**Date:** 05/08/2024

## Experiment 1
## Basic Mathematical Operations

***Aim***

i)Basic Mathematical Operations
Write a MATLAB script that:

- Defines two variables ‘a’ and ‘b’ with values 5 and 3, respectively.
- Computes and displays the sum, difference, product, and quotient of ‘a’ and ‘b’.
- Uses relational operators to compare ‘a’ and ‘b’ and displays the results of the comparisons (e.g., a>b, a==b, etc.).

ii) Complex Mathematical Expressions
Write a MATLAB script that:

- Defines three variables x=2, y=4, and z=6.
- Computes the value of the expression f(x,y,z)=x2+y2−z2+√xy and displays the result.

***Theoretical Background***

MATLAB (Matrix Laboratory) is a high-level programming language and interactive environment used for numerical computation, data analysis, visualization, and algorithm development.
Given below are some important commands:

1. `ans`
`ans` is a default variable used to store the result of a computation or command when no other variable is specified. It stands for "answer." For example, if you perform a calculation like `5 + 3` without assigning it to a variable, MATLAB stores the result in `ans`:

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

2. `diary`

record all the input and output of the MATLAB command window to a file. This is useful for keeping a log of your MATLAB session. It toggles both on and off.

3. Who

lists the names of defined variables

```
>>who
    Your variables are:
            a         b
```

4.Whos

: lists the names and sizes of defined variables

```
>>whos a
    Name      Size           Bytes  Class     Attributes
            a        1x1               8  double
```

5. Arithmetic Operators

MATLAB supports a range of arithmetic operators for performing mathematical operations:

i) Addition (`+`): Adds two numbers.

ii) Subtraction (`-`): Subtracts one number from another.

iii) Multiplication (`*`): Multiplies two numbers or matrices.

iv) Division (`/`): Divides one number by another. For element-wise division of matrices, use `./`.

v) Element-wise Multiplication (`.*`): Multiplies corresponding elements of two arrays or matrices.

vi) Element-wise Division (`./`)**: Divides corresponding elements of two arrays or matrices.

```
>>g = [8 16] ./ [2 4]
    g =
          [4 4]
```

vii) Power (`^`)**: Raises a number to the power of another. For element-wise power, use `.^`.

```
>>h = 2 ^ 3
```

```
    h =
          8
>>i = [2 3] .^ 2
    i =
          [4 9]
```

viii) Square Root (`sqrt`): Computes the square root of a number or each element of an array.

```
>> j = sqrt(9);
    j =
          3
>>k = sqrt([4 16])
    k =
          [2 4]
```

6. Relational Operators

i) Relational operators are used to compare values and return a logical result (`true` or `false`):

ii) Not equal to (`~=`)**: Checks if two values are not equal.

iii) Greater than (`>`): Checks if one value is greater than another.

iv) Less than (`<`): Checks if one value is less than another.

v) Greater than or equal to (`>=`): Checks if one value is greater than or equal to another.

vi) Less than or equal to (`<=`): Checks if one value is less than or equal to another. c = (4 <= 5);

***Code***

***i)***

```
a=5;

   b=3;
   a+b

ans =

   8
```

```
   a-b

ans =

   2

   a*b
```

```
ans =

   15

    a/b

ans =

   1.6667

x = (5 == 5)


x =

 logical

  1

    y = (4 ~= 5)

y =

 logical

  1

     z = (7 > 5)

z =

 logical
```

```
  1

    a = (3 < 5)

a =

 logical

  1

     b = (5 >= 5)

b =

 logical

  1

    c = (4 <= 5)

c =

 logical

  1

    x=(5==4)

x =

 logical

           0
```

**ii)**

```
>>x=2;
>>y=4;
>>z=6;
>>f(x,y,z)=x^2 + y^2 + z^2 + sqrt(x*y);
>>f(x,y,z)
             ans =
                          58.8284
```

## ***Conclusion***

The basic commands like whos,who and diary have been completed and basic arithmetic and relational expressions have also been completed .

**Date:** 12/08/2024

## Experiment 2
## Vectors and Matrices

### *Aim*

i) **Vector Operations**

Write a MATLAB script that:

- Creates two vectors A=[1,2,3] and B=[4,5,6].
- Computes and displays the dot product and cross product of vectors ‘A’ and ‘B’.

ii) **Matrix Operations**

Write a MATLAB script that:

- Creates two matrices M1=[1,2;3,4] and M2=[5,6;7,8].
- Performs and displays the result of matrix multiplication M1×M2,inverse of M1,Transpose of M2,matrix concatenation, Determinant of M2

### *Theoretical Background*

Matrix in MATLAB

A matrix in MATLAB is a two-dimensional array of numbers where each element is identified by its row and column index. MATLAB is particularly designed to handle matrices and matrix operations efficiently, making it a powerful tool for numerical computations.

*Matrix Representation*

- **Row Vector**: A 1xN matrix where N is the number of columns.
- **Column Vector**: An Nx1 matrix where N is the number of rows.
- **Square Matrix**: An NxN matrix with an equal number of rows and columns.
- **Diagonal Matrix**: A square matrix where only the diagonal elements (from the top left to the bottom right) are non-zero.
- **Identity Matrix**: A special diagonal matrix where all the diagonal elements are 1, denoted as I.

In MATLAB, matrices are defined using square brackets [ ], with semicolons ; separating the rows. For example:

```
>>A = [1 2 3; 4 5 6; 7 8 9];
```

This defines a 3x3 matrix A.

*Difference Between Vector and Scalar*

- **Scalar**: A single number, often representing magnitude or quantity, and is a 1x1 matrix in MATLAB. For example, a = 5;.
- **Vector**: A one-dimensional array of numbers. A vector can be either a row vector (1xN) or a column vector (Nx1) in MATLAB. For example:
    - Row Vector: v_row = [1 2 3];
    - Column Vector: v_col = [1; 2; 3];

*Vector Operations in MATLAB*

- **Addition and Subtraction**: Element-wise addition or subtraction of vectors of the same size.

```
>> v1 = [1 2 3];
   v2 = [4 5 6];
   result = v1 + v2;  % Result: [5 7 9]
```

- **Scalar Multiplication**: Multiplying each element of a vector by a scalar.

```
>> v = [1 2 3];
   scalar = 2;
   result = scalar * v;  % Result: [2 4 6]
```

- **Dot Product**: The sum of the products of corresponding elements of two vectors.

```
>> v1 = [1 2 3];
   v2 = [4 5 6];
   dot_product = dot(v1, v2);  % Result: 32
```

- **Cross Product**: A vector perpendicular to two vectors in three-dimensional space.

```
>> v1 = [1 2 3];
   v2 = [4 5 6];
   cross_product = cross(v1, v2);  % Result: [-3 6 -3]
```

*Matrix Functions in MATLAB*

- **Transpose (')**: Transposes a matrix, swapping its rows and columns.

```
>> A = [1 2 3; 4 5 6];
   A_transpose = A';  % Result: [1 4; 2 5; 3 6]
```

- **Inverse (inv)**: Computes the inverse of a square matrix.

```
>> A = [1 2; 3 4];
   A_inv = inv(A);
```

- **Determinant (det)**: Computes the determinant of a square matrix.
```
>> A = [1 2; 3 4];
    determinant = det(A);  % Result: -2
```

- **Eigenvalues and Eigenvectors (eig)**: Computes the eigenvalues and eigenvectors of a square matrix.
```
>> A = [1 2; 2 1];
    [V, D] = eig(A);  % V contains eigenvectors, D contains eigenvalues
```

- **Matrix Multiplication (*)**: Multiplies two matrices if their dimensions are compatible.
```
>> A = [1 2; 3 4];
    B = [5 6; 7 8];
    C = A * B;  % Result: [19 22; 43 50]
```

- **Element-wise Operations (.*, ./, .^)**: Perform element-wise multiplication, division, or exponentiation.
```
>> A = [1 2 3];
     B = [4 5 6];
     C = A .* B;  % Element-wise multiplication, Result: [4 10 18]
```

*Types of Matrices*

- **Zero Matrix:** A matrix filled entirely with zeros.
```
zeromatrix = zeros(3)
zeromatrix =
    0    0    0
    0    0    0
    0    0    0
```

- **Ones Matrix:** A matrix filled entirely with ones.
```
onematrix = ones(3)
onematrix =
    1    1    1
    1    1    1
    1    1    1
```

- **Size and Length of Matrix:** Size returns the dimensions of a matrix as [rows, columns] while Length returns the size of the largest dimension of the matrix..

  ```
  size(onematrix)
  ans =
     3   2

  length(onematrix)
  ans =
     3
  ```

- **Identity Matrix:** A square matrix with ones on the diagonal and zeros elsewhere.

  ```
  identity = eye(3)
  identity =
     1   0   0
     0   1   0
     0   0   1
  ```

- **Matrix Rank:** The number of linearly independent rows or columns in a matrix.

  ```
  rank(newmatrix)
  ans =
     2
  ```

- **Upper and Lower Triangular Matrices:** A matrix where all the elements below the main diagonal are zero is called Upper triangular matrix. A matrix where all the elements above the main diagonal are zero is called Lower triangular matrix

  ```
  triu(dotprod)
  ans =
     1   2   3
     0   4   6
     0   0   9

  tril(dotprod)
  ans =
     1   0   0
     2   4   0
     3   6   9
  ```

- **Diagonal Matrix:** A matrix with non-zero elements only on the main diagonal.
```
diag([1 2 3])
ans =
    1    0    0
    0    2    0
    0    0    3
```

- **Matrix Reshaping:** Changing the dimensions of a matrix without altering its elements.
```
shape = 1:10;
reshape(shape, [5,2])
ans =
    1    6
    2    7
    3    8
    4    9
    5   10
```

***Code***

**i)**

```
A = [1 2 3];
B = [4 5 6];

dotproduct = A .* B
dotproduct =
    4    10    18

crossproduct = cross(A, B)
crossproduct =
   -3     6    -3
```

**ii)**

```
M1 = [1 2; 3 4];
M2 = [5 6; 7 8];

product = M1*M2
product =
   19    22
   43    50
```

```
inverse = inv(M1)
inverse =
  -2.0000    1.0000
   1.5000   -0.5000

transpose = M2'
transpose =
    5     7
    6     8
concat = [M1, M2]
concat =
    1     2     5     6
    3     4     7     8


determinant = det(M2)
determinant = -2.0000

diary off
```

## ***Conclusion***

This experiment demonstrates the creation and manipulation of various types of matrices in MATLAB, including basic operations on vectors and matrices, as well as specific matrix functions such as identity, diagonal, and triangular matrices. All operations were successfully executed.