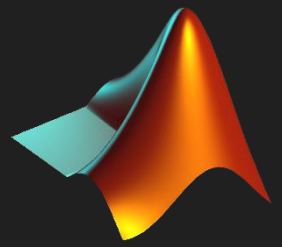


MATLAB

A Tool for Scientific Computing

Dr. Mahesh Goyani

Assistant Professor

Department of Computer Engineering

Government Engineering College, Modasa

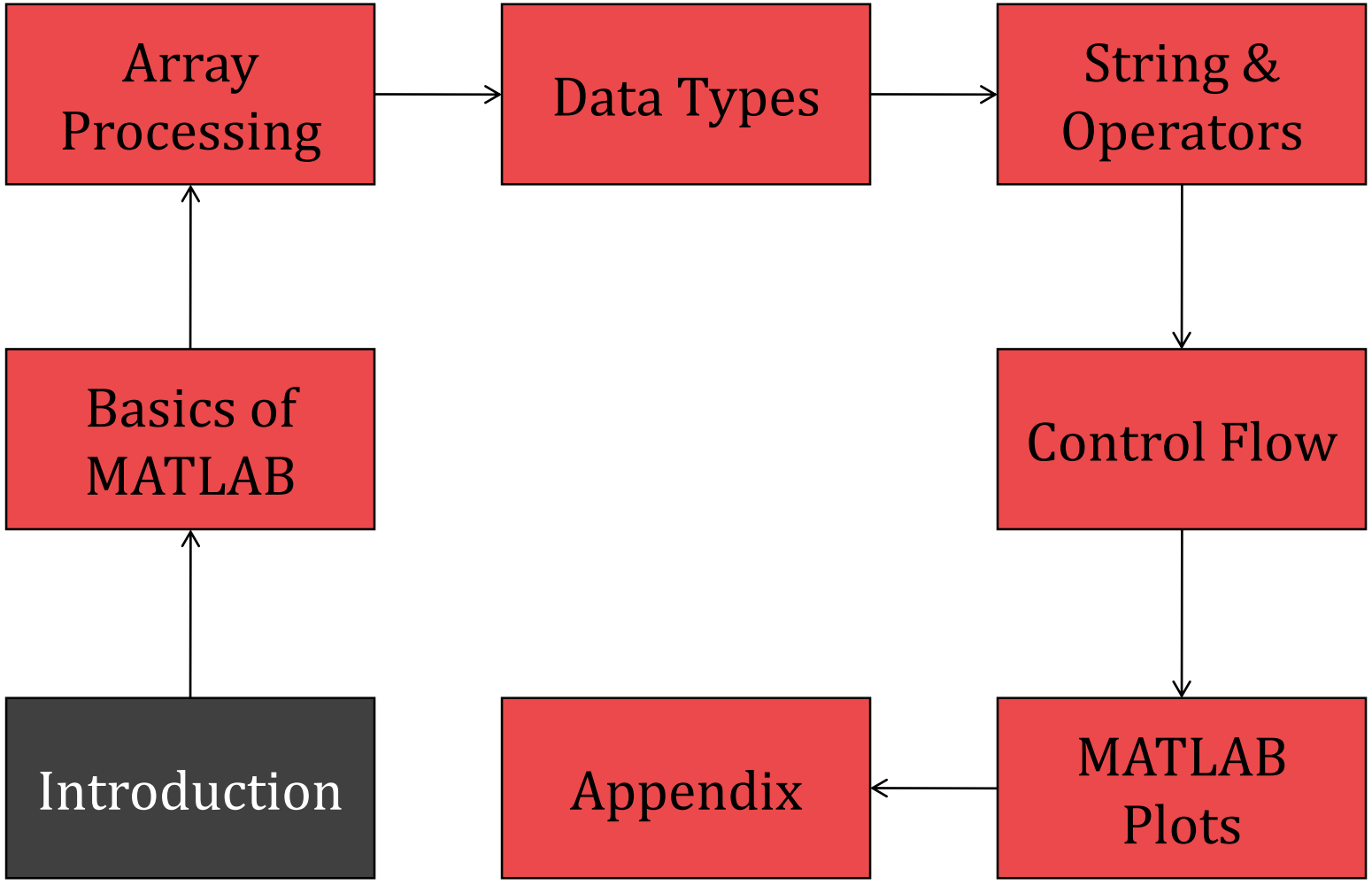
E-Mail: mgoyani@gmail.com

URL: www.maheshgoyani.in

PREREQUISITE

- Basics of programming language
- Coding skill in any of the following languages:
  - C
  - C++
  - JAVA
- And…
  - *Obviously* A Good Logic !! ☺

Operational Pipeline



MATLAB

- MATLAB stands for **MAT**rix **LAB**oratory
- Developed by Cleve Moler in 1970's to help students to learn linear algebra.
- MathWorks Inc company founded in 1984 for commercial development
- MATLAB is interpreted – code based system
- Basic data element of MATLAB is an *Array / Matrix* that does not require dimensioning.
- Add-on toolboxes for extra power.

SIMULINK

- Used to model, analyze and simulate dynamic systems using block diagrams.
- Fully integrated with MATLAB , easy and fast to learn and flexible.
- It has comprehensive block library which can be used to simulate linear, non–linear or discrete systems – excellent research tools.
- C codes can be generated from Simulink models for embedded applications and rapid prototyping of control systems.

TOOLBOXES

- Collections of functions to solve problems from several application fields.

- Digital Signal Processing Toolbox
- Image Toolbox
- Wavelet Toolbox
- Neural Network Toolbox

- Fuzzy Logic Toolbox
- Control Toolbox
- Multibody Simulation Toolbox
- And many others…

WHY MATLAB?

- Add-on toolboxes (60+ Toolboxes) for extra power.
- It frees you from coding in high-level languages
- Extensive HELP documentation
- Provides professional looking graphs
- Easy vector and matrix manipulation
- *Implicit loops for arrays*
- MATLAB M-files are completely portable across a wide range of platforms.

Pros & Cons

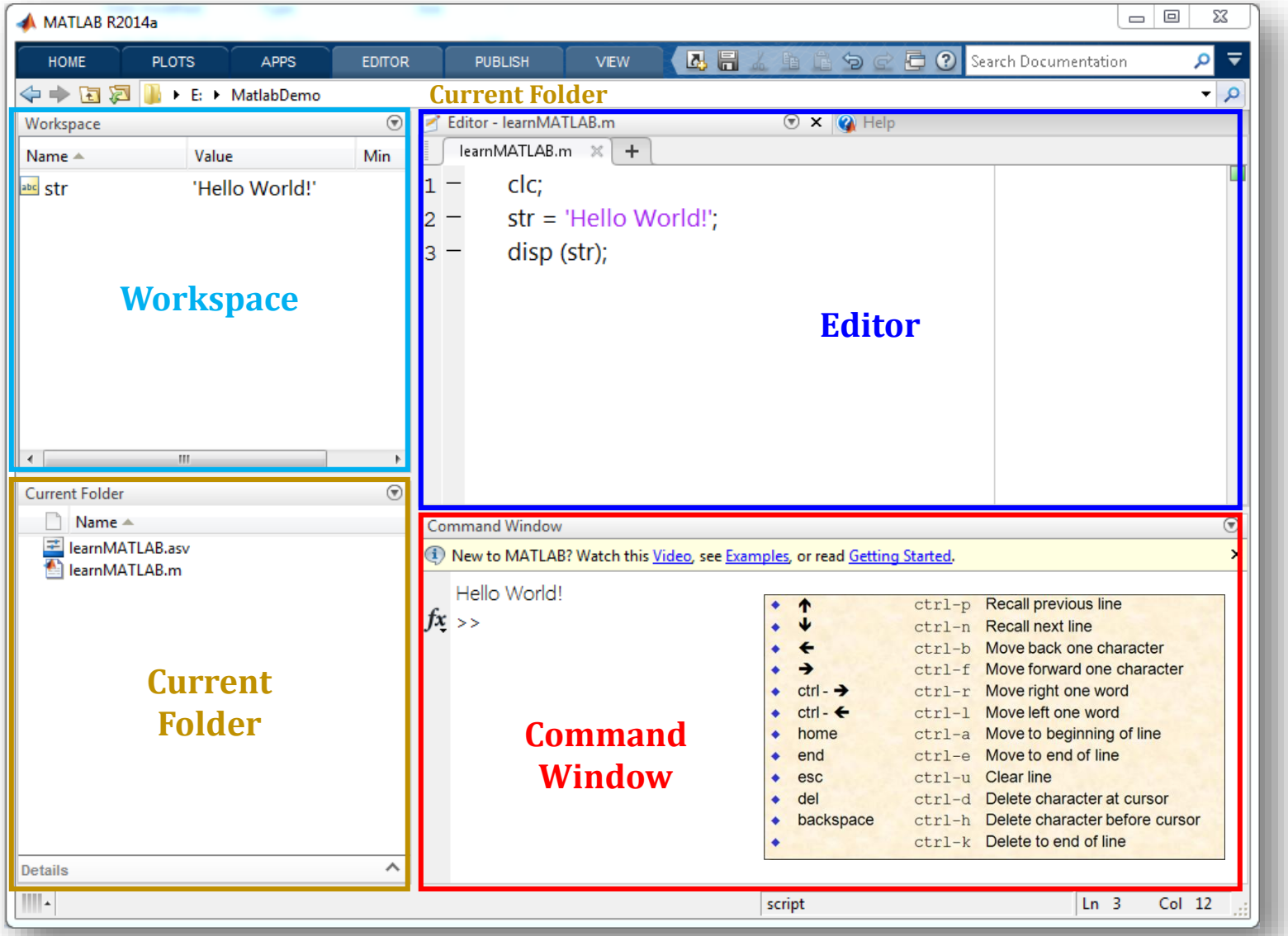
Advantages

- Ease of use
- Platform independent
- Predefined functions
- Parallelism
- Professional Plotting

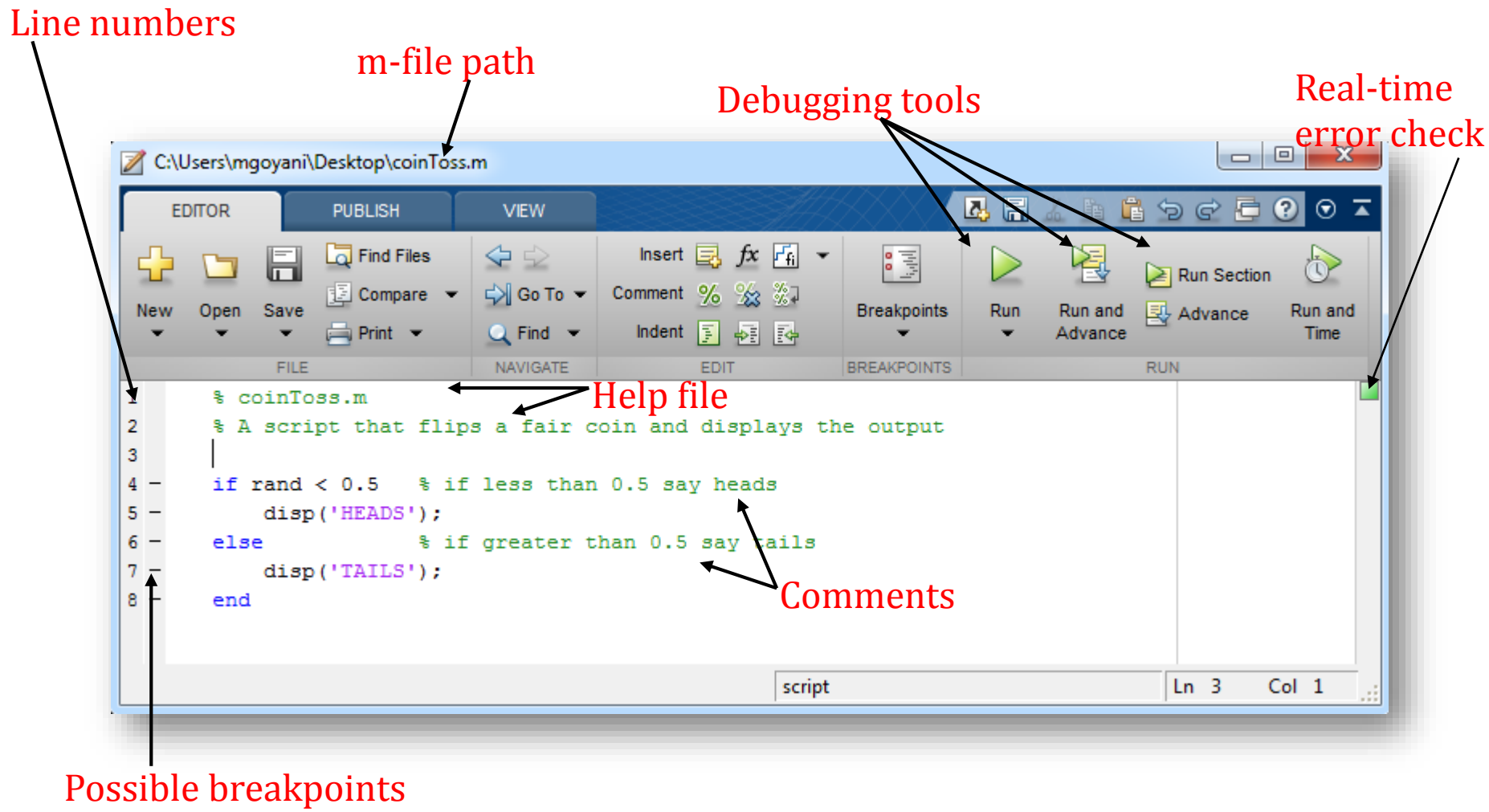
Disadvantages

- Can be slow
- Not geared to the Web
- Expensive
- Not designed for large scale system development

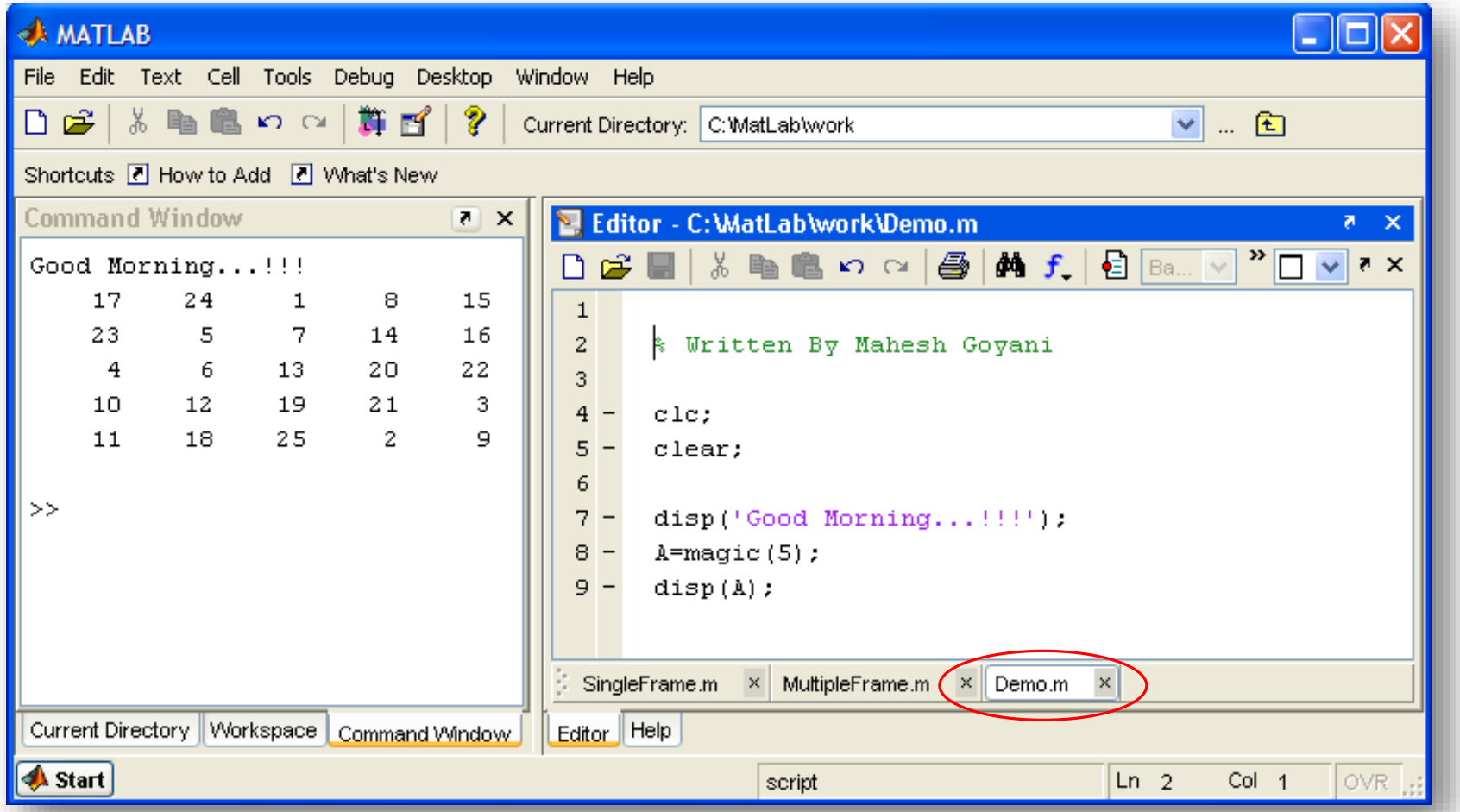
MATLAB Interface



EDITOR



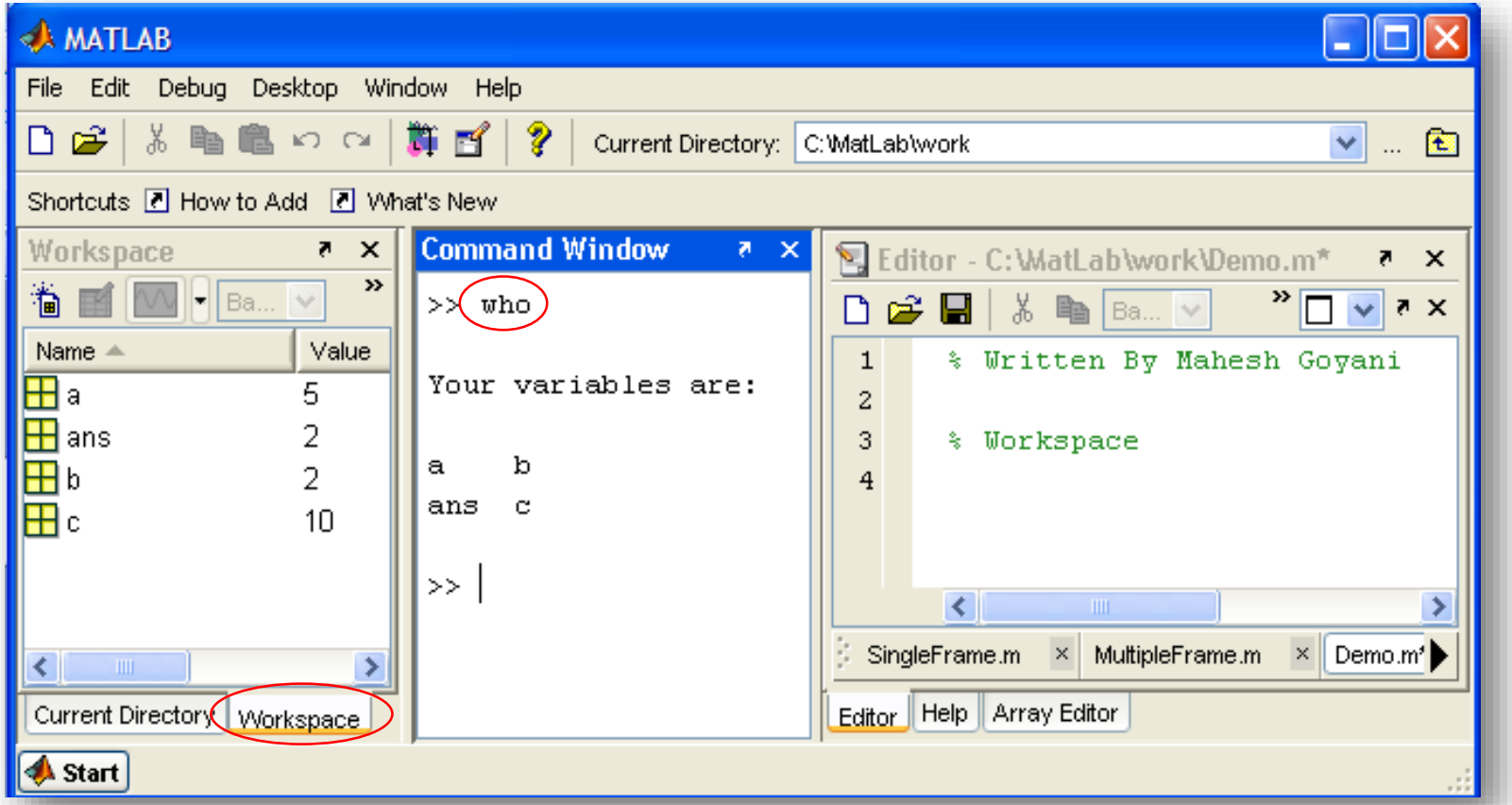
Command Window



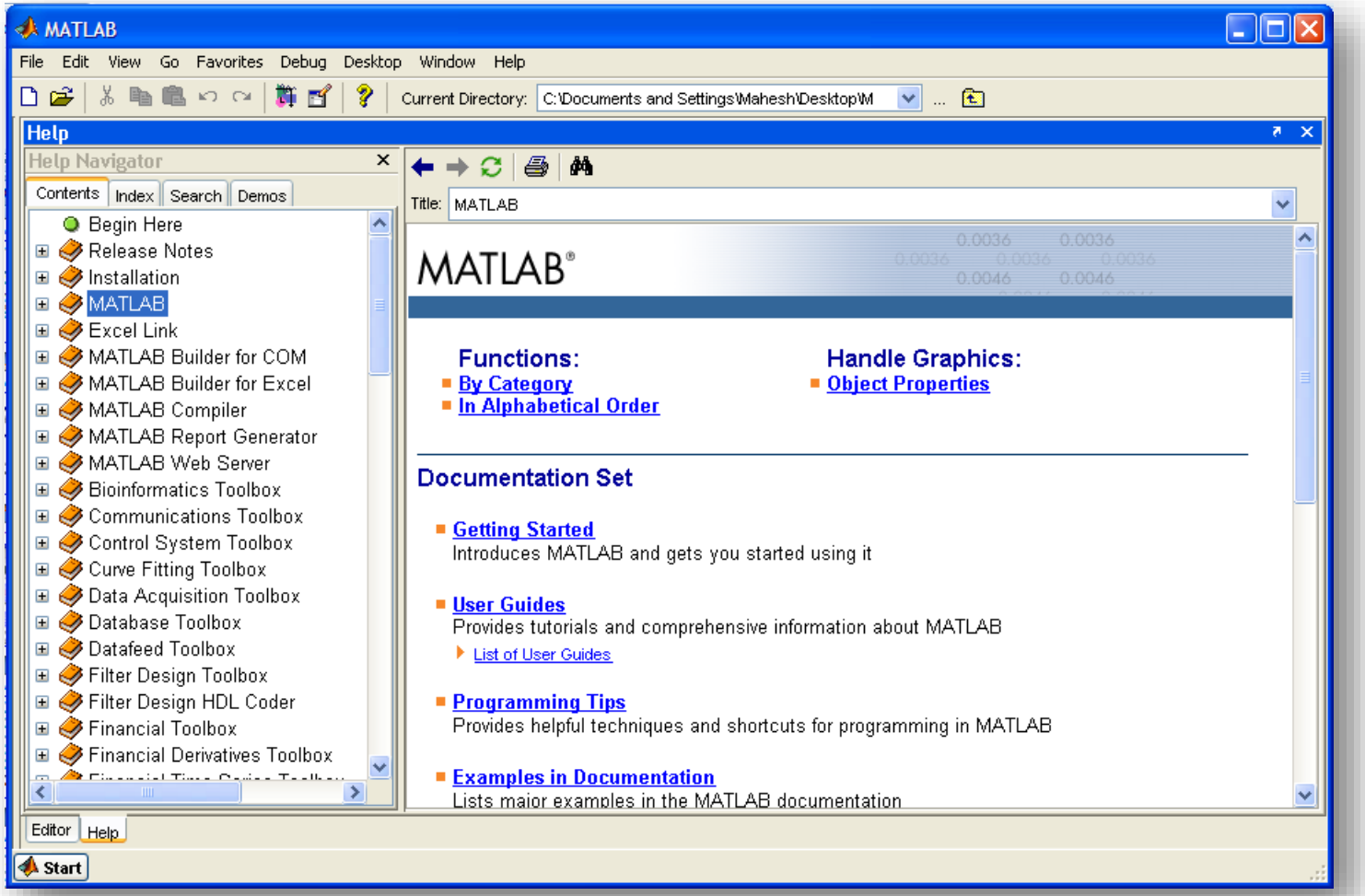
Press **F5** to Run

WORKSPACE

- Variables are saved in workspace
- Command *who* retrieves the variables available in workspace

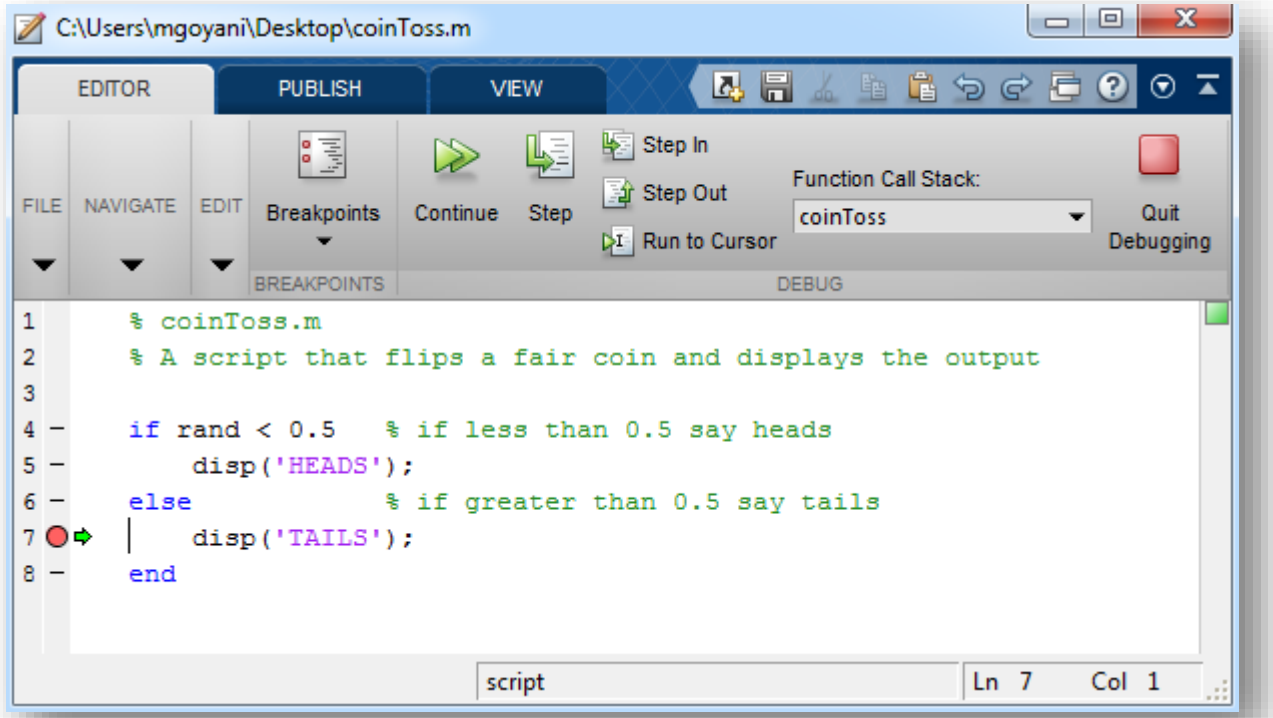


HELP

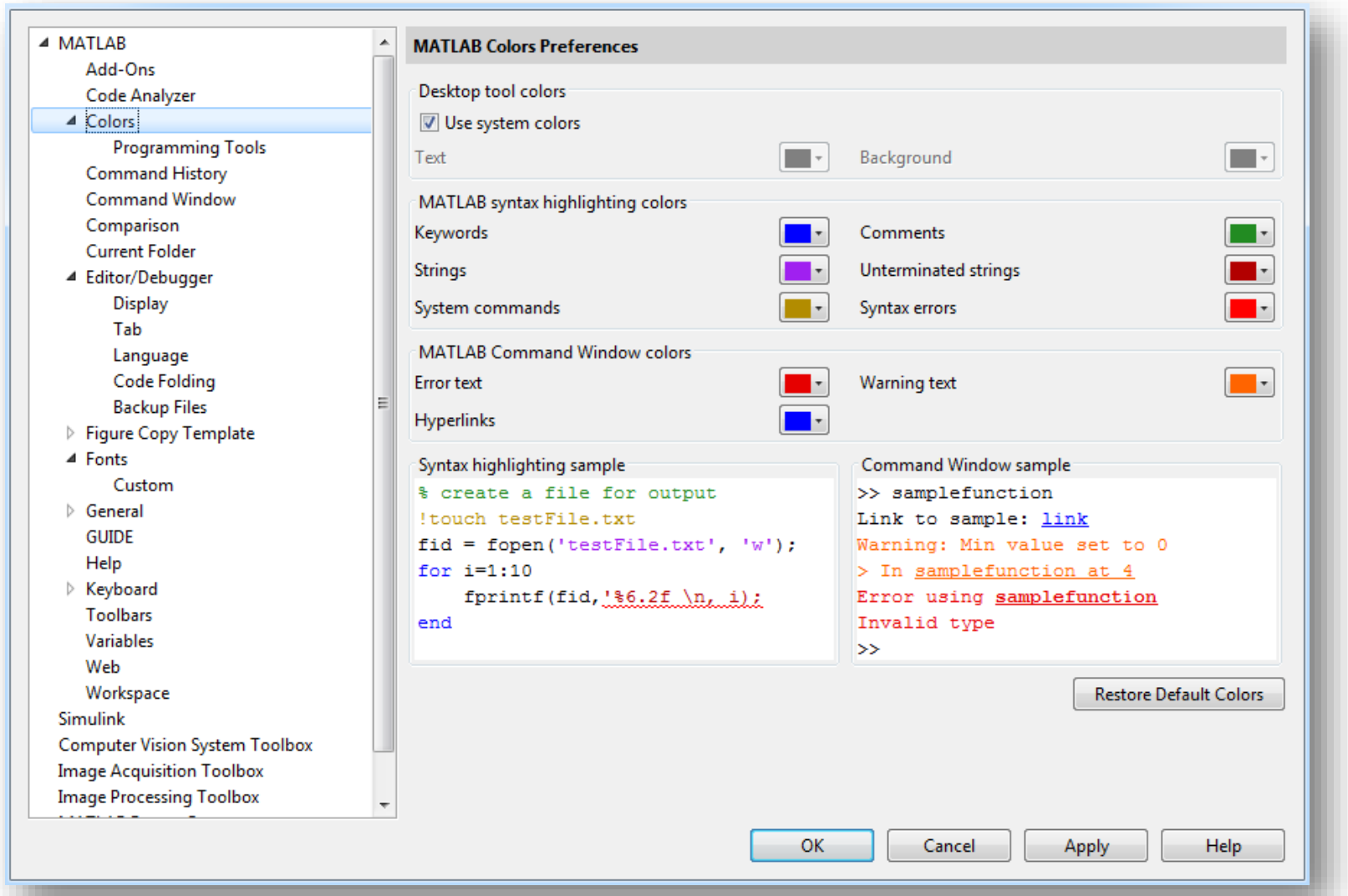


- help command (>>help)
- lookfor command (>>lookfor)
- Help Browser (>>doc)
- helpwin command (>>helpwin)
- Search Engine
- Printable Documents
- "Matlabroot\help\pdf_doc\"
- Link to The MathWorks

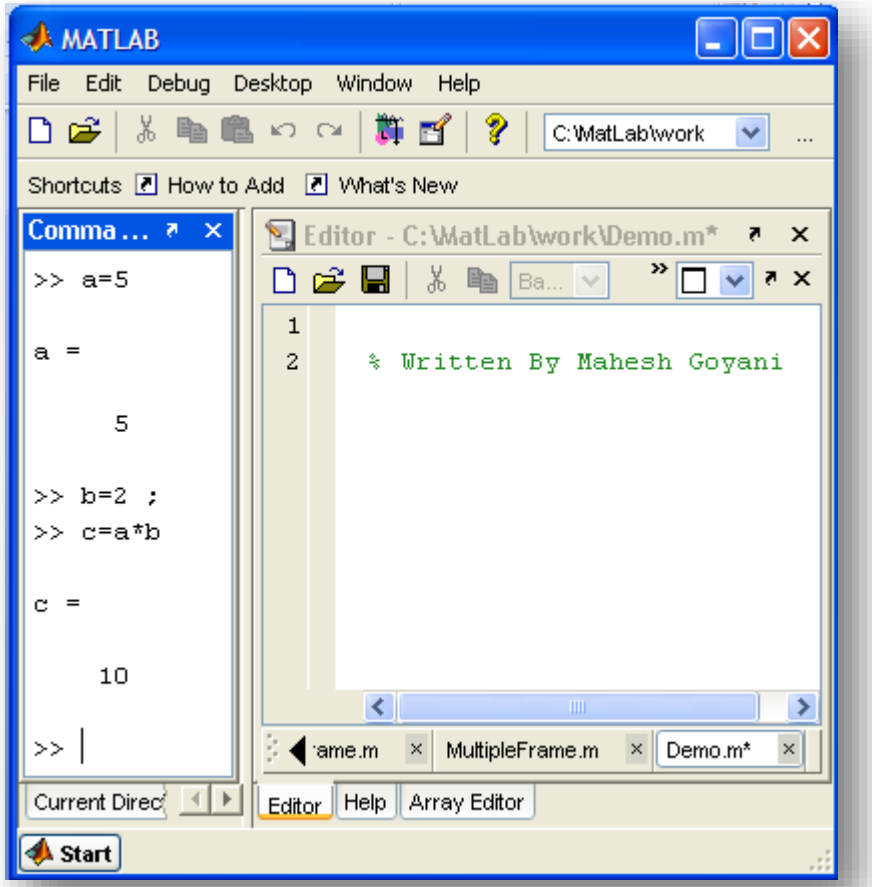
Visual Debugging

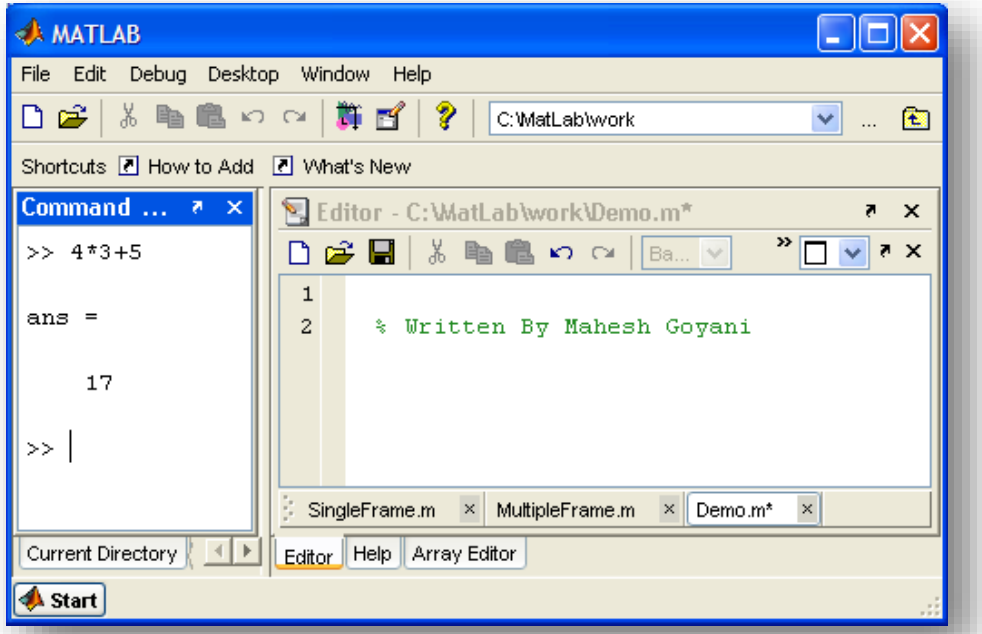


CUSTOMIZATION

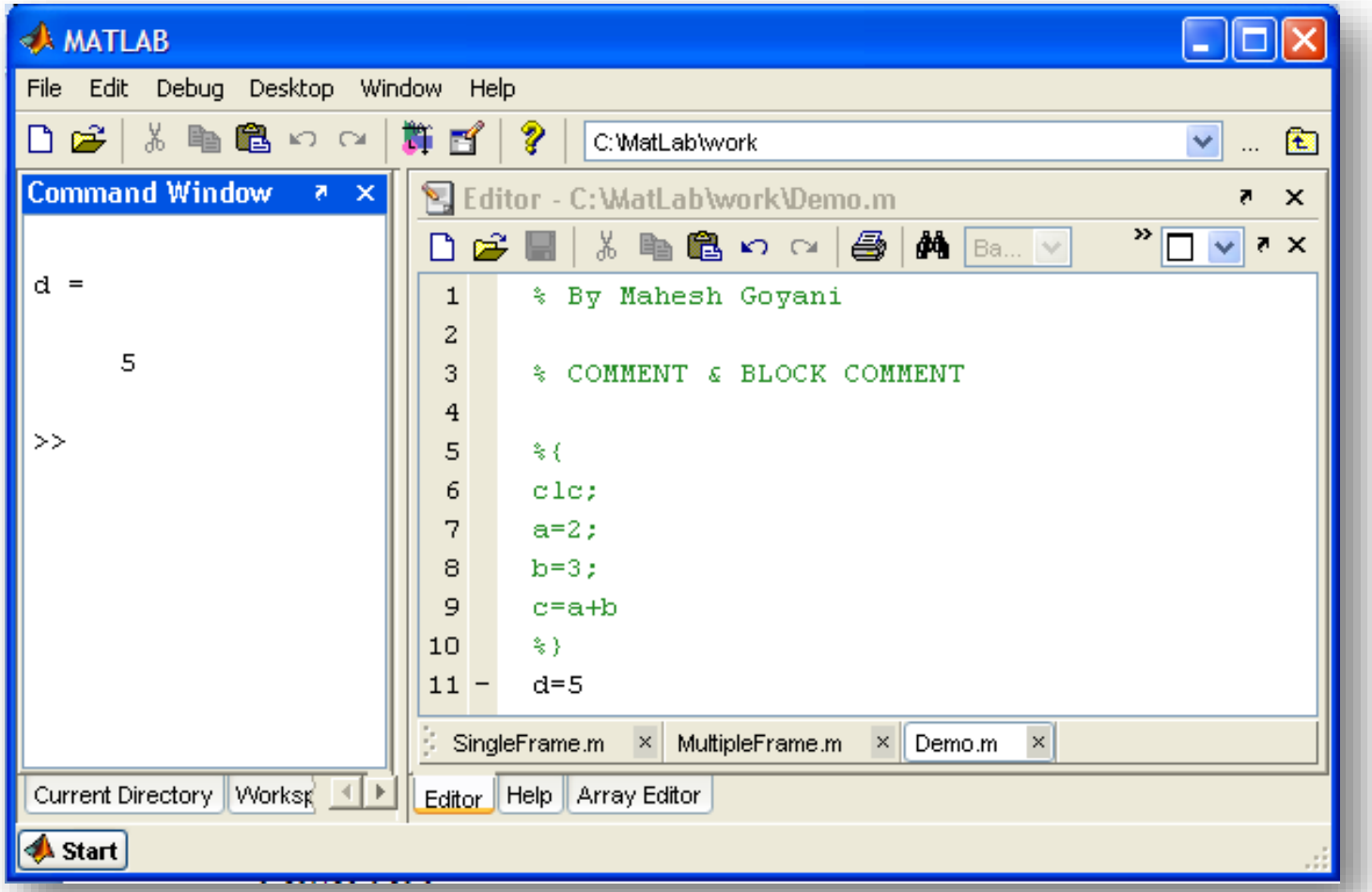


Basic Operation





COMMENTS

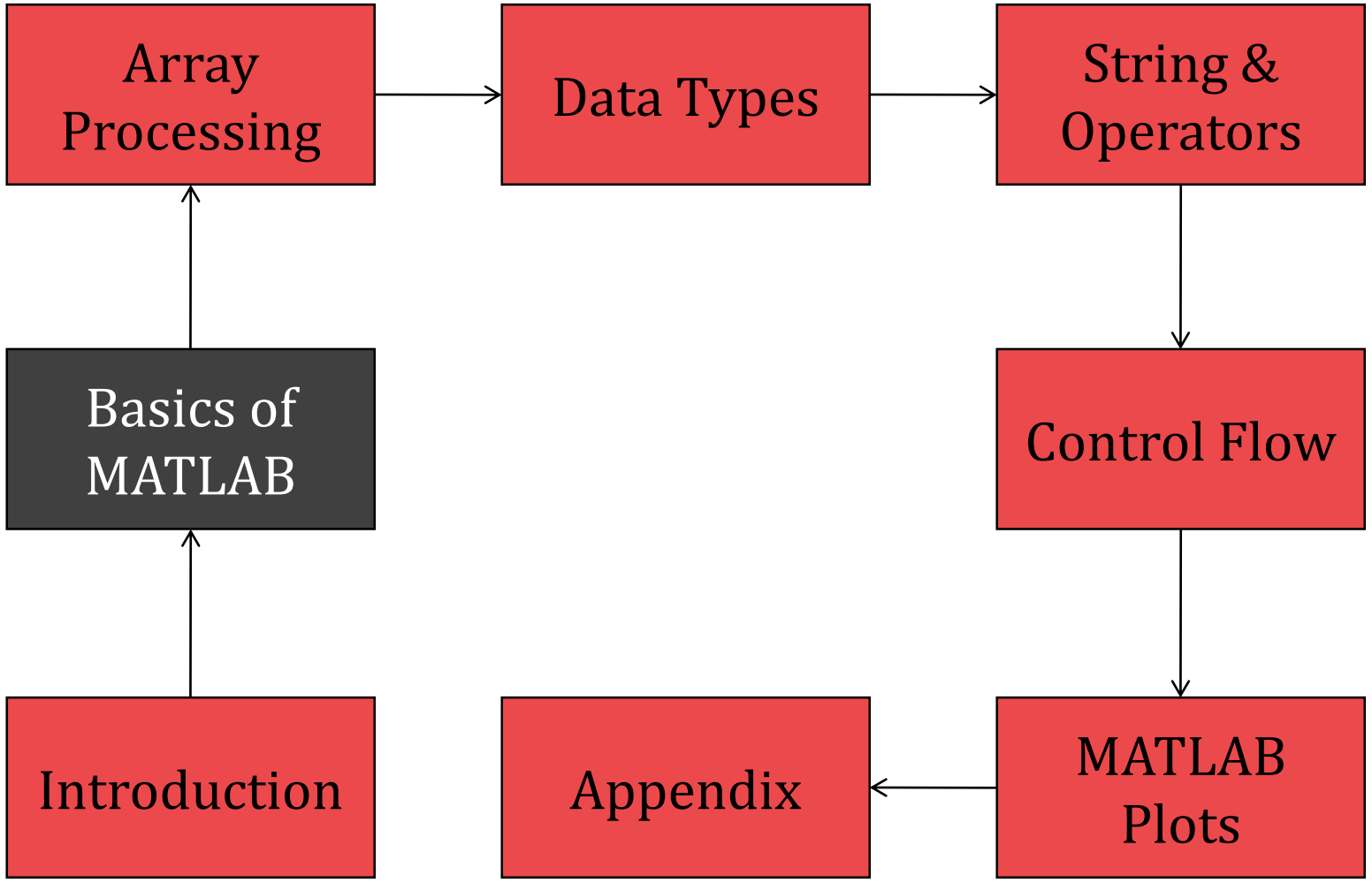


Managing Workspace

- MATLAB allocates contiguous memory to variables.

  - clc
  - clear a b c*
  - who
  - whos
  - save
  - pack

Operational Pipeline

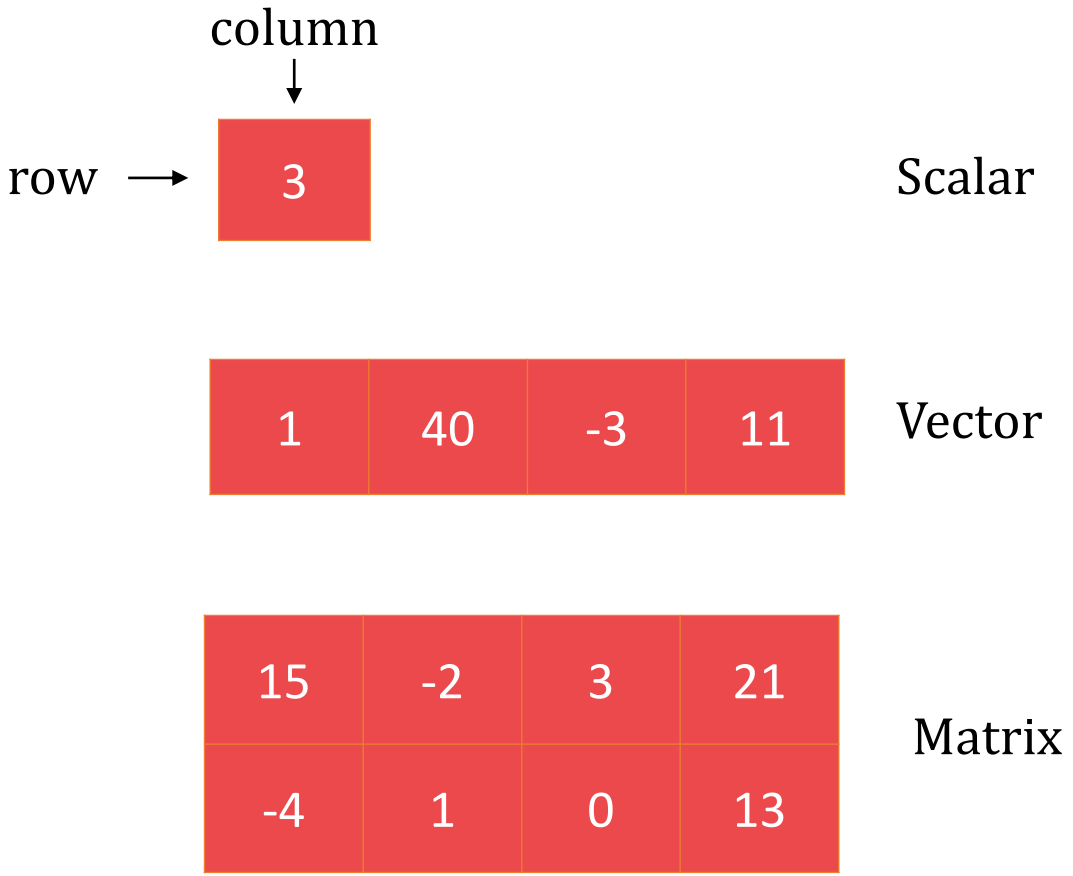


Rules to Declare Variables

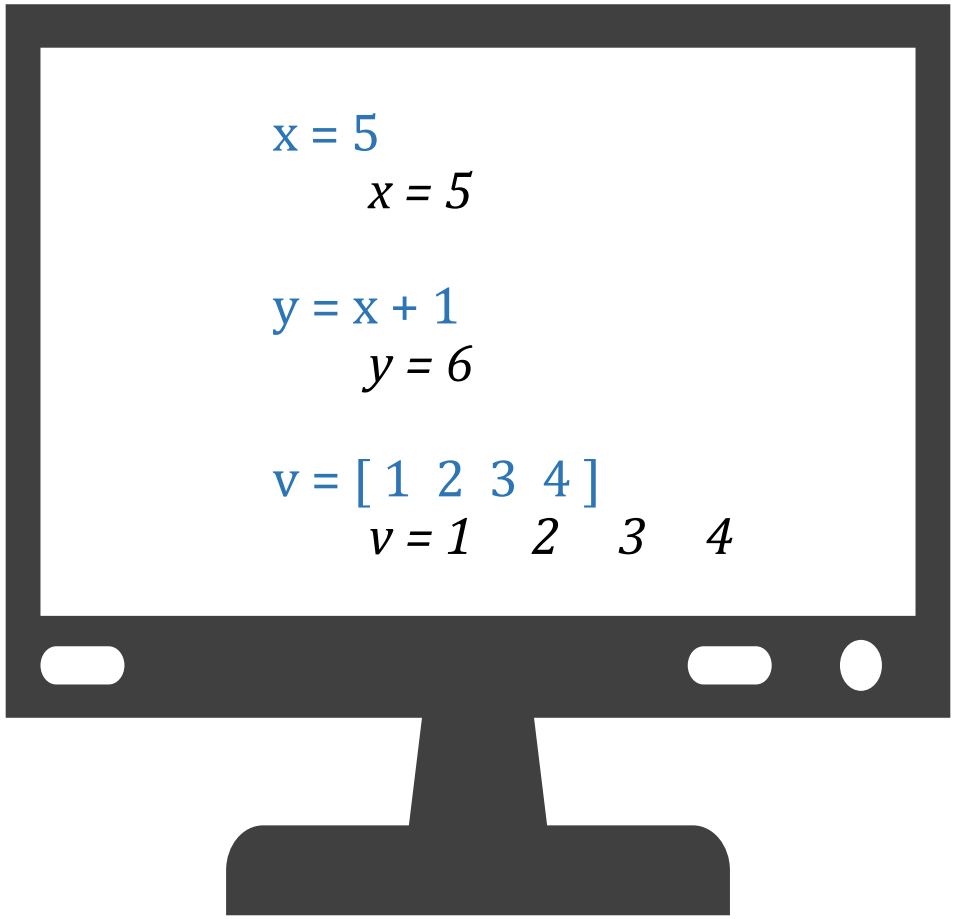
- Loosely Typed : Declaration is not required
- Case Sensitive
- Maximum Length : 63 Characters
- Must Start With Character
- Special characters are not allowed
- Special Variables are not allowed
- Keywords are not allowed

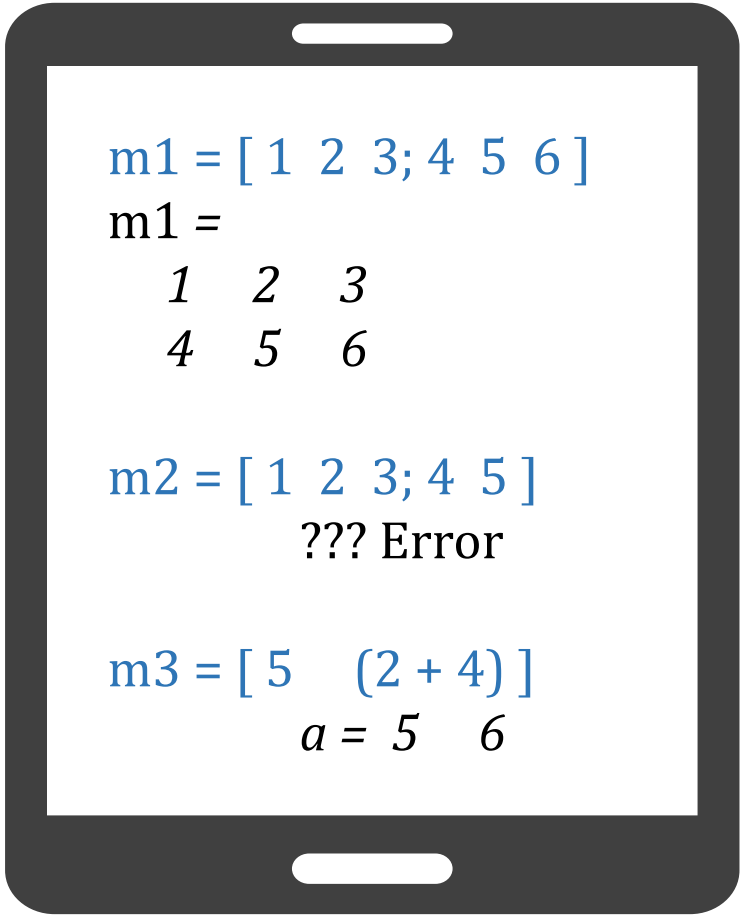
Fundamental Units

- The fundamental unit of data is an **array**



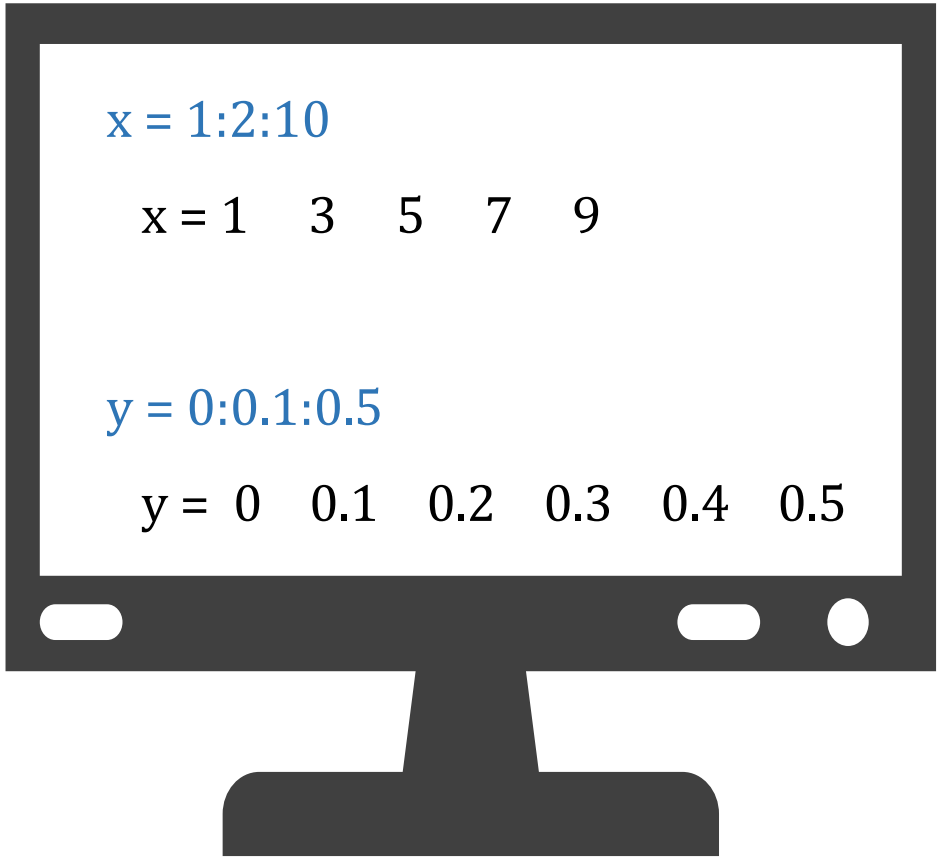
Initialization using Assignment



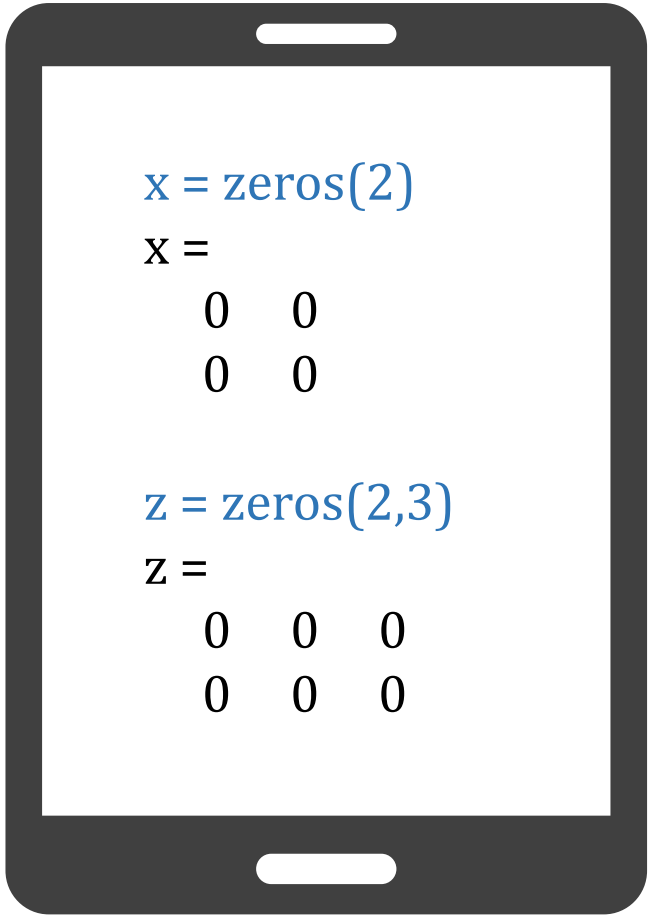


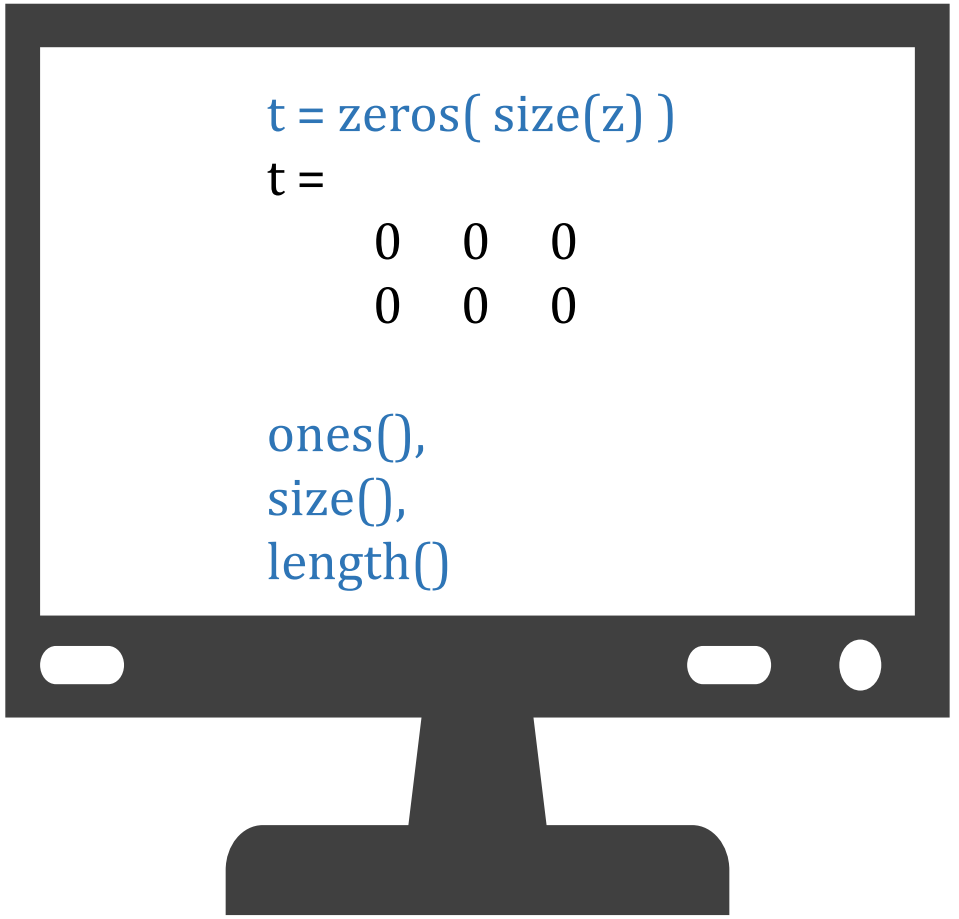
Initialization using Shortcut Statement

- Colon Operator → First : Increment : Last

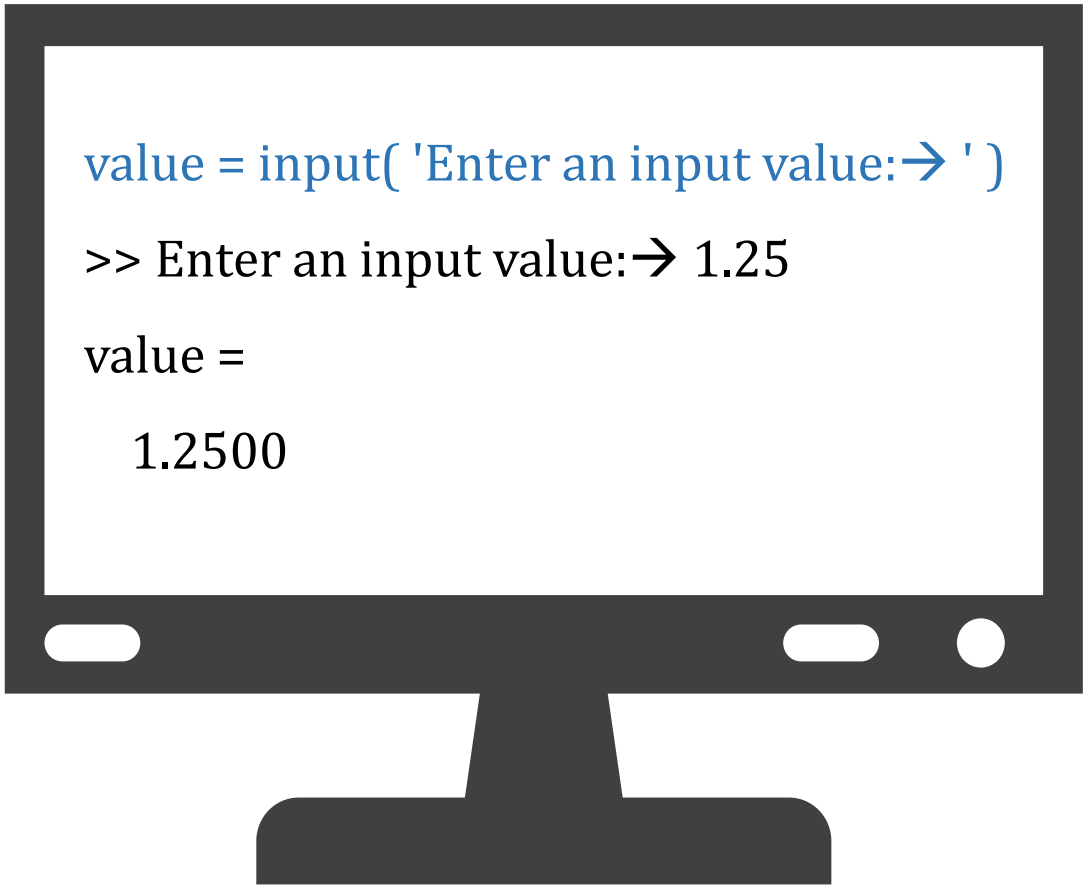


Initialization using Built in Functions





Initialization using User Input

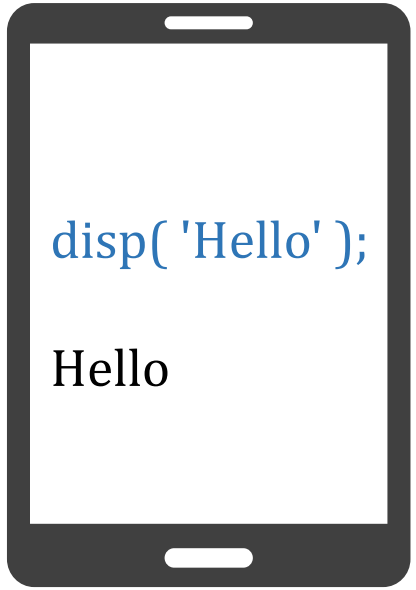


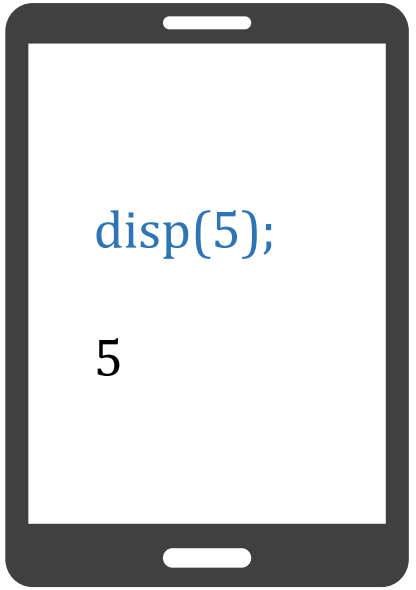
Number Formats

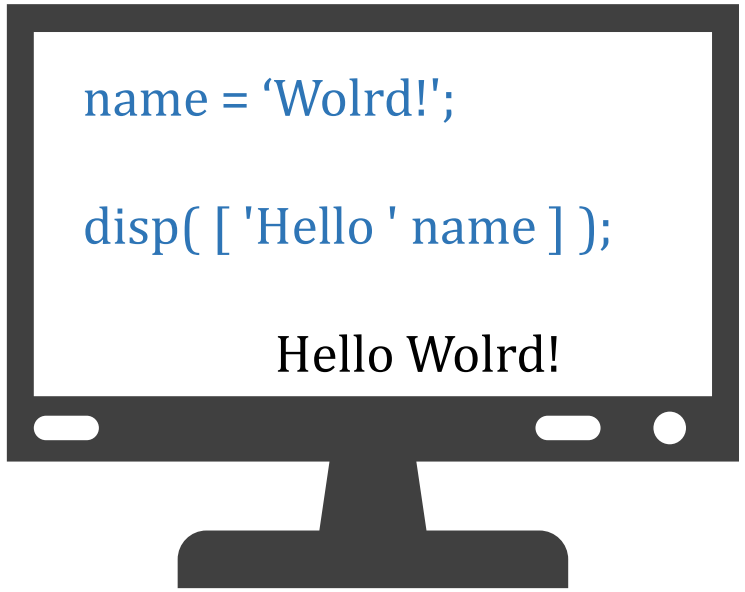
- Changing the data format
- value = 12.345678901234567

format short	→	*12.3457*
format long	→	*12.34567890123457*
format short e	→	*1.2346e+001*
format long e	→	*1.234567890123457e+001*
format rat	→	*1000/81*

DISPLAY DATA





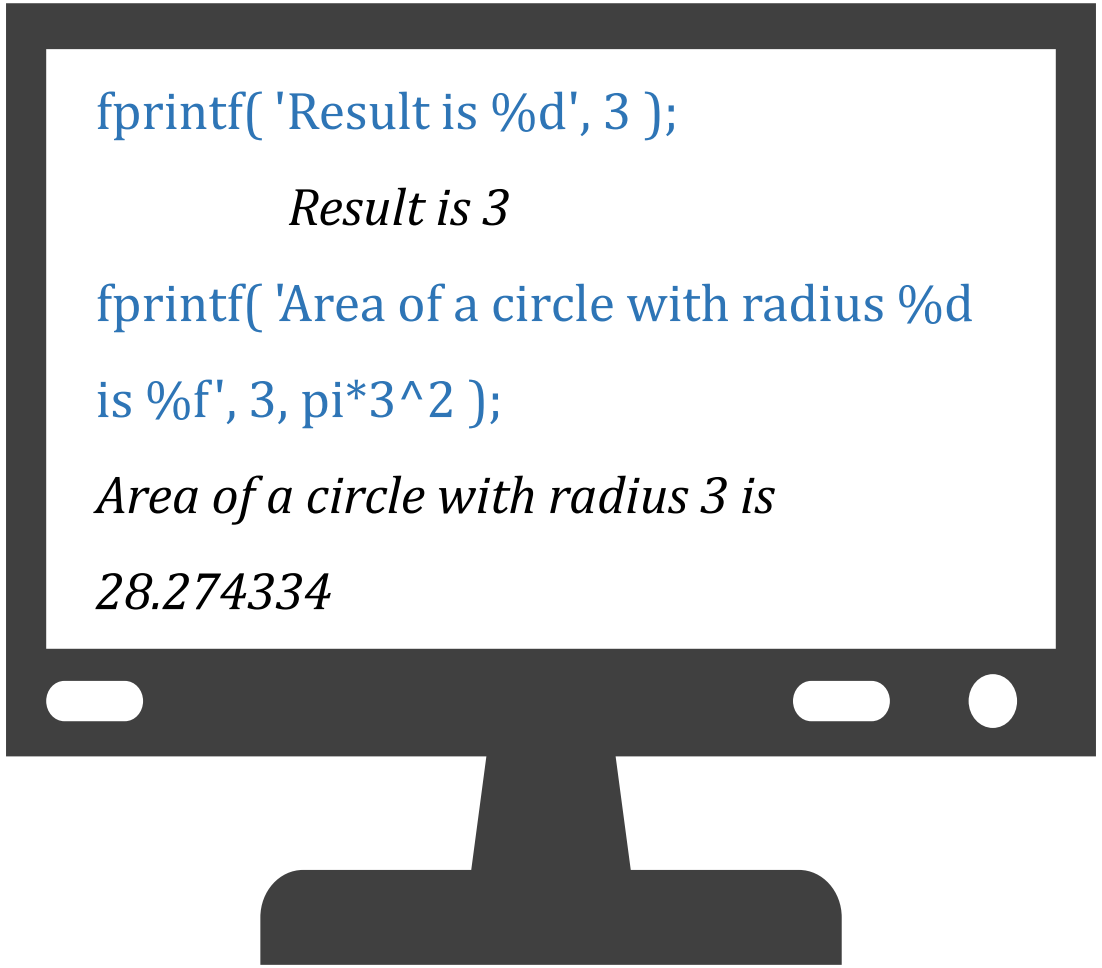


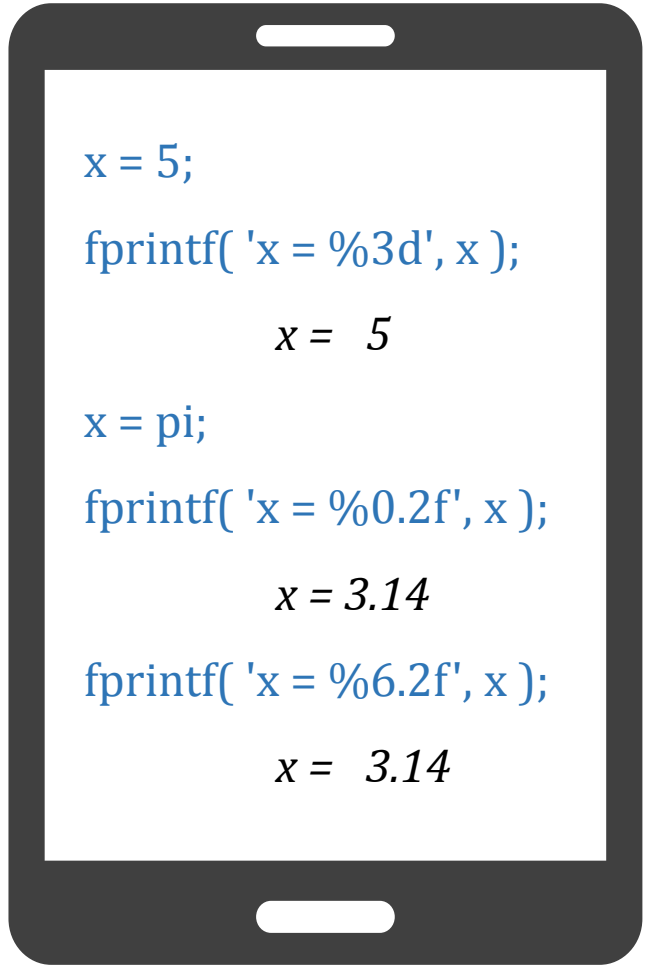
Display Data with Format

- The *fprintf(format, data)* function:

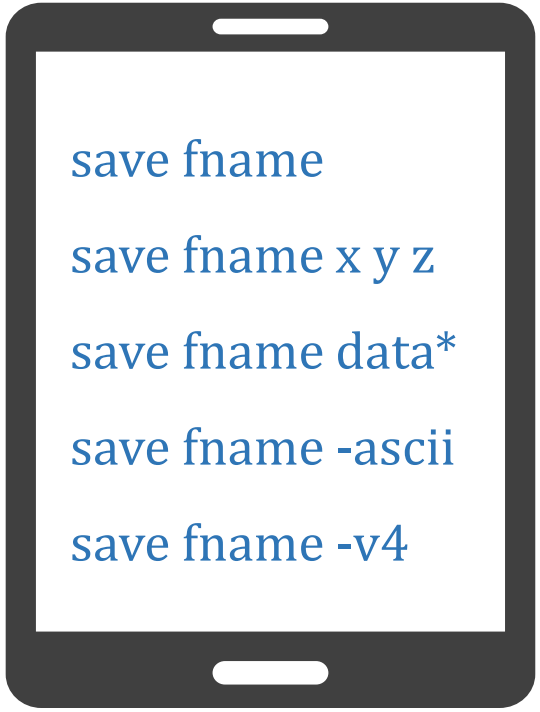
%d	*integer*
%f	*floating point format*
%e	*exponential format*
\n	*new line character*
\t	*tab character*

Output Formatting



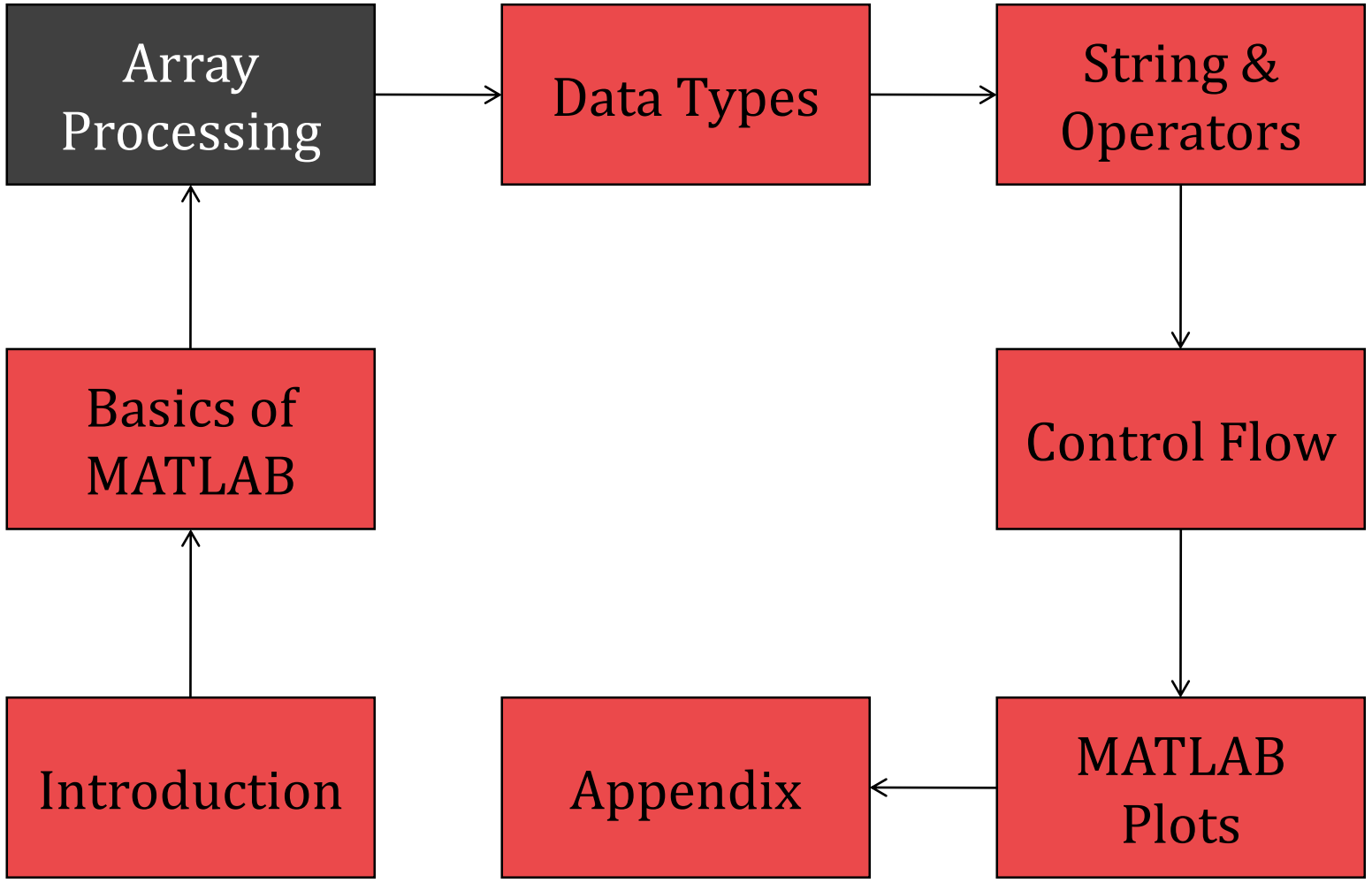


File I/O: Low Level Routines



- csvread - Comma-separated ASCII files
- dlmread - General ASCII delimited files
- textread - Read formatted data

Operational Pipeline



Array vs. Matrix Operation

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}, B = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \end{bmatrix}$$

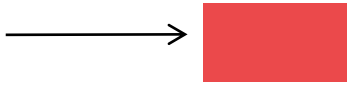
$$A.*B = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & a_{13}b_{13} \\ a_{21}b_{21} & a_{22}b_{22} & a_{13}b_{13} \end{bmatrix}$$

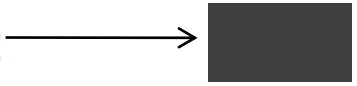
$$A./B = \begin{bmatrix} a_{11}/b_{11} & a_{12}/b_{12} & a_{13}/b_{13} \\ a_{21}/b_{21} & a_{22}/b_{22} & a_{13}/b_{13} \end{bmatrix}$$

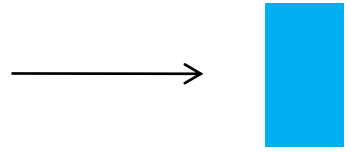
$$A.\hat{}\, n = \begin{bmatrix} a_{11}^n & a_{12}^n & a_{13}^n \\ a_{21}^n & a_{22}^n & a_{23}^n \end{bmatrix}$$

```
>> M = [1 2; 3 4]
M =
    1    2
    3    4

>> M^2
ans =
    7   10
   15   22

>> M.^2
ans =
    1    4
    9   16
```

Basics of Matrix

a = [1 2]; 

b = [3, 4]; 

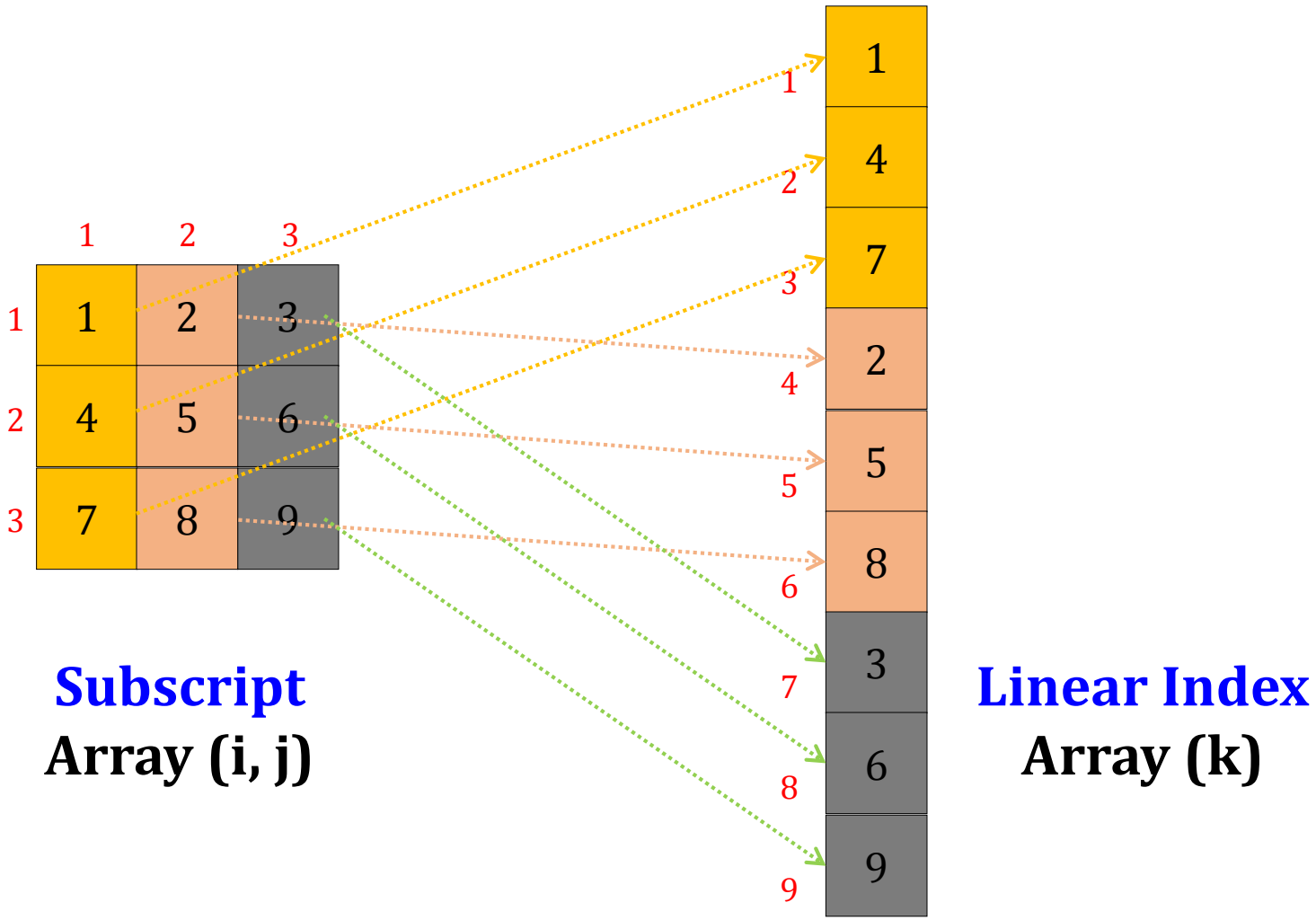
c = [5; 6]; 

d = [a; b]; 

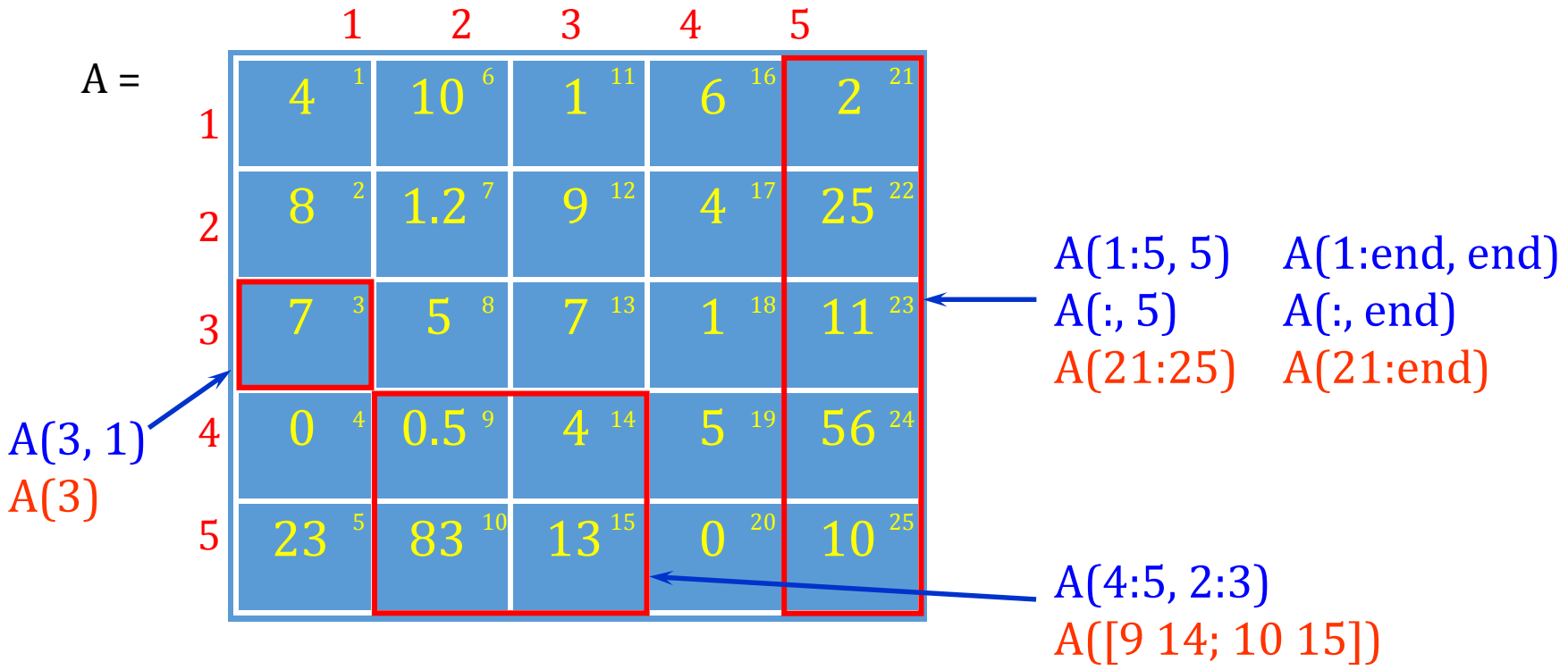
e = [d c]; 

f = [[e e]; [a b a]]; 

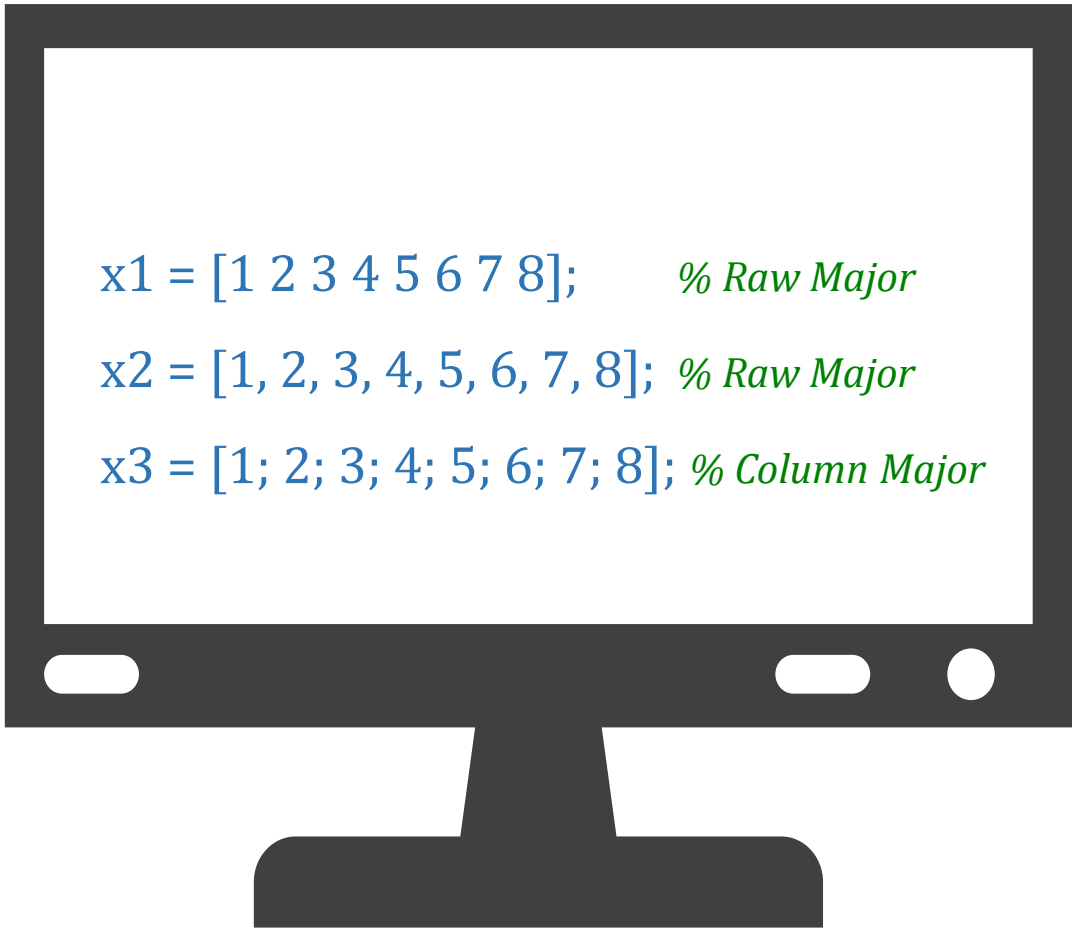
Array Indexing

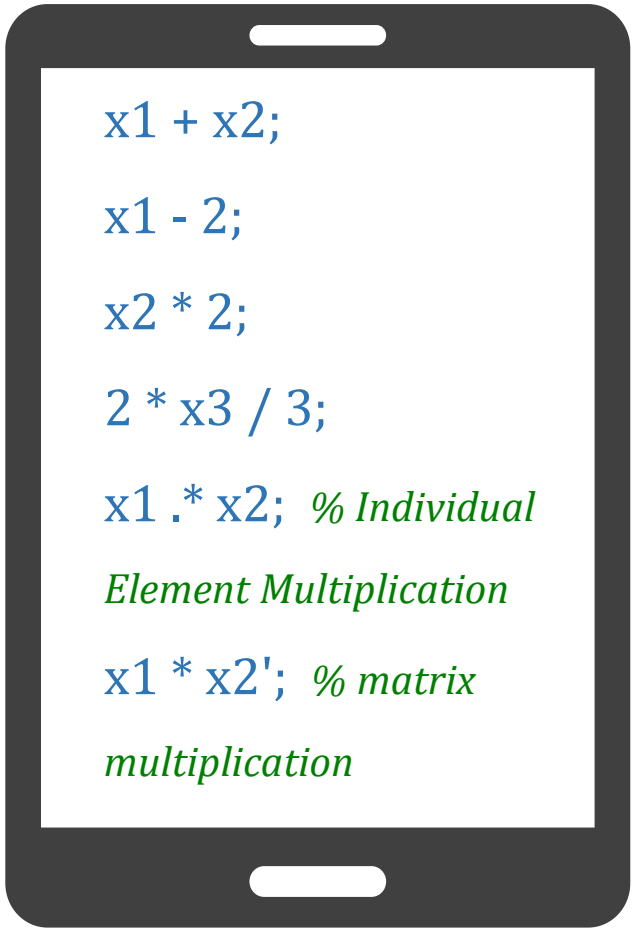


Array Indexing

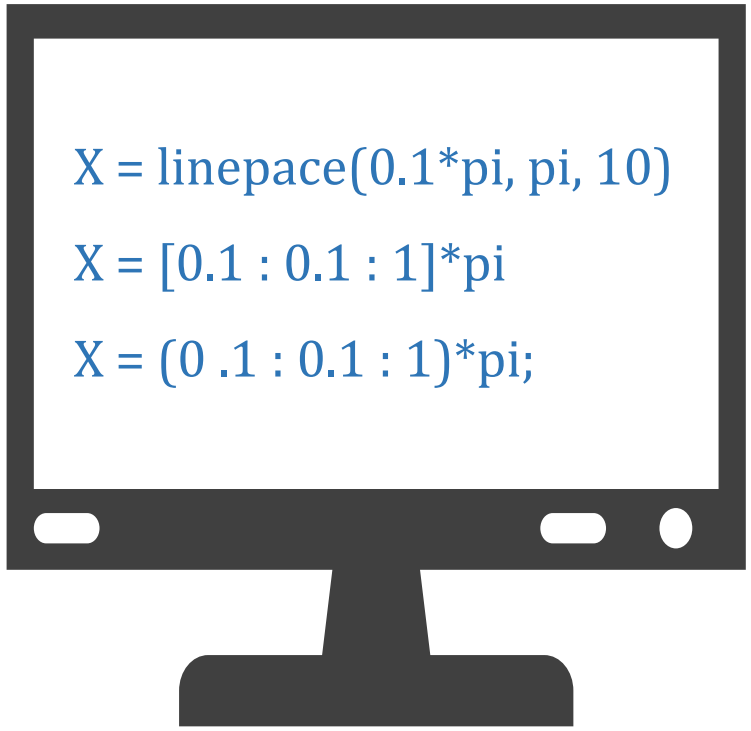


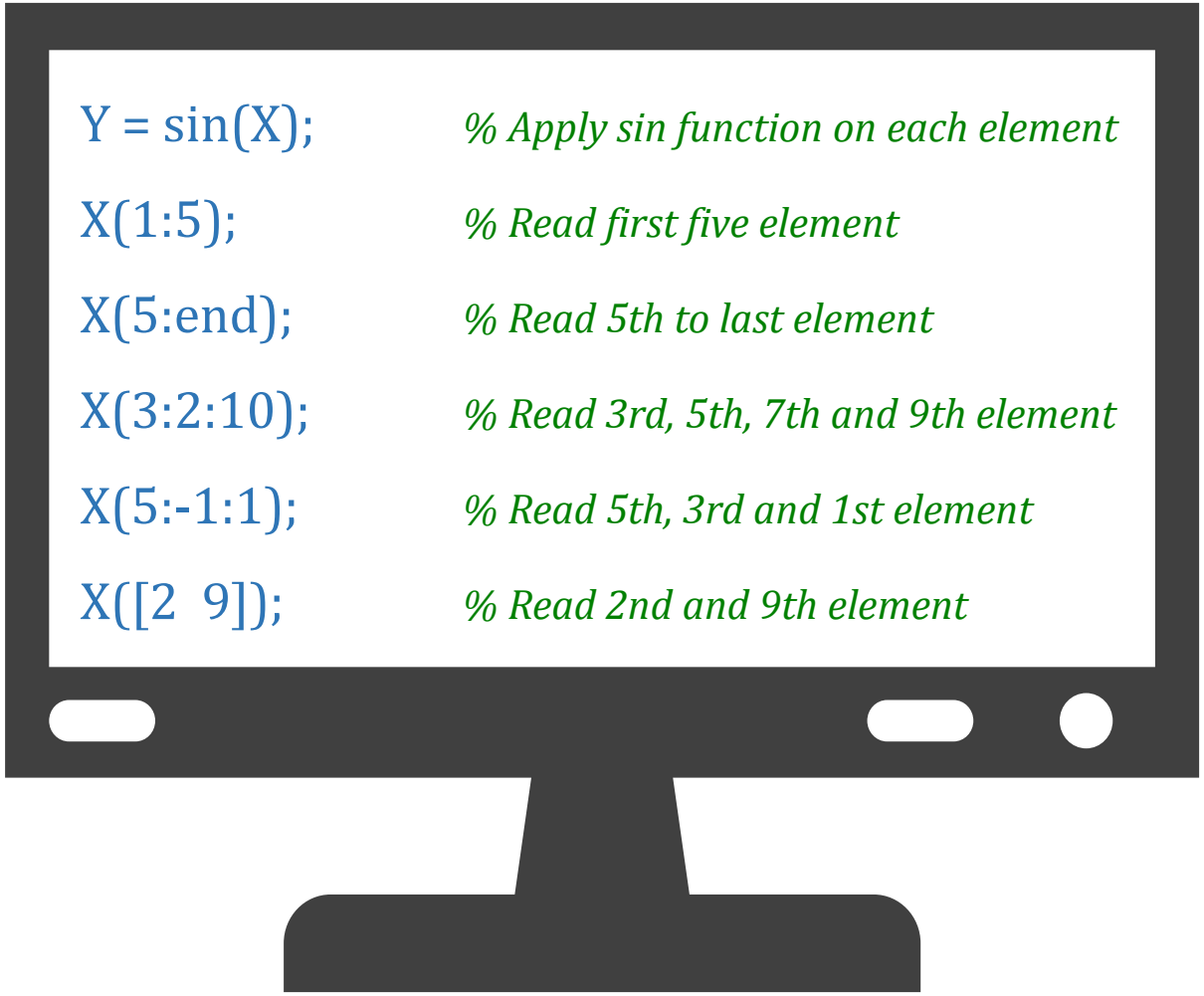
Array Basics



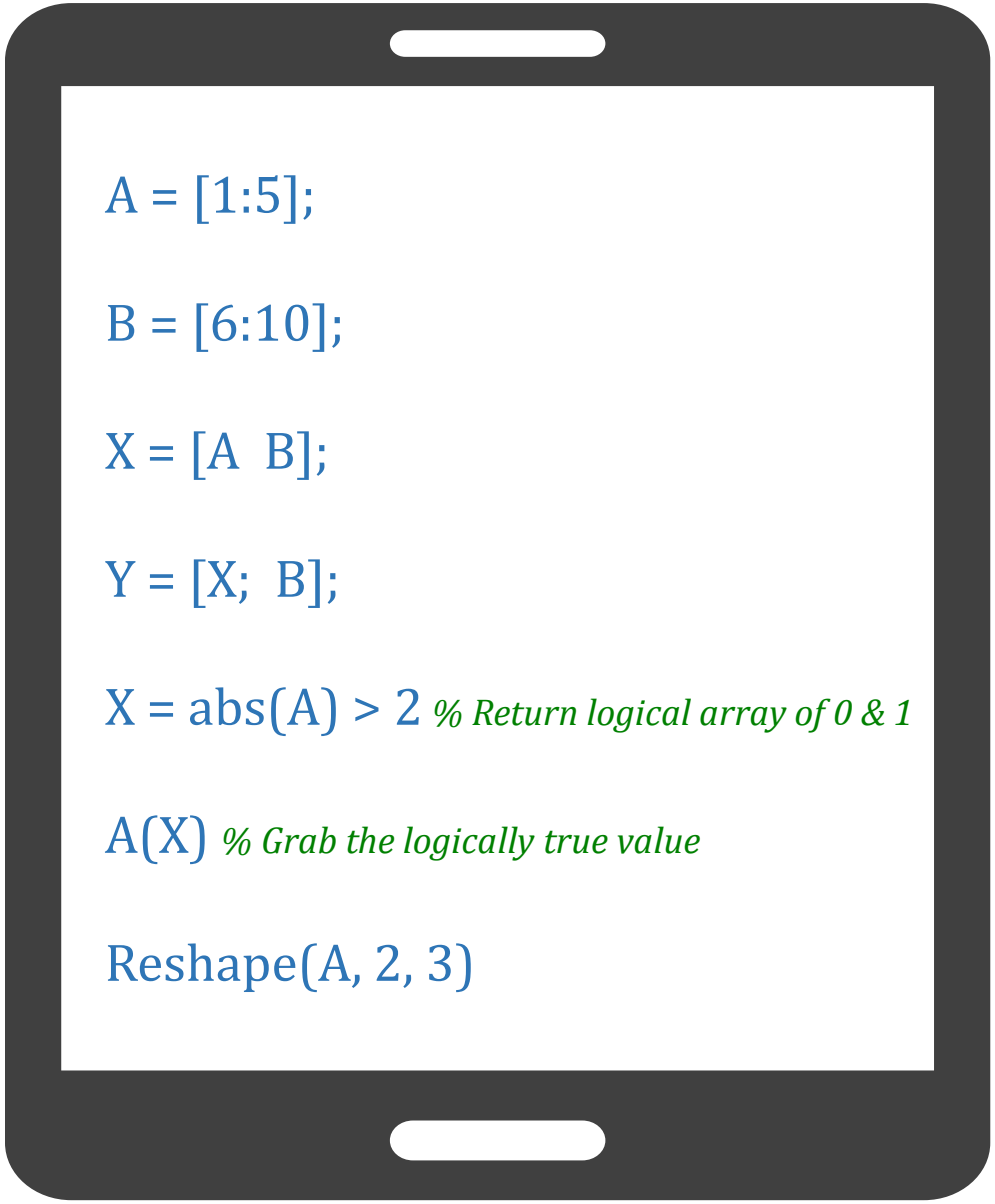


Array Initialization and Use





Array Basics



Array Indexing

- *Use () parentheses to specify index*
- *Colon operator (:) specifies range / ALL*
- *[ ] to create matrix of index subscripts*
- *'end' specifies maximum index value*
- *size(A) gives dimensions of array*

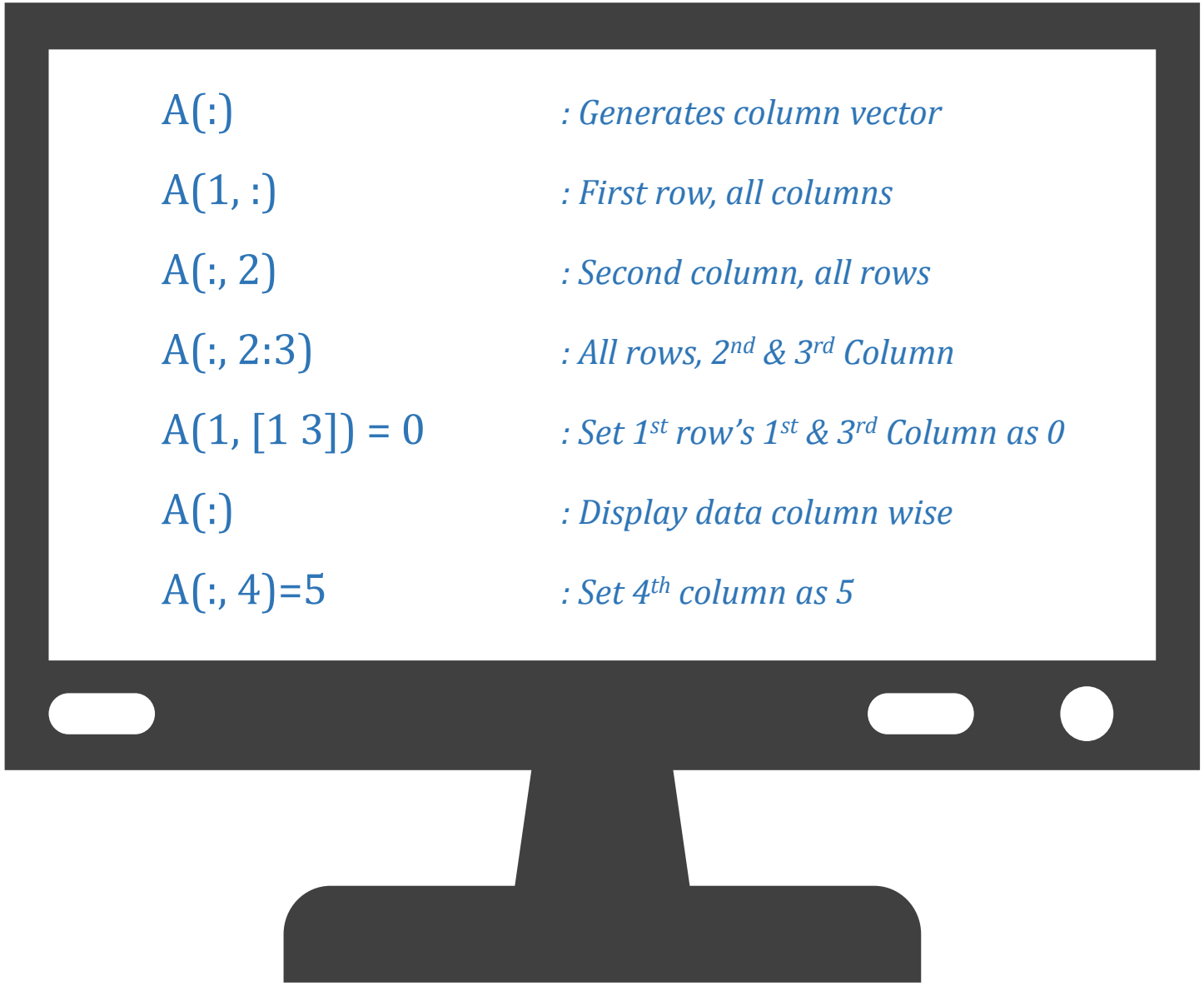
Standard Arrays

```
a=ones(5);
a=ones(5,3);

a=zeros(5);
a=zeros(5,3);

a=eye(5);
a=eye(5,3);
```

```
True(2,3);
False(2,3);

a=rand(5);
a=rand(5,3);

a=randn(5);
a=randn(5,3);
```

```
Magic(5 )

b=[1 2 3 4 5];

a=diag(b);
a=diag(b,2);
a=diag(b,-2);
```

Array Manipulation



Array Functions

A=[randperm(8); randperm(8); randperm(8)]

[data, index] = sort(X,' ascend')

sort(A(:, 4)) : *Sort 4th column*

find(A > 2) : *Return Index number on Logical true*

max(A) : *Return Max value from all column*

max(A(:, 1)) : *Return Max value from 1st Column*

[a b] = max(A) : *Return max value with index*

diag(A) : *Extract diagonal values*

triu(A) : *Upper triangular part*

tril(A) : *Lower triangular part*

repmat(A, 2) : *replicate matrix A by twice*

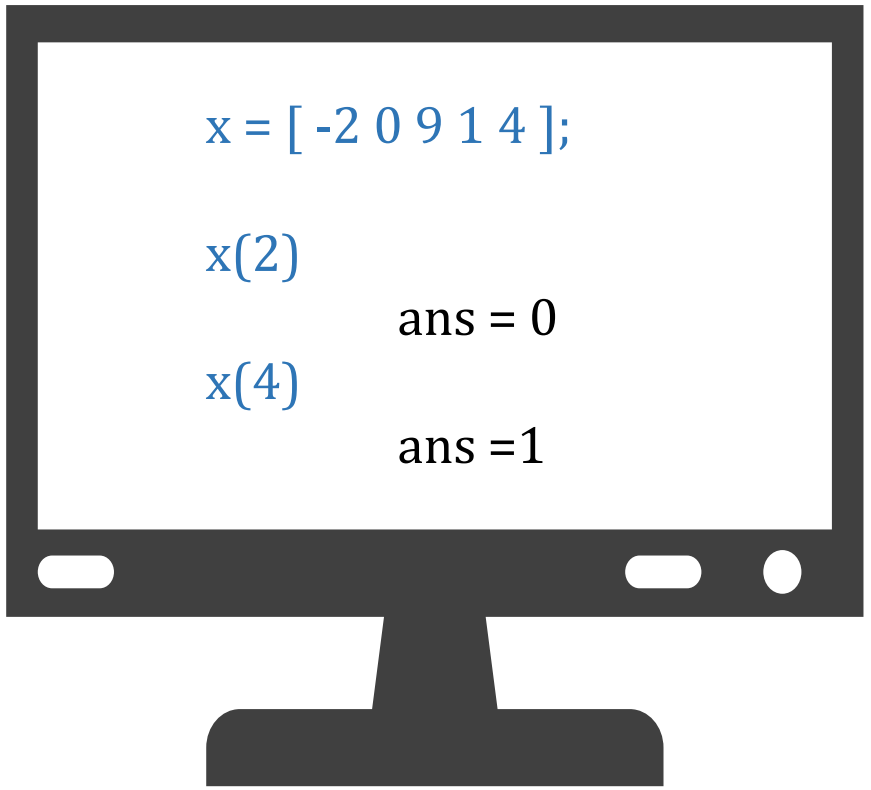
size(A) : *return dimension of an Array*

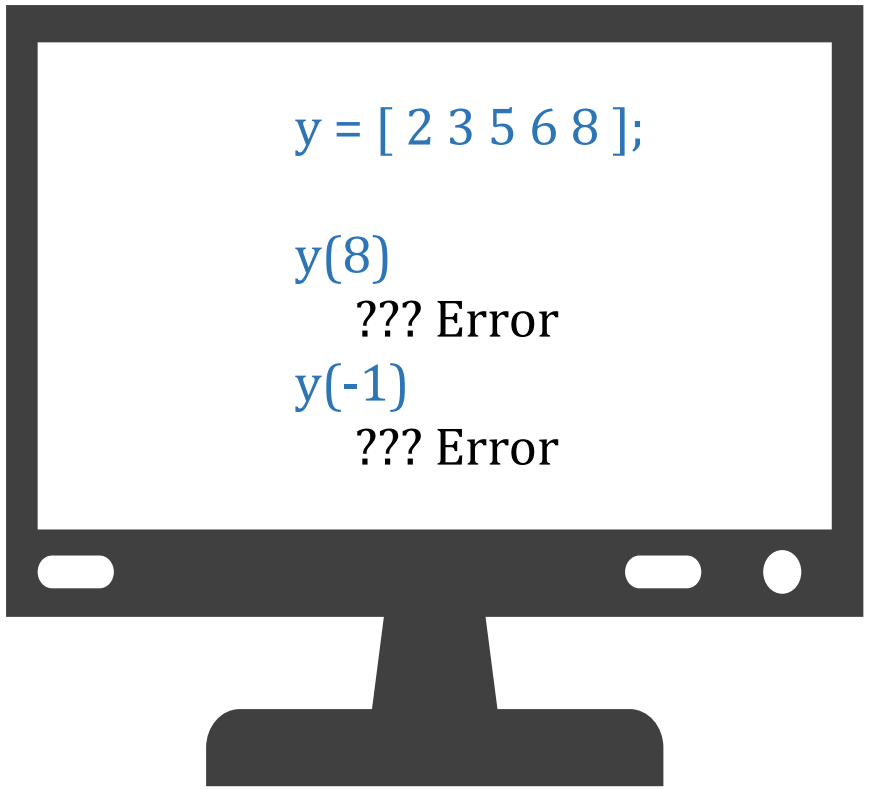
lenth(A) : *return no. of Columns*

numel(A) : *return total no. of elements in A*

Array Indexing

- Array indices start from 1





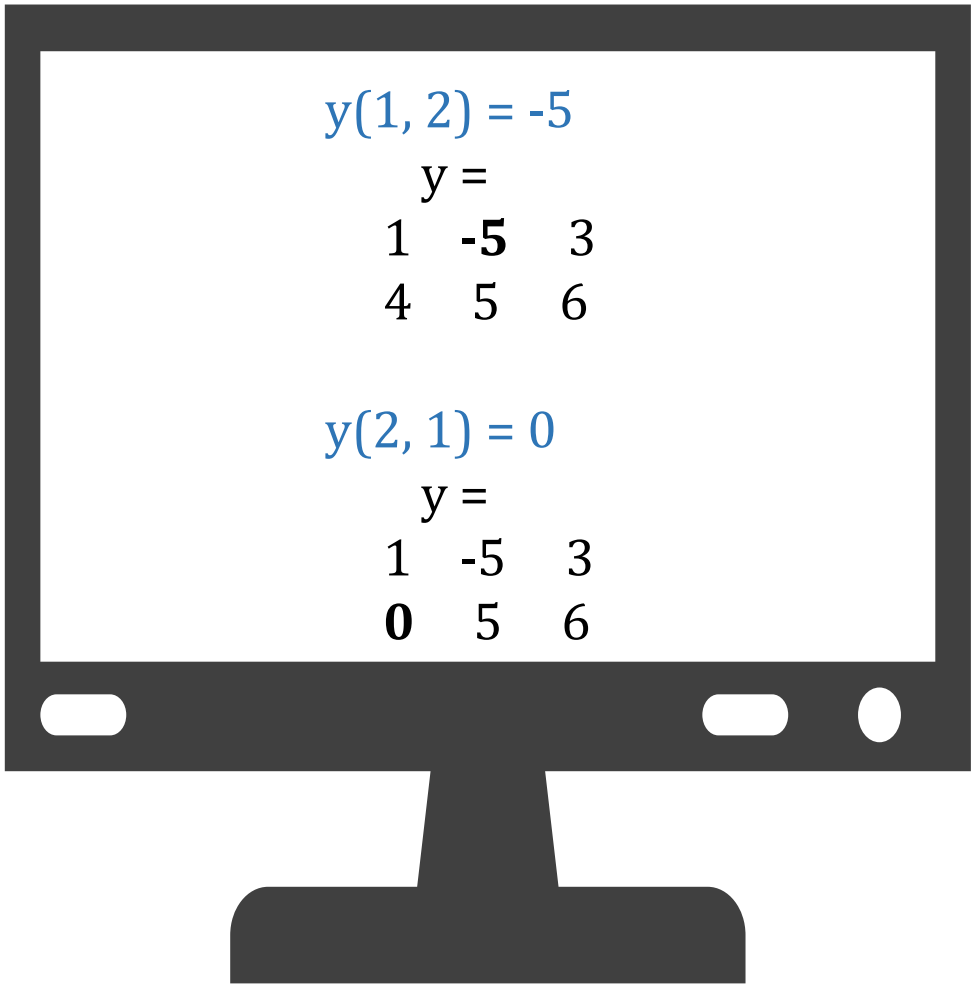
Array Indexing

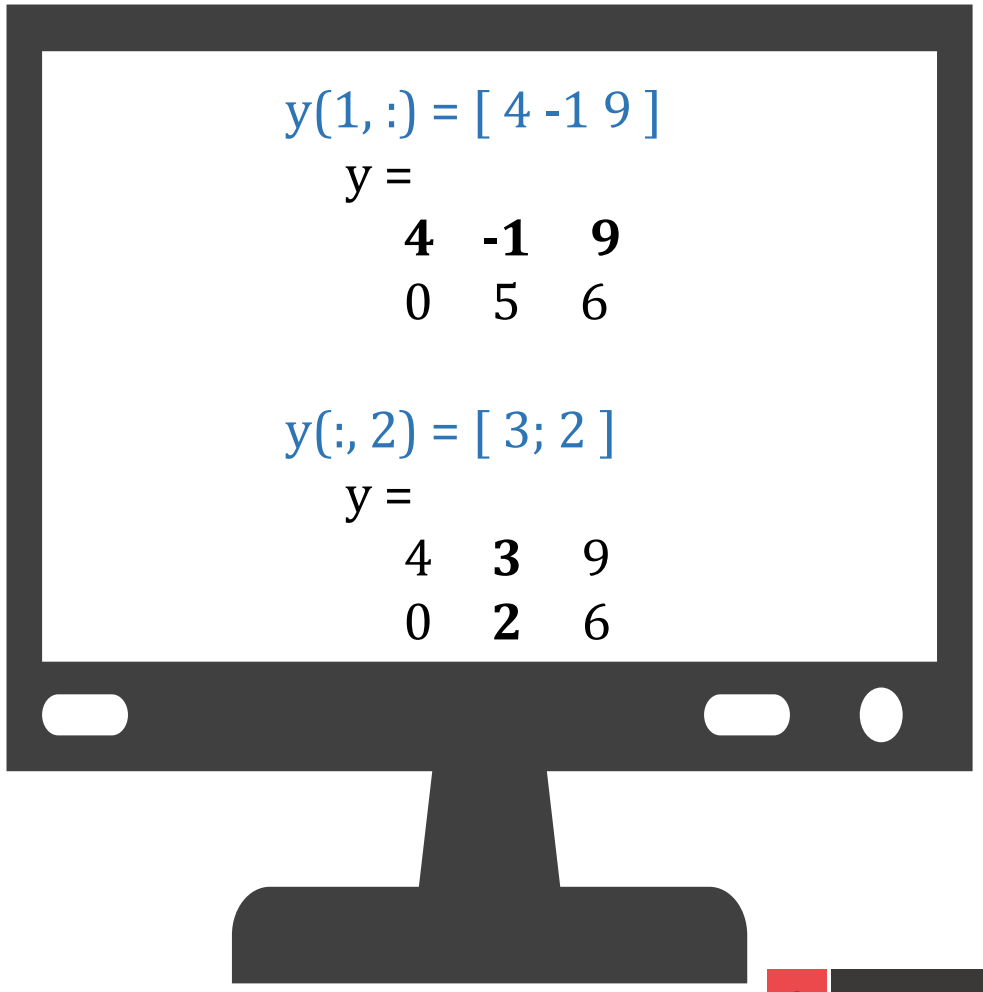
```
y = [ 1 2 3; 4 5 6 ];

y(1, 2)
        ans = 2

y(2, 3)
        ans = 6

y(2)
        ans = 4

y(1, 2:end)
        ans = 2     3
```

```
y(1, :)
        ans = 1     2     3

y(:, 2)
        ans =       2
                    5

y(2, 1:2)
        ans = 4     5

y(:, 2:end)
        ans = 2     3
              5     6
```

Array Indexing

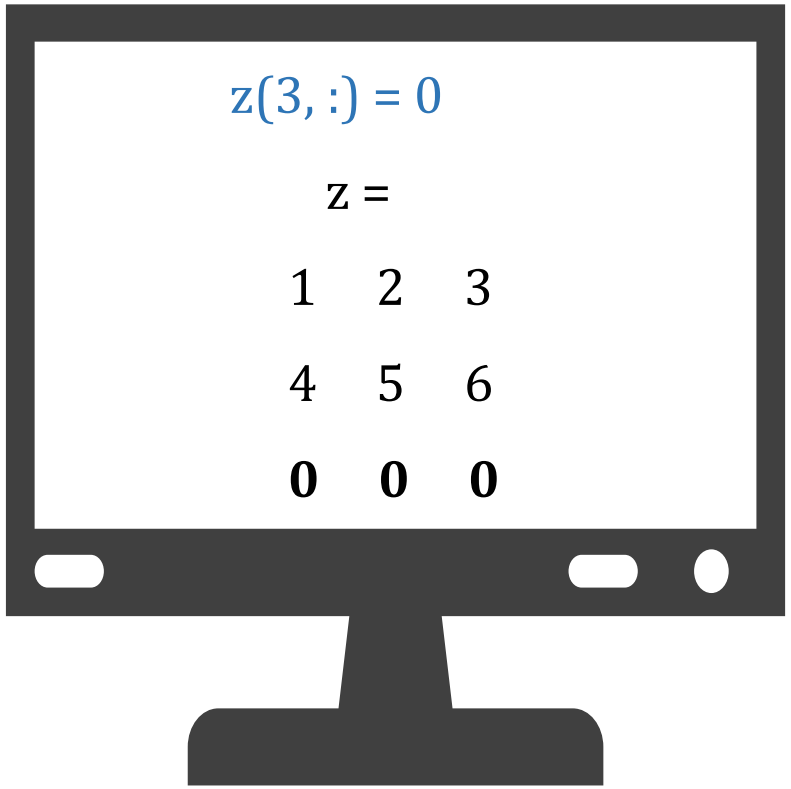
```
y = [ 1 2 3; 4 5 6 ];
```

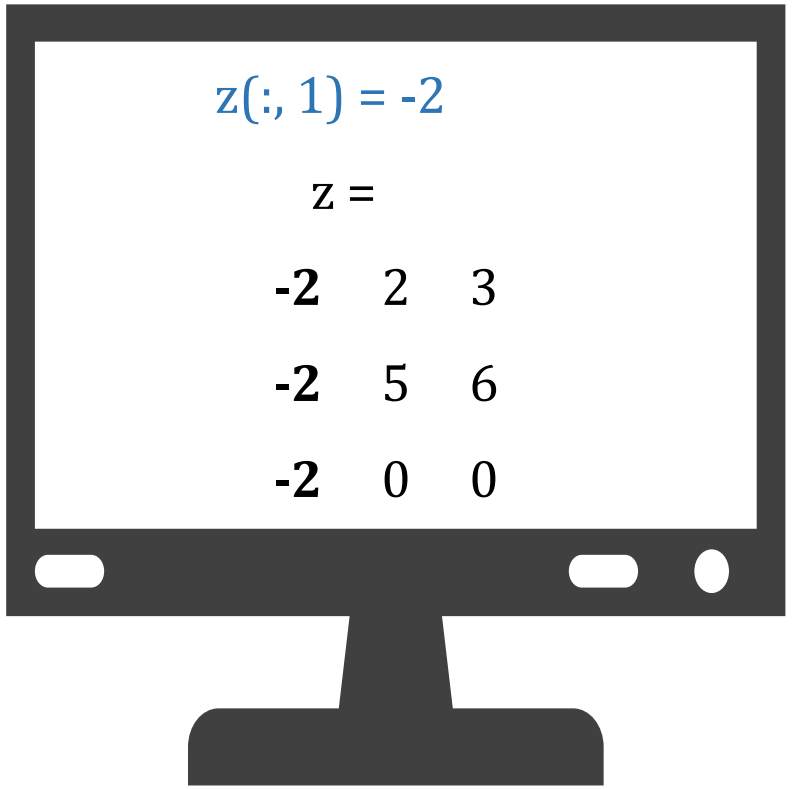


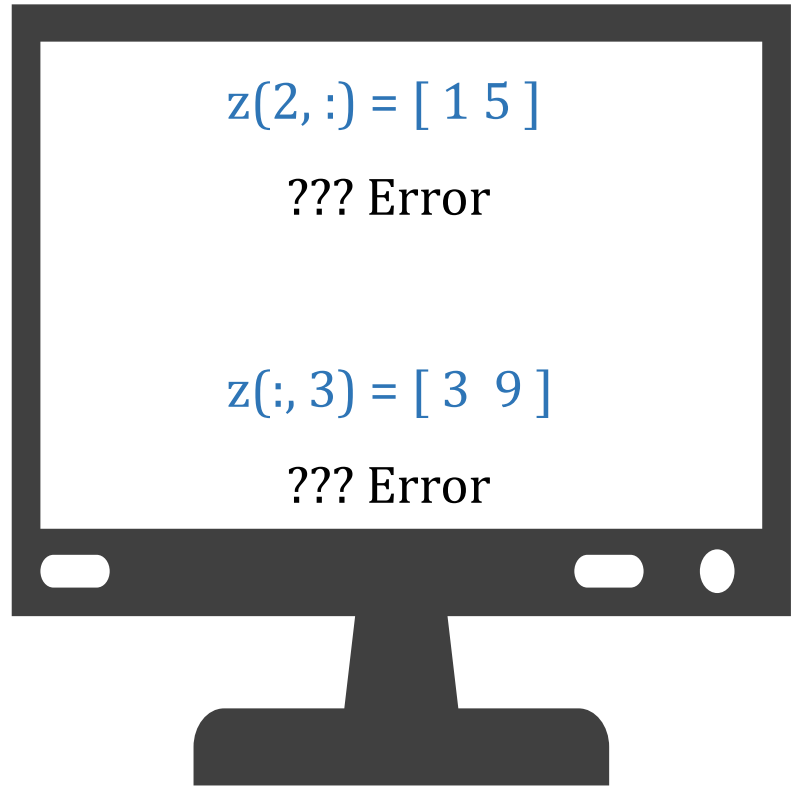


Array Indexing

z = [1 2 3; 4 5 6; 7 8 9];

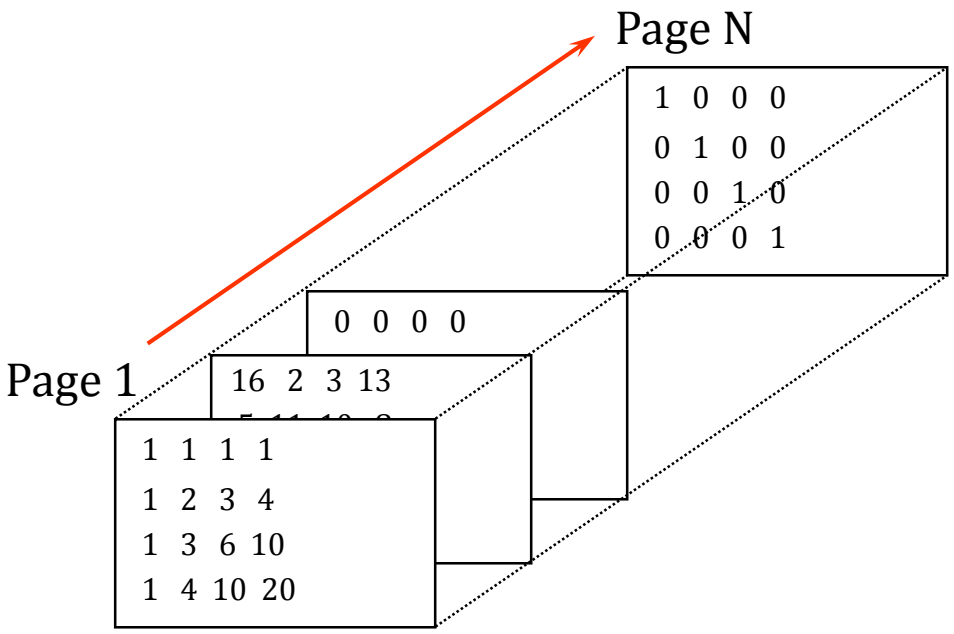






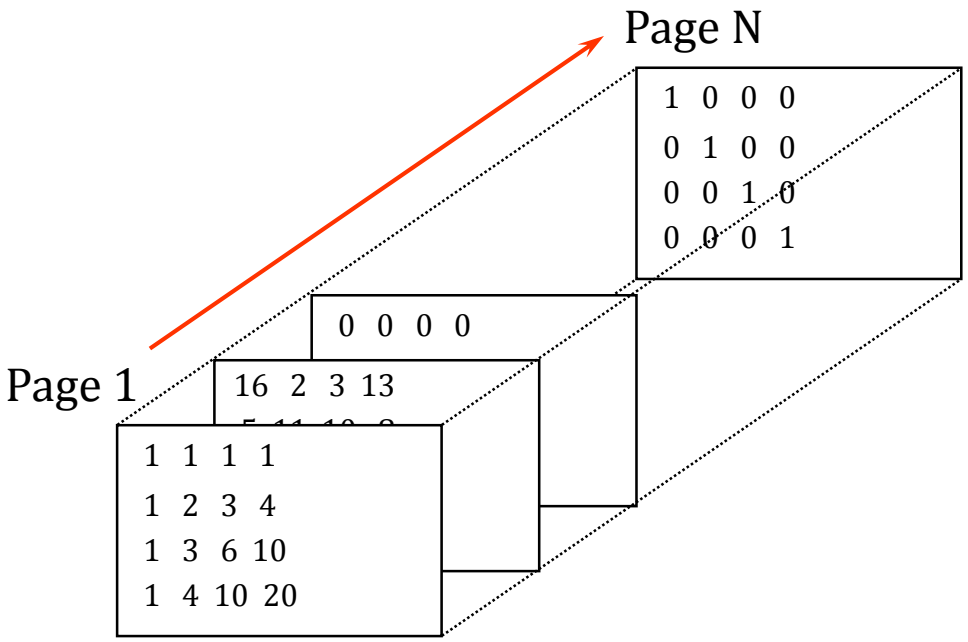
MULTIDIMENSIONAL ARRAY

- Generally Third dimension is referred as page, and higher dimensions have no generic name
- Each page contains 2D Array




```
A = pascal(4);

A(:, :, 2) = magic(4)

A(:, :, 3) = zeros(4)

.

A(:, :, N) = eye(4)
```

Displaying Multidimensional Array



A (:, :, 1) = 1 1 1 1
1 2 3 4
1 3 6 10
1 4 10 20

A(: ; : ; 2) = 16 2 3 13
5 11 10 8
9 7 6 12
4 14 15 1

A(2, :, 1) = 1 2 3 4

A(:, 1, 2) =

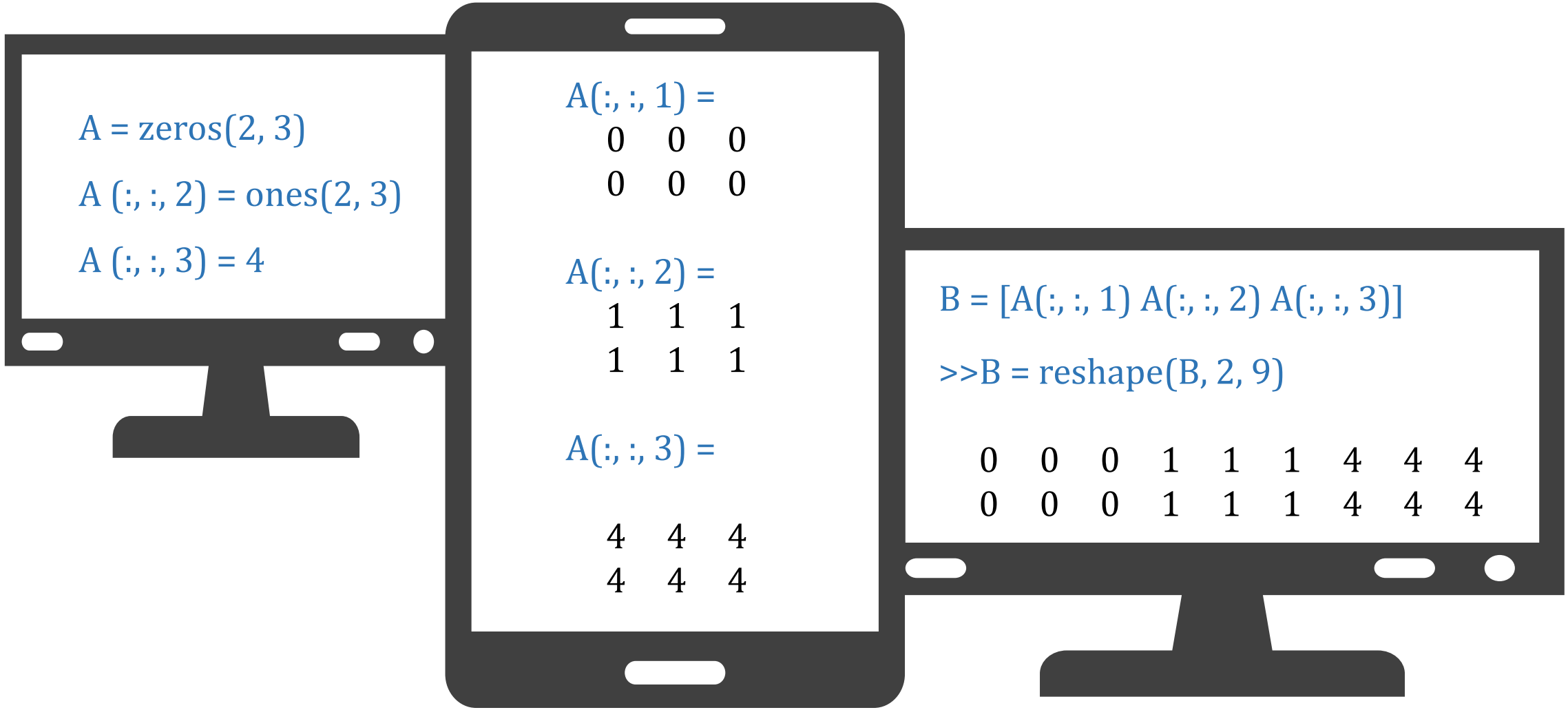
16

5

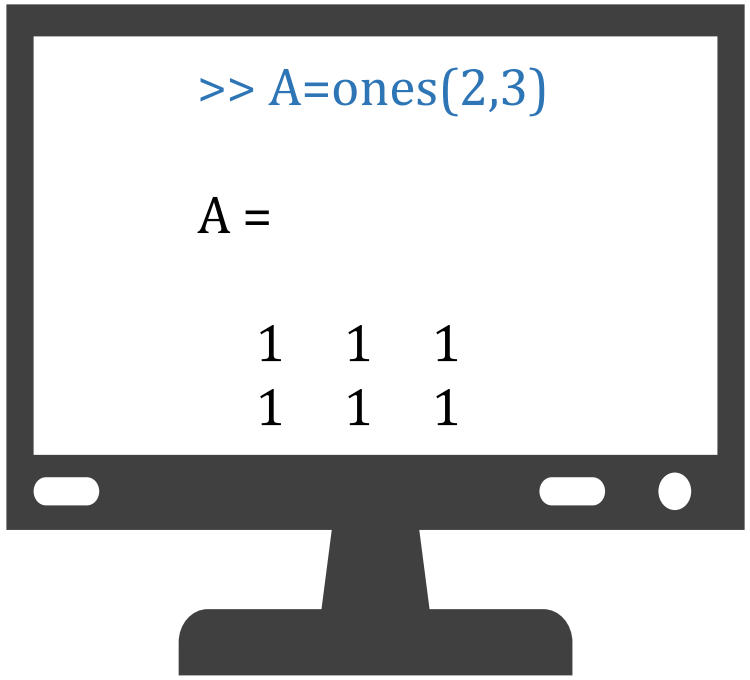
9

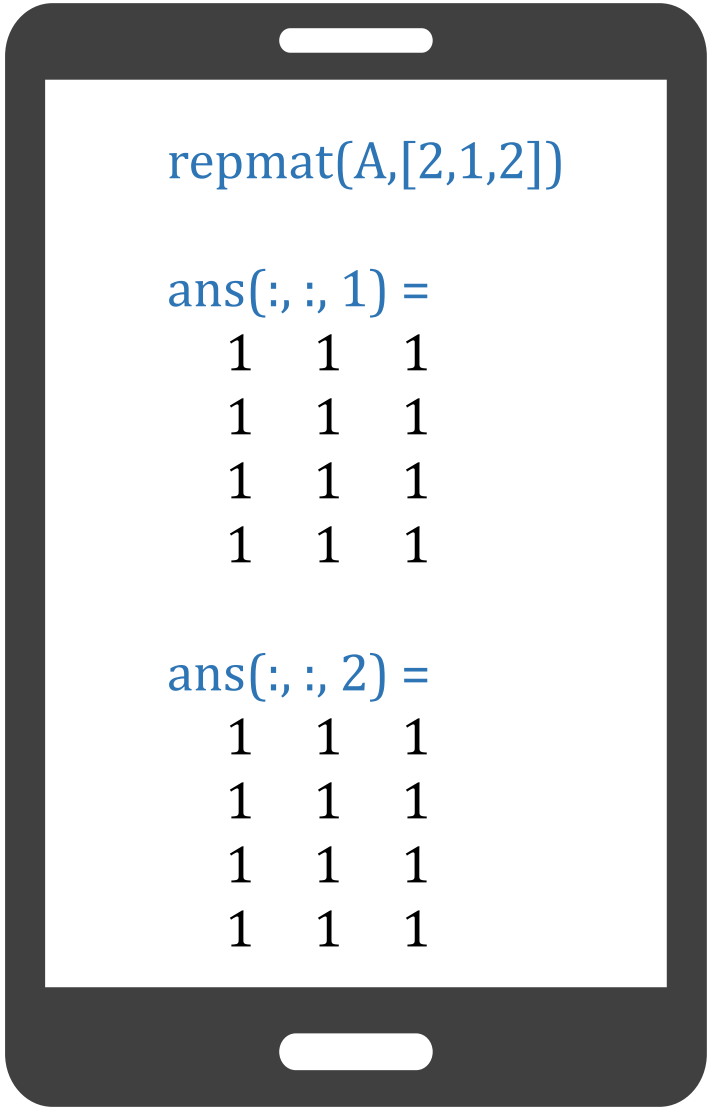
4

3D Array Manipulation

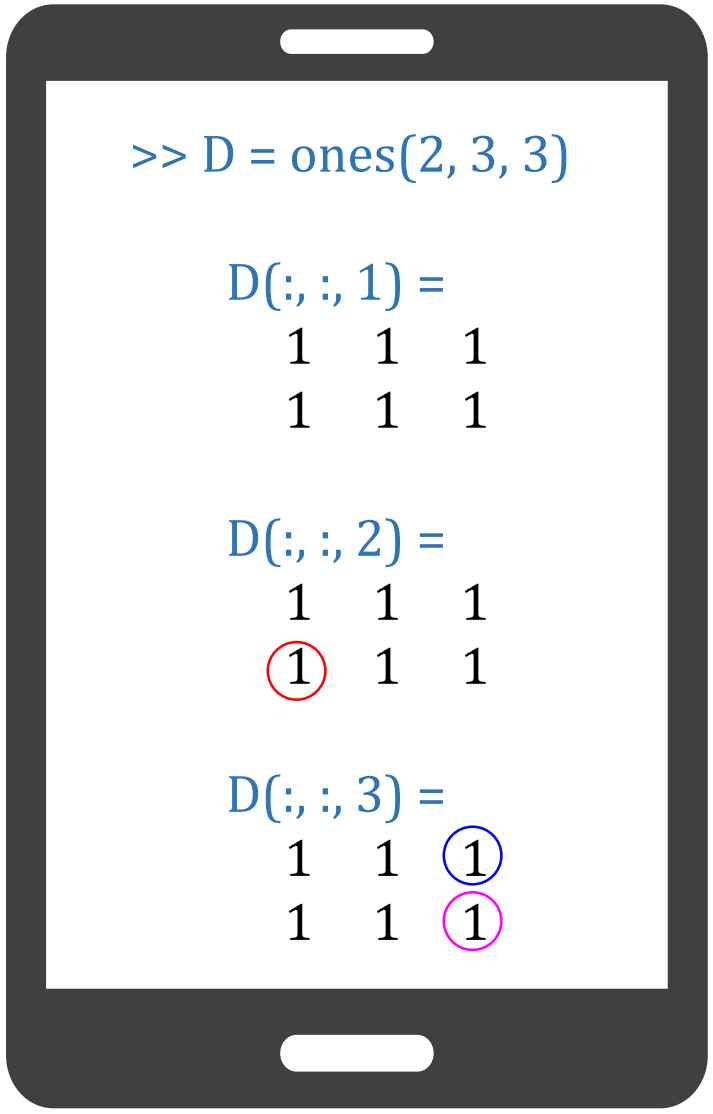


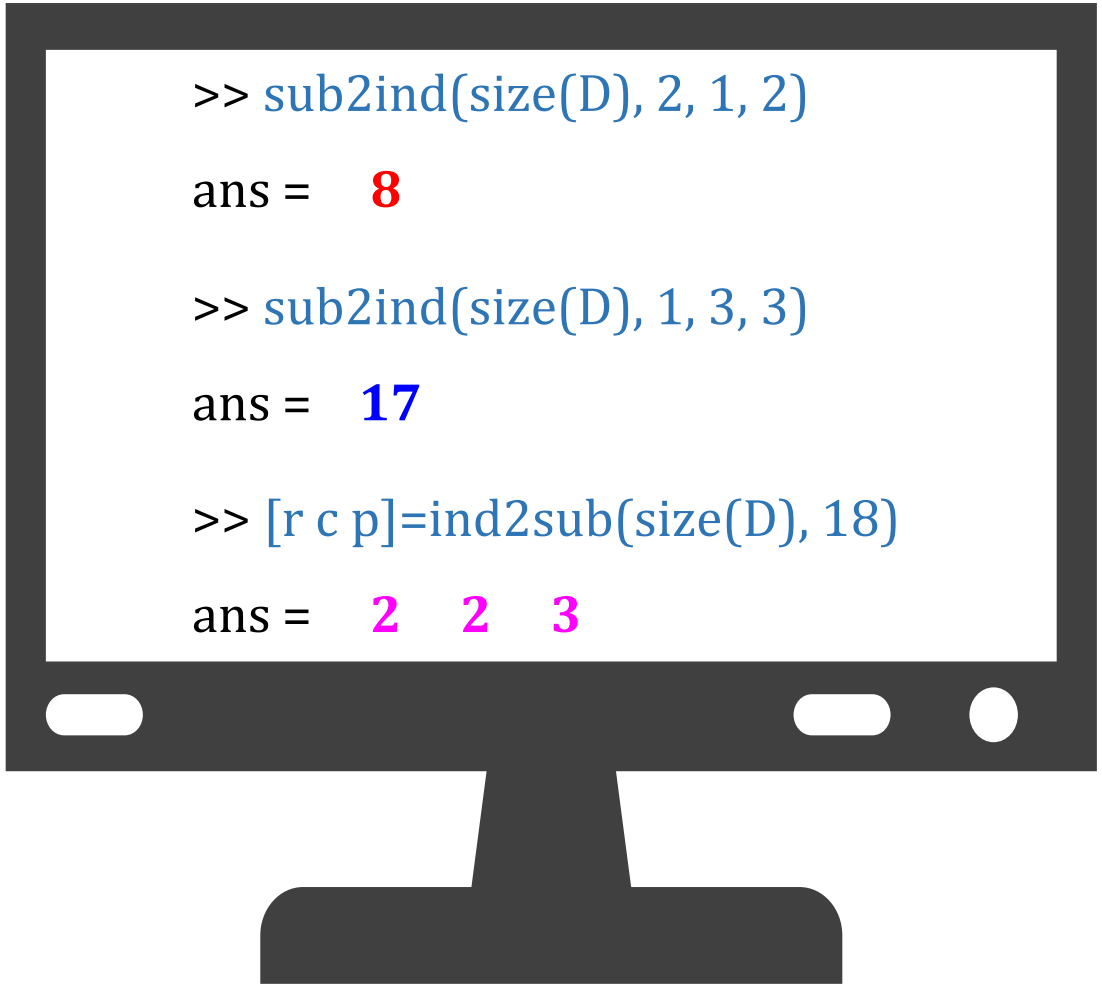
3D Array Manipulation





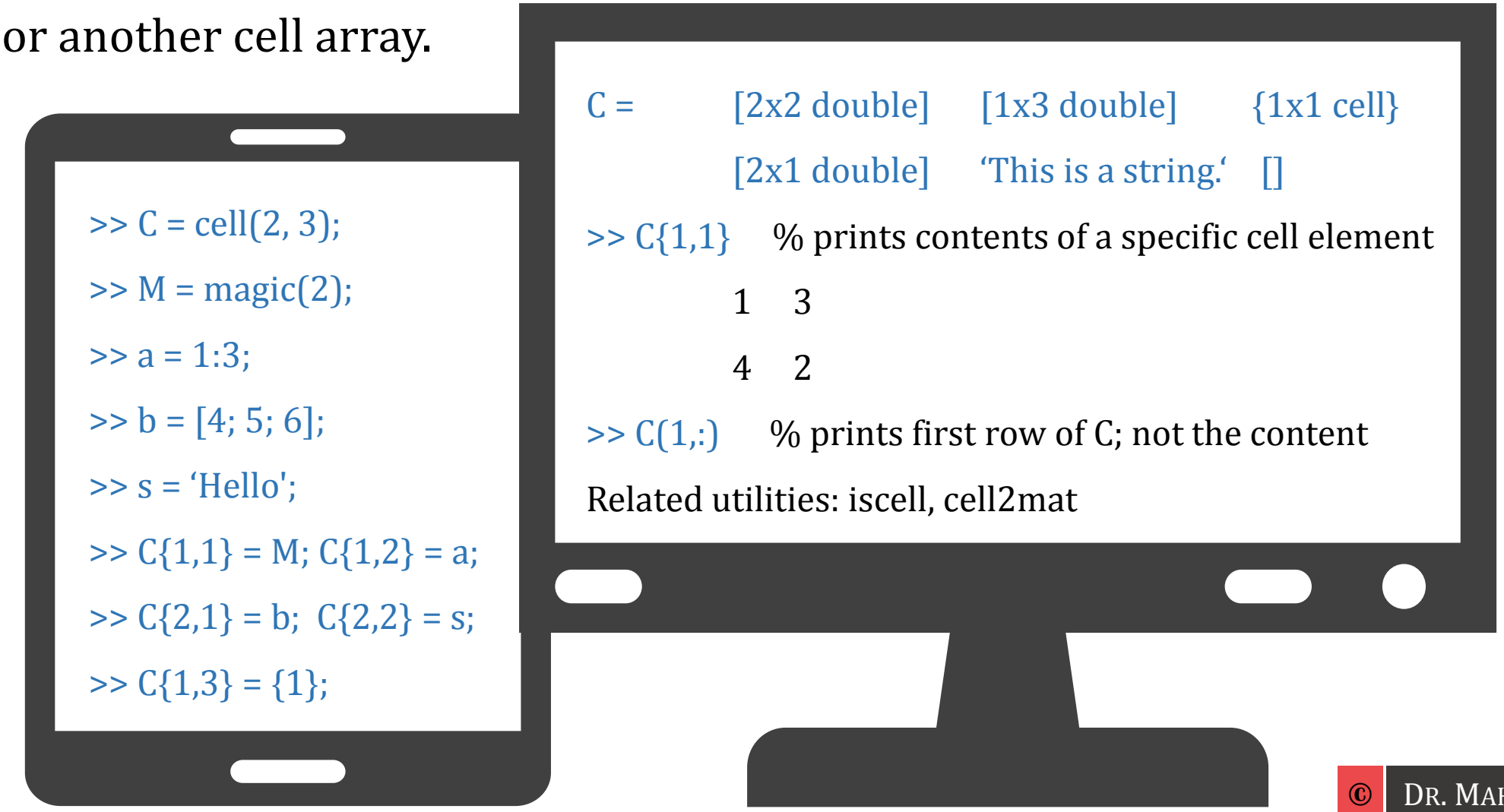
3D Array Indexing





Cell Arrays

- It is a special array of arrays. Each element of cell array may point to a scalar, an array, or another cell array.



STRUCTURES

```
>> name(1).last = 'Smith';    name(2).last  = 'Hess';
>> name(1).first = 'Mary';    name(2).first = 'Robert';
>> name(1).sex = 'female';    name(2).sex = 'male';
>> name(1).age = 45;          name(2).age = 50;
>> name(2)
```

```
    last: 'Hess'
    first: 'Robert'
    sex: 'male'
    age: 50
```

```
>> name = struct('last',{Smith','Hess'},
'first',{Mary','Robert'},...
        ('sex',{female','male'}, 'age',{45,50});
```

Related utilities: isstruct, fieldnames, getfield, isfield

Thank You!

Dr. Mahesh Goyani

mgoyani@gmail.com

www.maheshgoyani.in