```
% Basic code of MatLab
% vector ( matrix 1x5 )
>> v = [1 4 -7 10 -6]

v =

     1     4    -7    10    -6

% To run a Matlab without displaying the output a semicolon to the end of the command
>> v1 = [1 4 7 10 13 12 53 -6 8];
>> v1

v1 =

     1     4     7    10    13    12    53    -6     8

 % To reverse the order of the vector
>> flip_v1 = fliplr(v1)

flip_v1 =

     8    -6    53    12    13    10     7     4     1

% Or
>> flip_v2 = v1(end:-1:1)

flip_v2 =

     8    -6    53    12    13    10     7     4     1

% To list the defined variables in the Matlab workspace, type
>> who

Your variables are:

flip_v1  flip_v2  v        v1

% To get detailed information about these variables, type
>> whos
  Name          Size            Bytes  Class     Attributes

  flip_v1       1x9                72  double
  flip_v2       1x9                72  double
  v             1x5                40  double
  v1            1x9                72  double

% Column vectors are created with semicolons
>> w = [1;4;9;6;8;7;10]

w =

     1
     4
     9
     6
     8
     7
    10

% Or use matrix transpose
>> w1 = [1 5 9 6 7 10 5]'

w1 =

     1
     5
     9
     6
     7
    10
     5

% To access blocks of elements, we use MATLAB's colon notation (:)
>> w(1:3)

ans =

     1
     4
     9

% To access all elements from the third through the last elements
>> w(3:end)

ans =

     9
     6
     8
     7
    10

% To access all elements from the 3rd to the element before the end while skipping the n
>> w(3:2:end-1)

ans =

     9
     8

>> v(:)

ans =

     1
     4
    -7
    10
    -6

% To enter a matrix A
>> A = [1 2 3; 4 5 6; 7 8 9]

A =

     1     2     3
     4     5     6
     7     8     9

%We can then view a particular element in a matrix by specifying its location (row, colu

>> A(2,1)

ans =

     4

%Changing a value
>> A(3,3) = 0

A =

     1     2     3
     4     5     6
     7     8     0

 % Matrix Operations:
>> m1 =[1 5 9 6 3 9]

m1 =

     1     5     9     6     3     9

>> m2 = [2 5 8 7 3 9]

m2 =

     2     5     8     7     3     9

>> m1 = 3 * m1

m1 =

     3    15    27    18     9    27

>> m2 = m1*2-5

m2 =

     1    25    49    31    13    49

>> Q1 = ones(3,2)
```

```
Q1 =

     1     1
     1     1
     1     1

>> Q2 = ones(2)

Q2 =

     1     1
     1     1

>> Z1 = zeros(size(Q1))

Z1 =

     0     0
     0     0
     0     0

>> size(Q1)

ans =

     3     2
% randam value from 1 to 10 range for matrix [2 4]
>> A1 = randi(10,[2 4])
>> A1 = randi(10,[2 4])
>> A1 = randi(10,[2 4])

A1 =

     7     8     7     8
     8     4     2     1


A1 =

     3     1     7    10
     1     9     4     1


A1 =

     5     8     2     5
     4     8     5     7
 % To understand the function of randi, type help randi
 >> help randi
 randi - Uniformly distributed random integers
    This MATLAB function returns a random scalar integer between 1 and imax.

    Syntax
```

```
      X = randi(imax)
      X = randi(imax,n)
      X = randi(imax,sz1,...,szN)
      X = randi(imax,sz)

      X = randi(___,typename)
      X = randi(___,"like",p)

      X = randi([imin,imax],___)

      X = randi(s,___)

    Input Arguments
      imax - Largest integer in sample interval
        positive integer
      imin - Smallest integer in sample interval
        1 (default) | scalar integer
      n - Size of square matrix
        integer value
      sz1,...,szN - Size of each dimension (as separate arguments)
        integer values
      sz - Size of each dimension (as a row vector)
        integer values
      typename - Data type (class) to create
        "double" (default) | "single" | "int8" | "uint8" | "int16" |
        "uint16" | "int32" | "uint32" | "logical"
      p - Prototype of array to create
        numeric array | logical array
      s - Random number stream
        RandStream object

    Examples
      Square Matrix of Random Integers
      Random Integers Within Specified Interval
      Control Random Number Generation
      3-D Array of Random Integers
      Random Integers of Other Data Types
      Size Defined by Existing Array
      Size and Numeric Data Type Defined by Existing Array
      Random Complex Integers
      Random Logical Array

    See also rand, randn, rng, RandStream, randperm

    Introduced in MATLAB in R2008b
    Documentation for randi
    Other uses of randi

>> D1 = [2 5 6 ;4 7 6]

D1 =

     2     5     6
     4     7     6
```

```
>> R1 = [2 1 ;8 2 ;5 6]

R1 =

     2     1
     8     2
     5     6

% multiply D1 by R1
>> Result = D1*R1

Result =

    74    48
    94    54

% Multiply element by element of matrices using the dot “.” before “*”
>> [1 2 3] .* [4 5 6]

ans =

     4    10    18

% Element-wise operations using the dot “.” Before the operator
>> bw = 2 .^(7:-1:0)

bw =

   128    64    32    16     8     4     2     1

% Matlab functions bin2dec (binary number to decimal) and dec2bin (decimal number to bin
>> b_bin1 =dec2bin(185)

b_bin1 =

    '10111001'
>> b_bin2 =dec2bin(185, 12)

b_bin2 =

    '000010111001'

>> b_dec1 = bin2dec(b_bin1)

b_dec1 =

   185
```