The *longest diagonals* of a square matrix are defined as follows:

* the first *longest diagonal* goes from the top left corner to the bottom right one;
* the second *longest diagonal* goes from the top right corner to the bottom left one.

Given a square matrix, your task is to swap its *longest diagonals* by exchanging their elements at the corresponding positions.

**Example**

For

matrix = [[1, 2, 3],

[4, 5, 6],

[7, 8, 9]]

the output should be

swapDiagonals(matrix) = [[3, 2, 1],

[4, 5, 6],

[9, 8, 7]]

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] array.array.integer matrix**

*Constraints:*  
1 ≤ matrix.length ≤ 10,  
matrix.length = matrix[i].length,  
1 ≤ matrix[i][j] ≤ 1000.

* **[output] array.array.integer**

Matrix with swapped diagonals.

<https://codefights.com/arcade/code-arcade/list-backwoods/kfwTN8pdx9uP8bcHS>

static int[][] reverseOnDiagonals(int[][] matrix)

{

List<int> d1 = new List<int>();

for (int i = 0; i < matrix.Length; i++)

d1.Add(matrix[i][i]);

int f = 0, c = matrix[0].Length - 1;

List<int> d2 = new List<int>();

for (int i = 0; i < matrix.Length; i++)

d2.Add(matrix[f++][c--]);

for (int i = 0; i < matrix.Length; i++)

matrix[i][i] = d2[i];

f = 0; c = matrix[0].Length - 1;

for (int i = 0; i < matrix.Length; i++)

matrix[f++][c--] = d1[i];

return matrix;

}