**Report**

Subject: Determinant of any square matrix of height

1-else if (type == CHMAT\_INT) {

printf("%d ", id[y \* X + x]);

matrix[y][x] = id[y \* X + x]

CHMAT\_INT the 2D array, the elements of any square matrix of size that we create with Chmat in the control of the container. For this reason, we prepare the int matrix[30][30] parameter of the function to get the correct values.

2-else if (type == CHMAT\_DOUBLE) {

printf("%d ", dd[y \* X + x]);

matrix[y][x] = dd[y \* X + x];

}

CHMAT\_DOUBLE the 2D array, the elements of any square matrix of size that we create with Chmat in the control of the container. For this reason, we prepare the int matrix[30][30] parameter of the function to get the correct values.

3- else if (type == CHMAT\_FLOAT) {

printf("%d ", fd[y \* X + x]);

matrix[y][x] = fd[y \* X + x];

}

CHMAT\_FLOAT the 2D array, the elements of any square matrix of size that we create with Chmat in the control of the container. For this reason, we prepare the int matrix[30][30] parameter of the function to get the correct values.

4- else if (type == CHMAT\_UNSIGNED\_INT) {

printf("%d ", uid[y \* X + x]);

matrix[y][x] = uid[y \* X + x];

}

}

CHMAT\_FLOAT the 2D array, the elements of any square matrix of size that we create with Chmat in the control of the container. For this reason, we prepare the int matrix[30][30] parameter of the function to get the correct values.

5- int CHMAT::det\_matrix(int matrix[30][30],int n)

det\_matrix function calculates the determinant of any square matrix of size 30 or less.

The first parameter of the function contains a 2D array with elements of the matrix. The second parameter of the function contains the size of the matrix.

6- int det = 0;

The det integer variable will give the determinant result of the matrix. The first value is 0 because it is a determinedant value calculated at each stage and a determinant value from a child recursive function.

7- int submatrix[30][30];

Contains the array in which the elements to be processed are kept when calculating determinants within the elements of the matrix.

8- if (n == 2)

If the Matrix size is 2x2, it enters this condition for calculation.

9- return ((matrix[0][0] \* matrix[1][1]) - (matrix[1][0] \* matrix[0][1]));

Calculates the determinant of the 2x2-dimensional matrix.

10- else {

The condition used to calculate the determinant of matrices of more than 2x2 sizes.

11- for (int x = 0; x < n; x++) {

When calculating the matrix with a size of 3x3, determinant calculation is made based on the elements in the first line according to our code. In other words, since the values of the elements in the first row and column will be deleted and the matrix of the remaining elements will be calculated, the function must return as much as the Matrix size in order to move the elements in the entire first row.It is calculated in the same way in those that are larger than the 3x3-dimensional matrix.

12- int subi = 0;

The initial value of the subi must be 0 so that the elements to be processed in the calculation of the matrix can start from line 1.

13- for (int i = 1; i < n; i++){

This for loop has been opened to select the elements to be processed on the line.

14- int subj = 0;

The initial value of the subj must be 0 so that the elements to be processed in the calculation of the matrix can start from column 1.

15- for (int j = 0; j < n; j++) {

This for loop has been opened to select the elements to be processed in the column.

16- if (j == x), continue;

j==x equality contains elements that will not be processed in the determinant account, so it is passed by skip with continue and is not written into the submatrix.

17- submatrix[subi][subj] = matrix[i][j];

The elements to be processed in the determinant account are thrown into the submatrix

18- det = det + (pow(-1, x) \* matrix[0][x] \* det\_matrix(submatrix, n - 1));

Calculates the determinant of the matrix with a size of more than 2x2. Determinant is recursive because determinant is calculated first for one short size when calculating.

19- return det;

In the global determinant account, determinants of a lower size are returned to be kept and added to the determinant account.