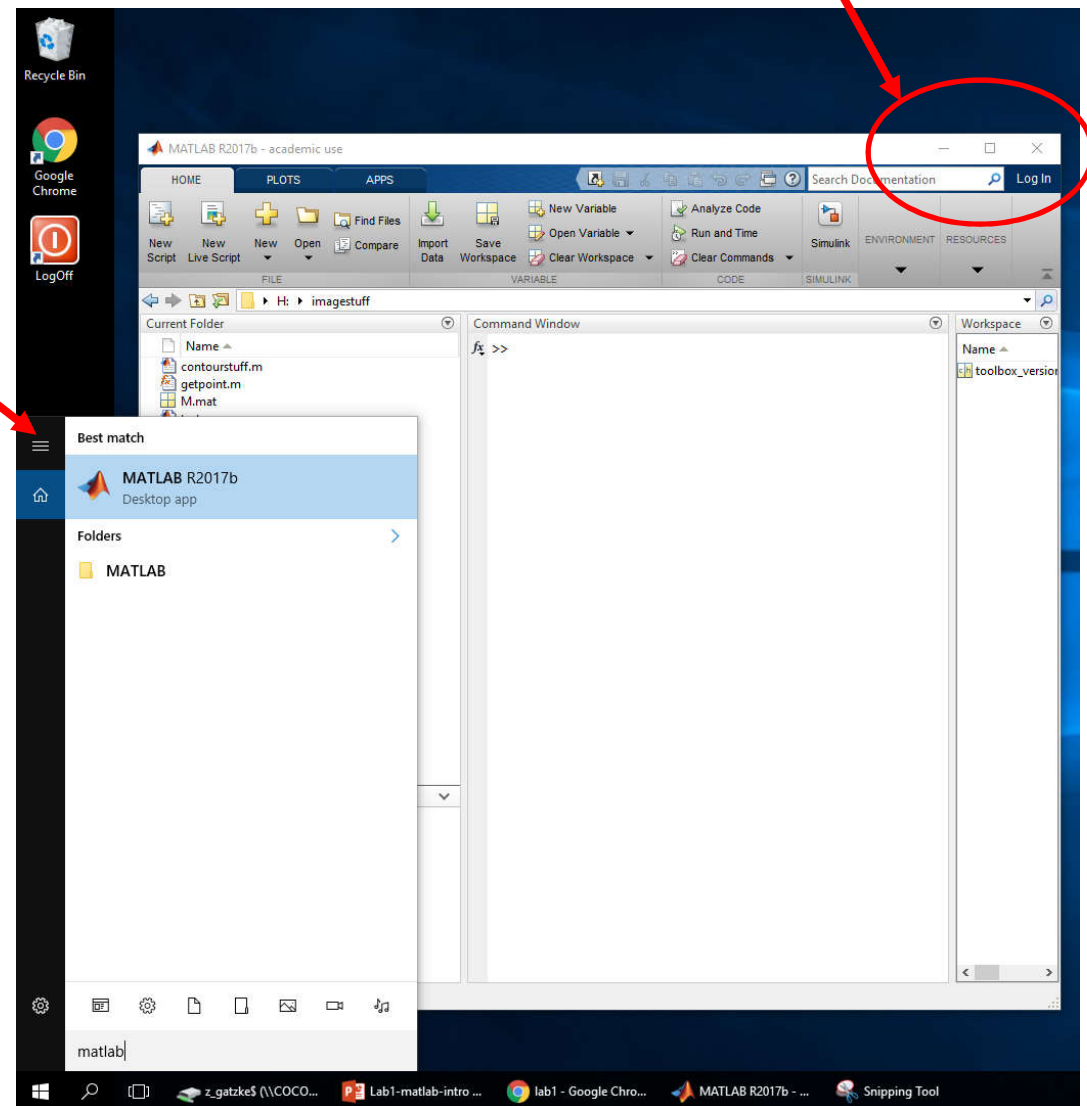


Lab 1

Introduction to MATLAB

Step 0 – Start MATLAB

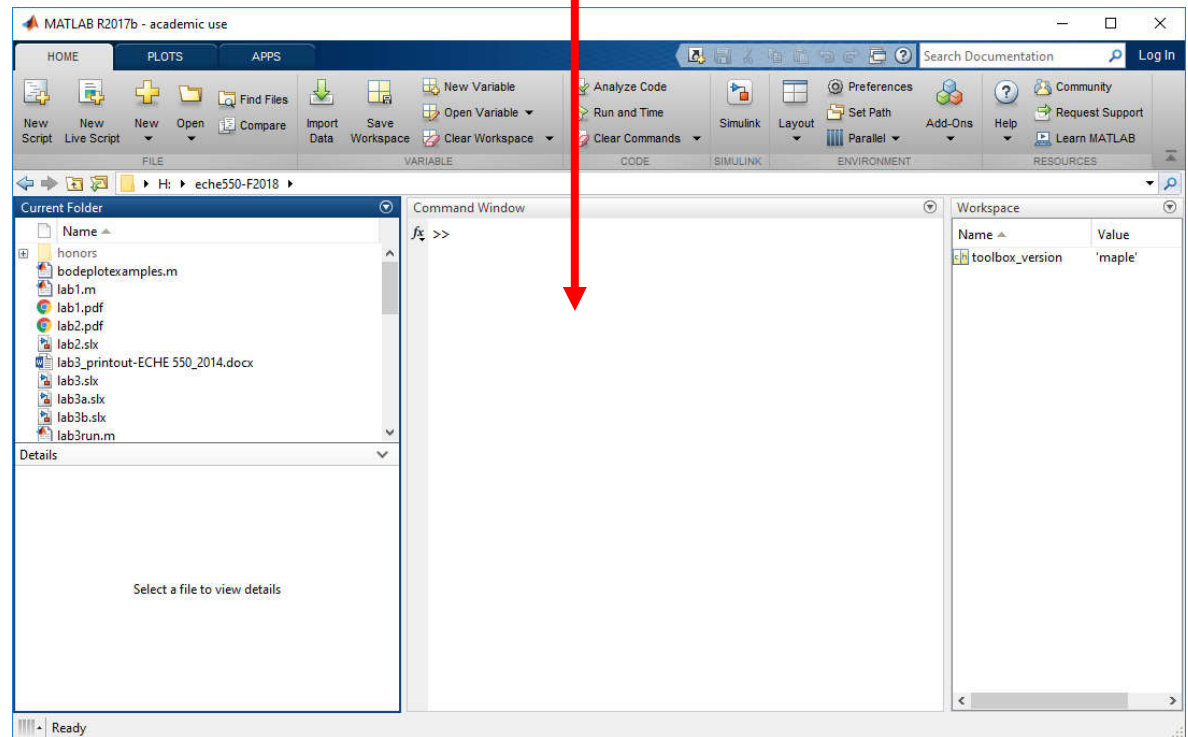
- Start MATLAB. Click the search icon in lower-left and type matlab into the search bar then click on Matlab
- Use ALT+Tab to switch between windows.
- Use ALT+Shift+Tab to switch the other way.
- You may want to maximize the MATLAB window and switch between MATLAB and this PDF file.
- You may want to have the PDF next to the MATLAB window.



Step 1 –MATLAB Interface

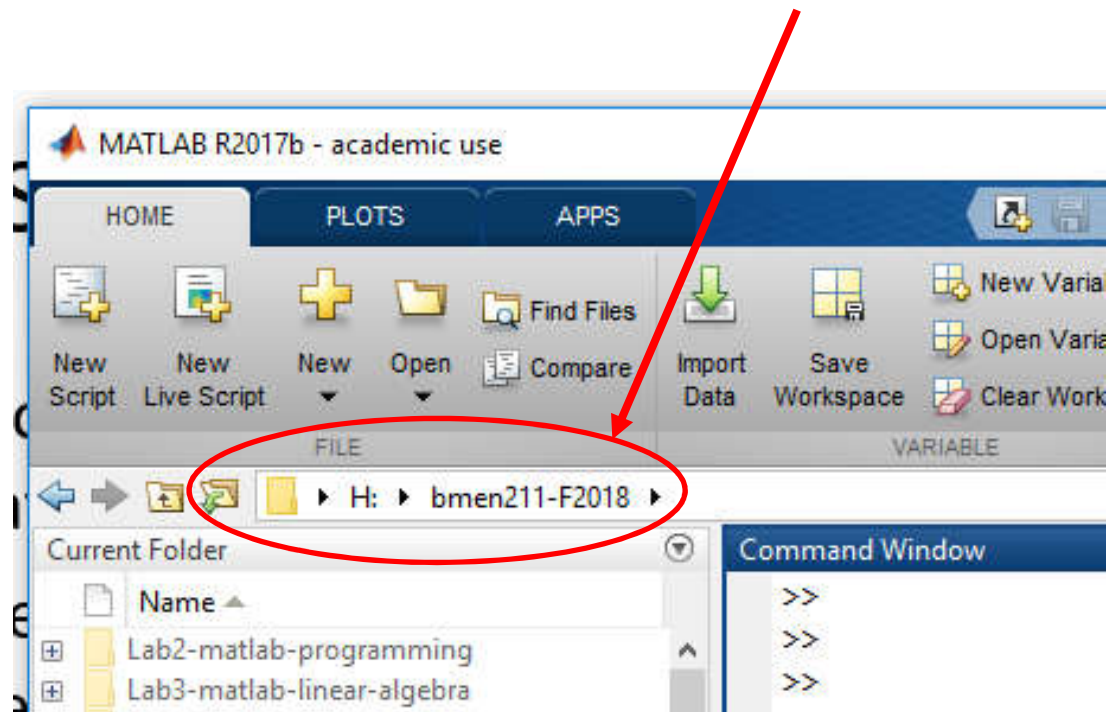
Command Window

- The main part of the MATLAB interface is the central Command Window
- You can type commands into this window
- At the command prompt `>>` in the command window type the command:
`pwd`
- and hit enter.
- This displays the current directory!



Step 2 – Set the Working Directory

- You need to set where MATLAB saves files!
- Click on the folder icon and select your working folder for this class.
- This is where you save your files!
- MATLAB files mostly have a “.m” extension.
- Example:
lab1.m
hw5.m

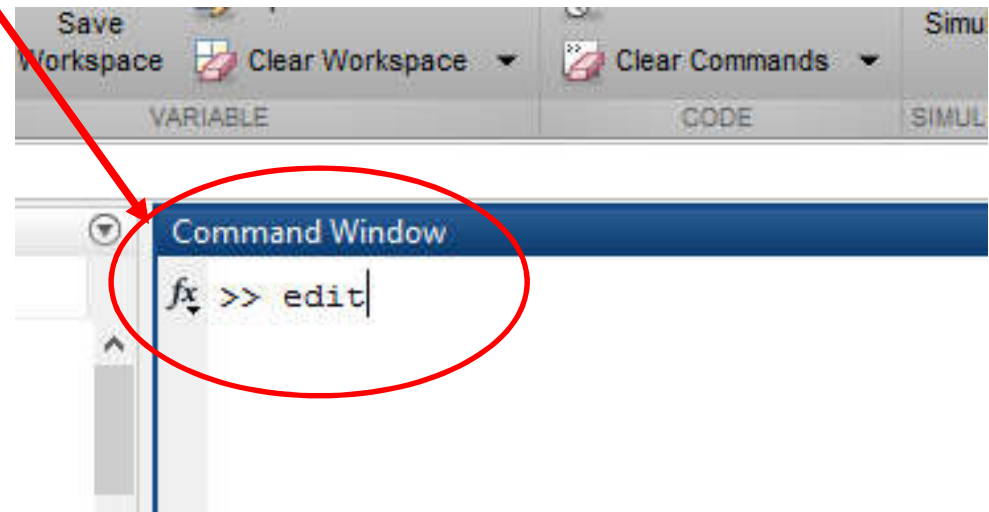


Step 3 – Start the Editor

- In the middle section where you see >> type the word:

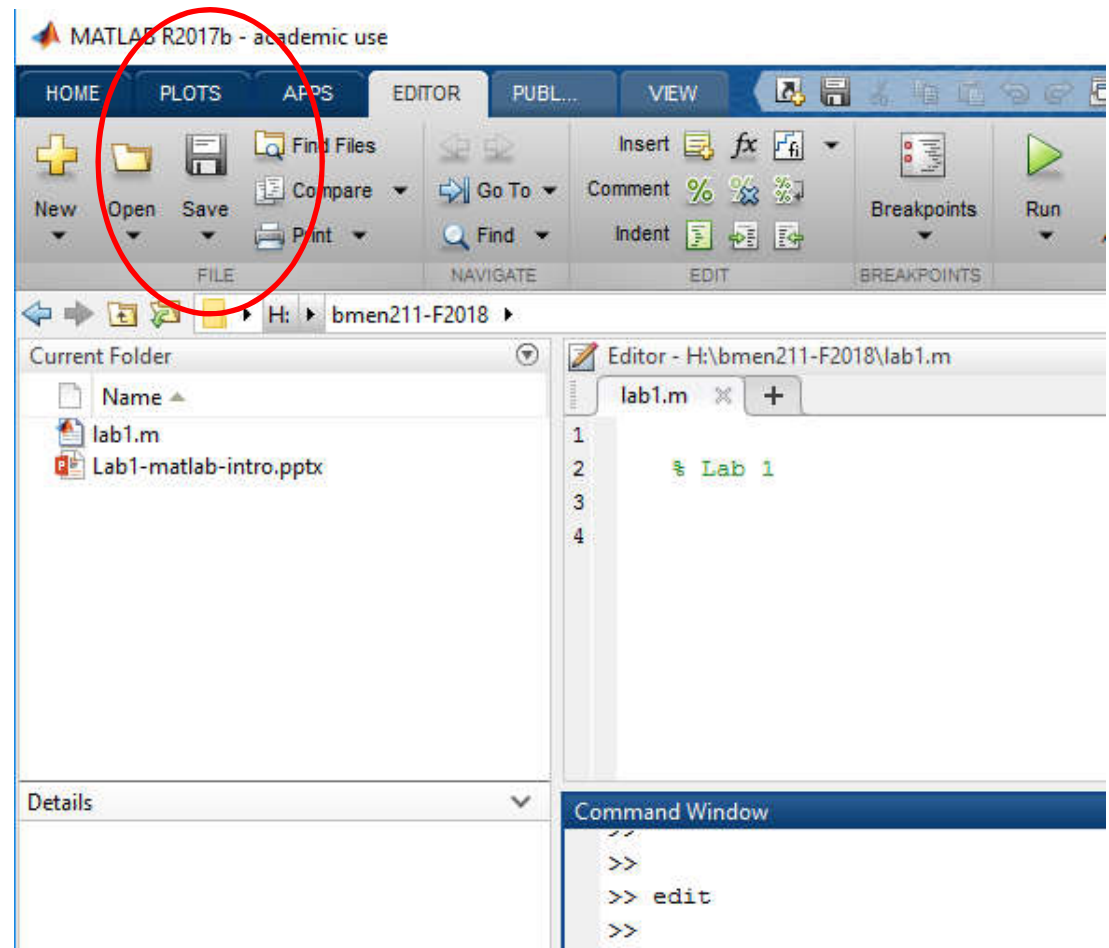
edit

- Hit return to start the editor.
- The editor is just a text file editor, but it has extra MATLAB functions.
- The editor starts “docked” in the central part of the window.
- The command window is at the bottom now.



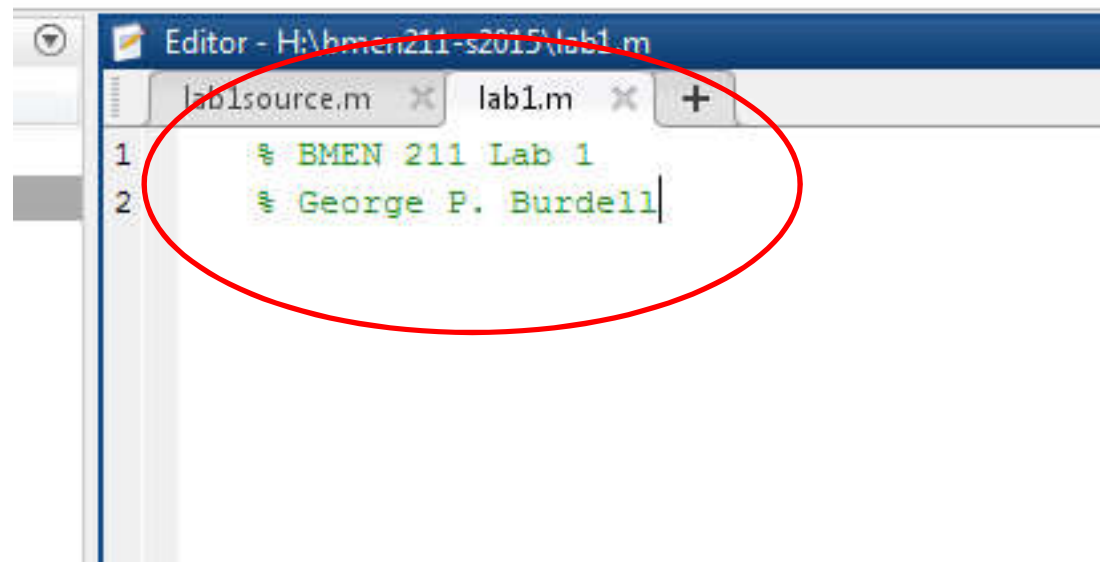
Step 4 – Save Your File

- Your MATLAB should now look like the image.
- Click on “Save” under the Editor tab to save your file.
- **YOU CANNOT USE SPACES IN FILE NAMES!**
- Give your file the name lab1
- The .m extension will be appended on automatically when it saves.



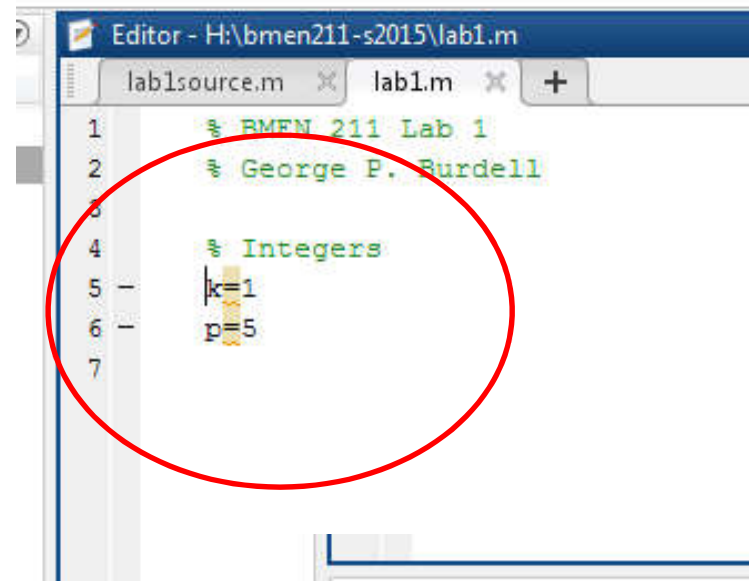
Step 5 – Comments

- Comments in MATLAB follow the % operator.
- Anything after a % will be ignored.
- Comments show in **green** in the editor.
- Add some comments to your file using the editor window.
- Obviously, use your name, not George's!



Step 6 – Make Variables with Integers

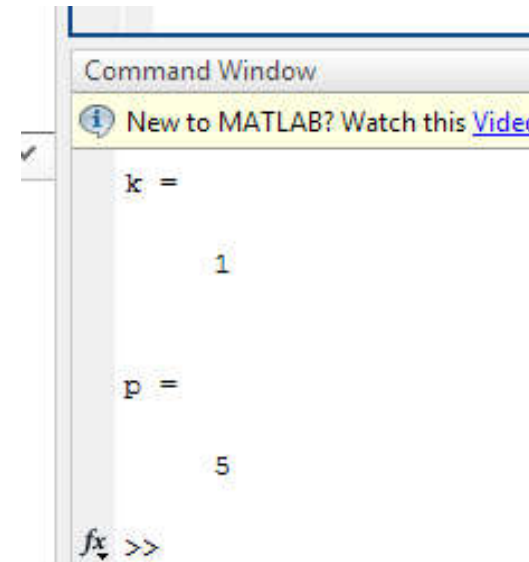
- MATLAB allows you to assign values to variables.
- In the editor window type the following two lines:
k=1
p=5
- Click the green triangle run button. This runs the script and you see the values in the command window.
- NOTE, the variables appear in the Workspace Window.



The image shows the MATLAB Editor window with the file 'lab1.m' open. The script contains the following lines:

```
1 % BMEN 211 Lab 1
2 % George P. Burdell
3
4 % Integers
5 k=1
6 p=5
7
```

A red circle highlights the lines 'k=1' and 'p=5'.



The image shows the MATLAB Command Window. It displays the output of the script:

```
k =
    1

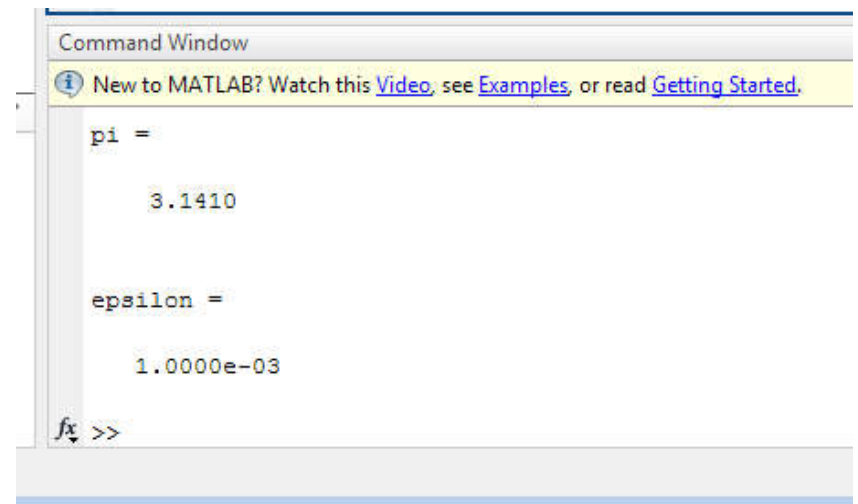
p =
    5
```

At the bottom, the prompt 'fx >>' is visible.

Step 7 – Make Variables with Real Values

- In the editor window type the following two lines:
pi=3.141
epsilon=0.001
- Click the green triangle run button or hit F5. This runs the script and you see the values in the command window.
- You could type commands in the command window, but you will make many mistakes.
- NOTE, the Workspace Window is always updated.

```
4 % Integers
5 k=1
6 p=5
7
8 % Real numbers (double precision)
9 pi=3.141
0 epsilon=0.001
```



The image shows the MATLAB Command Window after running the script. It displays the values of the variables 'pi' and 'epsilon'. 'pi' is assigned the value 3.1410, and 'epsilon' is assigned the value 1.0000e-03. The prompt 'fx >>' is visible at the bottom.

```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

pi =

    3.1410

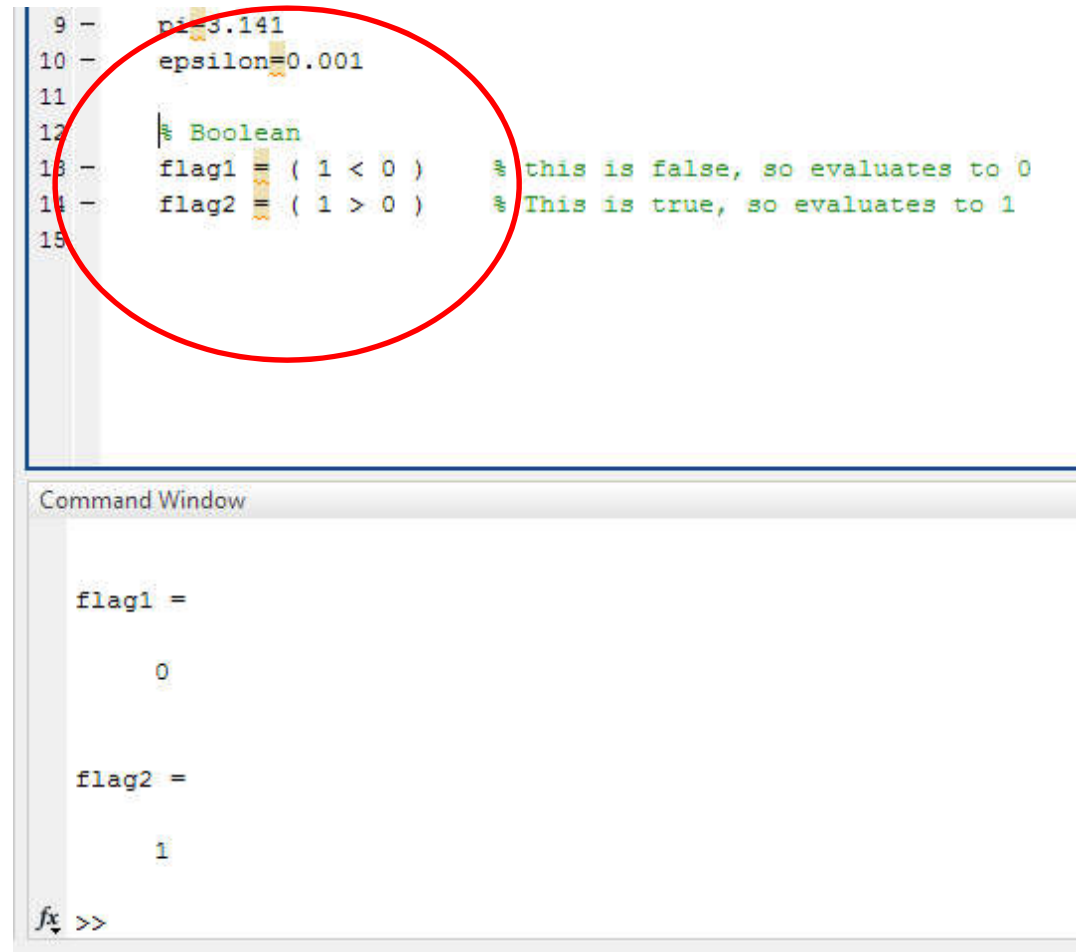
epsilon =

    1.0000e-03

fx >>
```

Step 8 – Boolean True/False Values

- Boolean values represent true or false.
- MATLAB uses 1 for TRUE and 0 for FALSE.
- In the editor window type the following two lines:
flag1 = (1 < 0)
flag2 = (1 > 0)
- Click the green triangle run button or hit F5.
- NOTE, the variables cannot have spaces or start with a number!



The image shows a MATLAB editor window with a script containing the following code:

```
9 - pi=3.141
10 - epsilon=0.001
11
12 % Boolean
13 - flag1 = ( 1 < 0 )    % this is false, so evaluates to 0
14 - flag2 = ( 1 > 0 )    % This is true, so evaluates to 1
15
```

A red circle highlights the two lines where the boolean flags are assigned. Below the editor is the Command Window, which displays the results of the execution:

```
Command Window

flag1 =

    0

flag2 =

    1

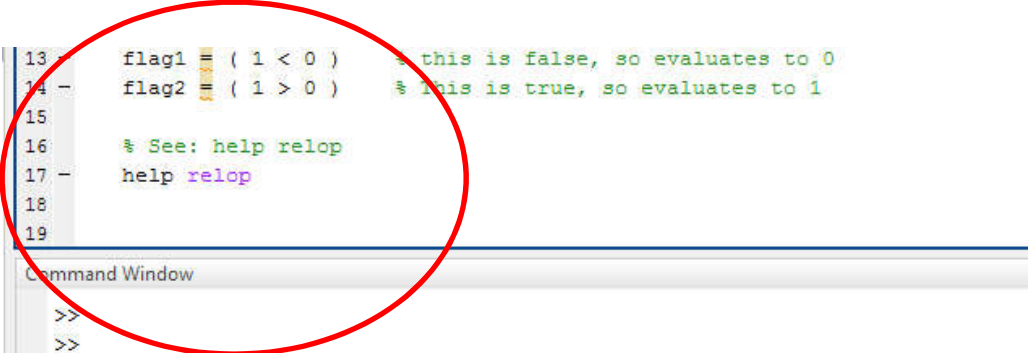
fx >>
```

Step 9 – Help

- There are many relational operators in MATLAB.
- In MATLAB the ampersand “&” means logical AND
- In MATLAB, the pipe “|” means logical OR
- MATLAB has a help command. In the editor window type the following line:

help relop

- Click the green triangle run button or just hit F5.
- Scroll up and down to read.
- Or type in **doc relop**



```
13 flag1 = ( 1 < 0 ) % this is false, so evaluates to 0
14 flag2 = ( 1 > 0 ) % this is true, so evaluates to 1
15
16 % See: help relop
17 help relop
18
19
```

Command Window

```
>>
>>
>>
>>
>> help relop
Relational operators.
< > Relational operators.
The six relational operators are <, <=, >, >=, ==, and ~=.
A < B does element by element comparisons between A and B
and returns a matrix of the same size with elements set to logical
1 (TRUE) where the relation is true and elements set to logical 0
(FALSE) where it is not. A and B must have the same dimensions
(or one can be a scalar).

& Element-wise Logical AND.
A & B is a matrix whose elements are logical 1 (TRUE) where both A
and B have non-zero elements, and logical 0 (FALSE) where either has
a zero element. A and B must have the same dimensions (or one can
be a scalar).

&& Short-Circuit Logical AND.
A && B is a scalar value that is the logical AND of scalar A and B.
This is a "short-circuit" operation in that MATLAB evaluates B only
if the result is not fully determined by A. For example, if A equals
```

Step 10 – Arrays

- Arrays - Arrays contain multiple pieces of data indexed along one or more dimensions.
- A vector can be seen as a 1D array of real numbers.
- A matrix can be seen as 2D array of real numbers.
- Type the following into the editor window:
b=[1 2 3 4 5 6]
b(1)
- Run the file. What does this do?

```
16 % See: help relop
17 - help relop
18
19 - b=[1 2 3 4 5 6] % This makes a row vector with six elements
20 - b(1) % You can access a single element of an array
21
22
23
```

Command Window

```
b =
    1     2     3     4     5     6

ans =
    1

fx Trial>>
```

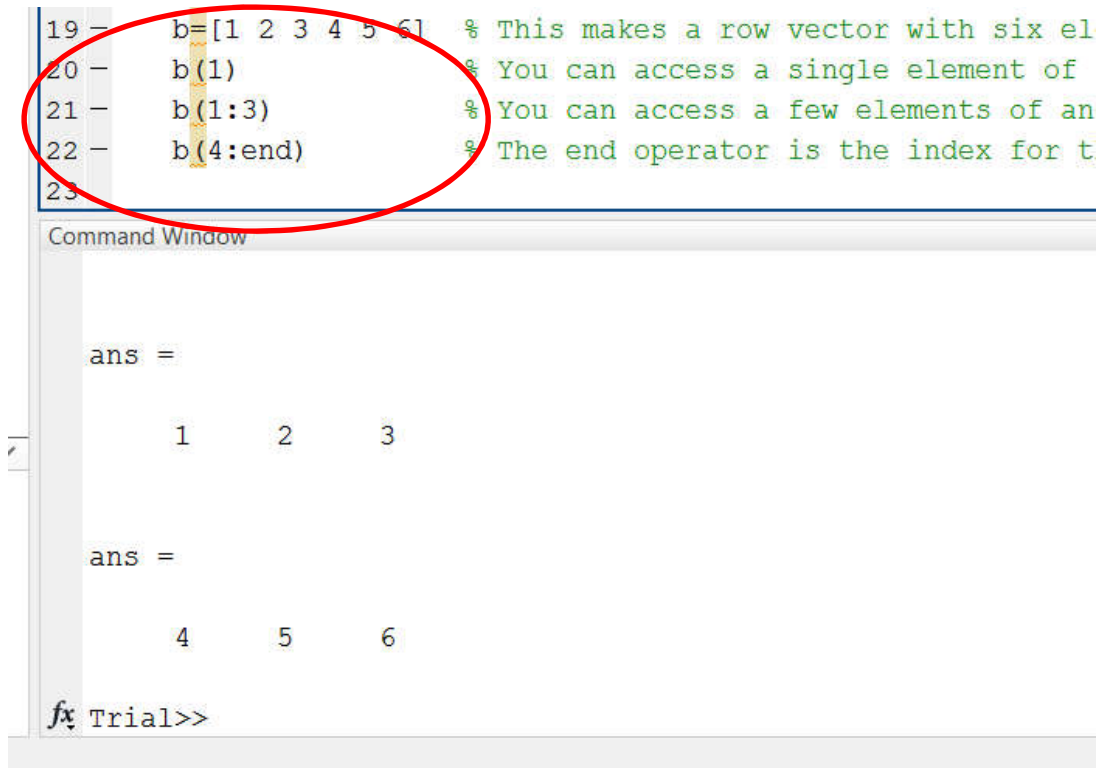
Step 11 – Array indexing

- You can make an array using square brackets.
- You can access information in an array using an index inside parenthesis.
- MATLAB starts the index counter at 1.
- Type the following into the editor window:

b(1:3)

b(4:end)

- Run the file. What does this do?



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

```
19 - b=[1 2 3 4 5 6] % This makes a row vector with six el
20 - b(1) % You can access a single element of
21 - b(1:3) % You can access a few elements of an
22 - b(4:end) % The end operator is the index for t
23
```

The Command Window displays the results of the commands:

```
ans =
     1     2     3

ans =
     4     5     6

fx Trial>>
```

A red circle highlights the first four lines of code in the editor window, and a yellow highlight is on the variable `b` in the first line.

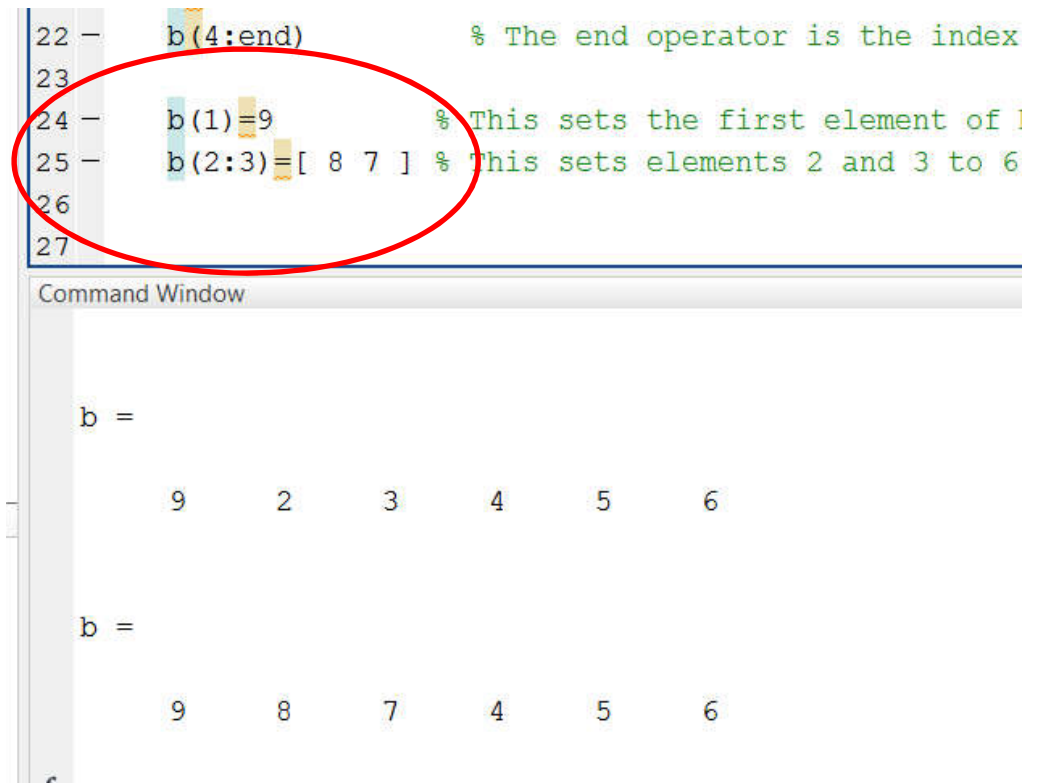
Step 12 – Changing Array Elements

- You can access multiple elements of an array.
- The end operator means the end of the array.
- You can also change elements in an array by assigning new values.
- Type the following into the editor window:

b(1)=9

b(2:3)=[8 7]

- Note: Both sides must be the same size!



The image shows a MATLAB editor window with the following code:

```
22 - b(4:end) % The end operator is the index
23
24 - b(1)=9 % This sets the first element of b to 9
25 - b(2:3)=[ 8 7 ] % This sets elements 2 and 3 to 8 and 7
26
27
```

A red circle highlights the lines 24 and 25. Below the editor is the Command Window showing the result of the operations:

```
b =
     9     2     3     4     5     6

b =
     9     8     7     4     5     6
```

Step 13 – Making New Rows

- To make new rows, use a semicolon between elements rather than spaces.
- Type the following into the editor window:
c=[1;2;3;4;5;6]
- This makes six rows, creating a column vector with six rows and one column.
- Spaces between values make new columns, semicolons make new rows.

```
24 - b(1)=9 % This sets the first element of b to 9
25 - b(2:3)=[ 8 7 ] % This sets elements 2 and 3 to 6 and 7
26
27 % Semicolons make new rows. You can create a column vector
28 - c=[1 ; 2 ; 3 ; 4 ; 5 ; 6]
29
```

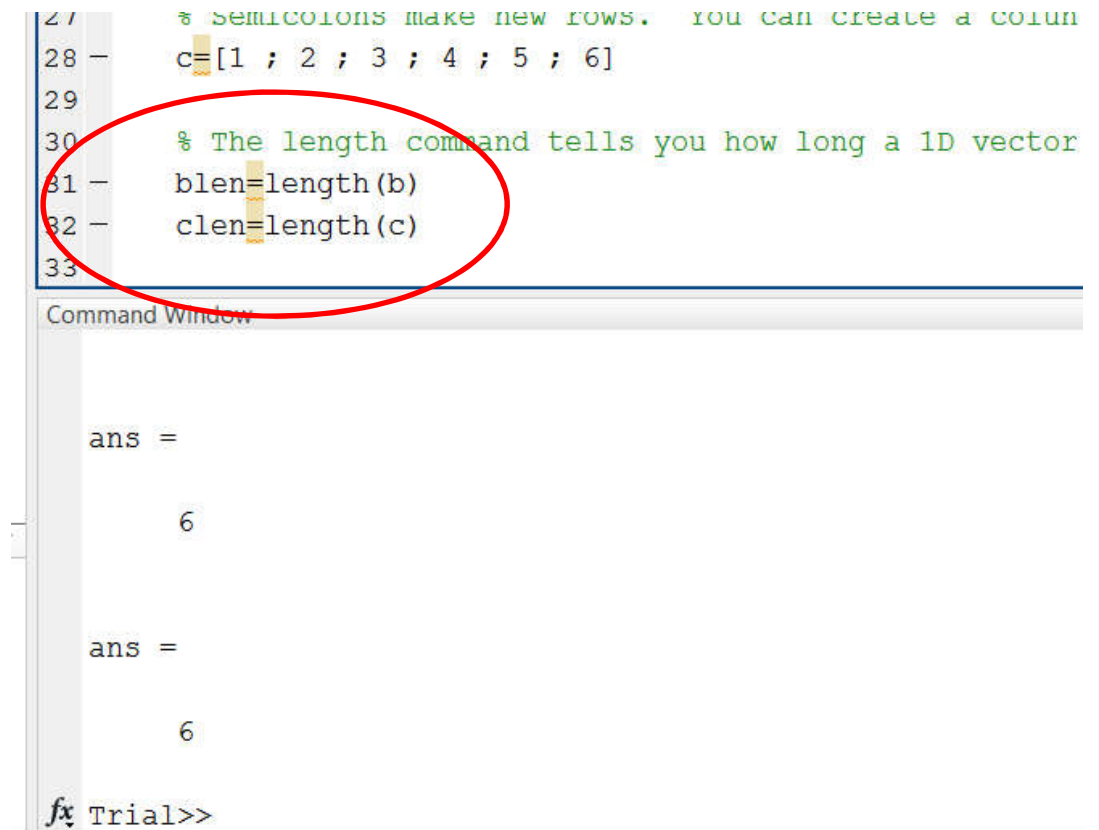
Command Window

```
c =
     1
     2
     3
     4
     5
     6

fx Trial>>
```

Step 14 – Using the “length” Function

- To find out how long an array is, use the length function.
- Type the following into the editor window:
blen=length(b)
clen=length(c)
- This makes two new variables containing the length of b and the length of c.

A screenshot of the MATLAB environment. The top part shows the MATLAB Editor with a script file. The script contains the following code: Line 27: % SEMICOLONS make new rows. You can create a column; Line 28: c=[1 ; 2 ; 3 ; 4 ; 5 ; 6]; Line 29: (empty line); Line 30: % The length command tells you how long a 1D vector; Line 31: blen=length(b); Line 32: clen=length(c); Line 33: (empty line). A red circle is drawn around lines 31 and 32. The bottom part shows the Command Window. It displays the results of the commands: 'ans = 6' for the first command and 'ans = 6' for the second command. The Command Window prompt is 'fx Trial>>'.

```
27 % SEMICOLONS make new rows. You can create a column
28 c=[1 ; 2 ; 3 ; 4 ; 5 ; 6]
29
30 % The length command tells you how long a 1D vector
31 blen=length(b)
32 clen=length(c)
33
```

Command Window

```
ans =
    6

ans =
    6

fx Trial>>
```

Step 15 – Using the “whos” Command

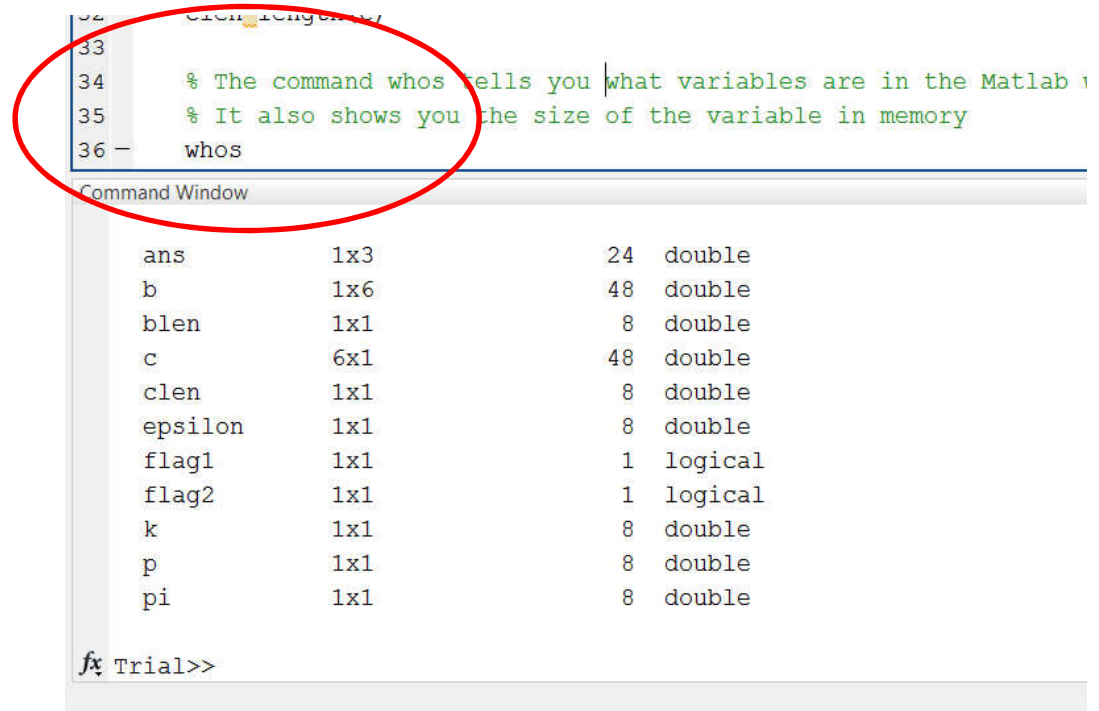
- To find out what variables are in the memory, the MATLAB workspace window gives some information.

- Type the following into the editor window:

whos

- This command shows what variables are defined, their size, and what type of information they contain.
- Note, you can get just some variables using wildcards like:

whos b*



```
33  
34 % The command whos tells you what variables are in the Matlab  
35 % It also shows you the size of the variable in memory  
36 whos
```

ans	1x3	24	double
b	1x6	48	double
blen	1x1	8	double
c	6x1	48	double
clen	1x1	8	double
epsilon	1x1	8	double
flag1	1x1	1	logical
flag2	1x1	1	logical
k	1x1	8	double
p	1x1	8	double
pi	1x1	8	double

fx Trial>>

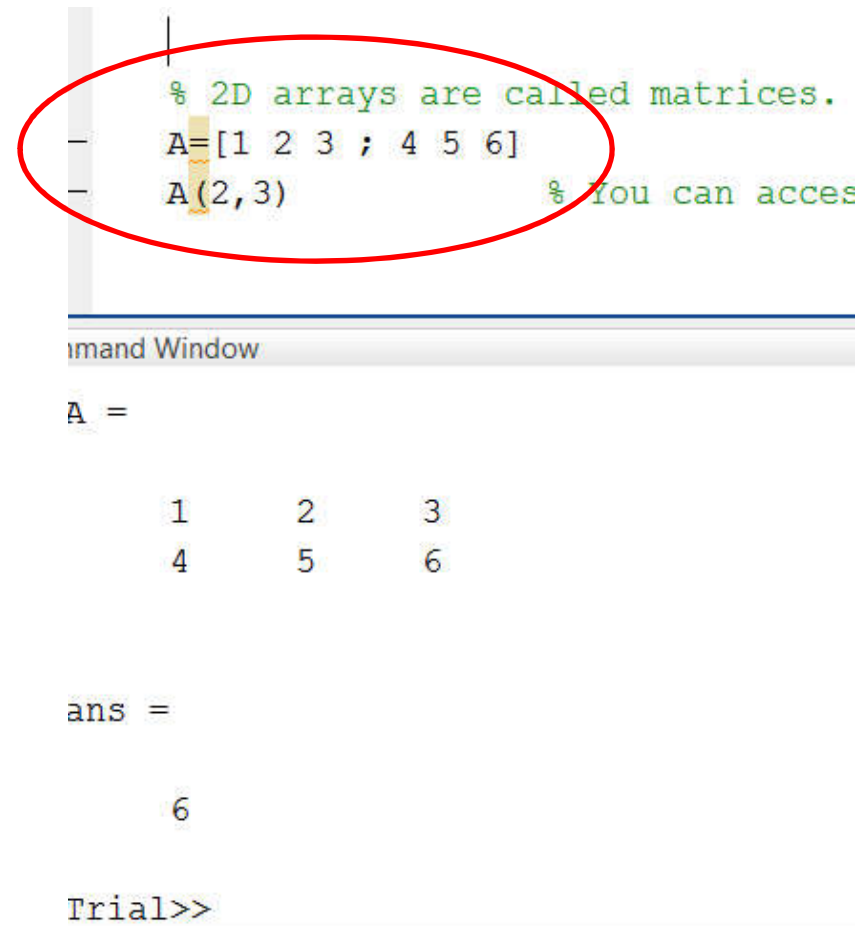
Step 16 – Making 2D Arrays (Matrices)

- 2D arrays are matrices.
- The matrix has two indices.
- The number of rows and number of columns define the size of the matrix.
- Type the following into the editor window:

A=[1 2 3 ; 4 5 6]

A(2,3)

- You can access individual element values from a matrix.



The image shows a MATLAB editor window with a red circle highlighting the first three lines of code: a comment, the matrix definition, and the element access command. Below the editor is the Command Window showing the resulting matrix A and the value of the expression A(2,3).

```
% 2D arrays are called matrices.  
A=[1 2 3 ; 4 5 6]  
A(2,3) % You can access individual elements
```

Command Window

```
A =  
  
     1     2     3  
     4     5     6  
  
ans =  
  
     6  
  
Trial>>
```

Step 17 – Accessing Multiple Matrix Elements

- The colon operator can be used to access multiple 2D array values.
- Type the following into the editor window:
A(:,1)
A(2,:)
A(1:2,2:3)
- Just using a colon access all the rows or all the columns.
- You can use numbers with the colon to access parts of the matrix.

```
40 - A(2,3)      % You can access a single element with row
41 - A(:,1)      % You can get all the rows from column 1 using :
42 - A(2,:)      % You can get all the columns from row 2 using :
43 - A(1:2,2:3)  % This gets rows 1 to 2 from column 2 to column 3
44 -

Command Window

ans =

     1
     4

ans =

     4     5     6

ans =

     2     3
     5     6

fx Trial>>
```

Step 18 – Using the “size” Function

- Sometimes you do not know the size of a matrix.
- The size command can determine the size in one of the dimensions.
- Type the following into the editor window:
size(A,1)
size(A,2)
[rows,cols]=size(A)
- The variables rows and cols will contain the size of matrix A.

```
43 - A(1:2,2:3) % This gets rows 1 to 2 from column 2 to col
44
45 % The size command tells you the size of an array in th
46 % dimension. Row is first, column is second.
47 - size(A,1)
48 - size(A,2)
49 % The size command can also return two values at once
50 - [rows,cols]=size(A)
51
```

Command Window

ans =

2

ans =

3

rows =

2

cols =

3

f Trial>>

Step 19 – More Complex Arrays

- MATLAB can make arrays with more dimensions.
- A 3D array is like a stack of matrices.
- Type the following into the editor window:
DD(2,2,2,2)=5
- This makes a four dimensional array, a list of a stack of 2D matrices with four indices.

```
51  
52 % Note that you can use N dimensional arrays,  
53 % but some operations like matrix multiplication won't work:  
54 - DD(2,2,2,2)=5  
55 % A 3D array is like a stack of matrices. A 4D array is a gr  
56
```

Command Window

DD(:, :, 2, 1) =

0	0
0	0

DD(:, :, 1, 2) =

0	0
0	0

DD(:, :, 2, 2) =

0	0
0	5

fx Trial>>

Step 20 – Using Strings

- Strings are just arrays that contain characters instead of numbers.
- MATLAB uses single quotes (next to enter) to make a string array.
- Type the following into the editor window:

name='bubba'
city='Columbia'
name(2:4)

```
59 - clear D*
60 % The clear command without a variable name clears a
61
62 % Strings are really just a 1D array of single chara
63 - name='bubba'
64 - city='columbia'
65 - name(2:4)
66
```

Command Window

```
name =
bubba

city =
columbia

ans =
ubb

fx Trial>>
```

Step 21 – Using the “ones” and “zeros” Functions

- The ones command makes a matrix full of ones.
- The zeros command makes a matrix full of zeros.
- Type the following into the editor window:

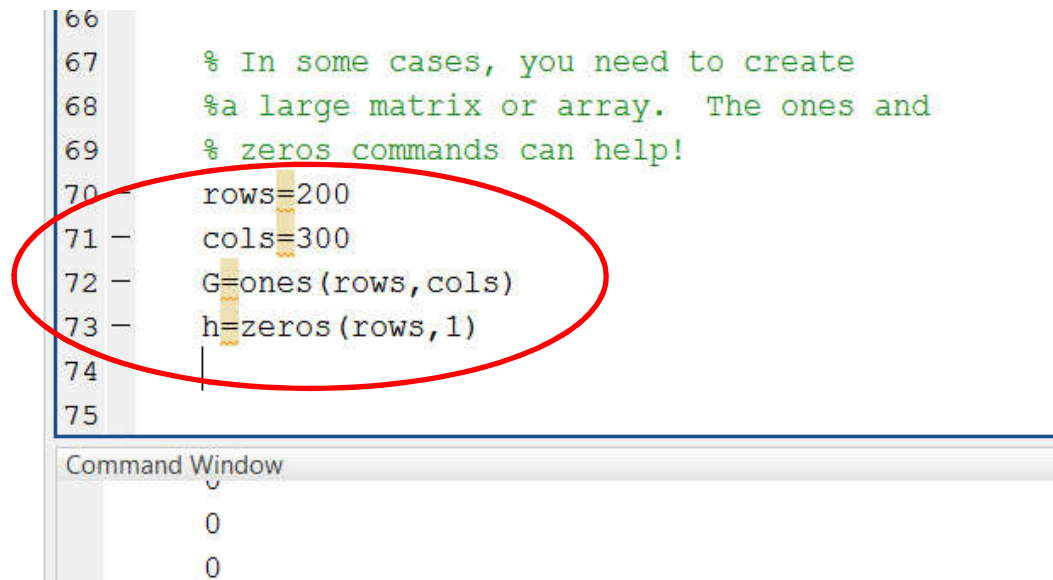
rows=200

cols=300

G=ones(rows,cols)

h=zeros(rows,1)

- Note: Adding a semicolon after a line suppresses command window output!

A screenshot of the MATLAB environment. The top part shows a code editor with line numbers 66 to 75. Lines 67-69 contain comments in green. Lines 70-73 contain code: rows=200, cols=300, G=ones(rows,cols), and h=zeros(rows,1). These four lines are circled in red. Line 74 has a vertical cursor. Below the code editor is a 'Command Window' pane showing two zeros, one on each line.

```
66
67 % In some cases, you need to create
68 % a large matrix or array. The ones and
69 % zeros commands can help!
70 rows=200
71 cols=300
72 G=ones(rows,cols)
73 h=zeros(rows,1)
74
75
```

Command Window

```
0
0
```

Step 22 – Using the “diary” Command

- The diary command writes stuff from the command window to a text file.
- Type the following into the editor window:

!erase lab1diary.txt

diary lab1diary.txt

str='Lab 1 – George Burdell, Sect. 1'

whos

diary off

- Run your file.

```
7      G=ones(rows,cols);
8      h=zeros(rows,1);
9      |
0      % The diary command can copy everything done in the editor to a text file
1      !erase lab1diary.txt           % This erases any old file
2      diary lab1diary.txt           % This turns on the diary
3
4      str='Lab 1 - George Burdell, Sect. 1'% Change this line to your info!
5      whos                          % This lists all variables in memory
6      diary off    % This stops the diary
7
8
```

Command Window

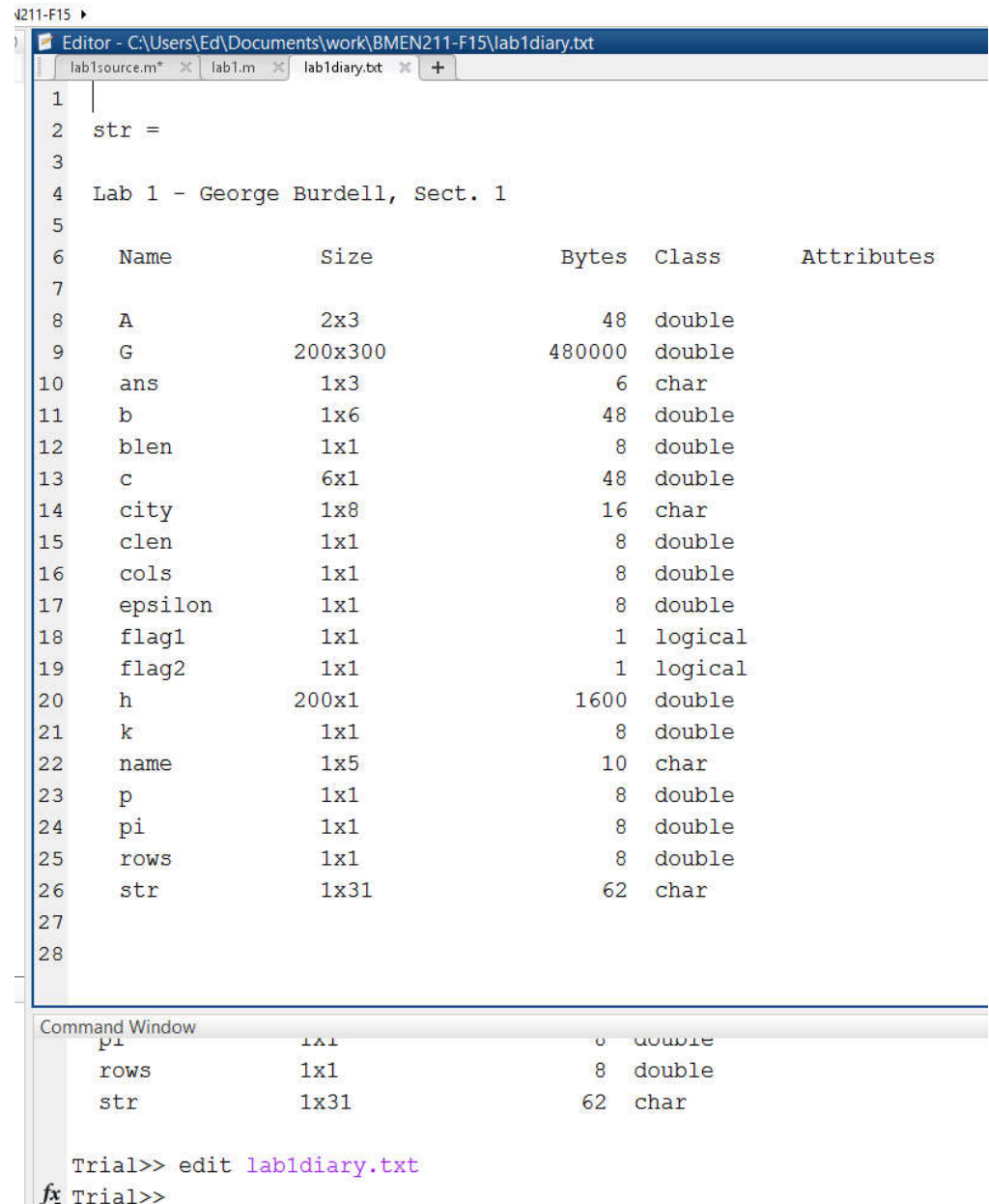
Step 23 – Print File

- In the command window, (not the editor window!) type the following

edit lab1diary.txt

and hit return.

- It should contain something similar to the image.
- Print out your diary file to turn in, showing that you completed this tutorial. DO NOT print out the .m file



The screenshot shows the MATLAB Editor window with the file `lab1diary.txt` open. The file contains a table with 6 columns: Name, Size, Bytes, Class, and Attributes. The table lists various parameters and their properties. Below the editor, the Command Window shows the command `edit lab1diary.txt` being entered.

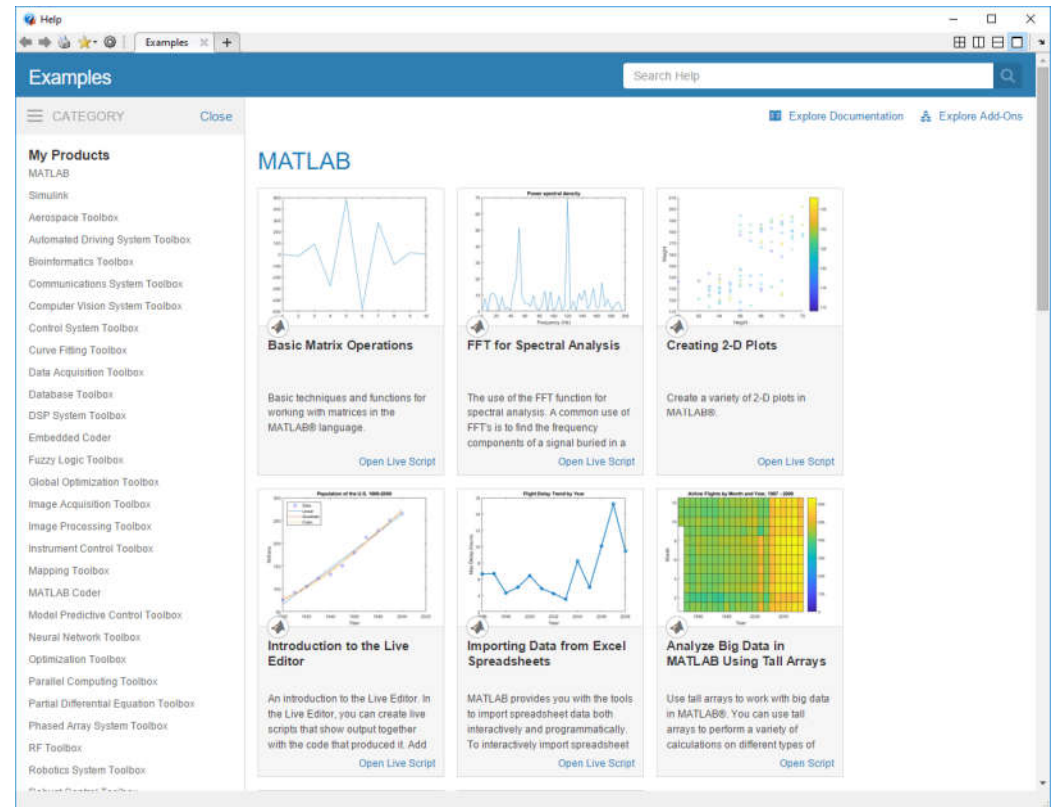
```
1  
2 str =  
3  
4 Lab 1 - George Burdell, Sect. 1  
5  
6 Name          Size          Bytes  Class    Attributes  
7  
8 A              2x3              48  double  
9 G             200x300      480000 double  
10 ans            1x3               6  char  
11 b              1x6              48  double  
12 blen           1x1               8  double  
13 c              6x1              48  double  
14 city           1x8              16  char  
15 clen           1x1               8  double  
16 cols           1x1               8  double  
17 epsilon        1x1               8  double  
18 flag1           1x1               1  logical  
19 flag2           1x1               1  logical  
20 h              200x1            1600  double  
21 k              1x1               8  double  
22 name           1x5              10  char  
23 p              1x1               8  double  
24 pi             1x1               8  double  
25 rows           1x1               8  double  
26 str            1x31             62  char  
27  
28
```

Command Window

```
pi              1x1               8  double  
rows            1x1               8  double  
str             1x31             62  char  
  
Trial>> edit lab1diary.txt  
fx Trial>>
```

Step 24 – Run Demos

- If you have extra time, go to the command prompt and type
demos
- In the command window and look through the various demos.
- You may want to check out the following:
 - Bioinformatics Toolbox
 - Image Processing Toolbox
 - SimBiology
 - Neural Network Toolbox



MATLAB R2017b - academic use

H... P... A... E... P... V... Search Documentation Log In

H: bmen211-F2018

Editor - H:\bmen211-F2018\lab1source.m

lab1source.m

```
1 % Comments vary from language to language.
2 % Use comments to explain what your code is doing.
3 % In Matlab, anything to the right of a % is treated as a comment!
4 % In the Matlab editor, everything that is a comment is in green
5
6 % First, you have to make sure Matlab is in the correct directory.
7 % To print the current working directory, use pwd
8 pwd
9
10 % To change the current directory, you can use the cd command or the Matlab
11 % graphical user interface above the script area.
12
13 % DATA STRUCTURES
14
15 % We use variables to represent data of different types
16 % Traditional data structures include:
17
18 % Integers
19 k=1
20 p=5
21
22 % Real numbers (double precision)
23 pi=3.141
24 epsilon=0.001
25
26 % and boolean TRUE FALSE expressions.
27 % Note that in MATLAB, boolean is expressed as 0=false, 1= true.
28
29 flag1 = ( 1 < 0 ) % this is false, so evaluates to 0
30 flag2 = ( 1 > 0 ) % This is true, so evaluates to 1
31
32 % Also note that there are various Boolean operators you can use.
33 % These include <, >, <=, >=, ==, ~=.
34 % In Matlab the ampersand & means logical AND
35 % In Matlab, the pipe | means logical OR,
36 % ~ means complement, and xor is XOR
37 % See: help relop
38 help relop
39
```

script Ln 1 Col 1

MATLAB R2017b - academic use

H... P... A... E... P... V... Search Documentation Log In

H: \bmen211-F2018

Editor - H:\bmen211-F2018\lab1source.m

lab1source.m

```
40 % Special data structures
41
42 % Arrays - Arrays contain multiple pieces of data indexed
43 % along one or more dimensions for example, a vector can be
44 % seen as a 1D array of real numbers and a matrix can be
45 % described as a 2D array of real numbers. A 1D array is
46 % often called a vector
47
48 b=[1 2 3 4 5 6] % This makes a row vector with six elements
49 b(1) % You can access a single element of an array
50 b(1:3) % You can access a few elements of an array
51 b(4:end) % The end operator is the index for the end of the array
52
53 b(1)=7 % This sets the first element of b to 7
54 b(2:3)=[6 5] % This sets elements 2 and 3 to 6 and 5
55
56 % Semicolons make new rows. You can create a column vector this way
57 c=[1 ; 2 ; 3 ; 4 ; 5 ; 6]
58
59 % The length command tells you how long a 1D vector is
60 blen=length(b)
61 clen=length(c)
62
63 % The command whos tells you what variables are in the Matlab workspace
64 % It also shows you the size of the variable in memory
65 whos
66
67 % 2D arrays are called matrices. They have two indices for row and column.
68 A=[1 2 3 ; 4 5 6]
69 A(2,3) % You can access a single element with row
70 A(:,1) % You can get all the rows from column 1 using :
71 A(2,:) % You can get all the columns from row 2 using :
72 A(1:2,2:3) % This gets rows 2 to 3 from column 1 to column 2
73
74 % The size command tells you the size of an array in the specified
75 % dimension. Row is first, column is second.
76 size(A,1)
77 size(A,2)
78 % The size command can also return two values at once
79 [rows,cols]=size(A)
80
```

script Ln 1 Col 1

MATLAB R2017b - academic use

Search Documentation Log In

H: > bmen211-F2018

Editor - H:\bmen211-F2018\lab1source.m

```
lab1source.m
81 % Note that you can use N dimensional arrays,
82 % but some operations like matrix multiplication won't work:
83 DD(2,2,2,2)=5
84 % A 3D array is like a stack of matrices. A 4D array is a group of 3Ds...
85
86 % To forget the values of a variable, use the clear command
87 clear DD
88 clear D*
89 % The clear command without a variable name clears all the memory
90
91 % Strings are really just a 1D array of single characters
92 name='bubba'
93 city='columbia'
94 name(2:4)
95
96 % You can use arrays of strings
97
98
99 names={'Bobbie','Sue','Thom'}
100 names(2)
101
102 % Most languages handle strings and structures differently,
103 % so watch out when using other programs.
104
105 % In some cases, you need to create
106 % a large matrix or array. The ones and
107 % zeros commands can help!
108 rows=200
109 cols=300
110 G=ones(rows,cols)
111 h=zeros(rows,1)
112
113 % Note that Matlab displays all the contents. To suppress this output and
114 % speed up processing, use a semicolon
115
116 G=ones(rows,cols);
117 h=zeros(rows,1);
118
119 % The diary command can copy everything done in the editor to a text file
120 !erase labidiary.txt % This erases any old file
121 diary labidiary.txt % This turns on the diary
122
123 str='Lab 1 - George Burdell, Sect. 1' % Change this line to your info!
124 whos % This lists all variables in memory
125 diary off % This stops the diary
126
127
```

script Ln 42 Col 1