Morteza Kazemi | 97243054

Amir Masoud Shaker | 97243081

11/07/2021

Homework #2

# Q1

Input Arguments of cylinder2: (height, radius)

Output Arguments of cylinder2: [area, volume]

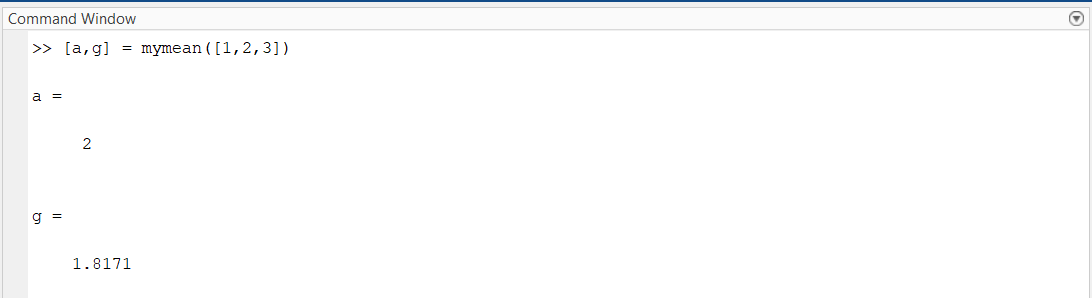
Name of a function must start with an alphabetic character to be valid.

|  |  |
| --- | --- |
| Name | Description |
| Local functions | In a function file, the first function is called the main function.  This function is visible to functions in other files and it can be called from the command line. Additional functions within the file are called local functions, and they can occur in any order after the main function.  Local functions are only visible to other functions in the same file. |
| Nested functions | A nested function is a function that is completely contained within a parent function.  The main difference between nested functions and other types of functions is that they can access and modify variables that are defined in their parent functions.  Notes:  To nest any function in a program file, all functions in that file must use an end statement.  A nested function can not be defined inside any control statements such as if/elseif/else, case/switch, try/catch, for, while.  A nested function can be called directly by name or using a function handle that you create with the @ operator.  All of the variables in nested functions or the functions that contain them must be explicitly defined. |
| Private functions | Private functions are useful to limit the scope of a function.  A function can be designated as private by storing it in a subfolder with the name private. So, the function is available only to functions and scripts in the folder immediately above the private subfolder. |
| Anonymous functions | An anonymous function is a function that is not stored in a program file, but is associated with a variable whose data type is function handle.  Anonymous functions can accept multiple inputs and return one output.  They can contain only a single executable statement. |

The examples are in the following:

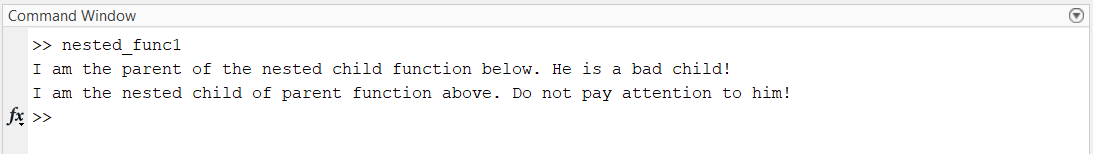
Local:





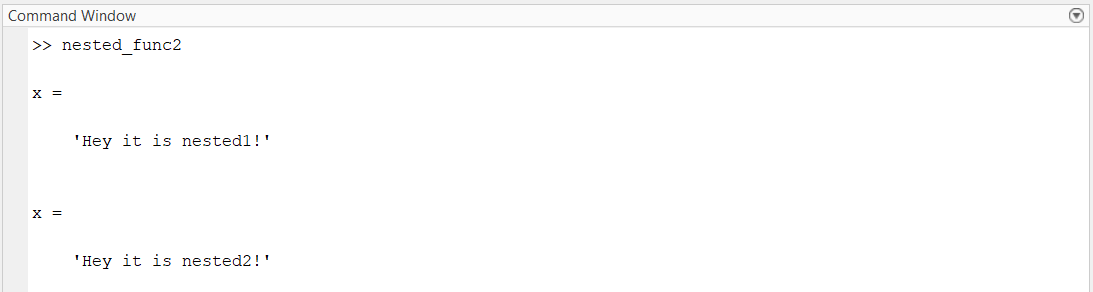
Nested1:



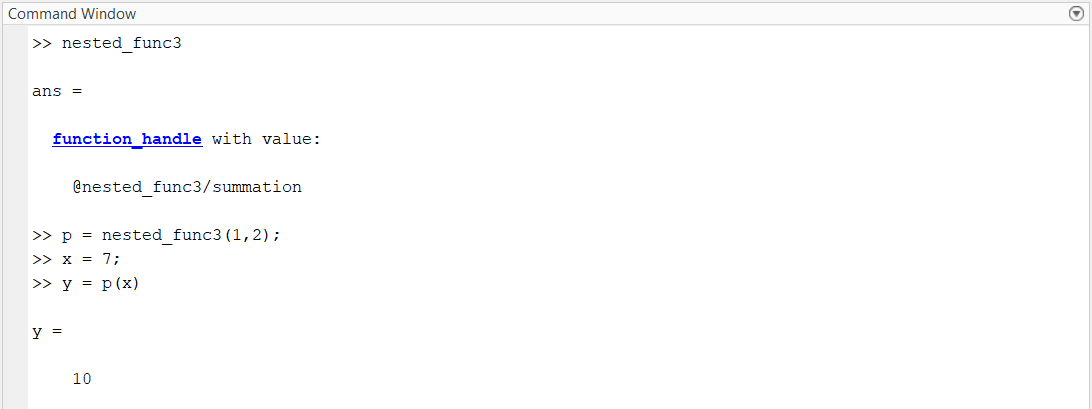


Nested2:



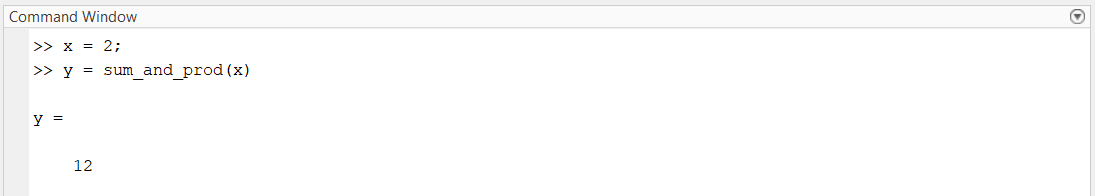


Nested3:



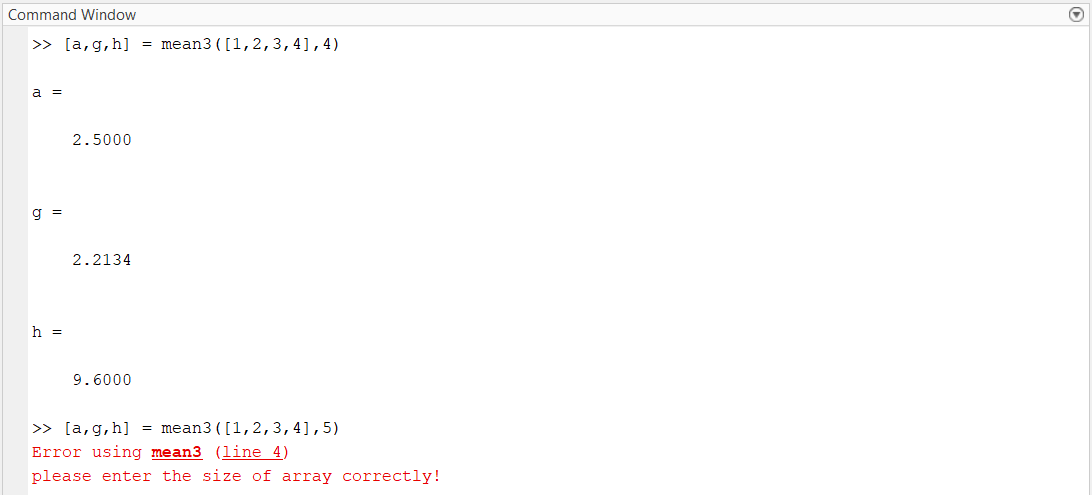
Anonymous:





# Q2





|  |  |  |
| --- | --- | --- |
| Syntax | Meaning | Description |
| 1. cd 2. cd newFolder 3. oldFolder = cd(newFolder) | Change current folder | 1. This syntax displays the current folder. 2. This syntax changes the current folder to newFolder 3. This syntax returns the existing current folder to oldFolder, and changes the current folder to newFolder. |



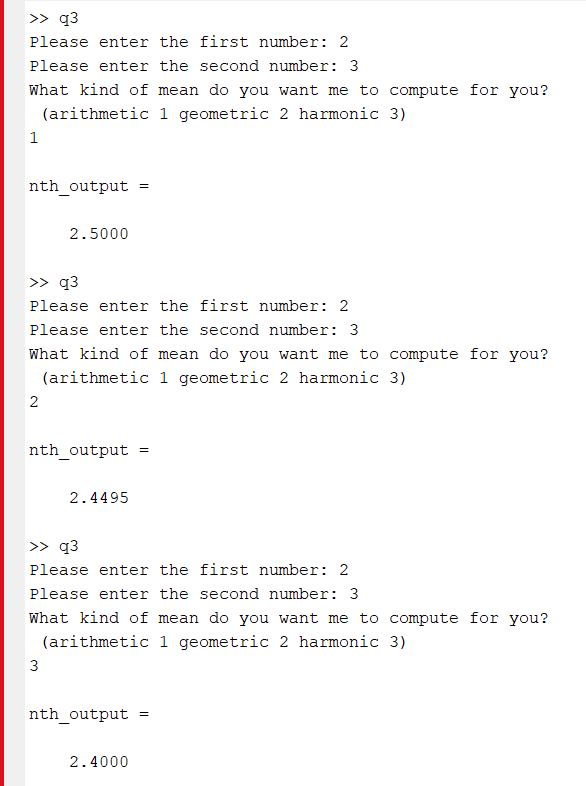
|  |  |
| --- | --- |
|  |  |



|  |  |
| --- | --- |
|  |  |

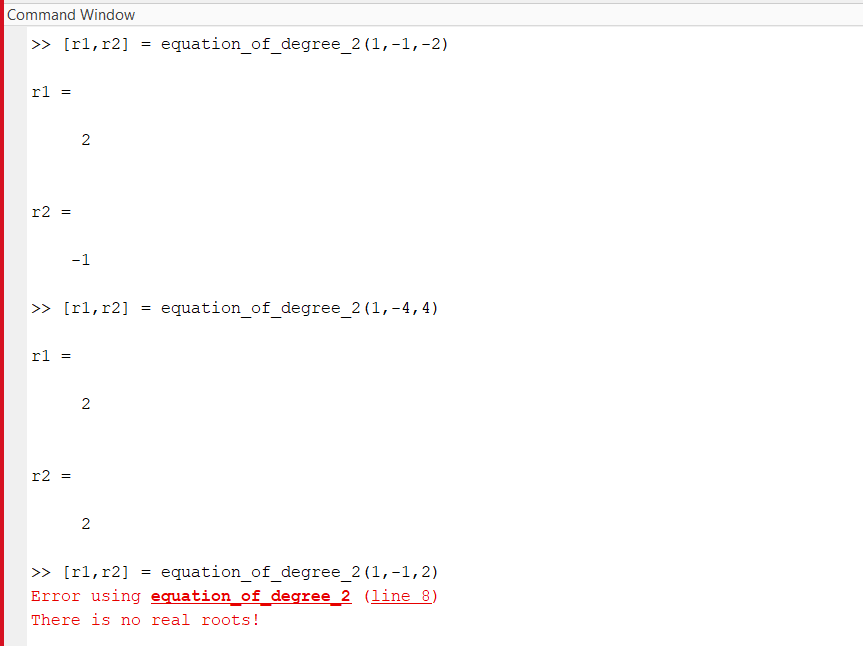
# Q3





# Q4



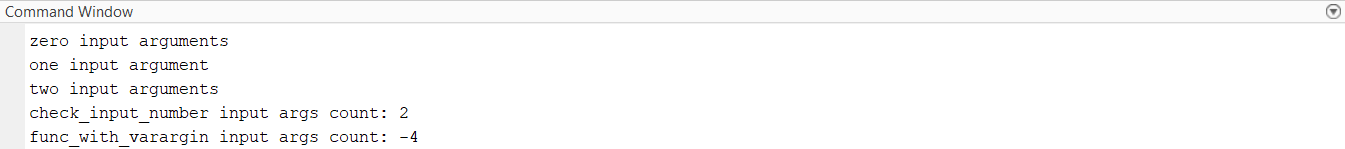


# Q5

|  |  |  |
| --- | --- | --- |
| Syntax | Meaning | Description |
| 1. nargin 2. nargin(fun) | Number of function input arguments | 1. nargin returns the number of function input arguments given in the call to the currently executing function. Use this syntax in the body of a function only. When using an arguments validation block, the value returned by nargin within a function is the number of positional arguments provided when the function is called. 2. nargin(fun) returns the number of input arguments that appear in the fun function definition. If the function includes varargin in its definition, then nargin returns the negative of the number of inputs. For example, if function myFun declares inputs a, b, and varargin, then nargin('myFun') returns -3.   If the input argument refers to a function that uses an arguments validation block, then the returned value is the number of declared positional arguments in the function definition as a non-negative value. |

Examples:

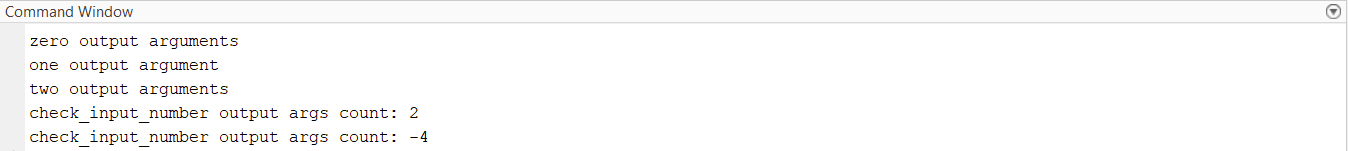




|  |  |  |
| --- | --- | --- |
| Syntax | Meaning | Description |
| 1. nargout 2. nargout(fun) | Number of function output arguments | 1. nargout returns the number of function output arguments specified in the call to the currently executing function. Use this syntax in the body of a function only. 2. nargout(fun) returns the number of outputs that appear in the fun function definition. If the function includes varargout in its definition, then nargout returns the negative of the number of outputs. For example, if function myFun declares outputs y, z, and varargout, then nargout('myFun') returns -3. |

Examples:





# Q6

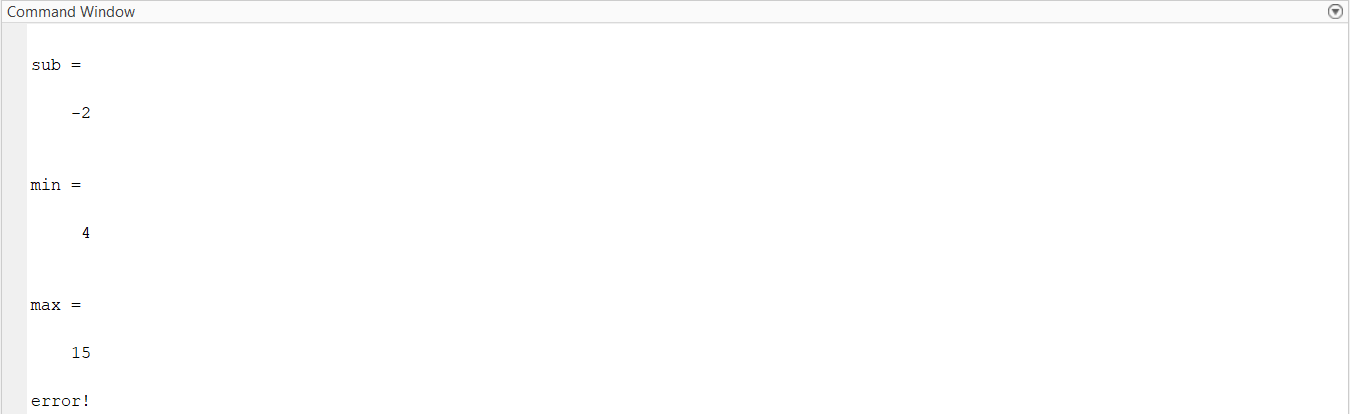
The details of varargin are explained in Q5 and Q8. The function is :



The script written to test the function is as follows:



The output of running the script in the command window:



# Q7

Here is the information we need for the commands. Examples of using these commands are also provided in the following.

|  |  |  |
| --- | --- | --- |
| Syntax | Meaning | Description |
| 1. tic 2. timerVal = tic | Start stopwatch timer | 1. tic works with the toc function to measure elapsed time. The tic function records the current time, and the toc function uses the recorded value to calculate the elapsed time. 2. timerVal = tic stores the current time in timerVal so that you can pass it explicitly to the toc function. Passing this value is useful when there are multiple calls to tic to time different parts of the same code. timerVal is an integer that has meaning only for the toc function. |

**Notes:**

* Consecutive tic calls overwrite the internally recorded starting time.
* The [clear](https://www.mathworks.com/help/releases/R2021a/matlab/ref/clear.html)(clear command line) function does not reset the starting time recorded by a tic function call!

|  |  |  |
| --- | --- | --- |
| Syntax | Meaning | Description |
| 1. toc 2. toc(timerVal) 3. elapsedTime = toc 4. elapsedTime = toc(timerVal) | Read elapsed time from stopwatch | 1. toc reads the elapsed time since the stopwatch timer started by the call to the tic function. Matlab reads the internal time at the execution of the toc function and displays the elapsed time since the most recent call to the tic function without an output. The elapsed time is expressed in seconds. 2. toc(timerVal) displays the elapsed time since the call to the tic function corresponding to timerVal. 3. elapsedTime = toc returns the elapsed time since the most recent call to the tic function. 4. elapsedTime = toc(timerVal) returns the elapsed time since the call to the tic function corresponding to timerVal. |

**Notes:**

* Consecutive toc  calls with no input return the elapsed time since the most recent call to tic. This property enables you to take multiple measurements from a single point in time.
* Consecutive calls to the toc function with the same timerVal input return the elapsed time since the tic function call that corresponds to timerVal.

**Input Arguments:**

Value of the internal timer saved from a previous call to the tic function and used by toc function are specified as a scalar of type uint64.

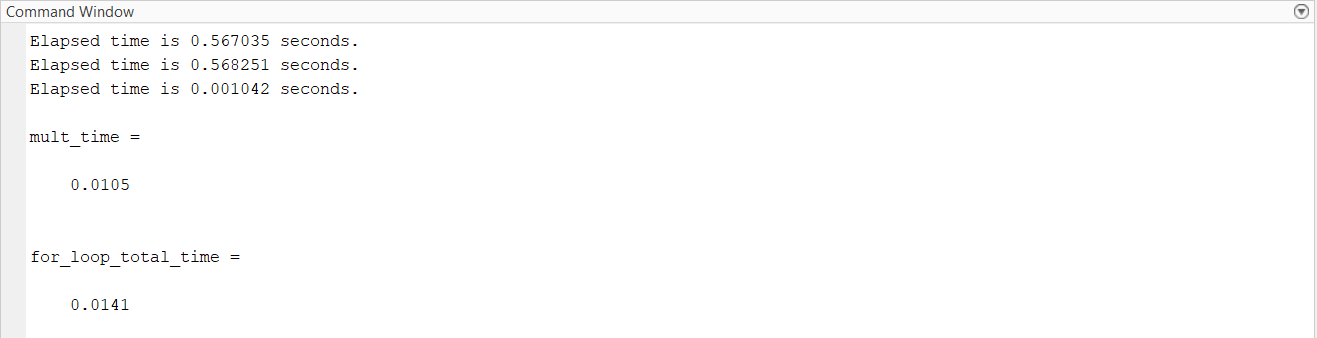
Examples:



All the details are explained in the code through comments.

The outputs are shown below:

Outputs:



# Q8

Here is the information we need for the commands. Examples of using these commands are also provided in the following.

|  |  |  |
| --- | --- | --- |
| Syntax | Meaning | Description |
| 1. [varargin](https://www.mathworks.com/help/releases/R2021a/matlab/ref/varargin.html?doclanguage=en&nocookie=true&prodfilter=ML%20SL%205G%20AE%20AT%20AA%20AU%20DR%20AS%20BI%20CM%20VP%20CT%20CF%20DA%20DB%20DF%20DD%20DH%20NN%20DO%20DS%20ET%20EC%20FH%20IT%20FI%20PO%20FL%20GD%20GC%20HD%20ES%20IE%20IA%20IP%20IC%20LP%20LS%20MG%20ME%20CO%20MJ%20MR%20MW%20AM%20MP%20MB%20MT%20NV%20OT%20OP%20DM%20PD%20AR%20BD%20BS%20CD%20CS%20PW%20PM%20RA%20RL%20RB%20RF%20RK%20RO%20RC%20RR%20SI%20TF%20SX%20SG%20SB%20SE%20SS%20LD%20PS%20SH%20MS%20VR%20VV%20CI%20RT%20SK%20SD%20CV%20SO%20DV%20WT%20PL%20XP%20SR%20RQ%20SZ%20HW%20EL%20SF%20ST%20SM%20ZC%20ID%20TA%20UV%20VE%20VN%20VT%20WA%20LH%20WL&docviewer=helpbrowser&docrelease=R2021a&s_cid=pl_webdoc&loginurl=https%3A%2F%2Flocalhost%3A31515%2Ftoolbox%2Fmatlab%2Fmatlab_login_framework%2Fweb%2Findex.html%3Fsnc%3DRWRRIS&searchsource=mw&snc=ZOUG6F&container=jshelpbrowser#bta08tf-1) | Variable-length input argument list | varargin is an input variable in a function definition statement that enables the function to accept any number of input arguments.  When the function executes, varargin is a 1-by-N cell array, where N is the number of inputs that the function receives after the explicitly declared inputs. If the function receives no inputs after the explicitly declared inputs, then varargin is an empty cell array.  Note: Specify varargin using lowercase characters, and include it as the last input argument after any explicitly declared inputs. |

|  |  |  |
| --- | --- | --- |
| Syntax | Meaning | Description |
| 1. varargout | Variable-length output argument list | varargout is an output variable in a function definition statement that enables the function to return any number of output arguments.  When the function executes, varargout is a 1-by-N cell array, where N is the number of outputs requested after the explicitly declared outputs. Inside of a function, varargout is an uninitialized variable and is not preallocated.  Note: Specify varargout using lowercase characters, and include it as the last output argument after any explicitly declared outputs. |

Examples:



All the details are explained in the code through comments.

The outputs are shown below:

Outputs:

