
```
clear all;
close all;
clc;
%2401030116
%Nishika
%Batch-B12
%postlab

%Exercise 1: Matrix Creation

matA = randi([-5 5],4,4);
disp('Matrix A:');
disp(matA);

diagB = diag(diag(matA));
disp('Diagonal Matrix B:');
disp(diagB);

upperMat = triu(matA);
lowerMat = tril(matA);

disp('Upper Triangular Matrix:');
disp(upperMat);

disp('Lower Triangular Matrix:');
disp(lowerMat);

% Exercise 2: Matrix Arithmetic and Logic

X = rand(3,3);
Y = rand(3,3);

disp('Matrix X:');
disp(X);
disp('Matrix Y:');
disp(Y);

mulMat = X * Y;
disp('Matrix Multiplication X*Y:');
disp(mulMat);

elemMul = X .* Y;
disp('Element-wise Multiplication X.*Y:');
disp(elemMul);

greaterXY = X > Y;
disp('X greater than Y (Logical Matrix):');
disp(greaterXY);

% Exercise 3: Vectorized Computations
```

```
x = -2:0.1:2;
y = x.^3 - 4*x + 1;

[maxVal, maxIndex] = max(y);
[minVal, minIndex] = min(y);

disp('Maximum value of y and its index:');
disp(maxVal);
disp(maxIndex);

disp('Minimum value of y and its index:');
disp(minVal);
disp(minIndex);

% Exercise 4: Matrix Indexing and Modification

M = randi([1 20],5,5);
disp('Original Matrix M:');
disp(M);

M(mod(M,2)==0) = 0;
disp('After replacing even elements with zero:');
disp(M);

M(M>10) = sqrt(M(M>10));
disp('After replacing elements greater than 10:');
disp(M);

% Exercise 5: Linear System of Equations

A = [3 2 -1; 2 -2 4; -1 0.5 -1];
B = [1; -2; 0];

X = A\B;
disp('Solution of AX = B:');
disp(X);

disp('Verification (A*X):');
disp(A*X);

% Exercise 6: Loop vs Vectorization
N = 100000;

tic;
sumLoop = 0;
for i = 1:N
    sumLoop = sumLoop + i^2;
end
timeLoop = toc;

tic;
sumVec = sum((1:N).^2);
timeVec = toc;
```

```

disp('Sum using for-loop:');
disp(sumLoop);
disp('Time using for-loop:');
disp(timeLoop);

disp('Sum using vectorization:');
disp(sumVec);
disp('Time using vectorization:');
disp(timeVec);

% Exercise 7: User-Defined Function (same file)

data = randi(50,1,10);
[mn, sd, mx, mnm] = statsCalc(data);

disp('Data Vector:');
disp(data);
disp('Mean:');
disp(mn);
disp('Standard Deviation:');
disp(sd);
disp('Maximum Value:');
disp(mx);
disp('Minimum Value:');
disp(mnm);

% Exercise 8: Conditional Logic

nums = randi([-10 10],1,20);

positiveCount = sum(nums > 0);
negativeCount = sum(nums < 0);
zeroCount = sum(nums == 0);

disp('Generated Numbers:');
disp(nums);
disp('Positive count:');
disp(positiveCount);
disp('Negative count:');
disp(negativeCount);
disp('Zero count:');
disp(zeroCount);

% Exercise 9: Plotting and Analysis
x = 0:0.01:5;
y = exp(-x).*sin(2*pi*x);

figure;
plot(x,y,'b','LineWidth',2);
grid on;
title('P2401030116_y = exp(-x) sin(2\pi x)');
xlabel('x');
ylabel('y');
legend('y = exp(-x) sin(2\pi x)');

```

```

[maxY, idx] = max(y);
hold on;
plot(x(idx), maxY, 'ro', 'MarkerSize', 8);
hold off;

% Exercise 10: Mini Challenge

randMat = randi([1 9], 6, 6);
disp('Original Matrix:');
disp(randMat);

if isequal(randMat, randMat')
    disp('Matrix is symmetric');
    symMat = randMat;
else
    disp('Matrix is not symmetric');
    symMat = (randMat + randMat')/2;
end

disp('Symmetric Matrix:');
disp(symMat);

% Function Definition

function [meanVal, stdVal, maxVal, minVal] = statsCalc(v)
meanVal = mean(v);
stdVal = std(v);
maxVal = max(v);
minVal = min(v);
end

```

Matrix A:

| | | | |
|----|----|----|----|
| 5 | -4 | -3 | -1 |
| 2 | 1 | -1 | 5 |
| 1 | 5 | -5 | -1 |
| -2 | -4 | 2 | 1 |

Diagonal Matrix B:

| | | | |
|---|---|----|---|
| 5 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 0 | -5 | 0 |
| 0 | 0 | 0 | 1 |

Upper Triangular Matrix:

| | | | |
|---|----|----|----|
| 5 | -4 | -3 | -1 |
| 0 | 1 | -1 | 5 |
| 0 | 0 | -5 | -1 |
| 0 | 0 | 0 | 1 |

Lower Triangular Matrix:

| | | | |
|---|---|----|---|
| 5 | 0 | 0 | 0 |
| 2 | 1 | 0 | 0 |
| 1 | 5 | -5 | 0 |

```

    -2    -4     2     1

Matrix X:
    0.1544    0.7581    0.6855
    0.3813    0.8711    0.2941
    0.1611    0.3508    0.5306

Matrix Y:
    0.8324    0.2992    0.3596
    0.5975    0.4526    0.5583
    0.3353    0.4226    0.7425

Matrix Multiplication X*Y:
    0.8113    0.6790    0.9878
    0.9366    0.6327    0.8419
    0.5216    0.4312    0.6478

Element-wise Multiplication X.*Y:
    0.1285    0.2268    0.2465
    0.2279    0.3943    0.1642
    0.0540    0.1483    0.3940

X greater than Y (Logical Matrix):
    0     1     1
    0     1     0
    0     0     0

Maximum value of y and its index:
    4.0720

    9

Minimum value of y and its index:
   -2.0720

   33

Original Matrix M:
    9     7     5     3    17
    9    14    16    14     4
    3    20    16    10     4
    1    19    15     5    14
    6    10    15     2    18

After replacing even elements with zero:
    9     7     5     3    17
    9     0     0     0     0
    3     0     0     0     0
    1    19    15     5     0
    0     0    15     0     0

After replacing elements greater than 10:
    9.0000    7.0000    5.0000    3.0000    4.1231
    9.0000         0         0         0         0

```

| | | | | |
|--------|--------|--------|--------|---|
| 3.0000 | 0 | 0 | 0 | 0 |
| 1.0000 | 4.3589 | 3.8730 | 5.0000 | 0 |
| 0 | 0 | 3.8730 | 0 | 0 |

Solution of AX = B:

1.0000
-2.0000
-2.0000

*Verification (A*X):*

1.0000
-2.0000
-0.0000

Sum using for-loop:

3.3334e+14

Time using for-loop:

0.0084

Sum using vectorization:

3.3334e+14

Time using vectorization:

4.7400e-04

Data Vector:

26 36 8 48 28 34 2 41 38 7

Mean:

26.8000

Standard Deviation:

15.9011

Maximum Value:

48

Minimum Value:

2

Generated Numbers:

Columns 1 through 13

1 -4 1 -2 -2 -7 -5 -10 9 3 9 -7 9

Columns 14 through 20

6 2 -1 -5 5 -6 -9

Positive count:

9

Negative count:

Zero count:

0

Original Matrix:

| | | | | | |
|---|---|---|---|---|---|
| 7 | 8 | 6 | 4 | 2 | 8 |
| 7 | 3 | 9 | 9 | 1 | 7 |
| 7 | 8 | 4 | 3 | 4 | 6 |
| 6 | 8 | 1 | 6 | 2 | 2 |
| 4 | 8 | 8 | 6 | 7 | 9 |
| 4 | 5 | 6 | 4 | 4 | 3 |

Matrix is not symmetric

Symmetric Matrix:

| | | | | | |
|--------|--------|--------|--------|--------|--------|
| 7.0000 | 7.5000 | 6.5000 | 5.0000 | 3.0000 | 6.0000 |
| 7.5000 | 3.0000 | 8.5000 | 8.5000 | 4.5000 | 6.0000 |
| 6.5000 | 8.5000 | 4.0000 | 2.0000 | 6.0000 | 6.0000 |
| 5.0000 | 8.5000 | 2.0000 | 6.0000 | 4.0000 | 3.0000 |
| 3.0000 | 4.5000 | 6.0000 | 4.0000 | 7.0000 | 6.5000 |
| 6.0000 | 6.0000 | 6.0000 | 3.0000 | 6.5000 | 3.0000 |

