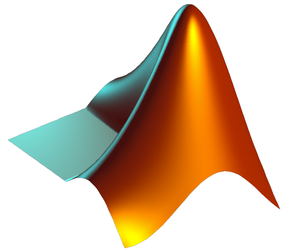
Discussion 1

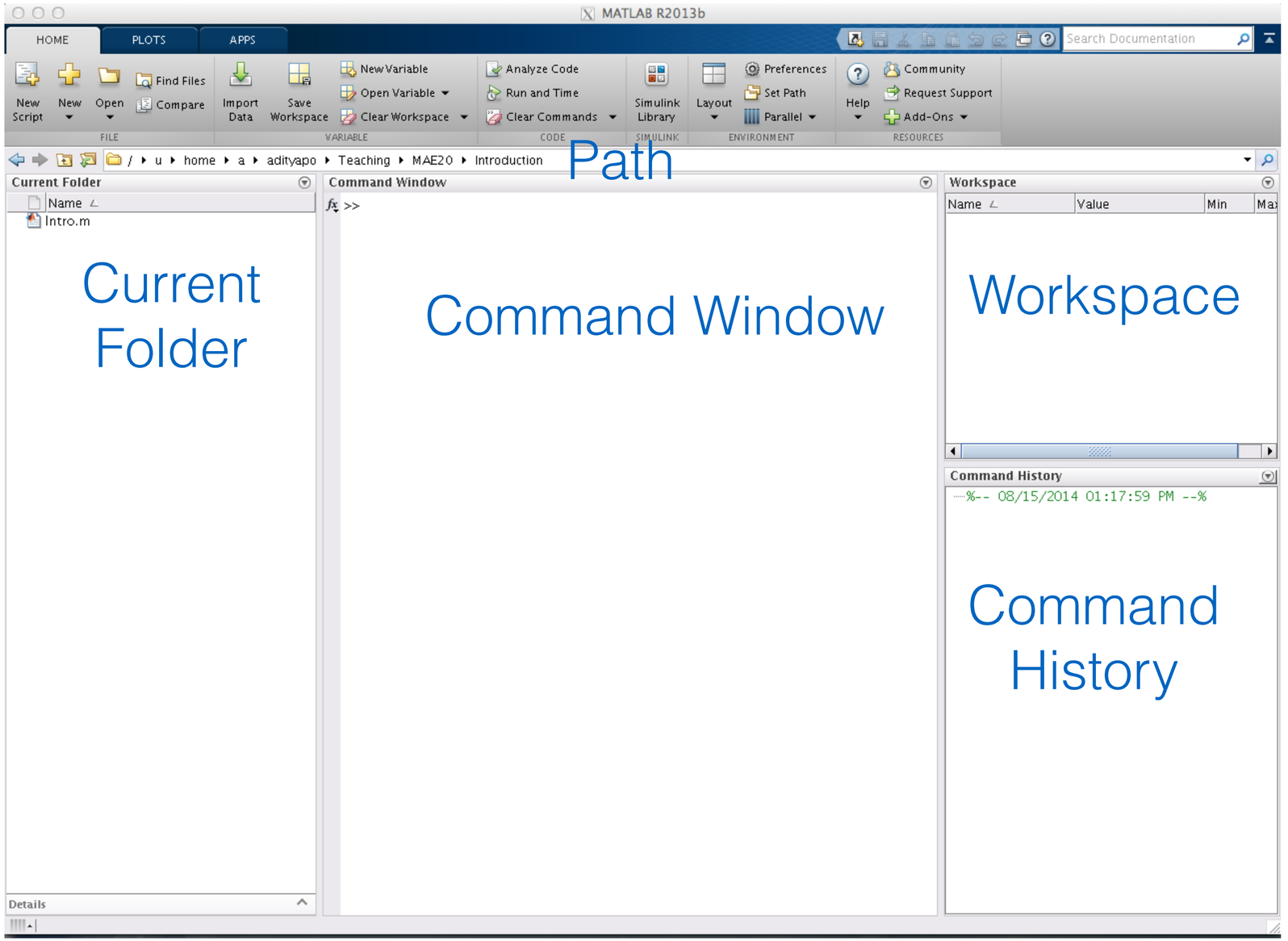
CEE/MAE M20
Introduction to Computer Programming with MATLAB

What is MATLAB?



- MATLAB = MATrix LABoratory
- MathWorks (the developers of MATLAB): “MATLAB® is a high-level language and interactive environment for numerical computation, visualization, and programming.”
- Provides toolboxes and packages for a wide range of problems including image processing, finance, biology, etc.
- Interpreted Language. (Not a Compiled Language)

MATLAB Interface



Toolstrip

Path

Current Folder

Command Window

Workspace

Command History

Windows

- Current Folder - directory contents of the current path
- Command Window - allows single line execution of commands
- Workspace - displays current variables stored in memory.
- Command History - shows most recent commands executed using the command line.
- Toolstrip - menus for opening/saving scripts, importing data, viewing documentation, etc.

Command Line

- We can use MATLAB as a simple calculator via the command line

```
>> 5 + 3
ans =
      8
```

- This assigns the value `'8'` to the variable `'ans'`, which is displayed in the workspace

- Adding a semicolon to the end of the expression suppresses the display of `'ans'`, but `'ans'` still stores the value `'8'`

```
>> 5 + 3;
```

- MATLAB has built-in functions, constants and variables

```
>> pi
ans =
    3.1416
```

```
>> cos(pi)
ans =
    -1
```

```
>> sin(pi)
ans =
    1.2246e-16
```

- Why did that happen?

- The function `'sin'` can only be approximated (for example, by a Taylor series)
- pi is an irrational constant made of infinite digits
- Computers cannot store infinitely long numbers

help

- To learn about MATLAB's built-in functions and variables, we can use the `help` feature

-
```
>> help sin
sin     Sine of argument in radians.
sin(X) is the sine of the elements of X…
```

-
```
>> help pi
pi      3.1415926535897....
pi = 4*atan(1) = imag(log(-1)) = 3.1415926535897….
```

- Also, `doc` is a very useful command, which brings up the MATLAB documentation window.

lookfor

- MATLAB's `lookfor` function searches through all help entries for a specific keyword

- ```
  >> lookfor exponent
  exp                 - Exponential.
  expm                - Matrix exponential.
  expint              - Exponential integral function.
  expmdemo            - Matrix Exponentials
  expmdemo1           - Matrix exponential via Pade
  approximation.
  …
  ```


  ```

Warnings/Errors

- Error messages are extremely helpful for debugging

- ```
  >> coss(pi)
  Undefined function 'coss' for input arguments
  of type 'double'.
  ```


  ```

Command Line History

- Instead of retyping `cos(pi)` with correct spelling, we can use the up arrow key
- The up arrow key browses through the history of previous commands
- Especially useful for lengthy commands
- We can also select previous commands directly from the Command History window

Basic Data Types

- Numeric —> double (real), integer
- Logical (boolean)
- Character —> a string is a 1D array of characters

Variables

- Variables can be assigned a value without declaring a type
- MATLAB implicitly converts a variable to a specific type
- The assignment operator, `'='`, is used to give a variable a defined value
-
```
% Below, the value 10 is assigned to variable
'distance'
distance = 10;
```

Variables

- A variable of one type can be assigned data of another type

```
distance = 10; % type double
distance = 'far'; % type char array (string)
```

- In MATLAB, scalars are actually defined as matrices of size 1x1

Clear and Close Commands

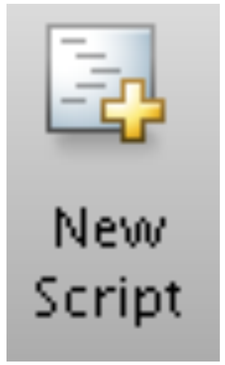
```
% Clear command window
clc;

% Clear variable from the workspace
clear;

% Close figure
close;

% Clear (but do not close) current figure
clf;

% Typically scripts begin with
clc; close all; clear all;
```



MATLAB Scripting - .m File

- A .m file is a MATLAB script file
- For programs longer than a few lines, use of the command line becomes inefficient
- To run an .m file saved as `filename.m`, simply type `filename` into the command line

```
>> filename
```

- Can also press F5 or Run (On the Toolstrip)

Comments

- Programmer-readable text embedded into the source code of a program
- Ignored by compilers and interpreters
- Primary purpose is to make source code easier to understand

- Comments help all users (including the original writer) to understand a script
- Use comments to concisely explain a piece of code
- Typical convention is to place a comment prior to the code snippet it explains

```
% Calculate area of a circle
area = pi*radius^2;
```

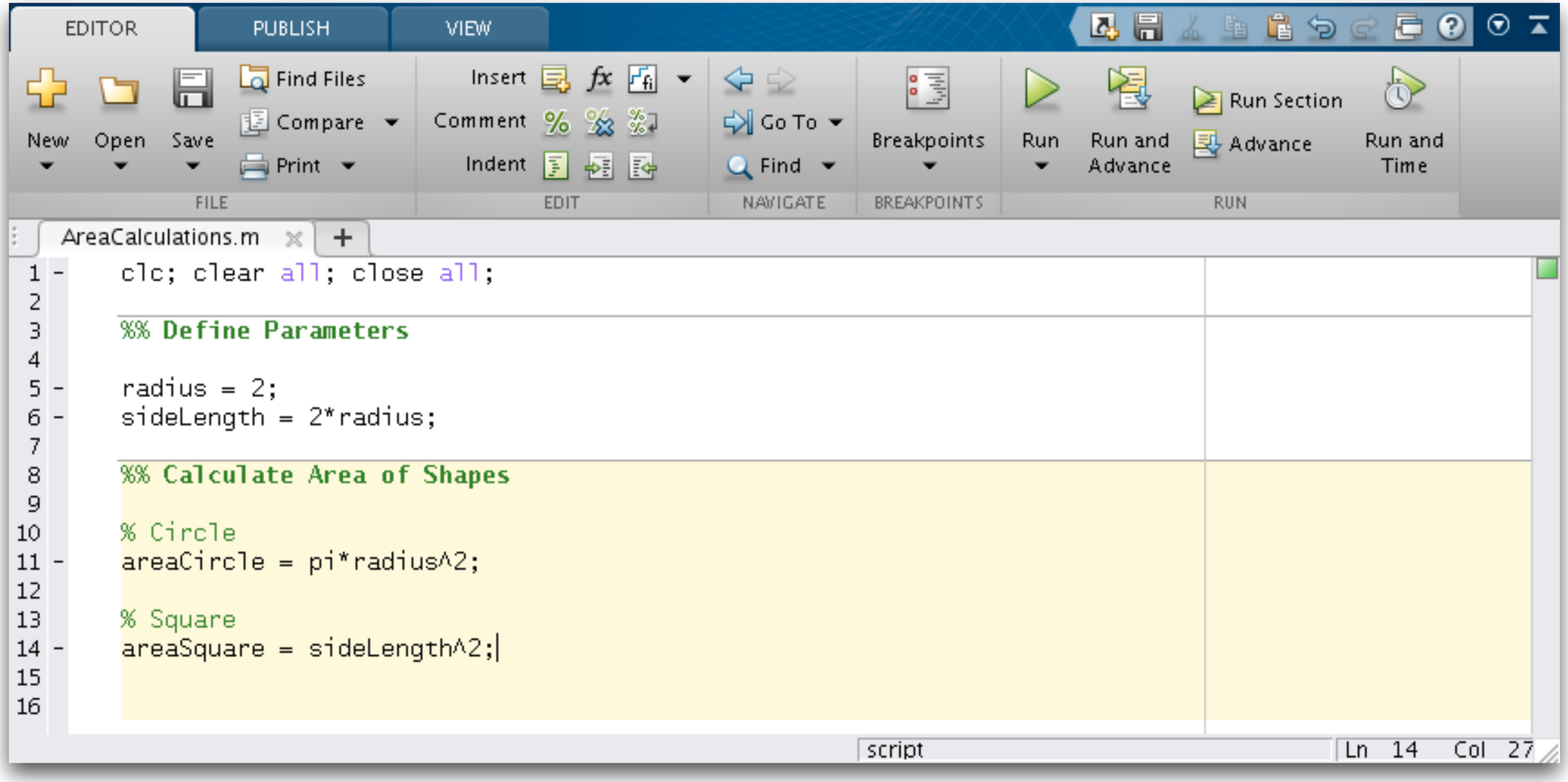
- In MATLAB, comment lines begin with `%`
- To comment multiple lines, use `%{ ... }%`
  - The first and final lines of a multi-line comment must be empty besides the delimiter
-
```
% This is a single line comment
%{
This is a multi-line comment
This text is green
%}
```

- Double percent signs (`%%`) define a code section, which can be followed by an optional title

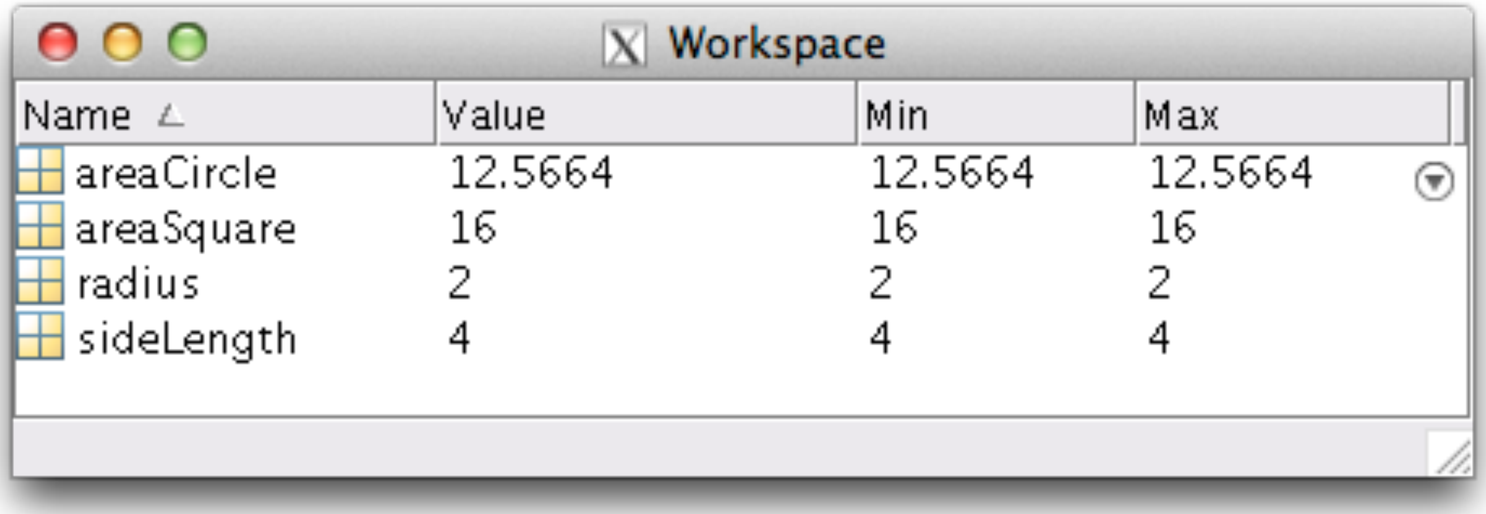
```
%% Calculate area of shapes
% Circle
areaCircle = pi*radius^2;
% Square
areaSquare = sideLength^2;
```

- Splitting code into sections allows the user to execute portions of code independently and keep code organized

Writing a short script in the editor



Once a script is executed, the variables appear in the workspace



Typing the script into the command window line by line will achieve the same result

Taking input from the command line

- Use MATLAB `input` function

```
% The code fragment below requests user input
and assigns the input to the variable
'firstName'
firstName = input('Enter your first name: ');
```

Printing to the command line

- `fprintf` allows you to control the way data are printed to the command window
- `disp` also prints to the command window but the output is not customizable

`fprintf` Formatting

- `fprintf(formatSpec,variable1,variable2,…,variableN)`
- *`formatSpec`* is a string that determines how data are printed

Format Specifier	Description
`'%s'`	String
`'%d'` or `'%i'`	Integer
`'%f'`	Floating Point (Real)
`'%e'`	Scientific Notation
`'\n'`	New Line
`'\t'`	Tab

```
% The following code prints 'myNumber' in
three different ways, skipping a line between
each output

myNumber = 100;

fprintf('%d\n',myNumber);
fprintf('%e\n',myNumber);
fprintf('%f\n',myNumber);
```

- *Field width* specifies the minimum number of characters to print

- Entered before format specifier

```
fprintf('%10f',someNumber)
```

-
```
% Prints myNumber with 3 empty spaces to the
left of '100'
myNumber = 100;
fprintf('%6d',myNumber)
```

- *Precision* specifies the number of digits to the right of the decimal point

- Entered after *field width*

```
fprintf('%10.6f',someNumber)
```

-

```
% Prints myNumber with 1 empty space and 1 zero after the decimal point. The decimal point counts as a space.
myNumber = 100;
fprintf('%6.1f',myNumber)
```