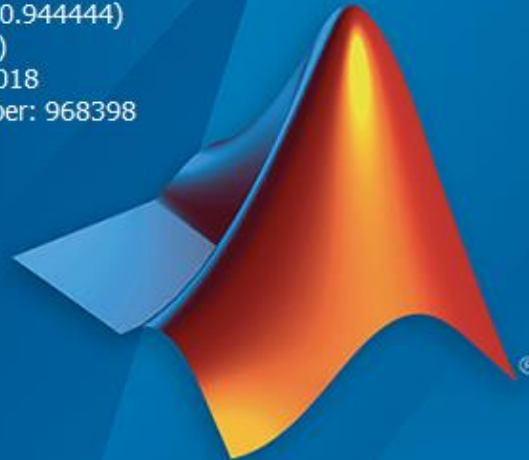


# Computer Programming



R2018b (9.5.0.944444)  
64-bit (win64)  
August 28, 2018  
License Number: 968398



# MATLAB®

Professional License

© 1984-2018 The MathWorks, Inc. Protected by U.S and international patents. See [mathworks.com/patents](http://mathworks.com/patents). MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](http://mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.



**R2018b**

# TIOBE Index for October 2018

The Importance Of Being Earnest  
Programming Community Index (TIOBE)

Oct 2018	Oct 2017	Change	Programming Language	Ratings	Change
1	1		Java	17.801%	+5.37%
2	2		C	15.376%	+7.00%
3	3		C++	7.593%	+2.59%
4	5	⬆	Python	7.156%	+3.35%
5	8	⬆	Visual Basic .NET	5.884%	+3.15%
6	4	⬇	C#	3.485%	-0.37%
7	7		PHP	2.794%	+0.00%
8	6	⬇	JavaScript	2.280%	-0.73%
9	-	⬆	SQL	2.038%	+2.04%
10	16	⬆	Swift	1.500%	-0.17%
11	13	⬆	MATLAB	1.317%	-0.56%
12	20	⬆	Go	1.253%	-0.10%
13	9	⬇	Assembly language	1.245%	-1.13%
14	15	⬆	R	1.214%	-0.47%
15	17	⬆	Objective-C	1.202%	-0.31%
16	12	⬇	Perl	1.168%	-0.80%
17	11	⬇	Delphi/Object Pascal	1.154%	-1.03%
18	10	⬇	Ruby	1.108%	-1.22%
19	19		PL/SQL	0.779%	-0.63%
20	18	⬇	Visual Basic	0.652%	-0.77%

Reference: <https://www.tiobe.com/tiobe-index/>

---

## Chapter 01:

# Starting with MATLAB

---

Starting MATLAB, MATLAB Windows

---

Working In The Command Window

---

Arithmetic Operations With Scalars

---

Display Formats

---

Elementary Math Built-in Functions

---

Defining Scalar Variables

---

Useful Commands For Managing Variables

---

Script Files

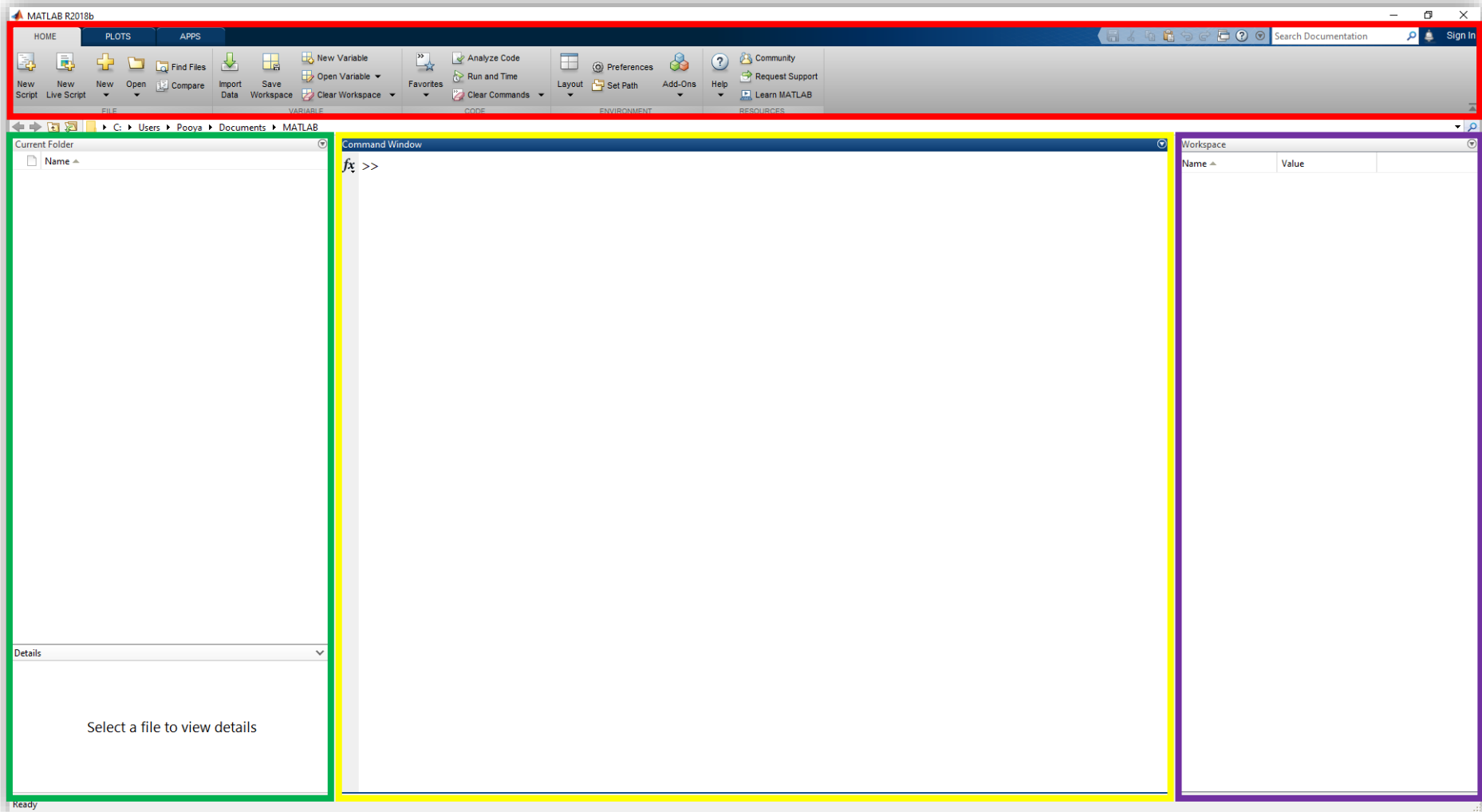
---

Examples Of MATLAB Applications

---

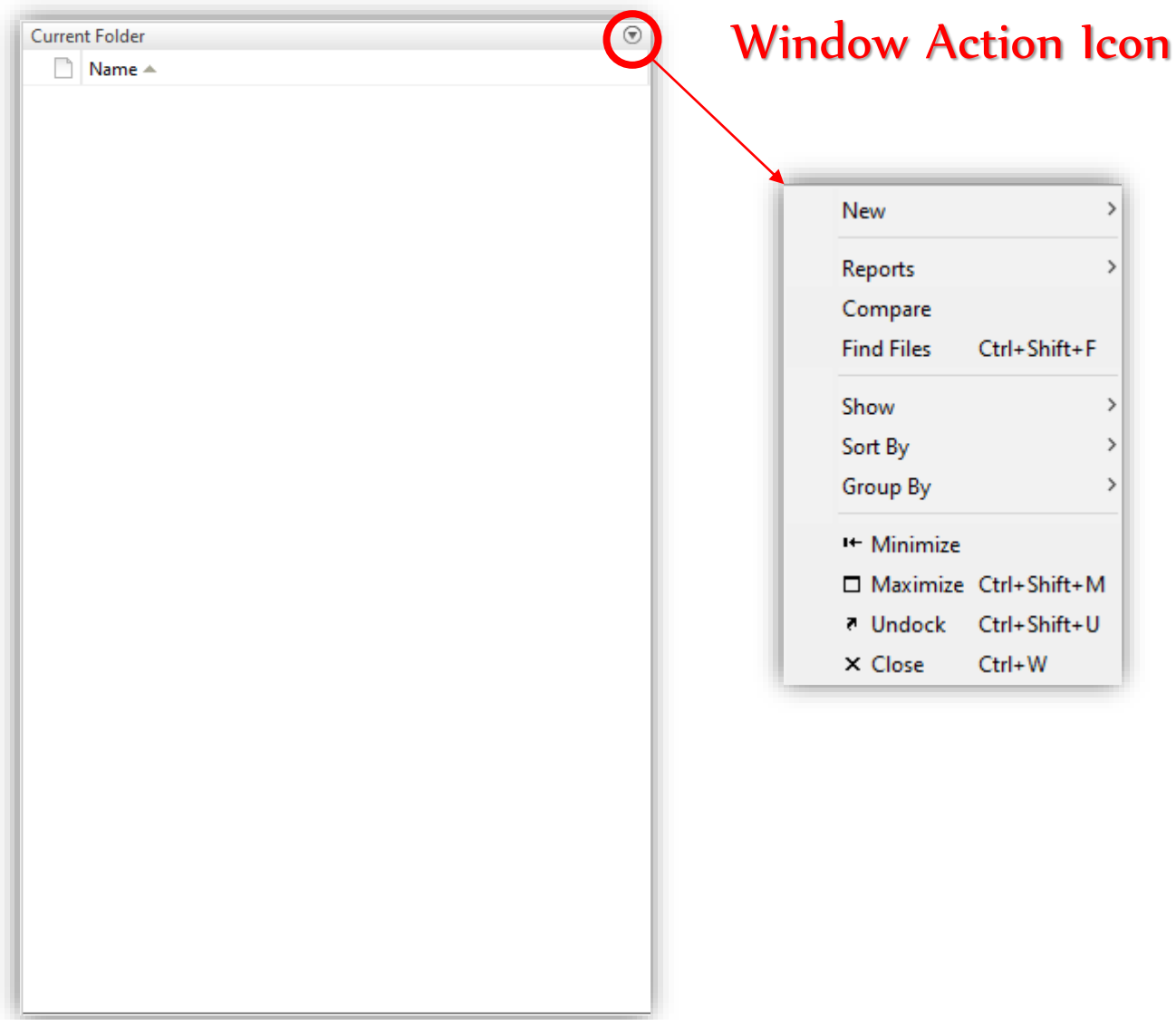
## Chapter 01

# Starting MATLAB, MATLAB Windows

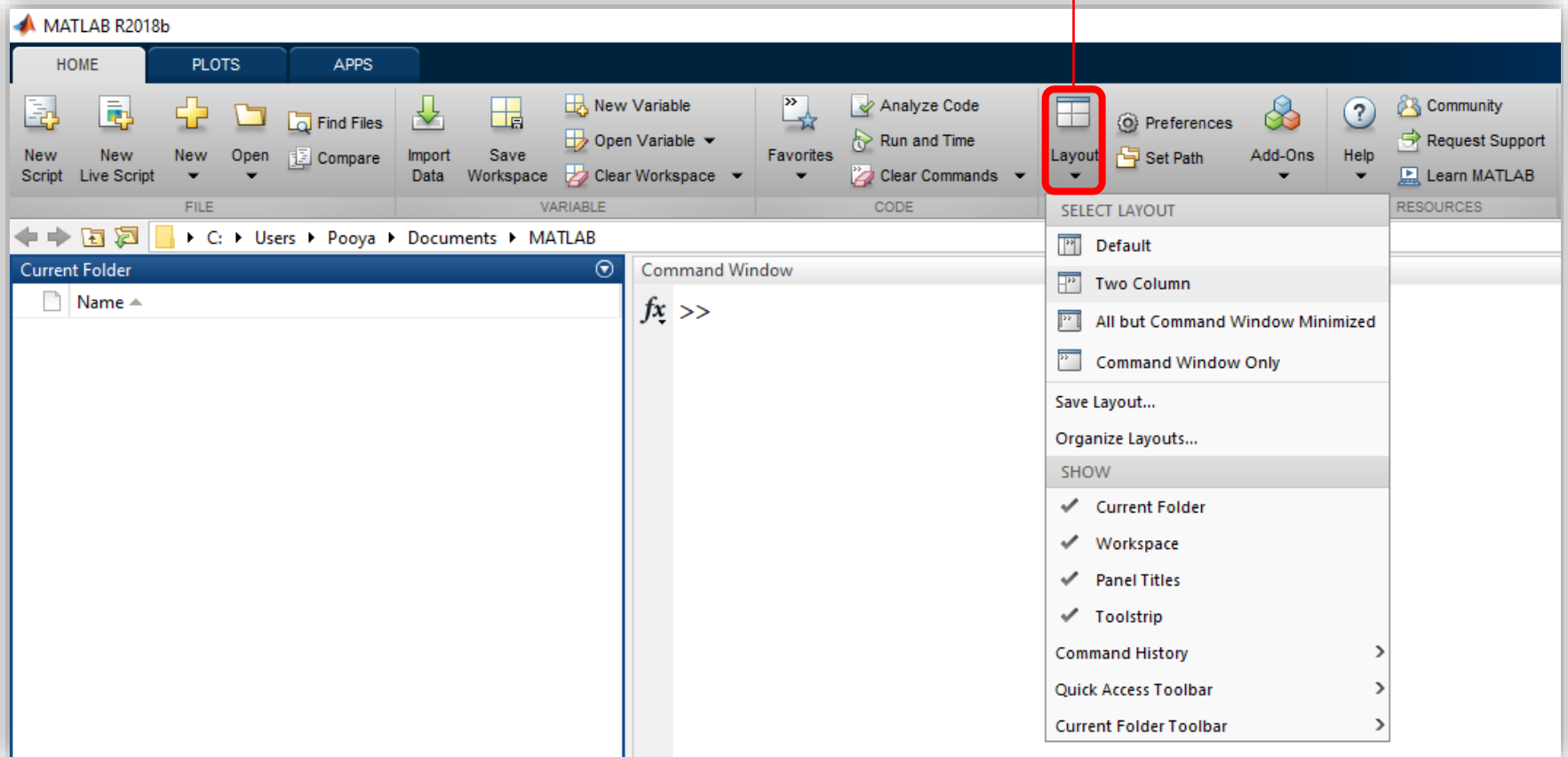


Window	Purpose
Current Folder Window	Shows the files in the current folder.
Command Window	Main window, enters variables, runs programs.
Workspace Window	Provides information about the variables that are stored.
Command History Window	Logs commands entered in the Command Window.
Editor Window	Creates and debugs script and function files.
Figure Window	Contains output from graphic commands.
Help Window	Provides help information.

# Starting MATLAB, MATLAB Windows



Layout Icon

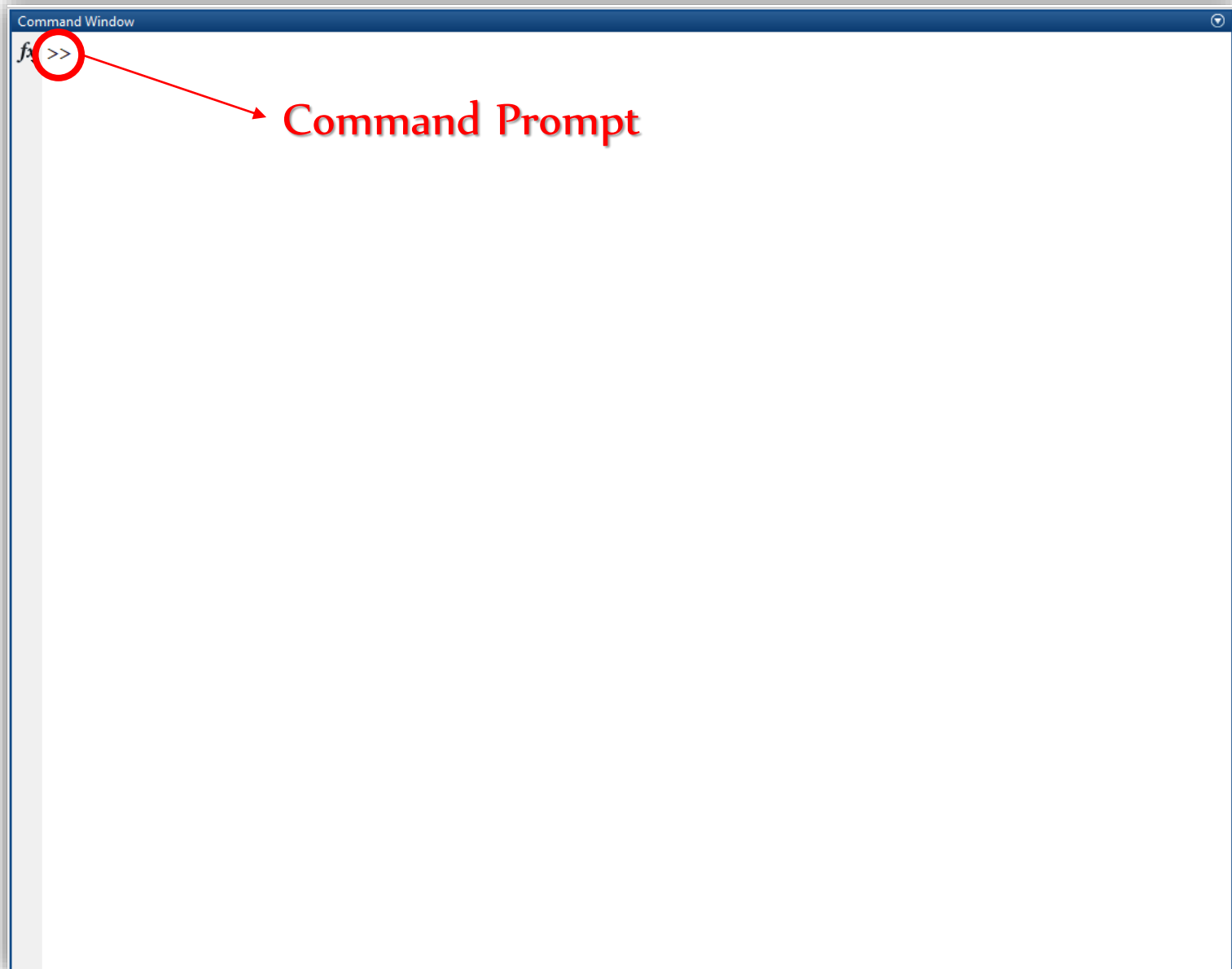


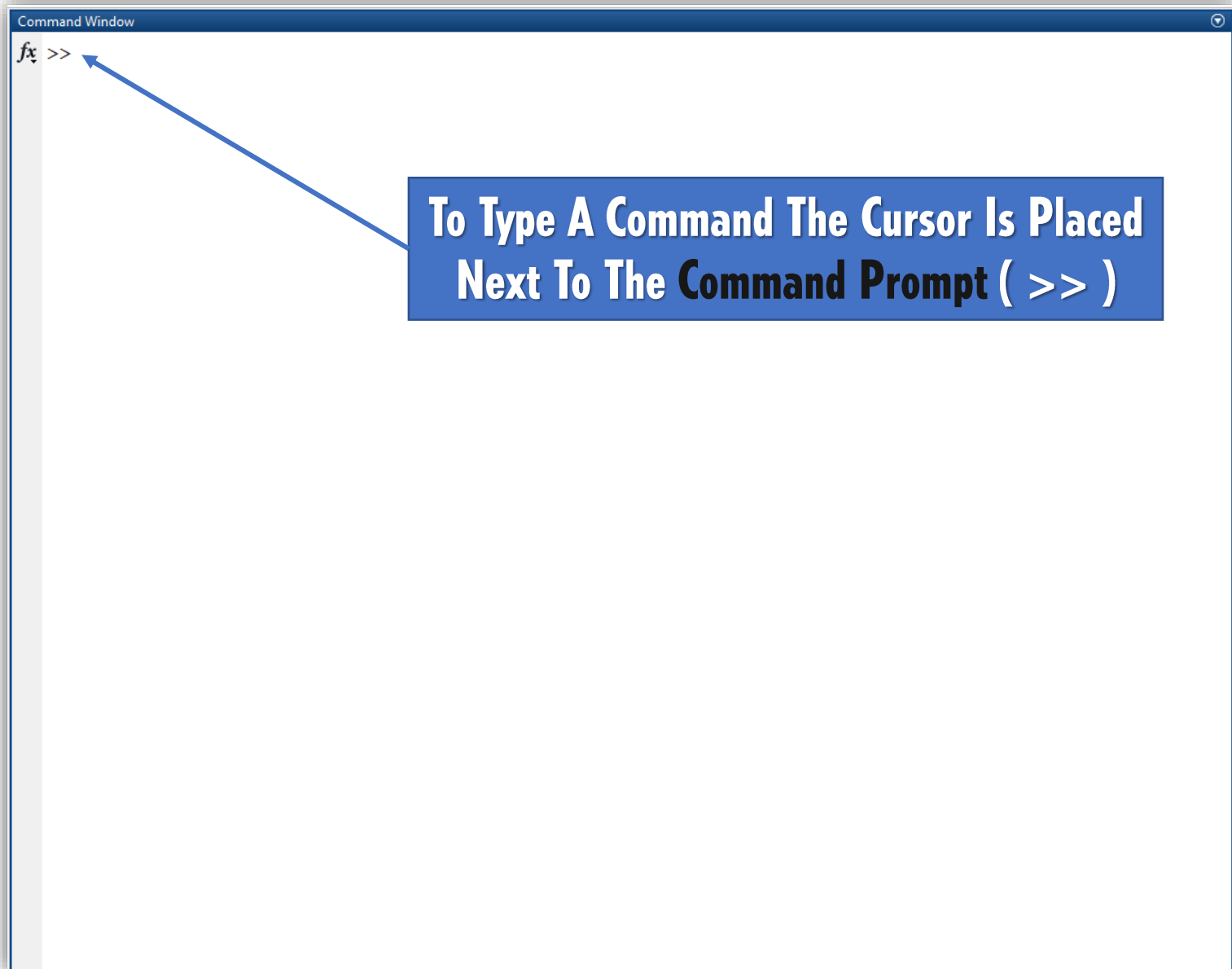


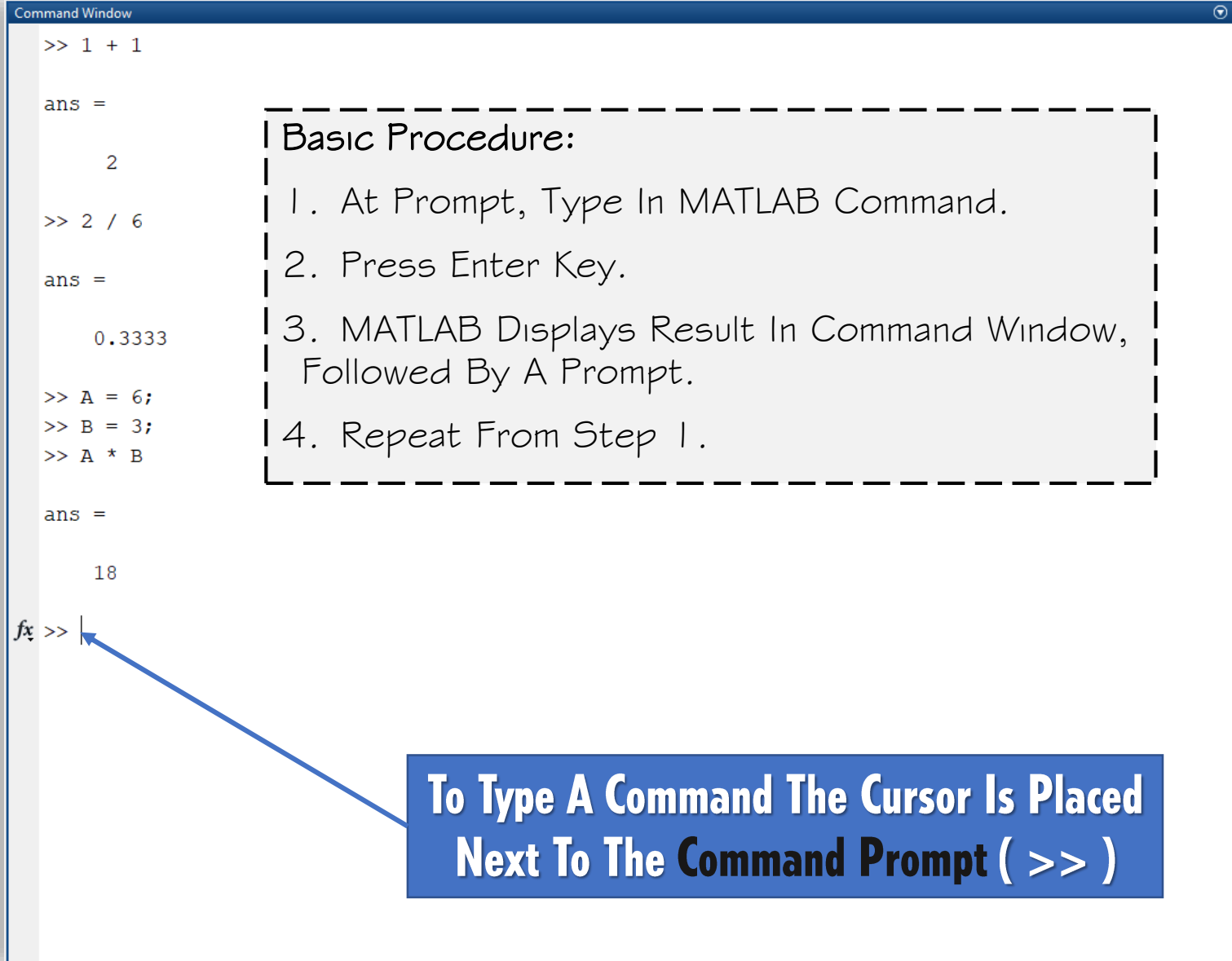
Command Window

 >>

- Execute Commands
- Open Other Windows
- Run Programs That You've Written
- Manage The MATLAB Software







The screenshot shows the MATLAB Command Window with the following text:

```
>> 1 + 1  
  
ans =  
  
2  
  
>> 2 / 6  
  
ans =  
  
0.3333  
  
>> A = 6;  
>> B = 3;  
>> A * B  
  
ans =  
  
18  
  
fx >> |
```

A dashed box highlights the following text:

**Basic Procedure:**

1. At Prompt, Type In MATLAB Command.
2. Press Enter Key.
3. MATLAB Displays Result In Command Window, Followed By A Prompt.
4. Repeat From Step 1.

A blue arrow points from a text box to the cursor in the Command Prompt.

**To Type A Command The Cursor Is Placed Next To The Command Prompt ( >> )**

The screenshot displays the MATLAB Command Window and Command History. The Command Window shows three separate calculations, each with its result assigned to the variable `ans`. The first calculation is `1 + 1` resulting in `2`. The second is `2 + 2` resulting in `4`. The third is `1 + 1, 2 + 2` resulting in `2`. The Command History window shows the same three commands. A context menu is open over the Command History, listing standard editing actions (Cut, Copy, Delete, Undo Delete) and MATLAB-specific actions (Evaluate Selection, Create Script, Create Live Script, Create Favorite, Set Error Indicator). At the bottom, a diagram shows four arrow keys (Up, Left, Down, Right) with an arrow pointing from the Down arrow key to the Command Window's input line, which is labeled `fx >>`.

```
Command Window
```

```
>> 1 + 1  
  
ans =  
  
     2  
  
>> 2 + 2  
  
ans =  
  
     4  
  
>> 1 + 1, 2 + 2  
  
ans =  
  
     2  
  
ans =  
  
     4  
  
>> 1 + 1, ...  
2 + 2  
  
ans =  
  
     2  
  
ans =  
  
     4
```

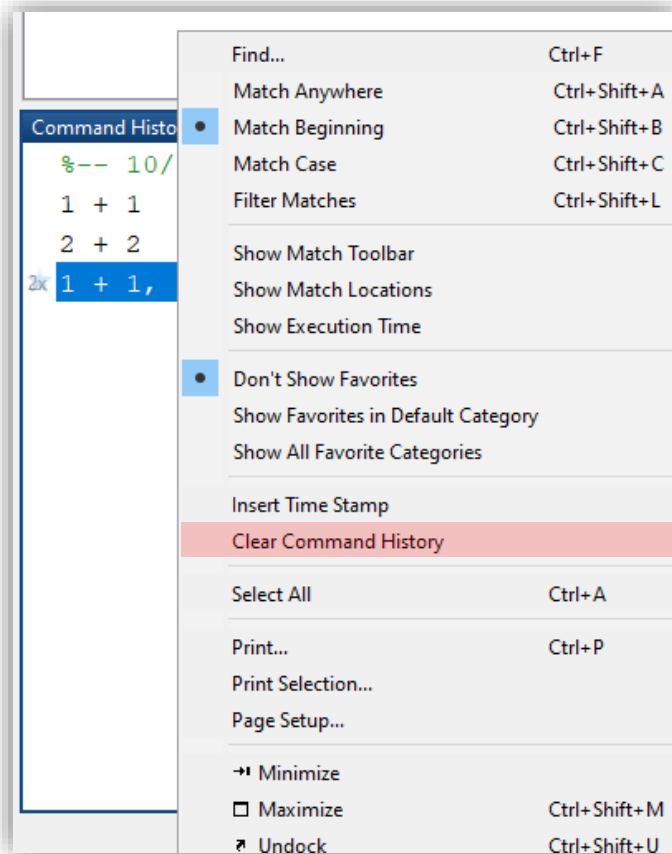
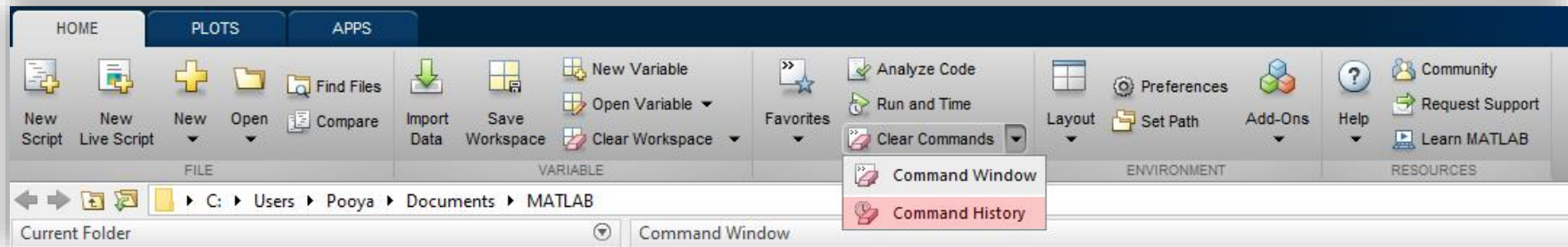
```
Command History
```

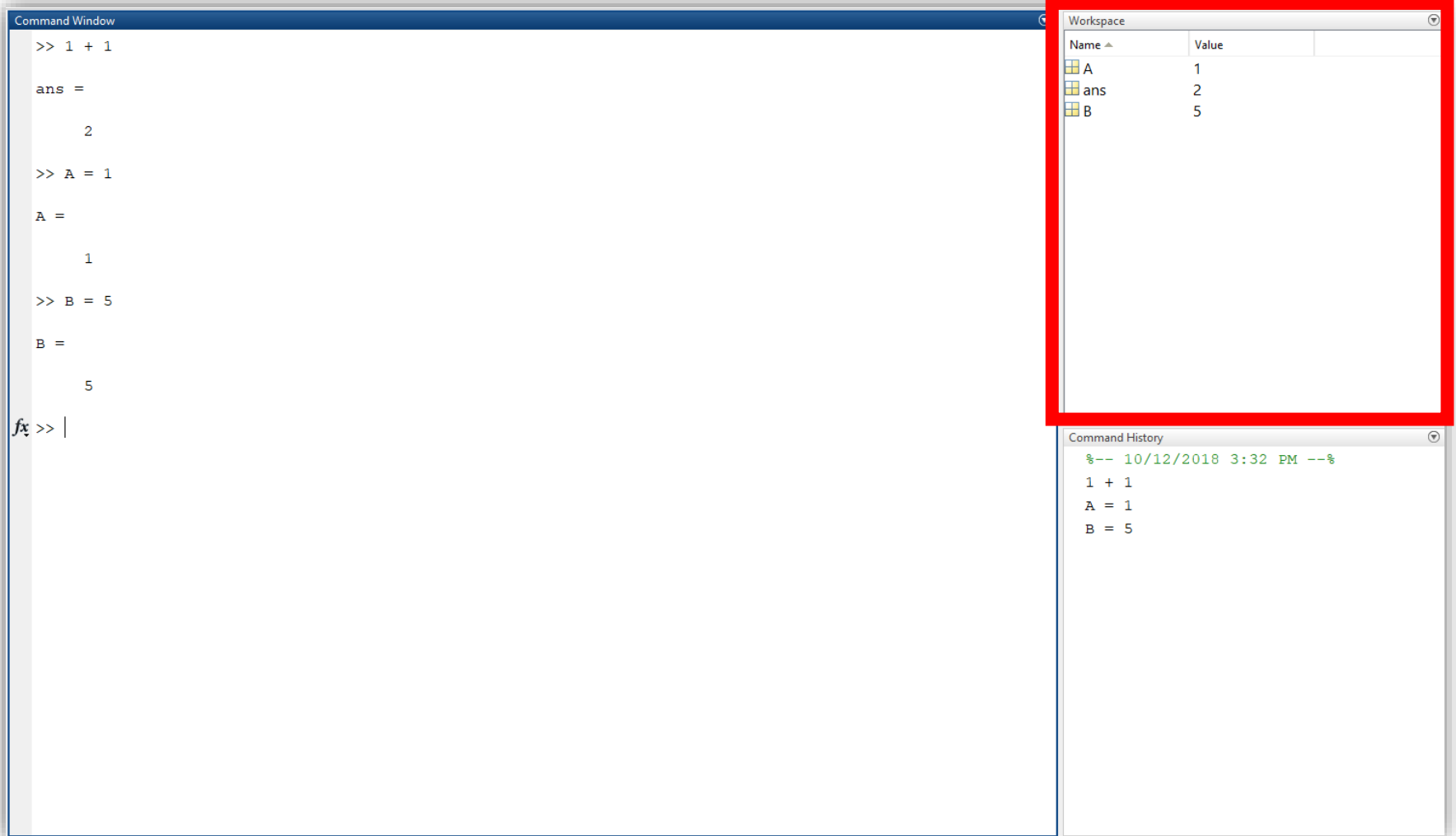
```
%-- 10/12/2018 2:30 PM --%  
1 + 1  
2 + 2  
1 + 1, 2 + 2
```

Cut	Ctrl+X
Copy	Ctrl+C
Delete	Delete
Undo Delete	Ctrl+Z
Evaluate Selection F9	
Create Script	
Create Live Script	
Create Favorite	
Set Error Indicator	

fx >>

↑  
← ↓ →





The image shows the MATLAB Command Window and its associated panels. The Command Window displays the following commands and their outputs:

```
>> 1 + 1  
  
ans =  
  
    2  
  
>> A = 1  
  
A =  
  
    1  
  
>> B = 5  
  
B =  
  
    5  
  
fx >> |
```

The Workspace panel, highlighted with a red border, shows the current workspace variables:

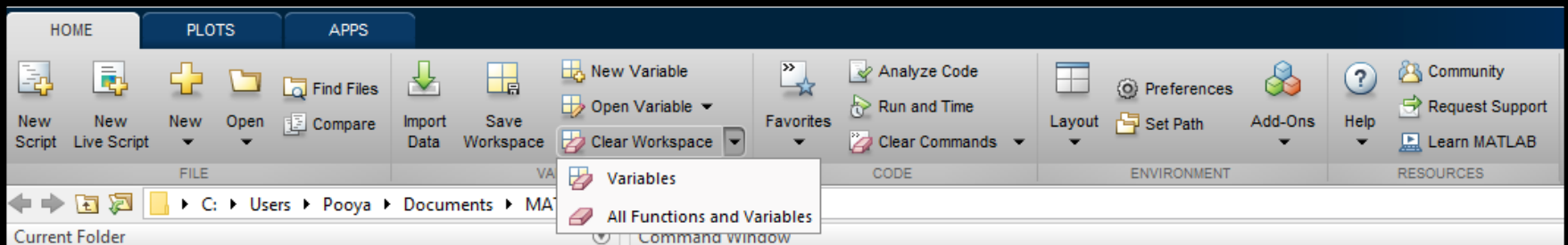
Name	Value
A	1
ans	2
B	5

The Command History panel shows the commands entered in the Command Window:

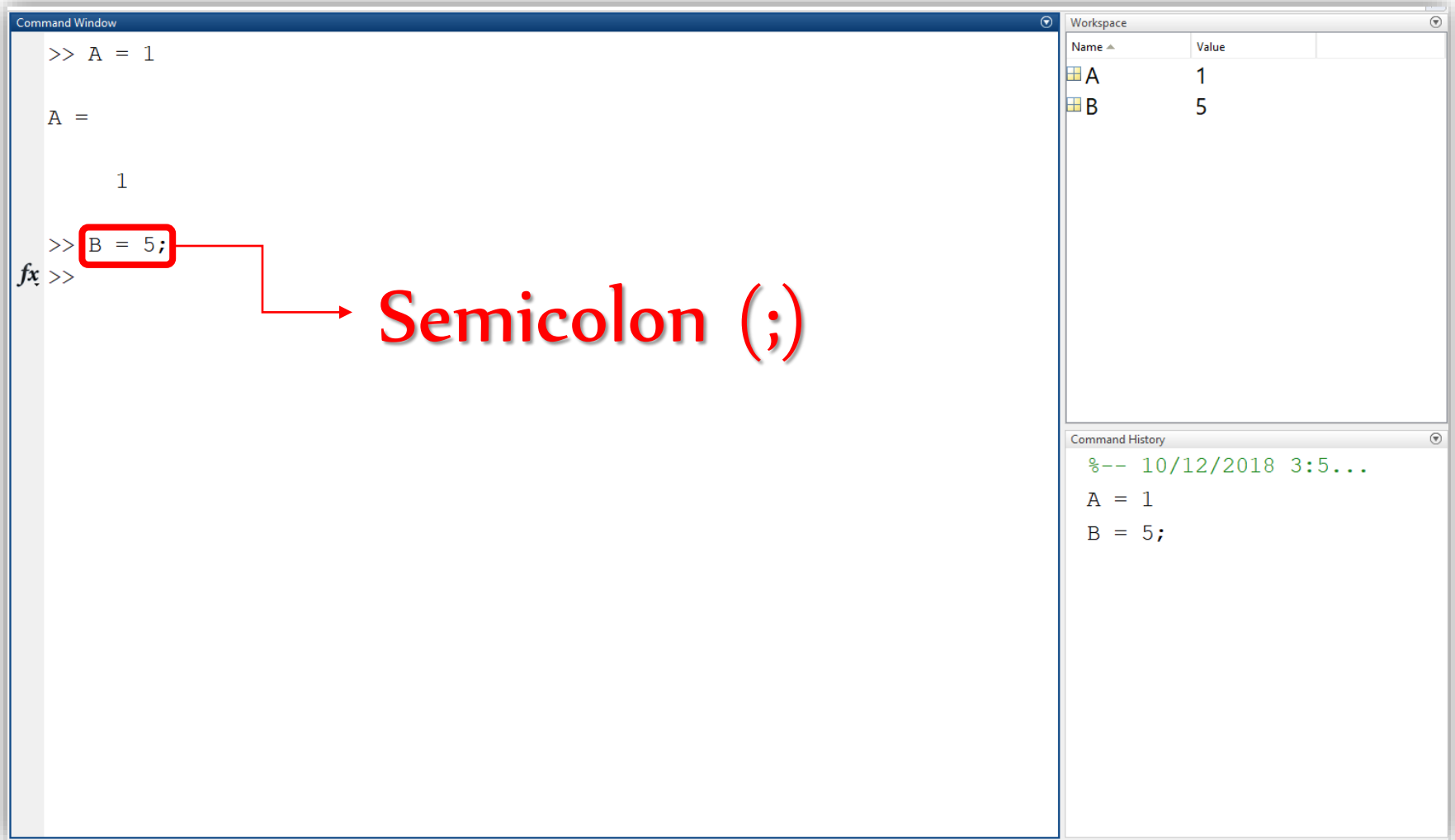
```
%-- 10/12/2018 3:32 PM --%  
1 + 1  
A = 1  
B = 5
```

## Chapter 01

# Working In The Command Window







The image shows a MATLAB Command Window and Workspace. The Command Window contains the following text:

```
>> A = 1  
  
A =  
  
1  
  
>> B = 5;  
fx >>
```

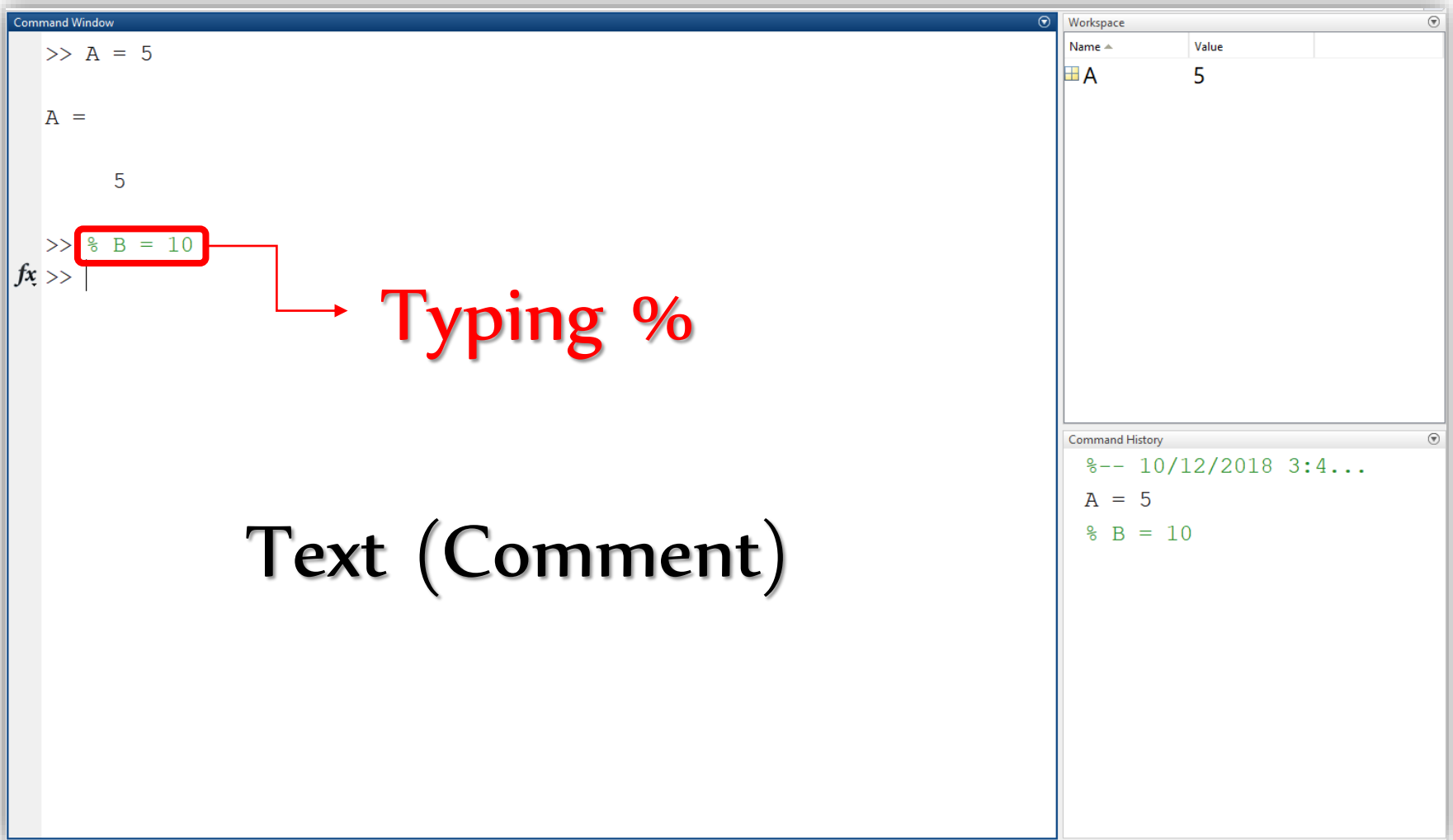
A red box highlights the semicolon in the command `B = 5;`. A red arrow points from this box to the text **Semicolon (;)**.

The Workspace window shows the following table:

Name	Value
A	1
B	5

The Command History window shows the following text:

```
%-- 10/12/2018 3:5...  
A = 1  
B = 5;
```



The image shows a MATLAB interface with three panels. The Command Window on the left contains the following text:

```
>> A = 5  
  
A =  
  
5  
  
>> % B = 10  
fx >> |
```

A red box highlights the text `% B = 10`, with a red arrow pointing to the text **Typing %**.

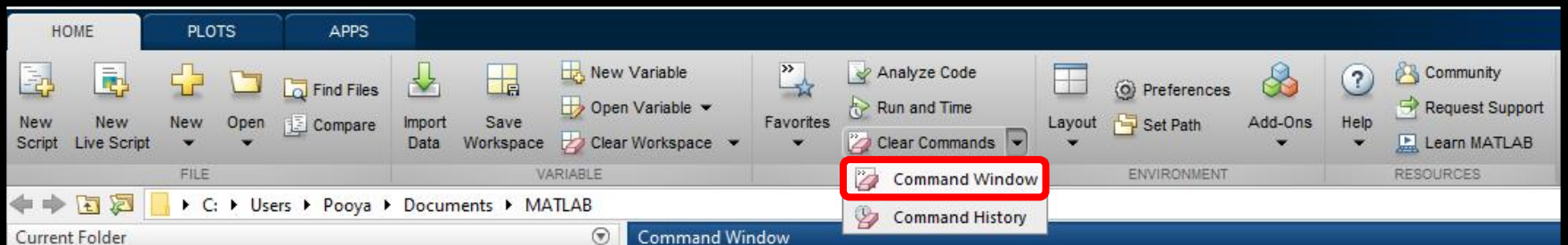
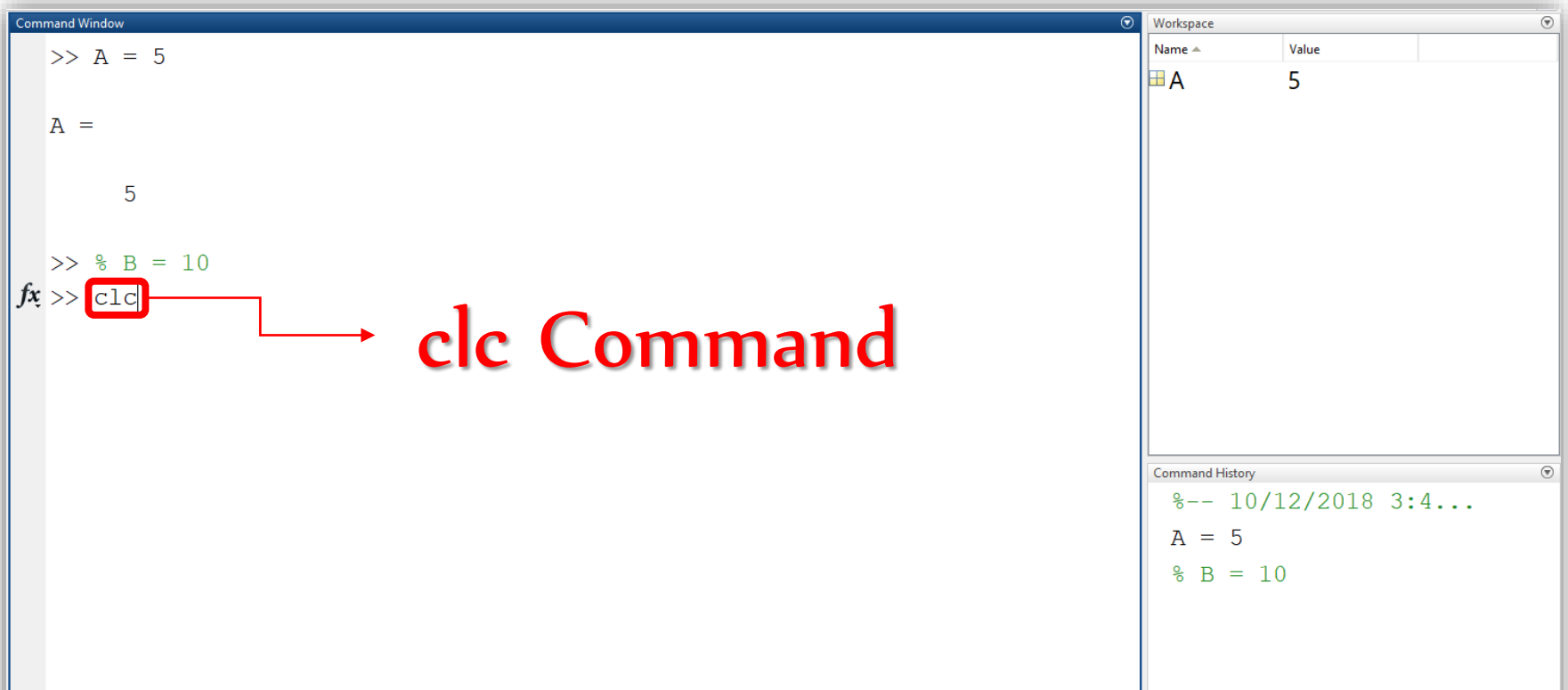
The Workspace panel on the right shows a table with two columns: Name and Value.

Name	Value
A	5

The Command History panel at the bottom right shows the following commands:

```
%-- 10/12/2018 3:4...  
A = 5  
% B = 10
```

**Text (Comment)**



Operation	Symbol	Example
Addition	+	$5 + 3$
Subtraction	−	$5 - 3$
Multiplication	*	$5 * 3$
Right Division	/	$5 / 3$
Left Division	\	$5 \setminus 3 = 3 / 5$
Exponentiation	^	$5 ^ 3$ (means $5^3 = 125$ )

## Order Of Precedence

Precedence	Mathematical Operation
First	Parentheses. For Nested Parentheses, The Innermost Are Executed First.
Second	Exponentiation.
Third	Multiplication, Division (Equal Precedence).
Fourth	Addition And Subtraction.

```
>> 7+8/2
```

```
ans =  
    11
```

← Type and press **Enter**.

8/2 is executed first.

```
>> (7+8)/2
```

```
ans =  
    7.5000
```

← Type and press **Enter**.

7+8 is executed first.

```
>> 4+5/3+2
```

```
ans =  
    7.6667
```

5/3 is executed first.

```
>> 5^3/2
```

```
ans =  
    62.5000
```

5^3 is executed first, /2 is executed next.

```
>> 27^(1/3)+32^0.2
```

```
ans =  
     5
```

1/3 is executed first, 27^(1/3) and 32^0.2 are executed next, and + is executed last.

```
>> 27^1/3+32^0.2
```

```
ans =  
    11
```

27^1 and 32^0.2 are executed first, /3 is executed next, and + is executed last.

```
>> 0.7854 - (0.7854)^3/(1*2*3) + 0.785^5/(1*2*3*4*5) ...  
- (0.785)^7/(1*2*3*4*5*6*7)
```

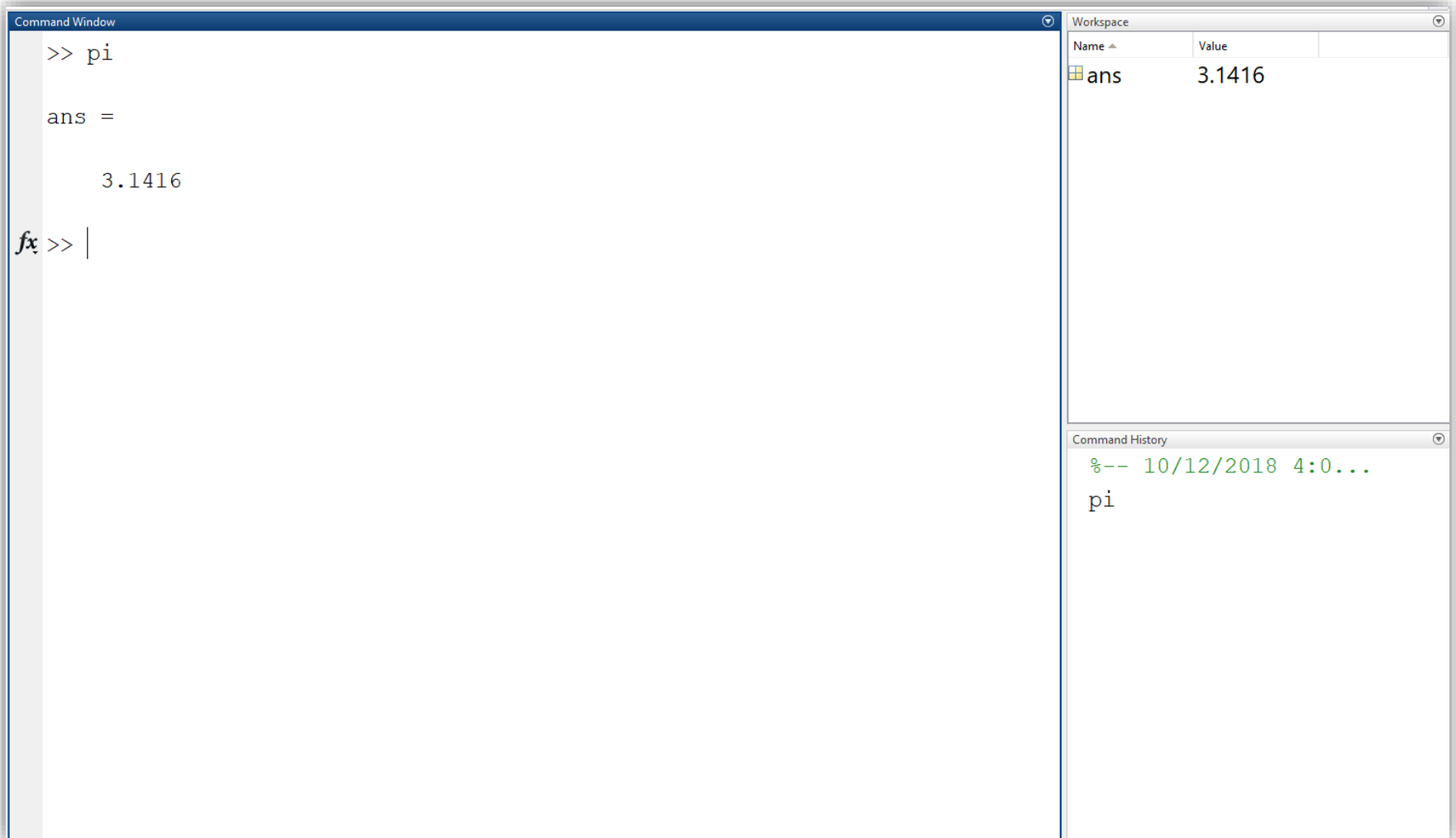
```
ans =  
    0.7071
```

← Type three periods ... (and press **Enter**) to continue the expression on the next line.

```
>>
```

The last expression is the first four terms of the Taylor series for  $\sin(\pi/4)$ .

The User Can Control The Format In Which MATLAB Displays Output On The Screen.



## The User Can Control The Format In Which MATLAB Displays Output On The Screen.

Command	Description	Example
<code>format short</code>	Fixed-point with 4 decimal digits for: $0.001 \leq \text{number} \leq 1000$ Otherwise display format short e.	<pre>&gt;&gt; 290/7 ans =     41.4286</pre>
<code>format long</code>	Fixed-point with 15 decimal digits for: $0.001 \leq \text{number} \leq 100$ Otherwise display format long e.	<pre>&gt;&gt; 290/7 ans =     41.428571428571431</pre>
<code>format short e</code>	Scientific notation with 4 decimal digits.	<pre>&gt;&gt; 290/7 ans =     4.1429e+001</pre>
<code>format long e</code>	Scientific notation with 15 decimal digits.	<pre>&gt;&gt; 290/7 ans =     4.142857142857143e+001</pre>
<code>format short g</code>	Best of 5-digit fixed or floating point.	<pre>&gt;&gt; 290/7 ans =     41.429</pre>
<code>format long g</code>	Best of 15-digit fixed or floating point.	<pre>&gt;&gt; 290/7 ans =     41.4285714285714</pre>
<code>format bank</code>	Two decimal digits.	<pre>&gt;&gt; 290/7 ans =     41.43</pre>
<code>format compact</code>	Eliminates blank lines to allow more lines with information displayed on the screen.	
<code>format loose</code>	Adds blank lines (opposite of compact).	



MATLAB Expressions Can Include **Functions**. You Can Think Of A Function As A **Black Box** That, In General, Takes **Inputs**, Does Some **Computations** With Them, And Produces **Outputs**.



MATLAB Expressions Can Include **Functions**. You Can Think Of A Function As A **Black Box** That, In General, Takes **Inputs**, Does Some **Computations** With Them, And Produces **Outputs**.

**Outputs** = Name (**Inputs**)

**Y** = sqrt (**X**)

$$Y = \text{sqrt}(X)$$

```
>> sqrt(64)
```

Argument is a number.

```
ans =  
      8
```

```
>> sqrt(50+14*3)
```

Argument is an expression.

```
ans =  
      9.5917
```

```
>> sqrt(54+9*sqrt(100))
```

Argument includes a function.

```
ans =  
      12
```

```
>> (15+600/4)/sqrt(121)
```

Function is included in an expression.

```
ans =  
      15
```

```
>>
```

# Elementary Math Built-in Functions

## Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example
<code>sqrt(x)</code>	Square root.	<pre>&gt;&gt; sqrt(81) ans =     9</pre>
<code>nthroot(x,n)</code>	Real $n$ th root of a real number $x$ . (If $x$ is negative $n$ must be an odd integer.)	<pre>&gt;&gt; nthroot(80,5) ans =     2.4022</pre>
<code>exp(x)</code>	Exponential ( $e^x$ ).	<pre>&gt;&gt; exp(5) ans =    148.4132</pre>

# Elementary Math Built-in Functions

## Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example
<code>abs(x)</code>	Absolute value.	<pre>&gt;&gt; abs(-24) ans =     24</pre>
<code>log(x)</code>	Natural logarithm. Base $e$ logarithm ( $\ln$ ).	<pre>&gt;&gt; log(1000) ans =     6.9078</pre>
<code>log10(x)</code>	Base 10 logarithm.	<pre>&gt;&gt; log10(1000) ans =     3.0000</pre>
<code>factorial(x)</code>	The factorial function $x!$ ( $x$ must be a positive integer.)	<pre>&gt;&gt; factorial(5) ans =     120</pre>

# Elementary Math Built-in Functions

## Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example
<code>sin(x)</code> <code>sind(x)</code>	Sine of angle $x$ ( $x$ in radians). Sine of angle $x$ ( $x$ in degrees).	<pre>&gt;&gt; sin(pi/6) ans =     0.5000</pre>
<code>cos(x)</code> <code>cosd(x)</code>	Cosine of angle $x$ ( $x$ in radians). Cosine of angle $x$ ( $x$ in degrees).	<pre>&gt;&gt; cosd(30) ans =     0.8660</pre>
<code>tan(x)</code> <code>tand(x)</code>	Tangent of angle $x$ ( $x$ in radians). Tangent of angle $x$ ( $x$ in degrees).	<pre>&gt;&gt; tan(pi/6) ans =     0.5774</pre>
<code>cot(x)</code> <code>cotd(x)</code>	Cotangent of angle $x$ ( $x$ in radians). Cotangent of angle $x$ ( $x$ in degrees).	<pre>&gt;&gt; cotd(30) ans =     1.7321</pre>

# Elementary Math Built-in Functions

## Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example
<code>round(x)</code>	Round to the nearest integer.	<pre>&gt;&gt; round(17/5) ans =      3</pre>
<code>fix(x)</code>	Round toward zero.	<pre>&gt;&gt; fix(13/5) ans =      2</pre>
<code>ceil(x)</code>	Round toward infinity.	<pre>&gt;&gt; ceil(11/5) ans =      3</pre>
<code>floor(x)</code>	Round toward minus infinity.	<pre>&gt;&gt; floor(-9/4) ans =     -3</pre>
<code>rem(x,y)</code>	Returns the remainder after $x$ is divided by $y$ .	<pre>&gt;&gt; rem(13,5) ans =      3</pre>
<code>sign(x)</code>	Signum function. Returns 1 if $x > 0$ , -1 if $x < 0$ , and 0 if $x = 0$ .	<pre>&gt;&gt; sign(5) ans =      1</pre>

# Defining Scalar Variables

## The Assignment Operator

Variable\_name = A numerical value, or a computable expression

```
>> x=15
```

The number 15 is assigned to the variable x.

```
x =  
    15
```

MATLAB displays the variable name and its assigned value.

```
>> x=3*x-12
```

```
x =  
    33
```

A new value is assigned to x. The new value is 3 times the previous value of x minus 12.

```
>>
```



# Defining Scalar Variables

## The Assignment Operator



THINK OF  $=$  AS MEANING “**ASSIGN TO**” OR “**STORE IN**”  
BUT NOT MEANING “**EQUALS**”

Why?

$x = x + 6$  has no meaning in math because it implies that  $0 = 6$ .

$x = x + 6$  is perfectly fine in MATLAB because it means “take whatever is in  $x$ , add 6 to that and store the result back into  $x$ ”

# Defining Scalar Variables

## The Assignment Operator

```
>> a=12;  
>> B=4;  
>> C=(a-B)+40-a/B*10;  
  
>> C  
C =  
    18
```

The variables a, B, and C are defined but are not displayed, since a semicolon is typed at the end of each statement.

The value of the variable C is displayed by typing the name of the variable.

```
>> a=12, B=4; C=(a-B)+40-a/B*10  
  
a =  
    12  
  
C =  
    18
```

The variable B is not displayed because a semicolon is typed at the end of the assignment.

# Defining Scalar Variables

## The Assignment Operator

```
>> ABB=72;
```

A value of 72 is assigned to the variable ABB.

```
>> ABB=9;
```

A new value of 9 is assigned to the variable ABB.

```
>> ABB
```

```
ABB =  
      9
```

The current value of the variable is displayed when the name of the variable is typed and the **Enter** key is pressed.

```
>>
```

```
>> x=0.75;
```

```
>> E=sin(x)^2+cos(x)^2
```

```
E =
```

```
      1
```

```
>>
```

??? Undefined function or variable 'x'

# Defining Scalar Variables

## Rules About Variable Names

A variable can be named according to the following rules:

- Must begin with a letter.
- Can be up to 63 characters long.
- Can contain letters, digits, and the underscore character.
- Cannot contain punctuation characters (e.g., period, comma, semicolon).
- MATLAB is case-sensitive: it distinguishes between uppercase and lowercase letters. For example, AA, Aa, aA, and aa are the names of four different variables.
- No spaces are allowed between characters (use the underscore where a space is desired).
- Avoid using the name of a built-in function for a variable (i.e., avoid using `cos`, `sin`, `exp`, `sqrt`, etc.). Once a function name is used to for a variable name, the function cannot be used.

## Predefined Variables and Keywords

```
break case catch classdef continue else elseif end for  
function global if otherwise parfor persistent return  
spmd switch try while
```

**ans** A variable that has the value of the last expression that was not assigned to a specific variable (see Tutorial 1-1). If the user does not assign the value of an expression to a variable, MATLAB automatically stores the result in **ans**.

**pi** The number  $\pi$ .

**eps** The smallest difference between two numbers. Equal to  $2^{(-52)}$ , which is approximately  $2.2204e-016$ .

**inf** Used for infinity.

**i** Defined as  $\sqrt{-1}$ , which is:  $0 + 1.0000i$ .

**j** Same as **i**.

**NaN** Stands for Not-a-Number. Used when MATLAB cannot determine a valid numeric value. Example:  $0/0$ .

---

Command	Outcome
<code>clear</code>	Removes all variables from memory
<code>clear x y z</code>	Removes only variables x, y, and z from memory
<code>who</code>	Displays a list of the variables currently in memory
<code>whos</code>	Displays a list of the variables currently in memory and their size, together with information about their bytes and class

---

So far, have run MATLAB commands by typing in single command, pressing ENTER, getting MATLAB's result, and then repeating this process for next command

Better Way:

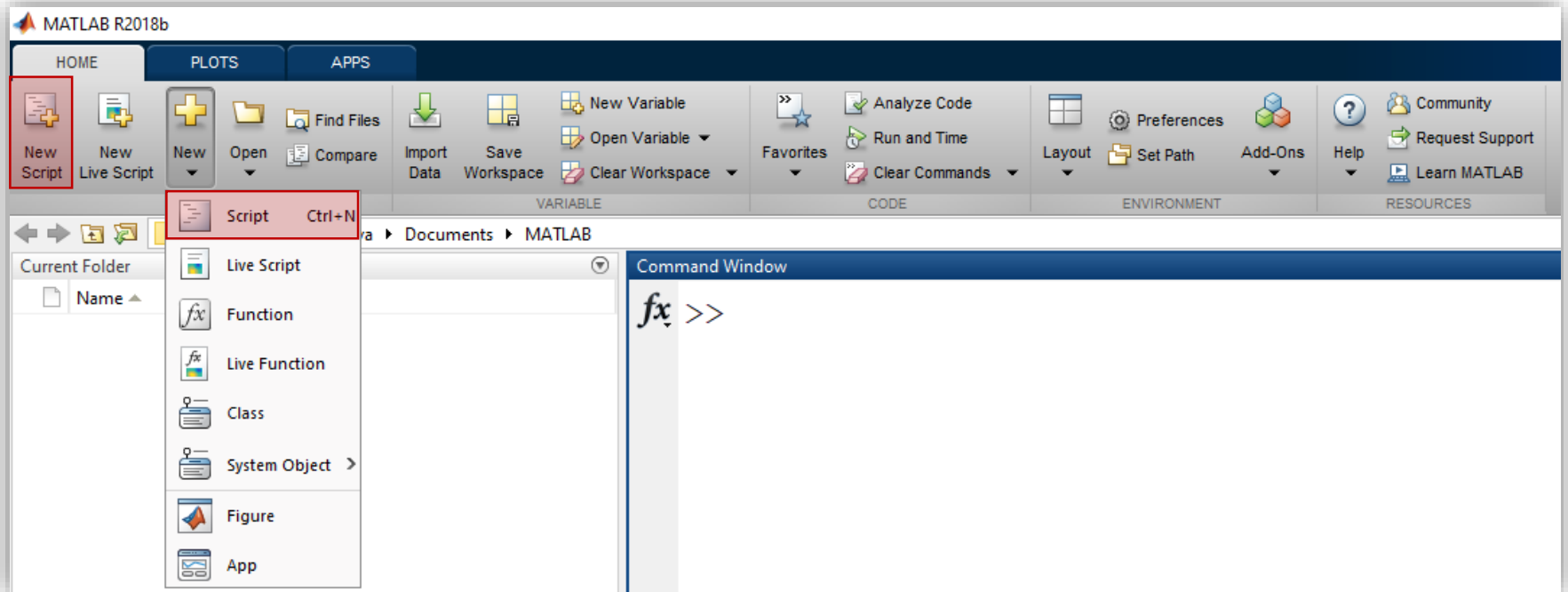
- A. Save All Commands In A File.
- B. With One Command In Command Window, Tell MATLAB To Run All Commands In File.

## Notes About Script Files

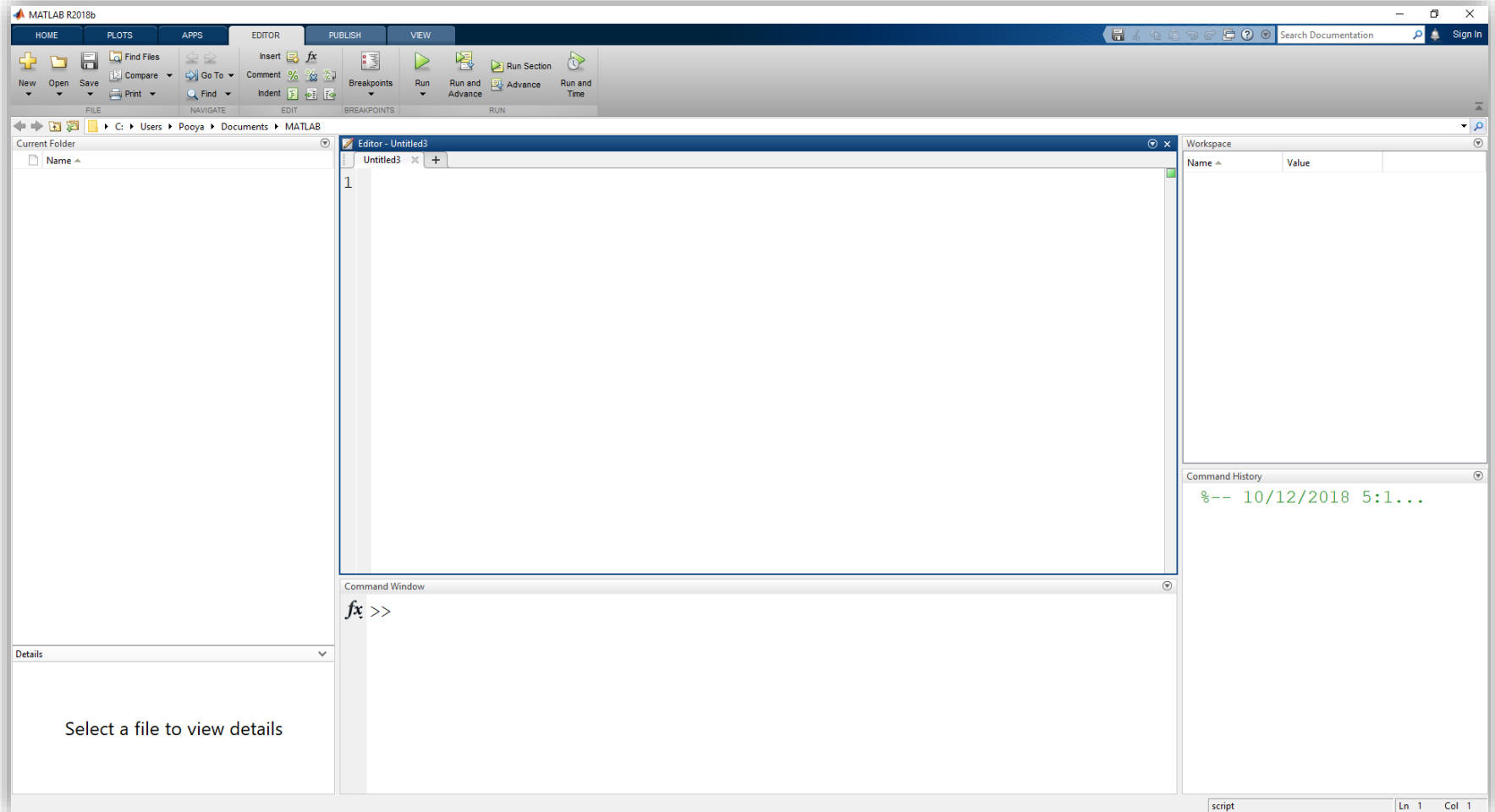
- i. A Script File Is A Sequence Of MATLAB Commands, Also Called A Program.
- ii. **When A Script File Runs, MATLAB Executes The Commands In The Order They Are Written, Just As If They Were Typed In The Command Window.**
- iii. When A Script File Has A Command That Generates An Output, The Output Is Displayed In The Command Window.
- iv. **Using A Script File Is Convenient Because It Can Be Edited And Executed Many Times.**
- v. Script Files Can Be Typed And Edited In Any Text Editor And Then Pasted Into The MATLAB Editor.
- vi. **Script Files Are Also Called M-files Because The Extension .M Is Used When They Are Saved.**



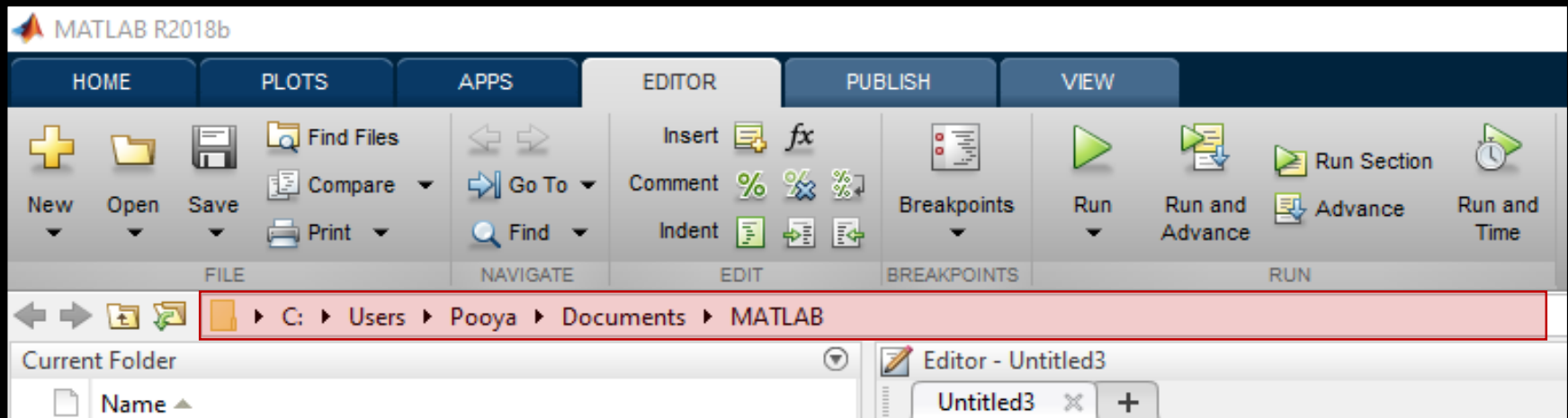
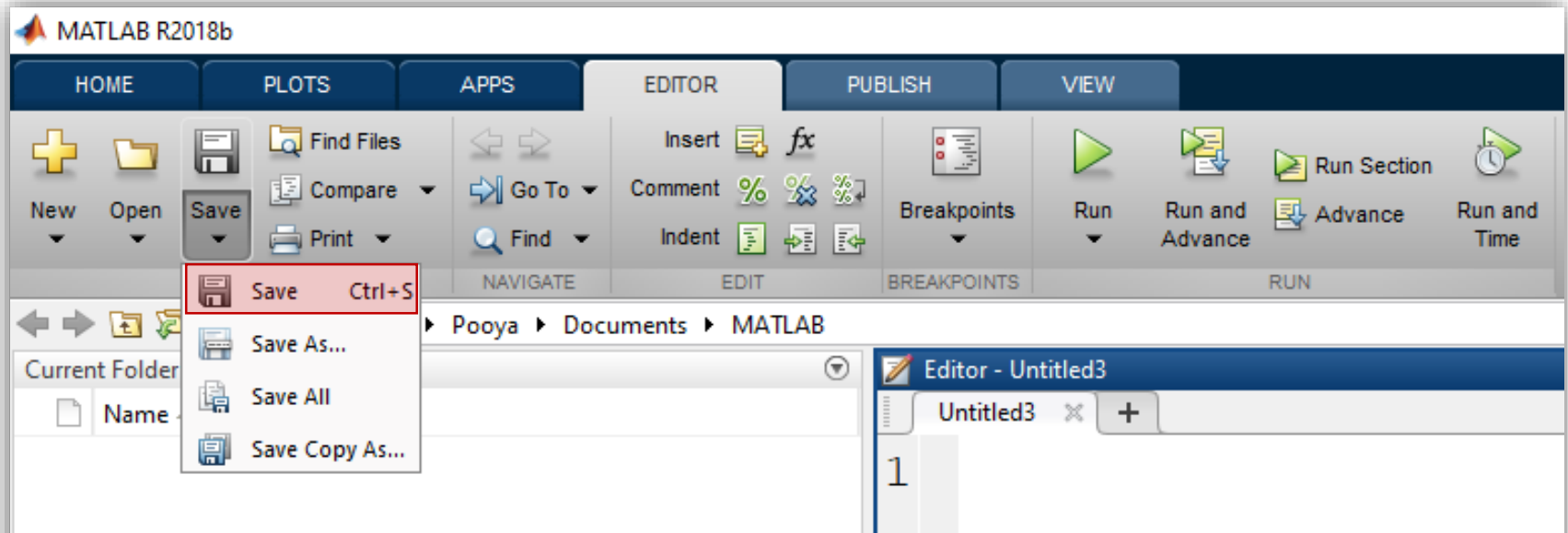
## Creating And Saving A Script File



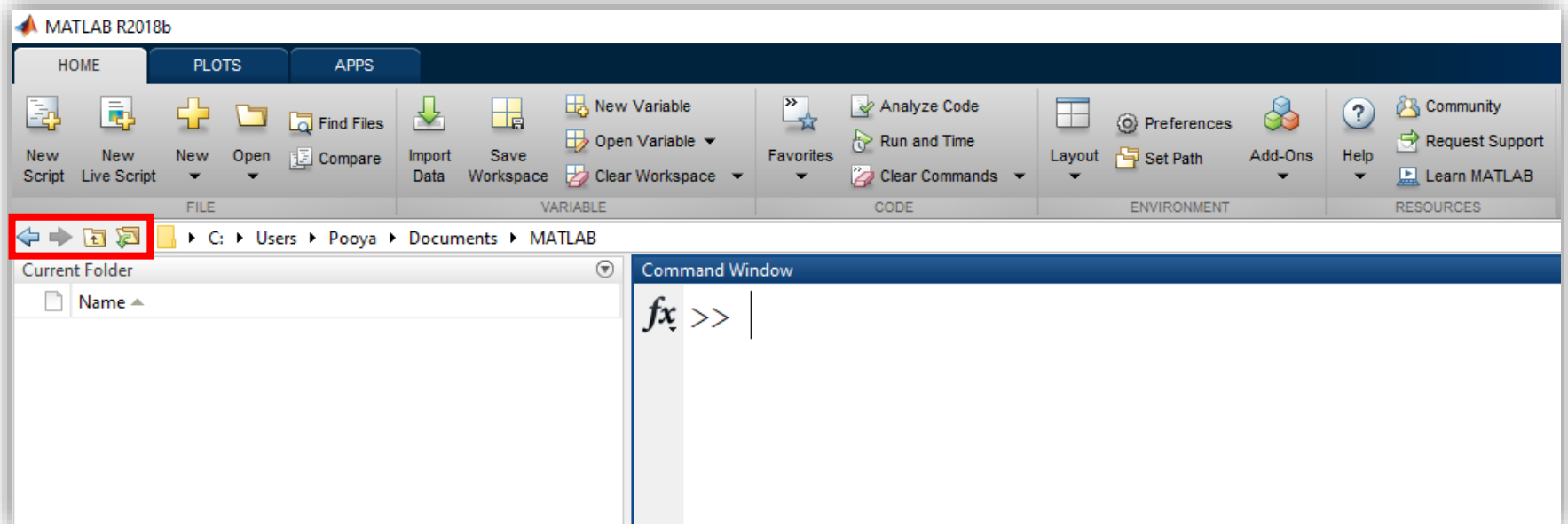
## Creating And Saving A Script File



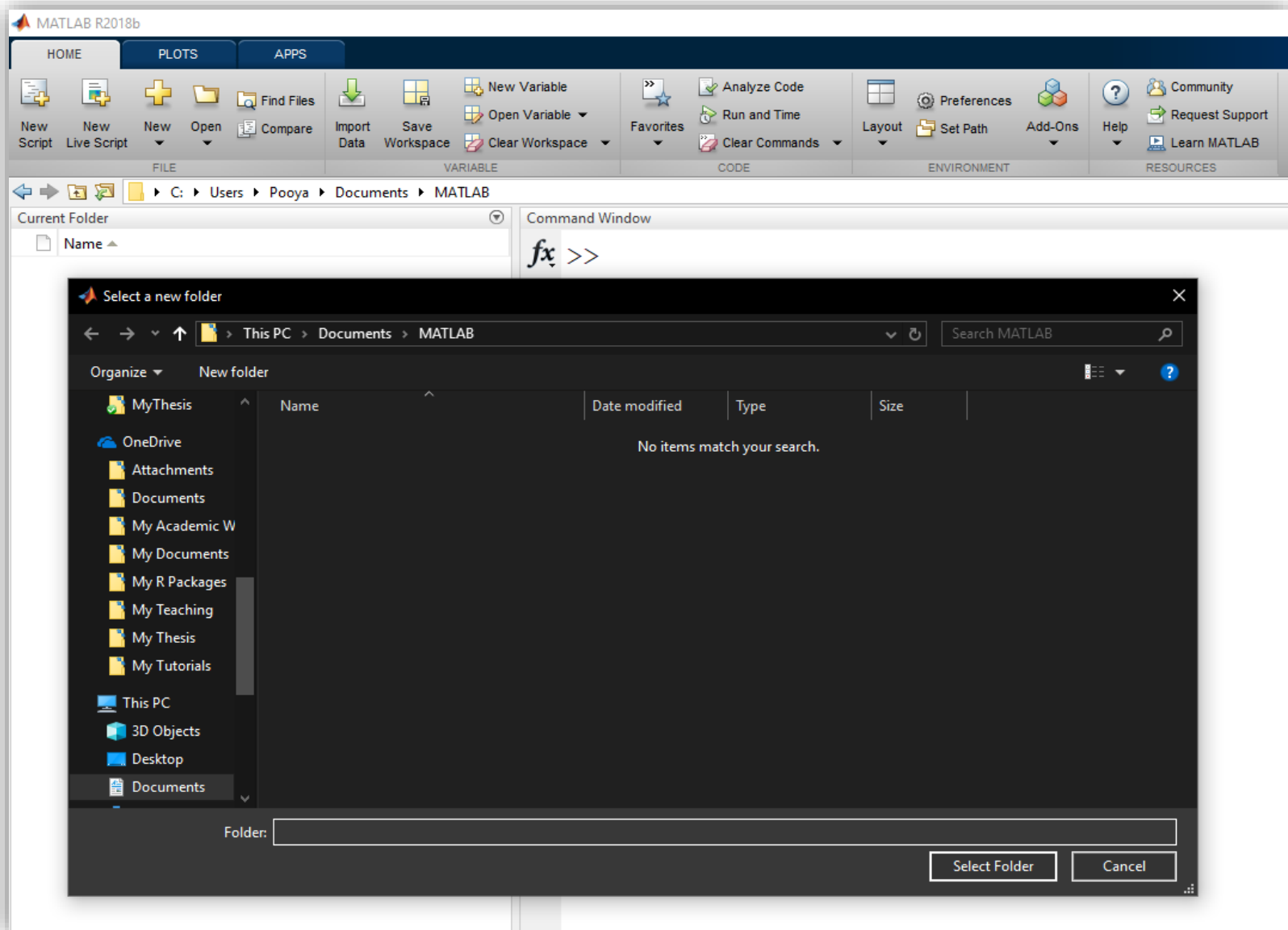
## Creating And Saving A Script File



## Creating And Saving A Script File



## Creating And Saving A Script File



## Creating And Saving A Script File

Command Window

&gt;&gt; pwd

ans =

'C:\Users\Pooya\Documents\MATLAB'

&gt;&gt; cd 'C:\Users\Pooya\Desktop'

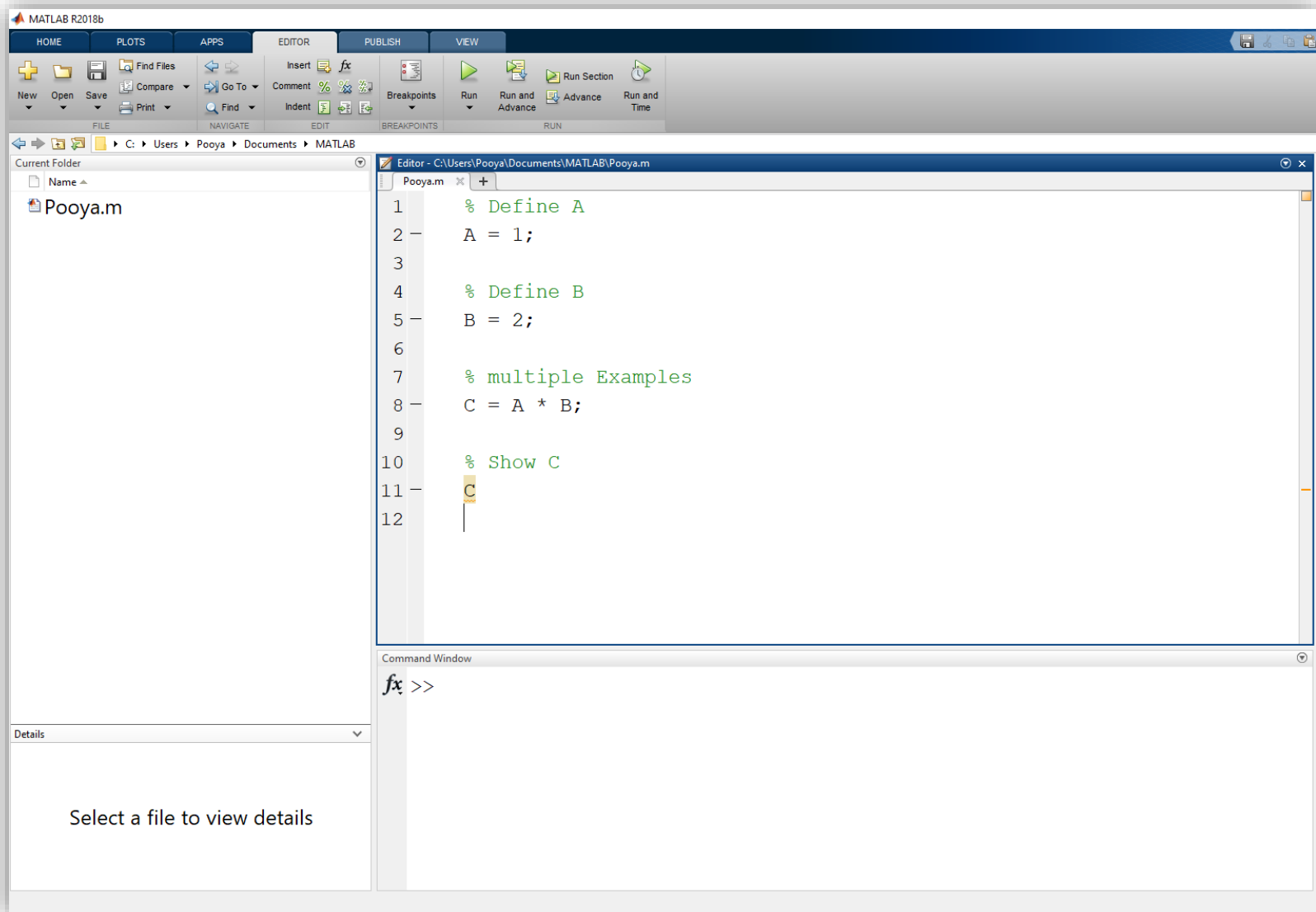
&gt;&gt; pwd

ans =

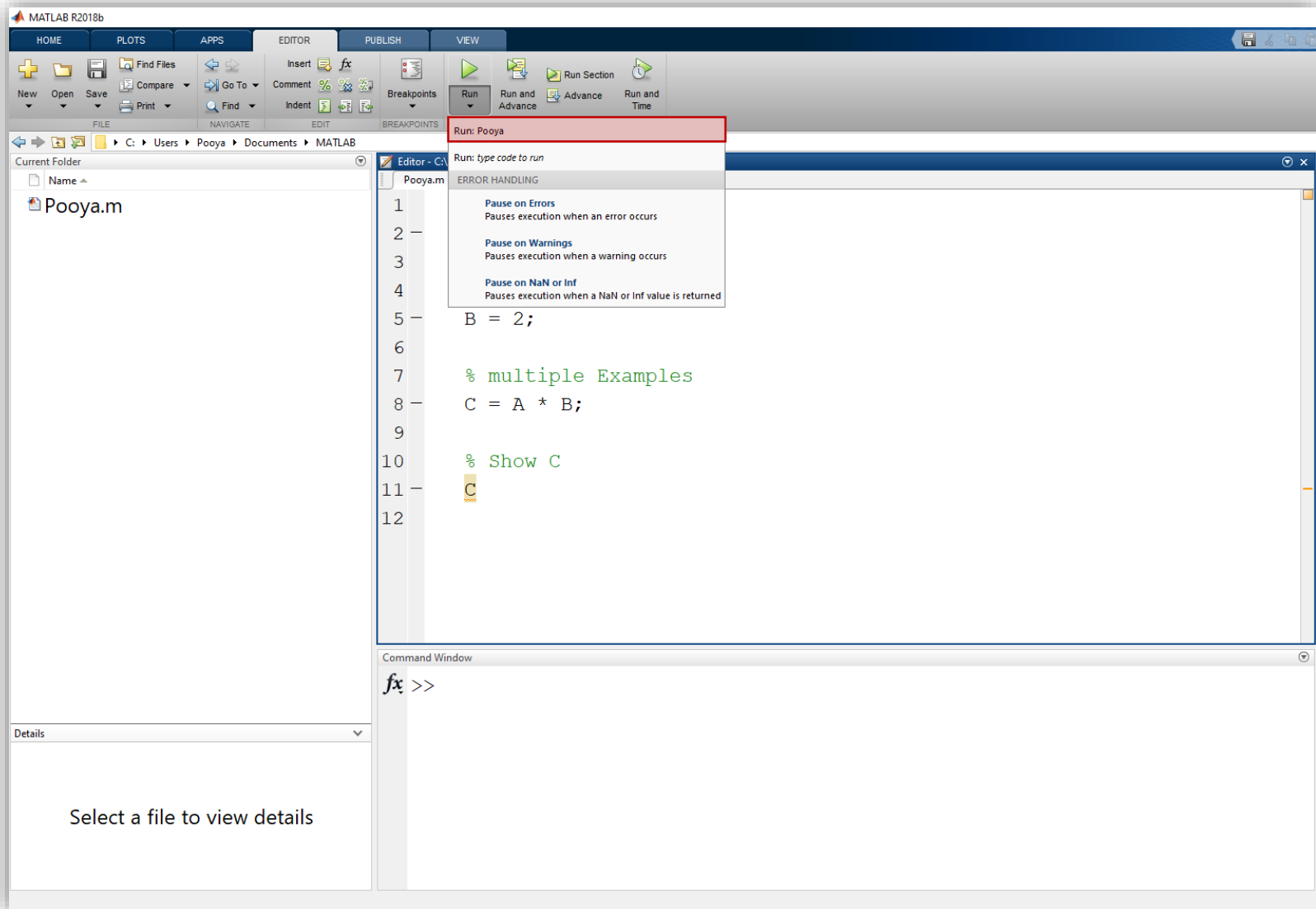
'C:\Users\Pooya\Desktop'

fx &gt;&gt; |

## Running (Executing) A Script File

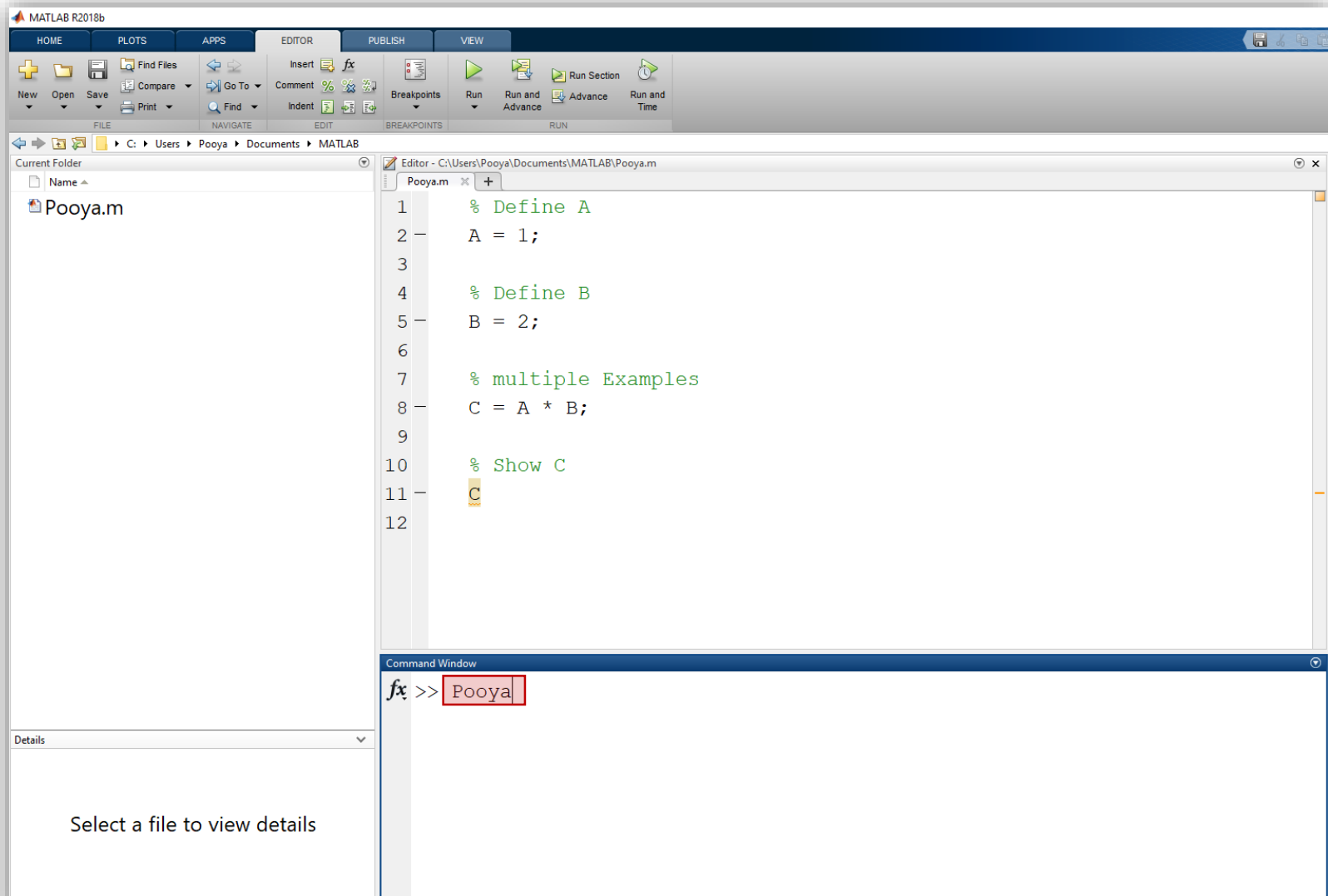


## Running (Executing) A Script File

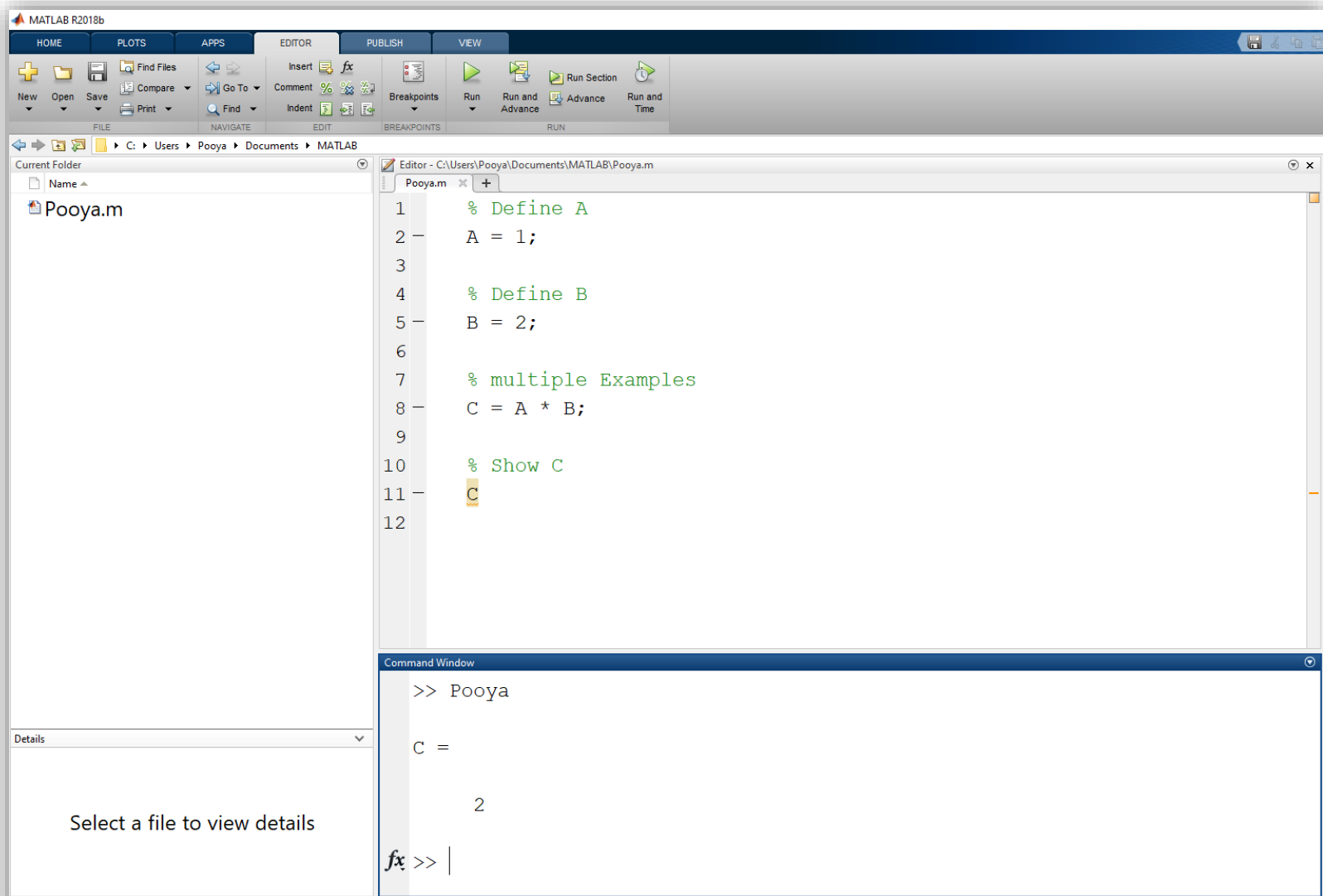




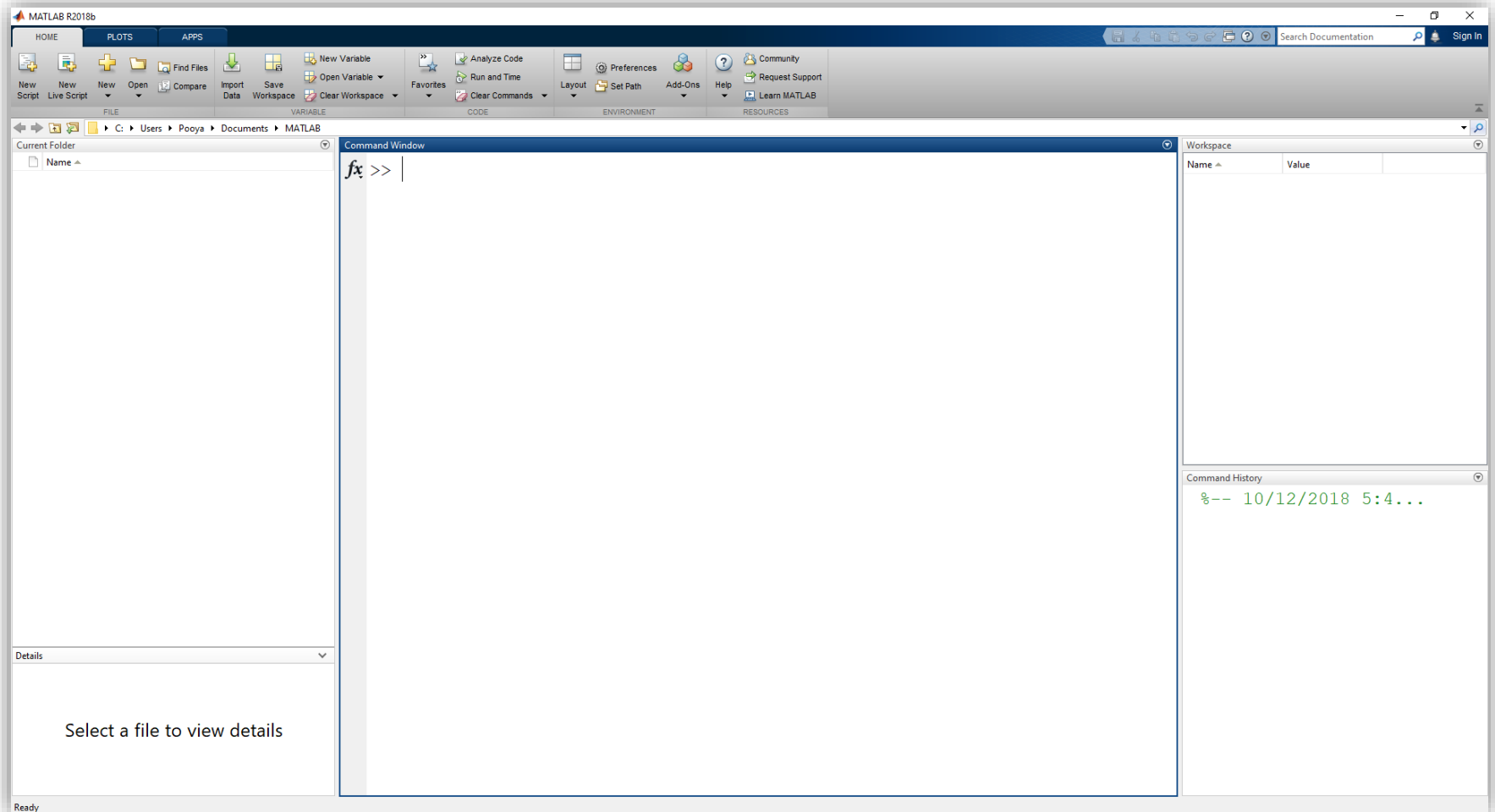
## Running (Executing) A Script File



## Running (Executing) A Script File

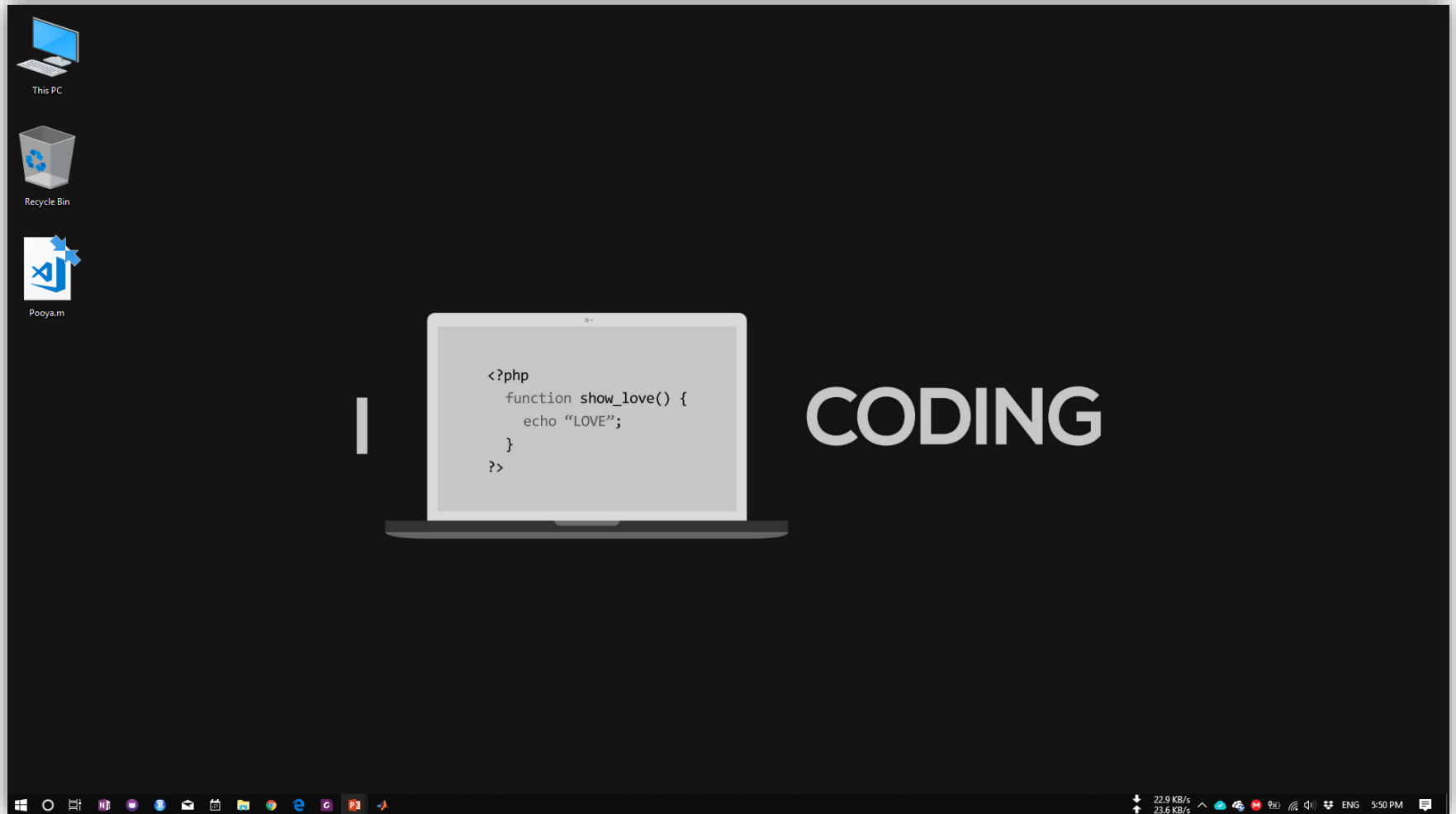


## Running (Executing) A Script File

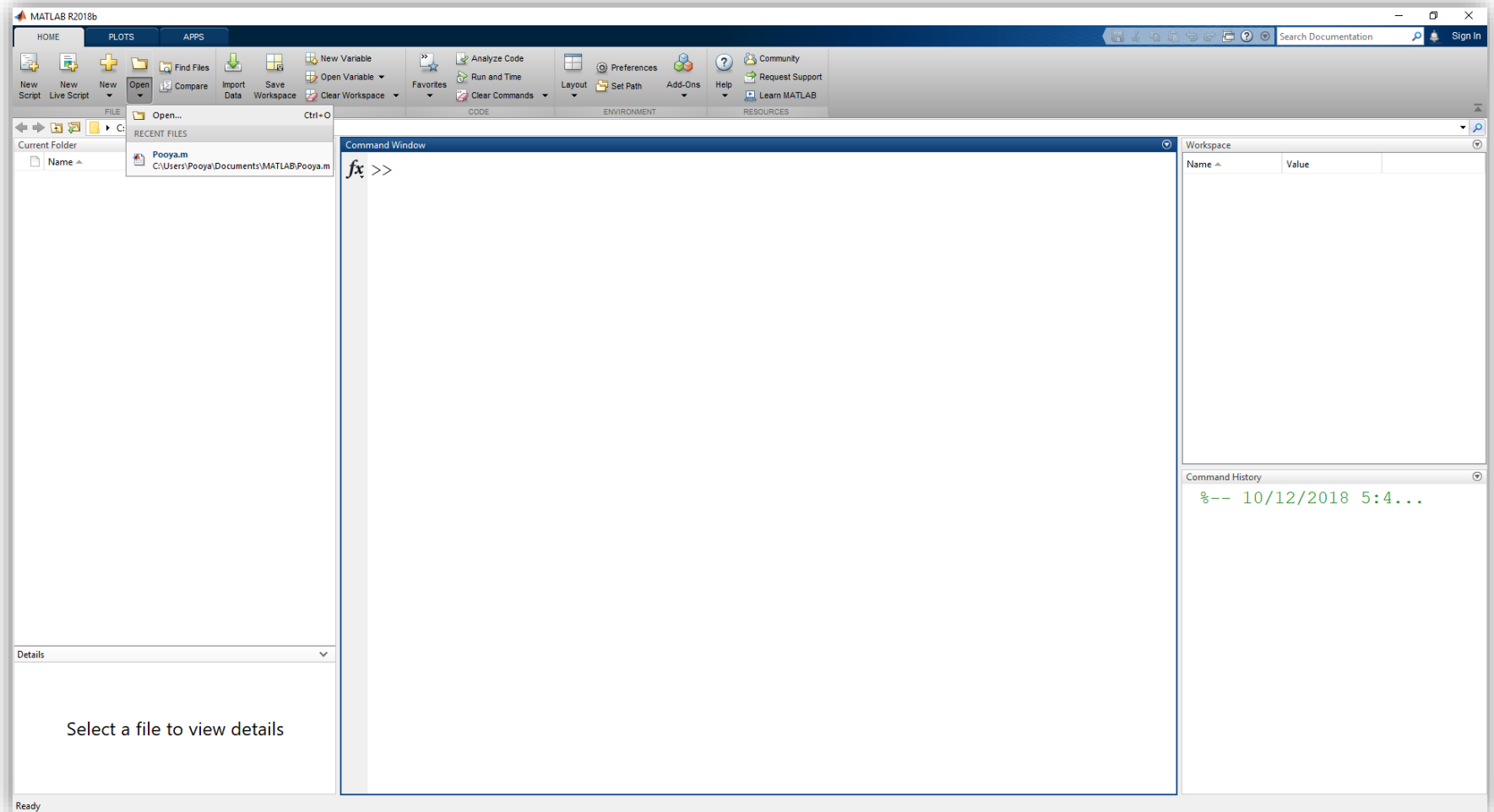


# Script Files

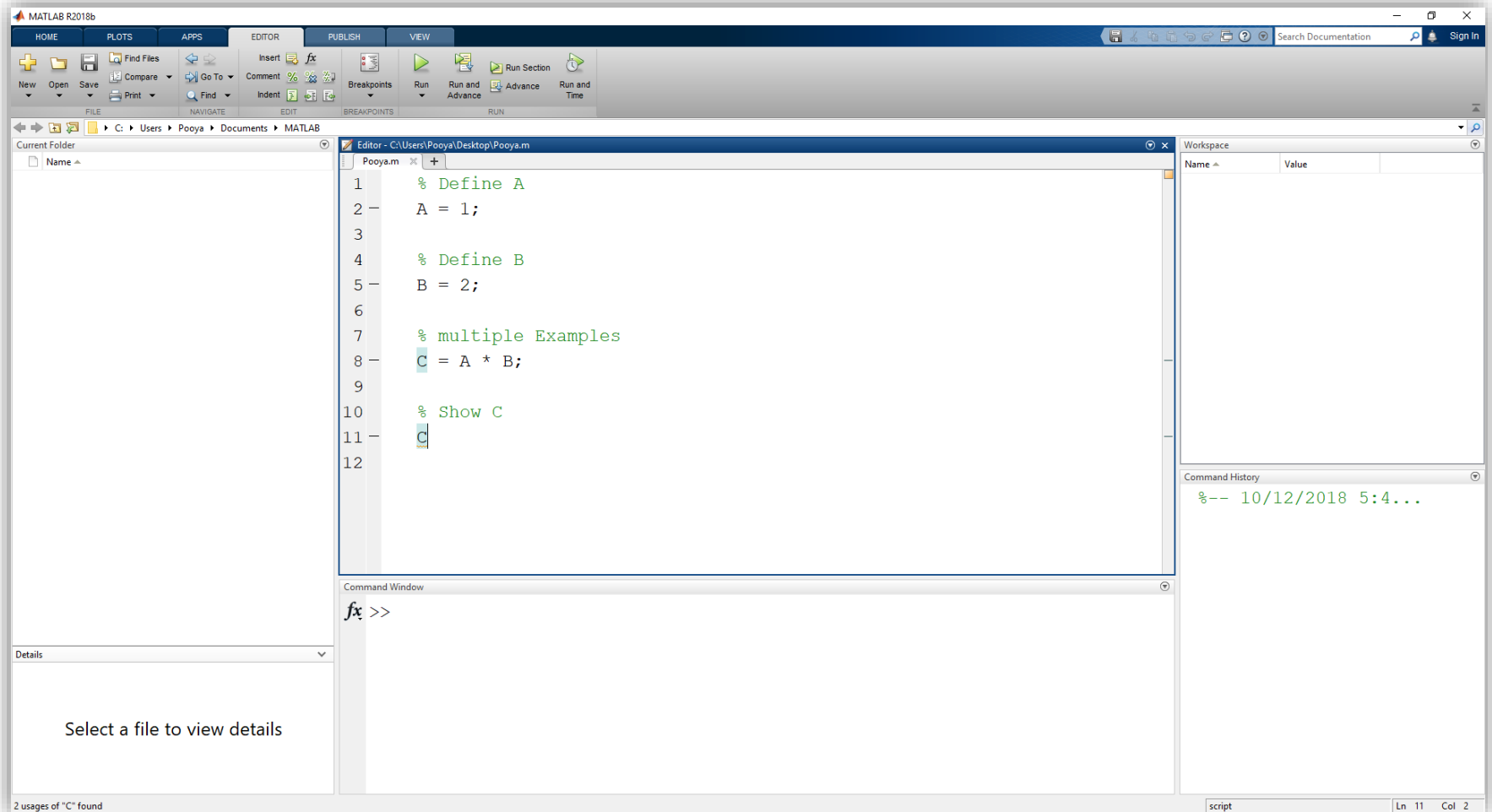
## Running (Executing) A Script File



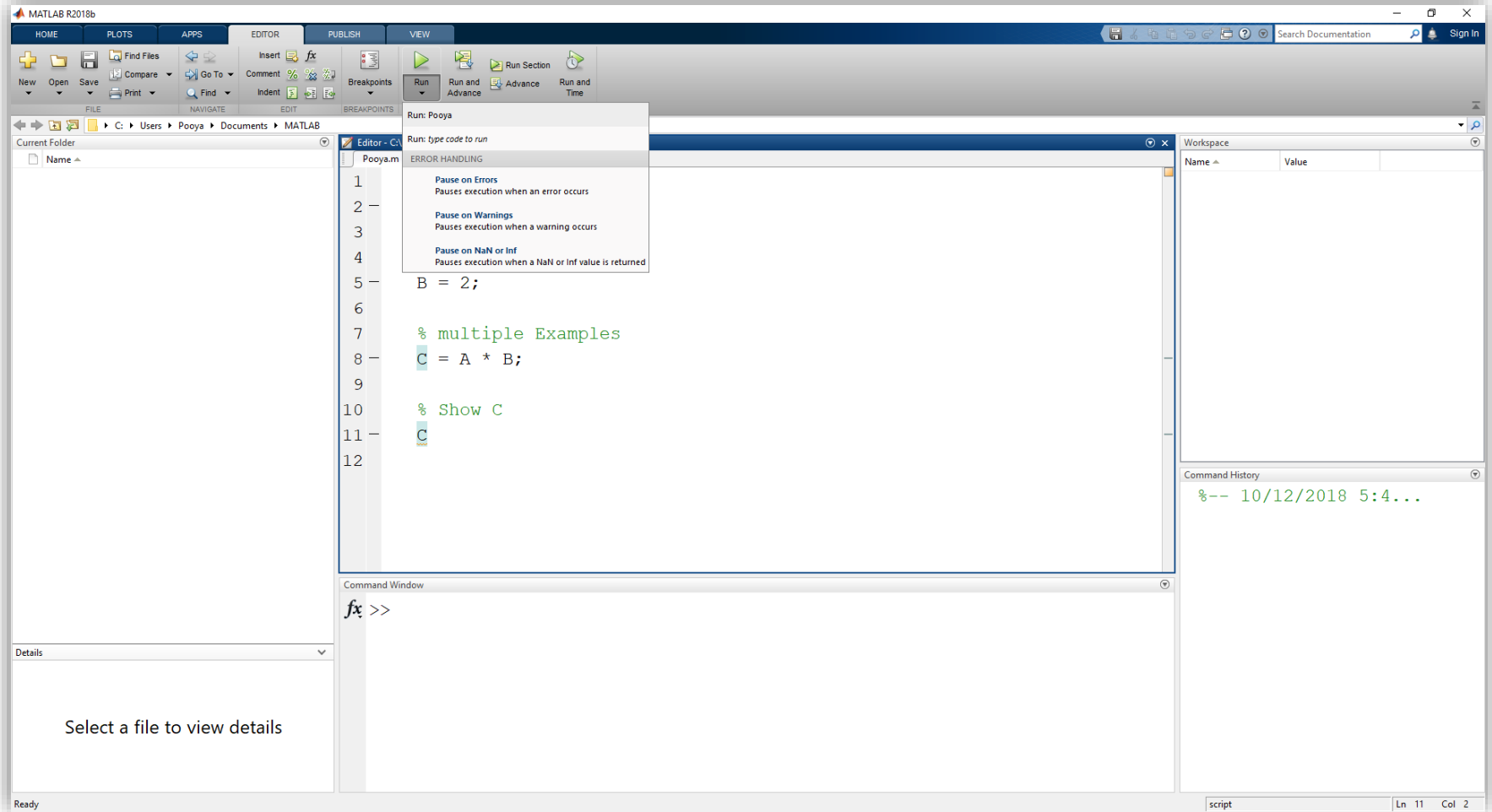
## Running (Executing) A Script File



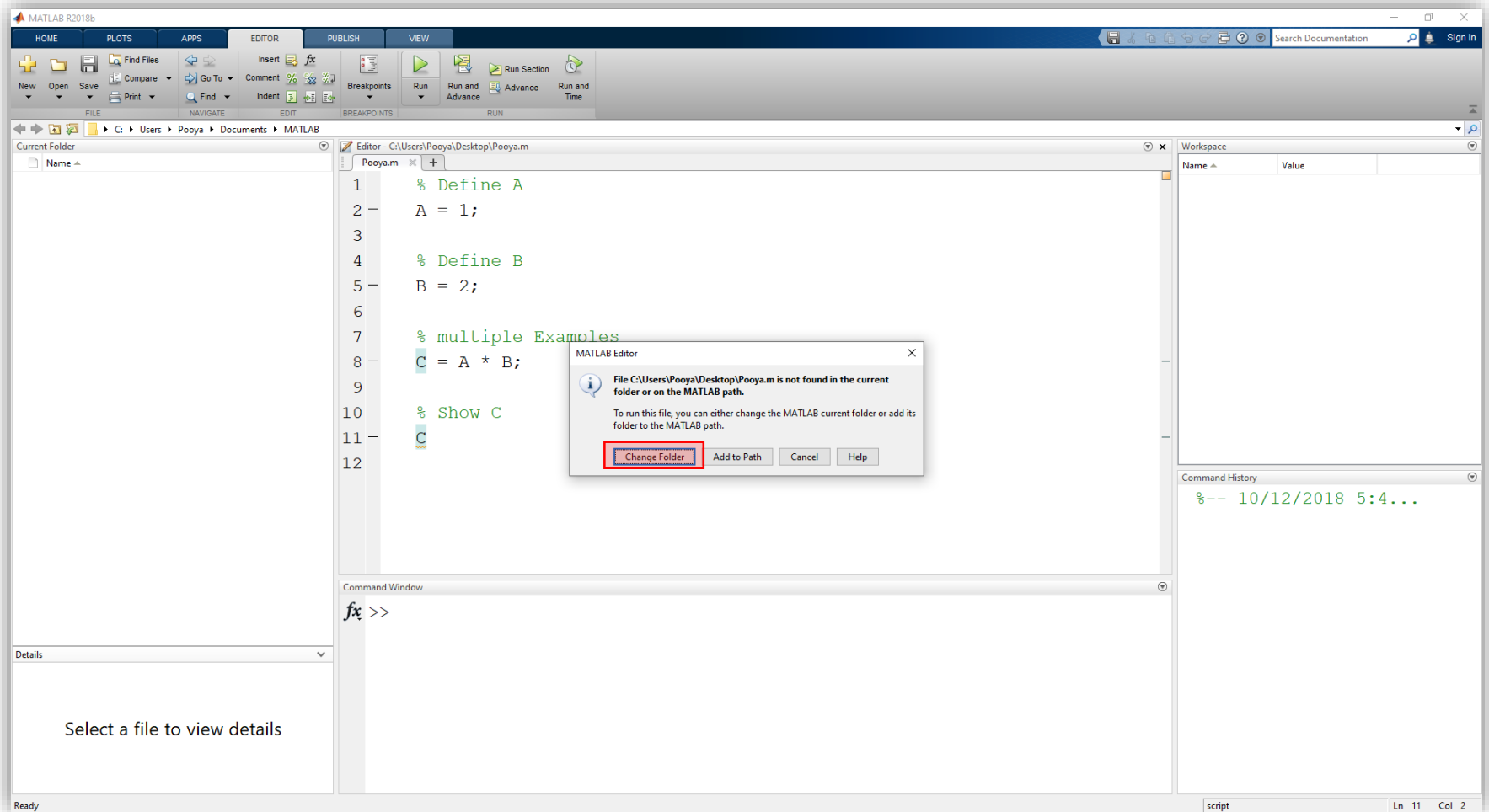
## Running (Executing) A Script File



## Running (Executing) A Script File

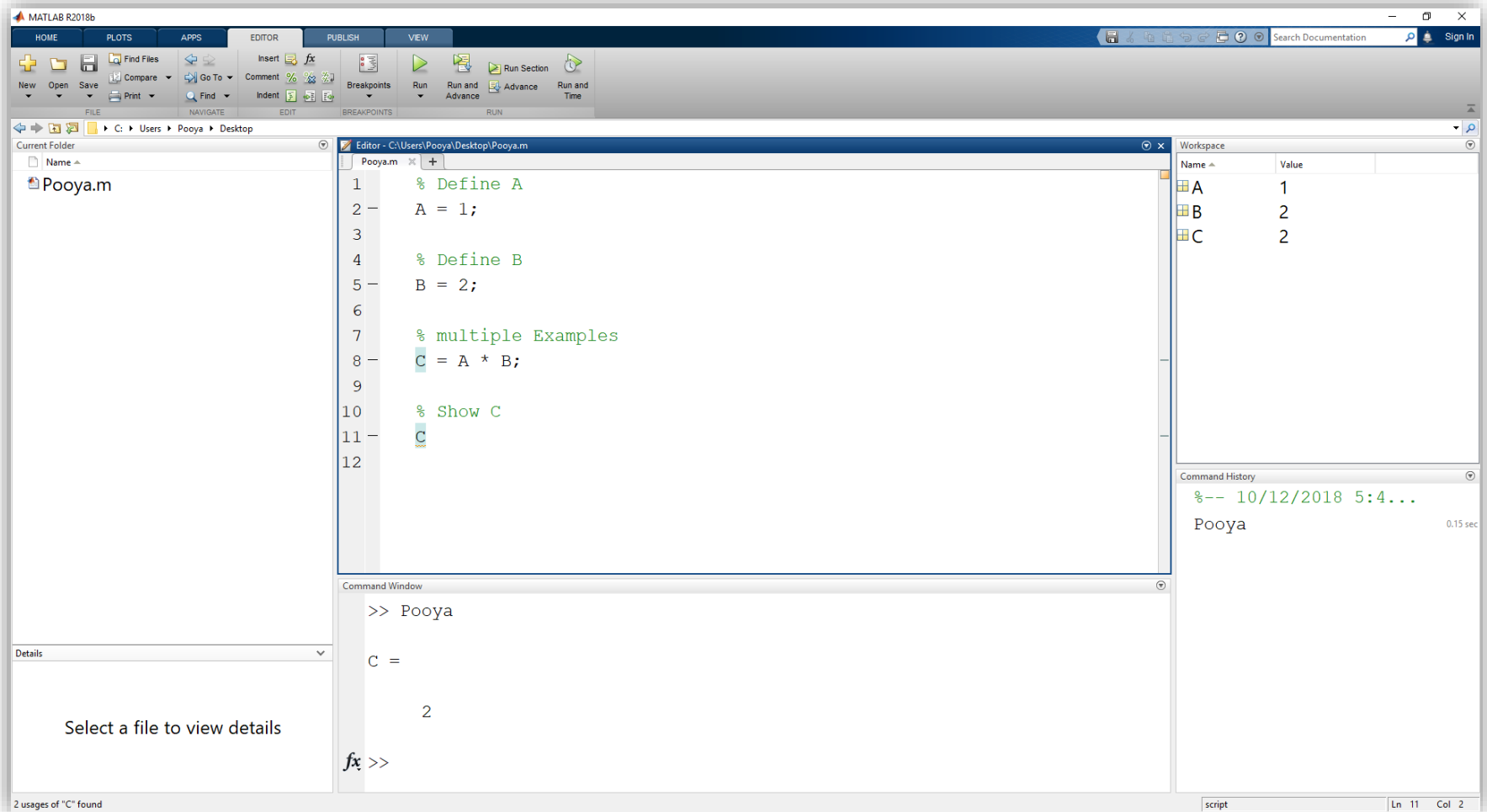


## Running (Executing) A Script File





## Running (Executing) A Script File



Sample Problem 01: **HEAT TRANSFER**

An object with an initial temperature of  $T_0$  that is placed at time  $t = 0$  inside a chamber that has a constant temperature of  $T_s$  will experience a temperature change according to the equation

$$T = T_s + (T_0 - T_s)e^{-kt}$$

where  $T$  is the temperature of the object at time  $t$ , and  $k$  is a constant. A soda can at a temperature of 120° F (after being left in the car) is placed inside a refrigerator where the temperature is 38° F. Determine, to the nearest degree, the temperature of the can after three hours. Assume  $k = 0.45$ . First define all of the variables and then calculate the temperature using one MATLAB command.

## Sample Problem 01: HEAT TRANSFER

The image displays the MATLAB R2018b environment. The main window shows a script named 'HeatTransfer.m' with the following code:

```
1 % Sample Problem 01: HEAT TRANSFER
2
3 % Refrigerator Temperature
4 Ts = 38;
5
6 % Soda Temperature
7 T0 = 120;
8
9 % Constant
10 k = 0.45;
11
12 % Time Passed
13 t = 3;
14
15 % Temperature Of Soda After 3 Hours
16 T = round(Ts + (T0 - Ts) * exp(-k * t));
17
```

The workspace on the right shows the following variables:

Name	Value
k	0.4500
t	3
T	59
T0	120
Ts	38

The Command History on the right shows the following commands:

```
%-- 10/12/2018 5:48 PM --%
Pooya
%-- 10/12/2018 6:13 PM --%
HeatTransfer
```

The Command Window at the bottom shows the result of the calculation:

```
T =
    59

fx >>
```

The Details pane on the left shows a message: "Select a file to view details".

Sample Problem 02: **TRIGONOMETRIC IDENTITY**

A trigonometric identity is given by:

$$\cos^2 \frac{x}{2} = \frac{\tan x + \sin x}{2 \tan x}$$

Verify that the identity is correct by calculating each side of the equation, substituting  $x = \frac{\pi}{5}$ .

Sample Problem 02: **TRIGONOMETRIC IDENTITY**

The image shows the MATLAB R2018b interface with a script editor, workspace, and command window.

**Script Editor:** The script is titled "TrigonometricIdentity.m" and contains the following code:

```
1 % Sample Problem 02: TRIGONOMETRIC IDENTITY
2
3 % Define x
4 x = pi/5;
5
6 % Calculate The Left-hand Side
7 LHS = cos(x/2)^2;
8
9 % Calculate The Right-hand Side
10 RHS = (tan(x) + sin(x))/(2 * tan(x));
11
12 % Show LHS And RHS
13 LHS
14 RHS
```

**Workspace:** The workspace displays the following variables:

Name	Value
LHS	0.9045
RHS	0.9045
x	0.6283

**Command Window:** The command window shows the output of the script:

```
LHS =
    0.9045

RHS =
    0.9045
```

**Command History:** The command history shows the following commands:

```
%-- 10/12/2018 6:3...
TrigonometricIdentity
```

MATLAB Assignment - Session 02

POOYA SHIRAZI

Blog General Irrigation Computer Programming Upload Your Files

## MATLAB Assignment - Session 02

This page contains your homework assignments. Please first download "Download MATLAB Assignment - Session 02", then based on the number of your group complete your assignments.

[Download MATLAB Assignment - Session 02](#)

Group	Problem Number				
A	1	9	17	25	33
B	2	10	18	26	34
C	3	11	19	27	35
D	4	12	20	28	36
E	5	13	21	29	37
F	6	14	22	30	38
G	7	15	23	31	39
H	8	16	24	32	40

© 2018 AgroCode, Co.

