



# Computer Aided Design (CAD)

## Lecture 1 Introduction

Dr. Sawsan Abdellatif

# Course Info

<b>Title</b>	<b>Computer Aided Design (CAD)</b>
<b>Lecturers</b>	Dr. Sawsan Abdellatif
<b>Lecturers webpages</b>	<a href="http://www.bu.edu.eg/staff/sawsanelsayed3">http://www.bu.edu.eg/staff/sawsanelsayed3</a>
<b>Email</b>	<a href="mailto:sawsan.abdellatif@feng.bu.edu.eg">sawsan.abdellatif@feng.bu.edu.eg</a>
<b>Teaching assistant (TA)</b>	---
<b>References</b>	Multiple references will be used
<b>Software packages</b>	Matlab/Simulink - Xilinx ISE
<b>Assessment (100)</b>	<ol style="list-style-type: none"><li>1. Final Term Exam (60)</li><li>2. Mid Term Exam</li><li>3. Assignments</li><li>4. Projects</li></ol>

# Course References

## First part: Matlab/Simulink

- 1) Matlab by Example: Programming Basics, Munther Gdeisat
- 2) Essential MATLAB® for Engineers and Scientists
- 3) Introduction to Simulink with Engineering Applications, Steven T. Karris

## Second part: design of digital circuits using Verilog

- 1) FPGA Prototyping by Verilog Examples (2008), Chu P. P.
- 2) Digital VLSI Systems Design: A Design Manual for Implementation of Projects on FPGAs and ASICs Using Verilog, Dr. S. Ramachandran

# Main Topics

- 1) **Matlab** as a software Environment for Modeling, Simulation, and Design.
- 2) **Simulink** as a software Environment for Modeling, Simulation, and Design.
- 3) Introduction on Programming of Field programmable gate arrays (FPGA) using hardware description languages (HDL) (e.g., VHDL/**Verilog**)

# CAD/CAE/CAM

- CAD is defined as the use of **Computer software** to help (**Aid**) the designer in the **Design** process (creation, modification, analysis, or optimization)
- CAEngineering (CAE) is a computer-based technique used to calculate the product's operational and functional parameters (**engineering analysis**).
- CAManufacturing (CAM) is a computer-based technique that is used to plan, manage, and control the **manufacturing** process.

# Design Process

## The design process

## Computer aided design

Recognition of need  
(customer order)

Problem definition  
(initial design)

Generation or  
synthesis

Analysis and  
optimization

Evaluation

Presentation

Geometric modelling

Engineering analysis  
and optimization

Design review and  
evaluation

Automated drafting

Production  
and  
marketing



# Design Process (cont'd)

## Geometric modelling

- Computer-compatible **mathematical description** of the geometry/functionality of an object.

## Engineering analysis

- Ex. Current and Voltage analysis, Static, Dynamic and Natural Frequency analysis, Heat transfer analysis, Fluid flow analysis, Motion analysis, Tolerance analysis, etc.

# Design Process (cont'd)

## Review and evaluation

- Checking the accuracy of the design



## Automated drafting

- Involves the creation of hard-copy engineering drawings directly from the CAD database.



# Potential Benefits of CAD

## ➤ Productivity Increase:

- Automation of repeated tasks
- Insert standard parts from database.

## ➤ Changeability:

- Don't have to redo entire drawing with each change.
- Keep track of previous design iterations.

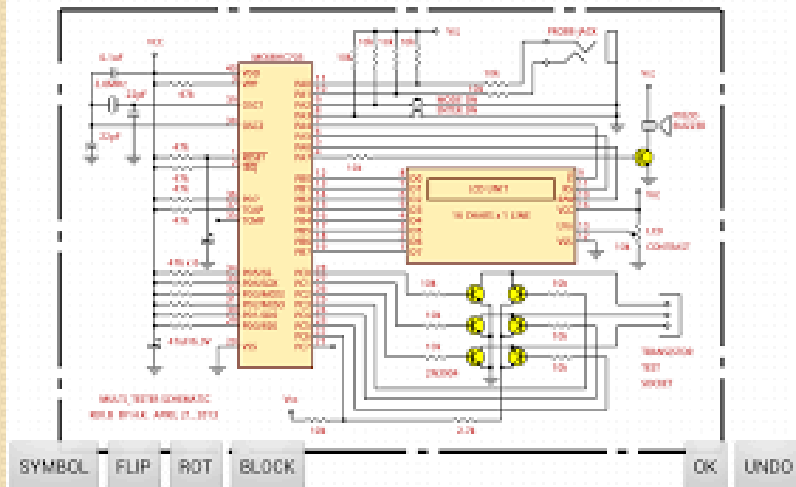
## ➤ Ease of Communication:

- With other teams/engineers, e.g. manufacturing, suppliers
- „With other applications (CAE, CAM)

## ➤ „Accurate, high quality drawings

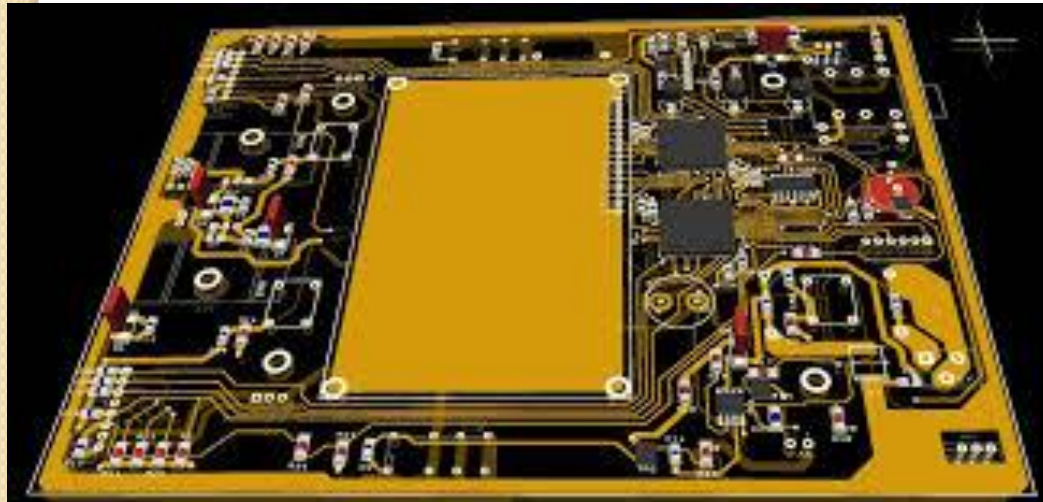
# Uses of CAD

- CAD is used in Several Fields e.g.,
  - Aerospace
  - Architecture
  - Automotive engineering
  - Machinery
  - Medical Design
  - Electronics & Communication



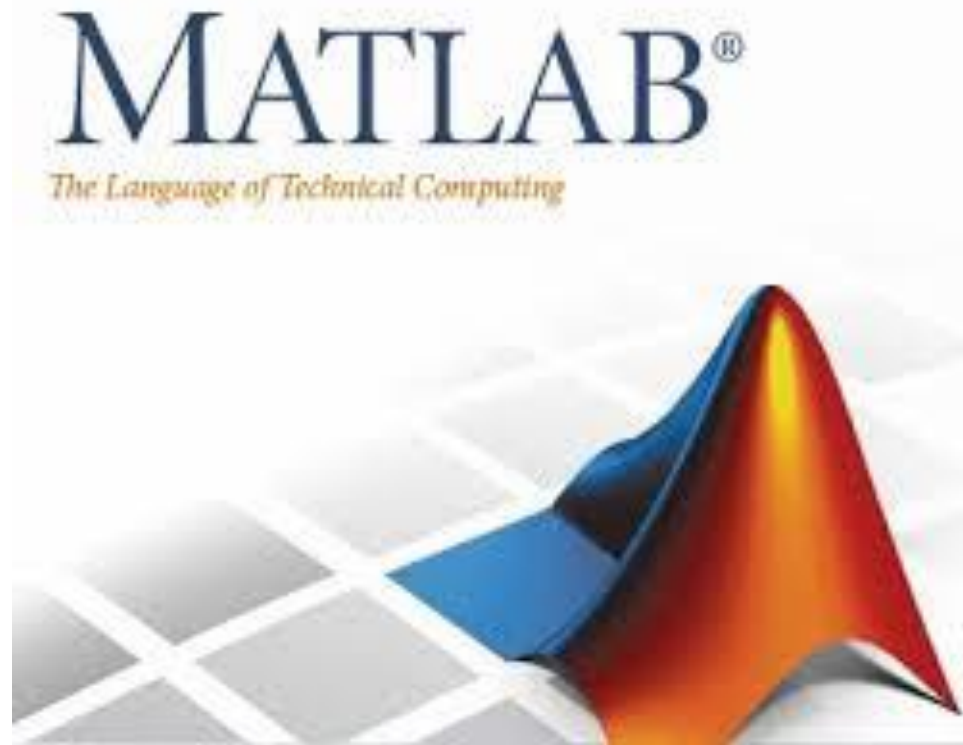
# Electronic design automation (EDA) or ECAD

- CAD can be used to design **electronic systems** such as printed circuit boards (PCBs) and integrated circuits (ICs).
- These designs can be used for thousands of electronic products and machinery.



# Design approaches

- **A top-down design:** proceeds from an **abstract**, high-level specification **to** a more and more **detailed** design by decomposition and successive refinement
- **A bottom-up design:** starts with **detailed** primitive blocks and combines them **into** larger and **more complex** functional blocks.
- Designs usually proceed from both directions simultaneously
  - Top-down design answers: **What are we building?**
  - Bottom-up design answers: **How do we build it?**



**Reference:**

**Matlab by Example: Programming Basics, Munther Gdeisat**

# Why Matlab?

- **Matrix Laboratory**
- Matlab is programming language and environment for **scientific computing** that is **centered on matrices**

## Common Uses of Matlab in Research

- Data Acquisition
- Multi-format data importing
- Analysis tools
- Statistics
- Graphing
- Modeling

# Matlab Environment

The image shows the MATLAB R2013a interface. The title bar reads "MATLAB R2013a". The ribbon includes tabs for HOME, PLOTS, and APPS. The HOME tab is active, showing options like New Script, New, Open, Compare, Import Data, Save Workspace, New Variable, Open Variable, and Clear Workspace. The current folder is "D:\Matlab\_CAD\_Work".

The Command Window is open and shows the prompt `>>`. The Workspace window is empty. The Command History window shows the following commands:

```
diff1=diff(y)
diff2=diff(di
diff2=diff(y,
sym f
x=f
```

The status bar at the bottom left shows "file1.m (MATLA)" and "Ready".

# Ways of writing Matlab Code

## 1) The Command Window

- The command window allows you to type commands directly and see the results immediately.

Ex:

```
>> a=[1 2]
```

- the output of the command

```
a =
```

```
1 2
```

## 2) The Editor Window (m-file)

- The Editor Window is a word processor specifically designed for Matlab commands.

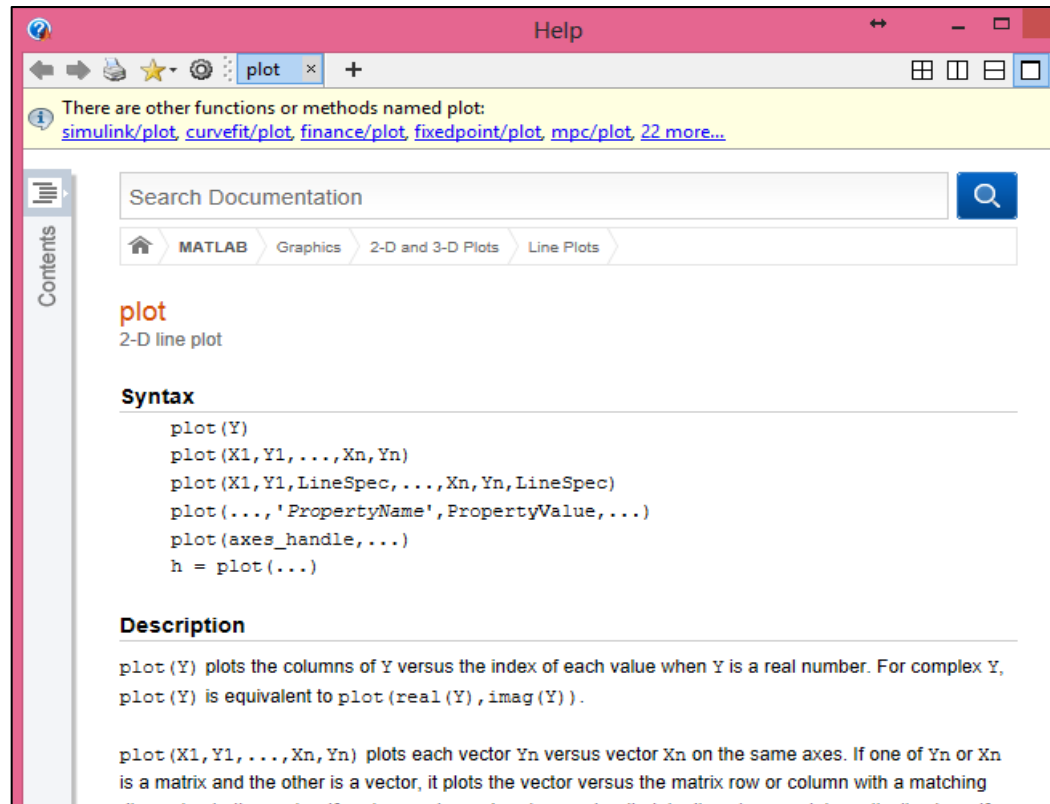


# Getting Help in Matlab

- The help is built into Matlab and can be accessed from the help menu.
- To get help on a command, type "doc commandname" in Command Window where *commandname* is the command of interest.

Ex:

```
>> doc plot
```



The screenshot shows the MATLAB Help window for the 'plot' function. The window title is 'Help' and the browser tab is 'plot'. A yellow banner at the top states: 'There are other functions or methods named plot: [simulink/plot](#), [curvefit/plot](#), [finance/plot](#), [fixedpoint/plot](#), [mpc/plot](#), [22 more...](#)'. Below this is a search bar labeled 'Search Documentation' and a breadcrumb trail: 'MATLAB > Graphics > 2-D and 3-D Plots > Line Plots'. The main content area is titled 'plot' and '2-D line plot'. Under the 'Syntax' section, the following code is listed:

```
plot(Y)
plot(X1,Y1,...,Xn,Yn)
plot(X1,Y1,LineStyle,...,Xn,Yn,LineStyle)
plot(...,'PropertyName',PropertyValue,...)
plot(axes_handle,...)
h = plot(...)
```

Under the 'Description' section, the text reads: 'plot(Y) plots the columns of Y versus the index of each value when Y is a real number. For complex Y, plot(Y) is equivalent to plot(real(Y), imag(Y)).' and 'plot(X1,Y1,...,Xn,Yn) plots each vector Yn versus vector Xn on the same axes. If one of Yn or Xn is a matrix and the other is a vector, it plots the vector versus the matrix row or column with a matching dimension to the vector. If Xn is a scalar and Yn is a vector, it plots discrete points vertically at Xn. If Yn is a scalar and Xn is a vector, it plots discrete points horizontally at Yn. If both Xn and Yn are scalars, it plots a single point at (Xn, Yn).'

# Creating Scalar Variables

- Matlab is a short name for Matrix laboratory.
- So, Matlab is a matrix-based software package. It considers the scalar variable to be a 1X1 matrix.
- A scalar here means a number such as “2” or “ - 100”

```
>> x = 1;
```

Command prompt

Scalar variable

Semicolon “;”

is used to direct Matlab not to display the value of the variable x in the Command Window.

# Creating Scalar Variables (cont'd)

Note the changes that happened in the Command Window, the Command History, and the Workspace windows.

The screenshot displays the MATLAB 7.12.0 (R2011a) interface. The Command History window shows the command `x=1;` executed on 12/06/2011 at 02:56. The Current Folder window shows the current folder is `C:\Matlab Programs`. The Workspace window shows a table with one variable `x` having a value of 1. The Command Window shows the command `>> x=1;` being entered.

Name	Value	Min	Max
x	1	1	1

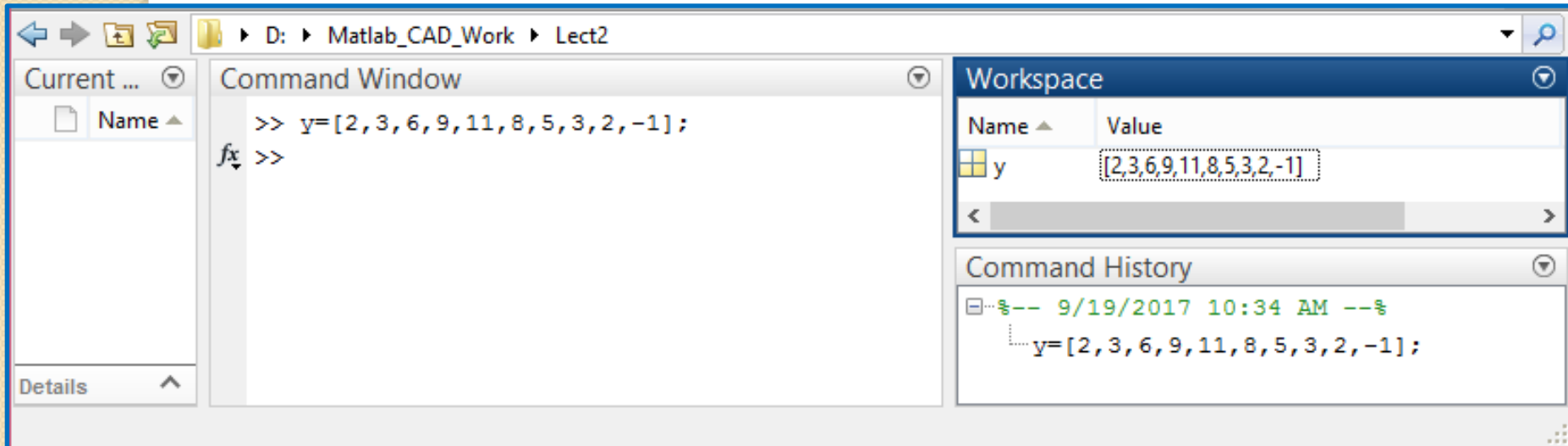
# Creating Vector Variables

## 1- Row vector

Type the Matlab Command:

```
>> y = [2,3,6,9,11,8,5,3,2,-1];
```

This creates **Row** vector with the values indicated in the command



The screenshot displays the MATLAB interface with the following components:

- Command Window:** Shows the command `>> y=[2,3,6,9,11,8,5,3,2,-1];` and the prompt `fx >>`.
- Workspace:** A table showing the variable `y` with the value `[2,3,6,9,11,8,5,3,2,-1]`.
- Command History:** Shows the command `y=[2,3,6,9,11,8,5,3,2,-1];` executed on 9/19/2017 at 10:34 AM.

Name	Value
y	[2,3,6,9,11,8,5,3,2,-1]

You can draw the vector `y` if you Right Click on the `y` variable in the Workspace and press `plot(y)`. (Check the results)

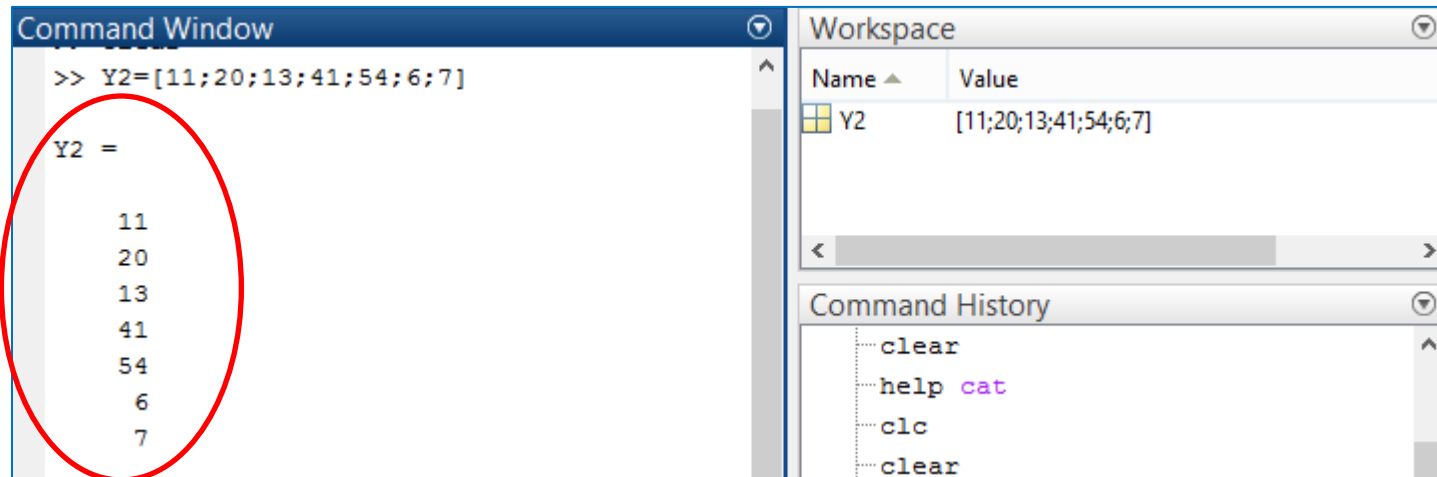
# Creating Vector Variables

## 2- Column vector

Type the Matlab Command:

```
>> Y2=[11;20;13;41;54;6;7]
```

This creates **Column** vector with the values indicated in the command



The screenshot shows the MATLAB Command Window and Workspace. In the Command Window, the command `>> Y2=[11;20;13;41;54;6;7]` is entered, and the output is a column vector `Y2 =` with values 11, 20, 13, 41, 54, 6, and 7. The output is circled in red. The Workspace window shows a variable `Y2` with the value `[11;20;13;41;54;6;7]`. The Command History window shows the command `clear`.

Note using **space** or **comma** “,” for creating **Row vector** and using **semicolon** “;” for creating **Column vector**

# Creating Array Variables

Type a Matlab Command:

```
>> Z = [1,2;3,4]
```

This creates an array variable with the following values:

The screenshot displays the MATLAB interface with three main windows:

- Command Window:** Shows the command `>> Z=[1,2;3,4]` and the resulting array:

```
Z =  
  
     1     2  
     3     4
```
- Workspace:** Shows a table with the following data:

Name	Value
Z	[1,2;3,4]
- Command History:** Shows the sequence of commands: `clear`, `clc`, and `Z=[1,2;3,4]`.

Note: “,” or space passes its next value to a new column **But** “;” passes its next value to a new row

Right Click on the Z variable in the Workspace and press mesh(Z).

(Check the results)

# Matlab Script Files

## Creating Script Files

- An M-file is a text file that contains a collection of commands that Matlab executes in a sequential order.
- A script file has the following **properties**:
  - It has no arguments (input data) and it does not return any values (outputs).
  - The commands executed in the script file have the same effect as if these commands were executed in the Command Window.
  - The variables created by the script file are displayed in the Workspace window.

Create a script file that contain the following commands:

```
x = 1;  
y = 2;
```

# Matlab Script Files

## Naming Script Files

➤ The following rules must be taken into consideration when a script file is named:

- The file name must **not** contain spaces or hyphens (-).
- The file name **must** start with an alphabetical character (a–z or A–Z).
- The file name **must** contain only alphabetical characters (a–z or A–Z), numbers (1–9) or underscores ( \_ ).
- Punctuation characters such as commas ( , ) or apostrophes ( ' ) are not allowed, because many of them have special meanings in Matlab.
- The file name must be neither a **Matlab variable** nor an existing **Matlab function**.
- The use of a **Matlab reserved word** as a file name is not allowed.

Remember: It is very helpful to use meaningful and descriptive names



# Matlab Script Files

## Naming Script Files

Remember Matlab Keywords and reserved words are not allowed to be used as file names

### Matlab reserved words examples

'name'	'across_variable'
'node'	'build'
'output'	'description'
'parameter'	'descriptor'
'setup'	'element'
'signal'	'input'
'source'	'interface_input'
'terminal'	'interface_node'
'through_variable'	'interface_output'
'variable'	'item_type'
	'local_variable'

### Matlab Keywords examples

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

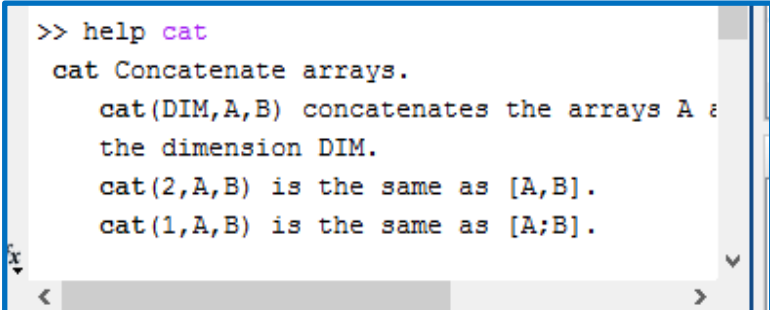
# Matlab Script Files

## Naming Script Files

- To check that the file name you have chosen is not a Matlab keyword or a Matlab function, you can use Matlab help.
- Example: If you want to name your file “cat.m”, write the following command on the command window:

```
>> help cat
```

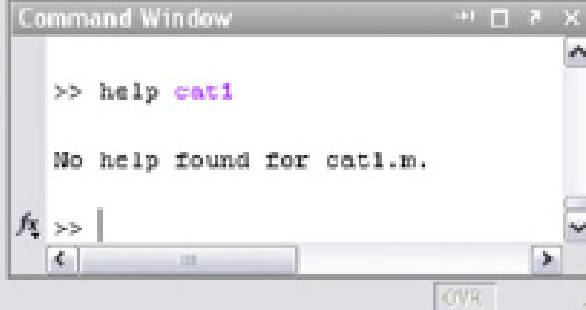
Matlab responds and informs you that there is already a function called “cat” that concatenates arrays



```
>> help cat
cat Concatenate arrays.
   cat(DIM,A,B) concatenates the arrays A and B along
   the dimension DIM.
   cat(2,A,B) is the same as [A,B].
   cat(1,A,B) is the same as [A;B].
```

- You can change the name to “cat1.m”,

```
>> help cat1
```



```
Command Window

>> help cat1

No help found for cat1.m.
```

# Matlab Script Files

## Commenting Matlab Code

- You can add a comment to Matlab code by inserting a percentage sign “%” at the beginning of the line. For example:

```
% Chris bought three rulers, two rubbers, and four books.  
% The price of a ruler is £6. The price of a rubber is £8.  
% The price for a book is £25.  
% This Matlab program calculates the total price paid by  
% Chris.
```

- A better method for commenting multiple lines of the code is performed by using **block commenting**.
- In this method, add the characters “%{” before the first line of the comments and add the characters “%}” after the last line of the comments.

```
%{  
Chris bought three rulers, two rubbers, and four books.  
The price of a ruler is £6. The price of a rubber is £8.  
The price for a book is £25. This Matlab program calculates the total  
price paid by Chris.  
%}
```

# Matlab Editor-Cell Mode

## Enabling Cell-mode (Section mode in Matlab 2013)

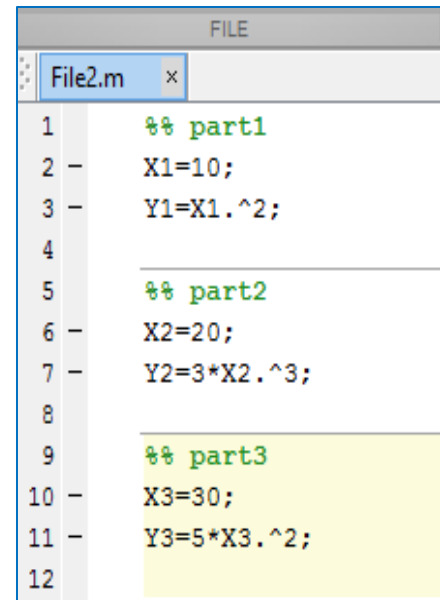
Code with no section mode

```
clear; clc;
%part 1
x1 = 10
y1 = x1.^2
%part 2
x2 = 20
y2 = 3*x2.^3
%part 3
x3 = 30
y3 = 5*x3.^2
```

Section mode enable you to divide your code into some sections that can be executed individually

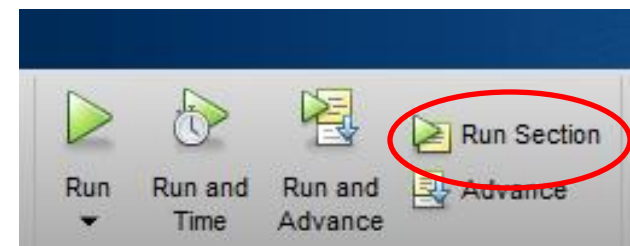
Two ways to create new section:

- Right Click in the m-file and press “Insert Section”.
- Type “%%” and space.



```
FILE
File2.m x
1 %% part1
2 - X1=10;
3 - Y1=X1.^2;
4
5 %% part2
6 - X2=20;
7 - Y2=3*X2.^3;
8
9 %% part3
10 - X3=30;
11 - Y3=5*X3.^2;
12
```

To run certain section, select this section in the m-file and press “Run Section”



# Required Tasks

1- Creating **Scalar** Variable

```
>> x = 1;
```

2- Creating **Vector** Variable

Row vector `>> y = [2, 3, 6, 9, 11, 8, 5, 3, 2, -1];`

Column vector `>> Y2 = [11; 20; 13; 41; 54; 6; 7]`

3- Creating **Array** Variable

```
>> Z = [1, 2; 3, 4]
```

4- Creating **Script file** that creates two vectors and add them



**Thanks for attention**