```
%%Given a mxn matrix A, write a script that sorts every row separately in ascending ↙
order

clc
A2 = randi(100,7,6);  %Generates a matrix of random integers between 1 and 100 ↙
inclusive with 7 rows and 6 columns
B2 = sort(A2, 2);        %Sorts the rows in the matrix in ascending order
disp('The matrix after sorting each row is : ');
disp(B2);

b2 = sort(A2, 1);
disp('The matrix after sorting each column is :');
disp(b2);

B2d = sort(A2, 2, "descend");
disp('The matrix after sorting each row in descending order is :');
disp(B2d);

b2d = sort(A2, 2, "descend");
disp('The matrix after sorting each column in descending order is :');
disp(b2d);

%% Given a mxn matrix A3, write a script that returns the matrix B3 made of rows ↙
whose means are smaller of the overall matrix mean of A3
%first claculate the overall mean
%Then calculate the mean of all rows
%Find the rows whose mean is less than that of the matrix
%Display the matrix B of rows whose mean is less
clc
A = randi(200, 8, 8);
meanOverall = mean(A, "all"); %Computes the mean of all the elments in the matrix
meanRows = mean(A, 2);        %Computes the mean of all rows
rowslessOverallMean = meanRows < meanOverall;
B = A(rowslessOverallMean, :);
disp('The matrix made of rows whose mean is less than the overall mean is :');
disp(B);


%% Given a mxn matxix A made of integers, write a script that returns the matrix made ↙
of rows of an integer matrix A such that every row returned has a multiple of 7 and a ↙
multiple of 5
%First iterate each row of the matrix
%Check each row to see if it contains at least one multiple of 5 and one
%multiple of 7
%Store rows that satisfy both conditions in a new matrix
clc
A4 = [105, 89, 86, 45, 76; 108, 92, 88, 90, 10; 77, 65, 100, 76, 45; 102, 52, 53, 19, ↙
2; 90, 89, 35, 10, 400];
[m, num] = size(A4);
```

```
resultmatrix = [];
for i = 1:m                             %iterates
    row = A4(i, :);
    hasMul5 = any(mod(row, 5) == 0);
    hasMul7 = any(mod(row, 7) == 0);

    if hasMul5 && hasMul7
        resultmatrix = [resultmatrix; row];
    end
end
disp(A4);
disp('The matrix with rows containing multiples of both 5 and 7 is :')
disp(resultmatrix);

%% Given the matrices A and B above, define C = B(:, 1:8), write matlab code to
% Cut the matrix in half by rows and columns to get the partition
% A = [A11 A12; A21 A22], then return the transpose A = [A11 A21; A12 A22].
% Find out if it is a symmetric matrix

clc
[m, n] = size(A);

%Identify matrices A and B
A;
B;
%Create a matrix C from B that contains the first 8 columns of B
C= B(:, 1:8);
disp('C');
disp(C)

%Cut the matrix in half both by rows abd columns to obtain the submatrices
%A11, A12, A21 and A22
m2 = floor(m/2);
n2 = floor(n/2);

A11 = C(1:m2, 1:n2);          %Top-left
A12 = C(1:m2, n2+1:end);      %Top-right
A21 = C(m2+1:end, 1:n2);      %Bottom-left
A22 = C(m2+1:end, n2+1:end); %Bottom-right

%Construct the matrix A from these submatrices
A5 = [A11 A21; A12 A22];     %Concatenates horizontally and then vertically

%Compute the transpose of A
A5_transpose = A5';

%Check if the resulting matrix is symmetric
is_symmetric = isequal(A5_transpose, A5);
```

```
disp('The matrix A5 is :')
disp(A5);
disp('The transposed matrix of A5 is :');
disp(A5_transpose);
disp(['Is the transposed matrix symmetric? ', num2str(is_symmetric)]);
%% REPEAT FOR C

clc

%Determine the size of the matrix C
[mC, nC] = size(C);

%Cut the matrix in half both by rows and columns
mC2 = floor(mC/2);
nC2 = floor(nC/2);

%Extraction of the submatrices
AC11 = C(1:mC2, 1:nC2);         %Top-left
AC12 = C(1:mC2, nC2+1:end);     %Top-right
AC21 = C(mC2+1:end, 1:nC2);     %Bottom-left
AC22 = C(mC2+1:end, nC2+1:end); %Bottom-right

%The matrix from concatenation of the submatrices
AC = [AC11 AC21; AC12 AC22];

%The transpose of AC
AC_transpose = AC';

%Check for symmetry
is_symmetric = isequal(AC_transpose, A);

disp('The matrix AC is :')
disp(AC);
disp('The transposed matrix of AC is :');
disp(AC_transpose);
disp(['Is the transposed matrix symmetric? ', num2str(is_symmetric)]);


%% Consiider the following Matrix A6 constructed using A6=(hilb(n)).^0.1.
%Plot the following (using subplot command to generate 6 plots in one
%figure

clc

%Define the matrix A6
A6 = (hilb(20)).^0.1

%Extract the third row
row3 = A6(3, :)
```

```
figure

%Plot the third row of A6 for n=20
subplot(3, 1, 1);
plot(row3);
title('Plot 1: Third row of A6');
xlabel('Column Index');
ylabel('Value');
grid on;


%Plot the third row of A6 verses the second column of A6
%Extract second column
col2 = A6(:, 2);

%Plot
subplot(3, 1, 2);
plot(row3, col2);
title('Plot 2: 3rd row vs 2nd column')
xlabel('3rd Row');
ylabel('2nd Column');
grid on;

%Plot the 1st, 3rd and 5th rows of A6 verses the 2nd Column of A6
%Extract the remaining rows
row1 = A6(1, :);
row5 = A6(6, :);

%Plot
subplot(3, 1, 3);
plot(row1, col2, row3, col2, row5, col2);
title('Plot 3: 1st, 3rd and 5th rows vs 2nd column');
xlabel('Rows');
ylabel('2nd Column');
grid on;

%% Consider the following matrix A7 constructed using A7=(hilb(n).0.1
clc
A7 = (hilb(n)).^0.1;    %Define A7
n = 20;                %Initialize the value of n
disp(A7);

%Compute the singular values of A7
singularValues = svd(A7);


%Plot the singular values
subplot(3, 2, 1);
```

```
plot(singularValues);
ylabel('Singular values');
xlabel('Index');
title('Plot 1 : Singular values of A7');
grid on

%Plot the third subdiagonal of A
subDiagonal3 = diag(A7, -3);
subplot(3, 2, 2);
plot(subDiagonal3);
title('Plot 2: Third Sub-diagonal');
xlabel('Index');
ylabel('Sub-diagonal Elements');
grid on;

%Plot the lower left 4x4 sublock of A7 made into a column vector

%Extract the lower left sublock
lowerLeftSublock = A7(end-3:end, 1:4);  %Selects the last 4 rows and last 4 columns

%Reshape the sublock into a vector
sublockVector = reshape(lowerLeftSublock, [], 1);

%Plot
subplot(3, 2, 3);
plot(sublockVector);
title('Plot 3: Column vector');
xlabel('Index');
ylabel('Column Vector');
grid on;

%Plot the pivots used by the LU decomposition

%Perform LU Decomposition
%LU decomposition factors a matrix A7 into a lower triangular matrix L and
%an upper triangular matrix U such that A7=P*L*U where P is a permutation
%matrix. The pivots are the diagonal elements of the U matrix after
%decomposition of the rows that are swapped due to the permutation matrix P
[L, U, P] = lu(A7);

%Extract the diagonal pivots from the matrix U
pivots = diag(U);

%Plot the pivots
subplot(3, 2, 4);
plot(pivots);
title('Plot 4: Pivots from LU Decomposition');
xlabel('Index');
ylabel('Pivots');
```

```
grid on;

%Plot the maximum element of evry row
%Find the maximum element of each row using the max funtion
maxElements = max(A7, [], 2)

%Plot the values
subplot(3, 2, 5);
plot(maxElements);
title('Plot 5: Maximum elements in each row');
xlabel('Index');
ylabel('MaxElements');
grid on;

%Plot the maximum element of the ith row vs. the minimum element of the ith
%column

%Find the minimum elements in each row
minElementRow = min(A7, [], 2);

%Find the minimum elements in each column
minElementCol = min(A7, [], 1);

%Convert the minElementCol to a column vector
minElementCol = minElementCol';

%Plot minElementRow vs minElementCol
subplot(3, 2, 6);
plot(minElementRow, minElementCol);
title('Plot 6 : minimum elements in each row vs column');
xlabel('minElementRow');
ylabel('minElementCol');
grid on;


%% Consider the following matrix A8 constructed using A8=(hilb(n).0.1
clc

%Define the matrix
A8 = (hilb(30)).^0.1;    %Define A8
[m, n] = size(A8);

%Generate the grid for the x and y coordinates
[X, Y] = meshgrid(1:n, 1:m);

%Plot the surface
subplot(2, 2, 1);
surf(X, Y, A8);
```

```
title('3D surface plot of Matrix A8');
xlabel('Column Index');
ylabel('Row Index');
zlabel('Matrix Value');
colorbar;                  %Shows the scale of values
%% SPARSITY PATTERN
%Plot the sparsity pattern of all elemts less that
sparsePattern = A8 < 0.2;
A9 = rand(8, 9);
sparsePattern2 = A9 < 0.2

subplot(2, 2, 2);
spy(sparsePattern2);
title('Sparsity Pattern of Elements in A9 less than 0.2');
xlabel('Column index');
ylabel('Row Index');
grid on;

%% Plot the sparsity pattern of all elements between 0.45 and 0.55

sparsePattern3= (A9 > 0.45) & (A9 < 0.55)

subplot(2, 2, 3);
spy(sparsePattern3);
title('Sparsity Pattern of Elements in A9 between 0.45 and 0.55');
xlabel('Column index');
ylabel('Row Index');
grid on;

%% Plot the histogram of all elements

matrixElements = A8(:);

subplot(2, 2, 4);
histogram(matrixElements);
title('Histogram of all elements in matrix A8');
xlabel('Column index');
ylabel('Row Index');
grid on;

%% RESHAPE
clc
A12 = 1:20;
B12 = reshape(A12, [5,4]);

A13 = magic(4);;
B13 = reshape(A13, [], 2);  %Reshapes the matrix A13 into a matrix that has 2 columns ↙
[] give for the first dimension to let reshape automatically calculate the ↙
appropriate number of rows
```

```
%Reshape multidimensional array into matrix
rng default  %Resets the random number generator setting to worker default values
A14 = rand(2,2,3);  %Generates a 3D array A14 of random numbers between 0 and 1,↙
2rows, 2columns and 3 number of slices/layers
B14 = reshape(A14, 6,2);

%% FLIP - Flips order of elements
clc
A15 = randi(100, 6, 6)
B15 = flip(A15)   %returns an array the same size as A15 but with the order of↙
elements reversed
```