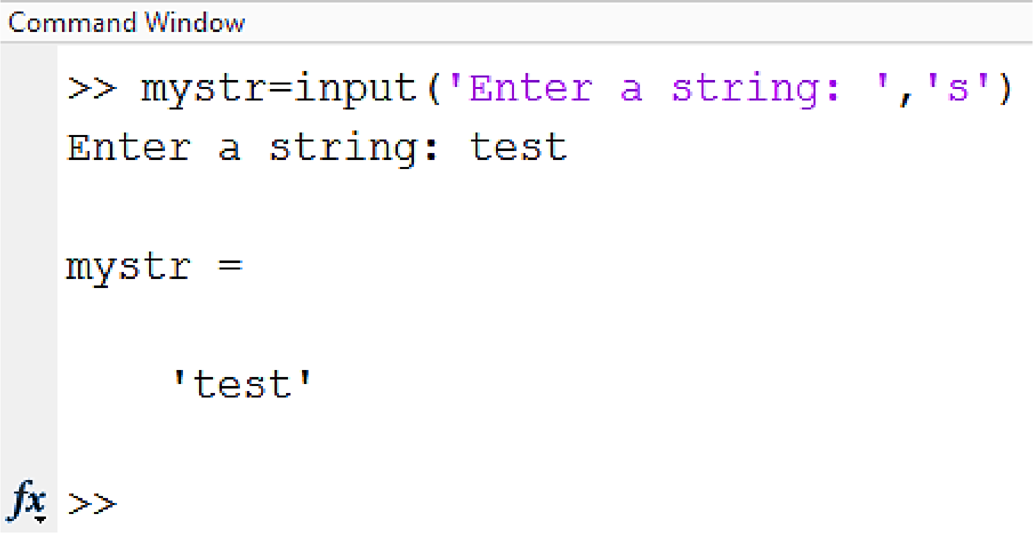
The simplest input function in MATLAB is called input. The input function is used in an assignment statement. To call it, a string is passed, which is the prompt that will appear on the screen, and whatever the user types will be stored in the variable named on the left of the assignment statement. To make it easier to read the prompt, put a **colon (:)** and then a **space** after the prompt.

EX1: Input statements read in values from the default or standard input device (usually the keyboard)



```
>> r = input ('Enter the radius: ')
Enter the radius: 7

r =

     7

>>
```

EX2: This method only works well if the user actually enters a number (or the name of a variable)

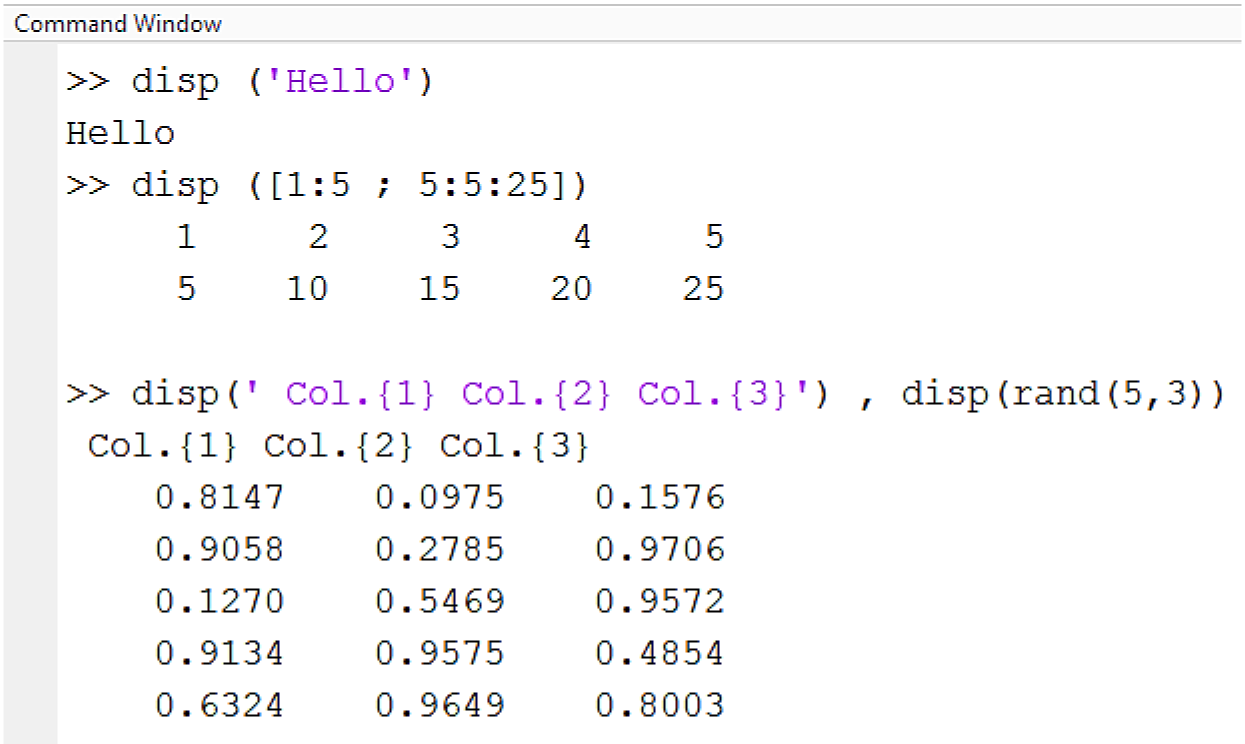


```
>> mystr=input('Enter a string: ','s')
Enter a string: test

mystr =

    'test'

>>
```

Output Statements: disp

The simplest output function in MATLAB is **disp**, which is used to display the result of an expression or a string *without* assigning any value to the default variable *ans*. However, **disp** does *not* allow *formatting*.

For examples:



```
>> disp ('Hello')
Hello
>> disp ([1:5 ; 5:5:25])
     1     2     3     4     5
     5    10    15    20    25

>> disp(' Col.{1} Col.{2} Col.{3}') , disp(rand(5,3))
 Col.{1} Col.{2} Col.{3}
    0.8147    0.0975    0.1576
    0.9058    0.2785    0.9706
    0.1270    0.5469    0.9572
    0.9134    0.9575    0.4854
    0.6324    0.9649    0.8003
```

Output Statements: fprintf

The ***fprintf*** function, first a string (called the ***format string***) is passed, which contains any text to be printed as well as formatting information for the expressions to be printed.

In this example, the **%d** is an example of format information. The **%d** is sometimes called a ***placeholder***; it specifies where the value of the expression that is after the string is to be *printed*. The character in the placeholder is called the ***conversion character,*** and it specifies the *type of value* that is being printed.

Placeholder	**Description**	**Example**
%d	Format as a *integer*	**>> fprintf ('%d', 4^5)** **1024>>**
%f	Format as a *floating point* value	**>> fprintf ('%f', sqrt(90.25))** **9.500000>>**
%g	Format as the form (*no trailing zero most compact*)	**>> fprintf ('%g', sqrt(90.25))** **9.5>>**
%e	Format as a floating point value in *scientific* notation	**>> fprintf ('%e', pi)** **3.141593e+000>>**
%s	Format as a *string*	**>> fprintf ('%s', '3*8/2')** **3*8/2>>**
\n	Insert a *new line* in the output string	**>> fprintf ('Welcome! \n this is MATLAB')** **Welcome!** **this is MATLAB>>**
\t	Insert a *tab* in the output string	**>> fprintf ('Welcome! \t this is MATLAB')** **Welcome! this is MATLAB>>**

Notes:

1- Its important adding the character \n at the end of output string in order to avoid the prompt (>>) from stick to the result as shown before. The character \n can also use in input function, for example:

```
>> h = input ('Enter \n The shape height :')
Enter
The shape height :
```

2- The character \n in form ‘\n\n’ use to get blank line in output, for example:

```
>> fprintf ('Hello \n\n This is MATLAB \n')
Hello


This is MATLAB
```

H.W 4

If x=[1 5 9; 2 7 4], then

a) display the last two elements by using **disp** command.

b) display the sum of each row as show below

The sum of 1st row =

The sum of 2nd row =