



Full Name: Giovanni Acireale  
Email: gio.acireale@gmail.com  
Test Name: Mock Test  
Taken On: 21 Jan 2025 00:15:17 IST  
Time Taken: 9 min 24 sec/ 24 min  
Invited by: Ankush  
Invited on: 21 Jan 2025 00:14:59 IST

Skills Score:

Tags Score:

- Algorithms 90/90
- Constructive Algorithms 90/90
- Core CS 90/90
- Greedy Algorithms 90/90
- Medium 90/90
- Problem Solving 90/90
- problem-solving 90/90

100%

90/90

scored in **Mock Test** in 9 min 24 sec on 21 Jan 2025 00:15:17 IST

Recruiter/Team Comments:

No Comments.

Plagiarism flagged

We have marked questions with suspected plagiarism below. Please review it in detail here -

	Question Description	Time Taken	Score	Status
Q1	Flipping the Matrix > Coding	9 min 14 sec	90/ 90	!

QUESTION 1



Needs Review

Score 90

Flipping the Matrix > Coding Algorithms Medium Greedy Algorithms Constructive Algorithms problem-solving Core CS Problem Solving

QUESTION DESCRIPTION

Sean invented a game involving a  $2n \times 2n$  matrix where each cell of the matrix contains an integer. He can reverse any of its rows or columns any number of times. The goal of the game is to maximize the sum of the elements in the  $n \times n$  submatrix located in the upper-left quadrant of the matrix.

Given the initial configurations for  $q$  matrices, help Sean reverse the rows and columns of each matrix in the best possible way so that the sum of the elements in the matrix's upper-left quadrant is maximal.

### Example

$matrix = [[1, 2], [3, 4]]$

```
1 2
3 4
```

It is  $2 \times 2$  and we want to maximize the top left quadrant, a  $1 \times 1$  matrix. Reverse row 1:

```
1 2
4 3
```

And now reverse column 0:

```
4 2
1 3
```

The maximal sum is **4**.

### Function Description

Complete the `flippingMatrix` function in the editor below.

`flippingMatrix` has the following parameters:

- `int matrix[2n][2n]`: a 2-dimensional array of integers

### Returns

- `int`: the maximum sum possible.

### Input Format

The first line contains an integer  $q$ , the number of queries.

The next  $q$  sets of lines are in the following format:

- The first line of each query contains an integer,  $n$ .
- Each of the next  $2n$  lines contains  $2n$  space-separated integers  $matrix[i][j]$  in row  $i$  of the matrix.

### Constraints

- $1 \leq q \leq 16$
- $1 \leq n \leq 128$
- $0 \leq matrix[i][j] \leq 4096$ , where  $0 \leq i, j < 2n$ .

### Sample Input

STDIN	Function
-----	-----
1	q = 1
2	n = 2
112 42 83 119	matrix = [[112, 42, 83, 119], [56, 125, 56, 49], \
56 125 56 49	[15, 78, 101, 43], [62, 98, 114, 108]]
15 78 101 43	
62 98 114 108	

### Sample Output

```
414
```

### Explanation

Start out with the following  $2n \times 2n$  matrix:

$$matrix = \begin{bmatrix} 112 & 42 & 83 & 119 \\ 56 & 125 & 56 & 49 \\ 15 & 78 & 101 & 43 \\ 62 & 98 & 114 & 108 \end{bmatrix}$$

Perform the following operations to maximize the sum of the  $n \times n$  submatrix in the upper-left quadrant:

2. Reverse column 2 ( $[83, 56, 101, 114] \rightarrow [114, 101, 56, 83]$ ), resulting in the matrix:

$$matrix = \begin{bmatrix} 112 & 42 & 114 & 119 \\ 56 & 125 & 101 & 49 \\ 15 & 78 & 56 & 43 \\ 62 & 98 & 83 & 108 \end{bmatrix}$$

3. Reverse row 0 ( $[112, 42, 114, 119] \rightarrow [119, 114, 42, 112]$ ), resulting in the matrix:

$$matrix = \begin{bmatrix} 119 & 114 & 42 & 112 \\ 56 & 125 & 101 & 49 \\ 15 & 78 & 56 & 43 \\ 62 & 98 & 83 & 108 \end{bmatrix}$$

The sum of values in the  $n \times n$  submatrix in the upper-left quadrant is  $119 + 114 + 56 + 125 = 414$ .

#### CANDIDATE ANSWER

Language used: C#

```

1
2 class Result
3 {
4
5     /*
6      * Complete the 'flippingMatrix' function below.
7      *
8      * The function is expected to return an INTEGER.
9      * The function accepts 2D_INTEGER_ARRAY matrix as parameter.
10     */
11
12     public static int flippingMatrix(List<List<int>>> matrix)
13     {
14         // given a 2n x 2n matrix, where each cell contains an integer, we
15         can perform one of two operations on the matrix:
16         // 1. Choose a 2 x 2 submatrix and rotate it 90 degrees clockwise
17         // 2. Choose a 2 x 2 submatrix and rotate it 90 degrees
18         counterclockwise
19         // we want to find the maximum sum of the integers in the matrix
20         after performing the operations
21         // we can do this by iterating through the matrix and selecting the
22         maximum value from each pair of corresponding elements
23         // then summing the maximum values
24
25         int n = matrix.Count / 2; // the number of rows in the matrix
26         int sum = 0; // initialize the sum
27
28         for (int i = 0; i < n; i++)
29         {
30             // i is the row index of the submatrix
31             for (int j = 0; j < n; j++)
32             {
33                 // j is the column index of the submatrix

```

```

34         // We are in.
35         // select the maximum value from each pair of corresponding
36 elements
37         sum += Math.Max(Math.Max(matrix[i][j], matrix[i][2 * n - j -
38 1]), Math.Max(matrix[2 * n - i - 1][j], matrix[2 * n - i - 1][2 * n - j -
39 1]));
40         // this is the maximum value from the four elements in the
41 submatrix
42         // the four elements are matrix[i][j], matrix[i][2 * n - j -
43 1], matrix[2 * n - i - 1][j], matrix[2 * n - i - 1][2 * n - j - 1]
         // 2 * n - j - 1 is the column index of the element in the
         same row but in the second half of the matrix
         // 2 * n - i - 1 is the row index of the element in the same
         column but in the second half of the matrix
     }
 }

 return sum;
}

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✔ Success	0	0.0504 sec	24.3 KB
Testcase 2	Easy	Hidden case	✔ Success	15	0.1086 sec	41.1 KB
Testcase 3	Easy	Hidden case	✔ Success	15	0.1136 sec	41.2 KB
Testcase 4	Easy	Hidden case	✔ Success	15	0.093 sec	38.6 KB
Testcase 5	Easy	Hidden case	✔ Success	15	0.1063 sec	41 KB
Testcase 6	Easy	Hidden case	✔ Success	15	0.1208 sec	41.1 KB
Testcase 7	Easy	Hidden case	✔ Success	15	0.1053 sec	41 KB
Testcase 8	Easy	Sample case	✔ Success	0	0.0718 sec	24.4 KB

No Comments