



Full Name: Kaviya Ramesh Email: kaviya09052004@gmail.com Test Name: **Mock Test** Taken On: 7 Aug 2025 19:04:34 IST 21 min 10 sec/ 40 min Time Taken: Invited by: Ankush Invited on: 7 Aug 2025 19:04:18 IST Skills Score: Tags Score: Algorithms 160/195 Constructive Algorithms 90/90 Core CS 160/195 Easy 70/105 Greedy Algorithms 90/90 Medium Problem Solving 160/195 70/105 Search Sorting 70/105

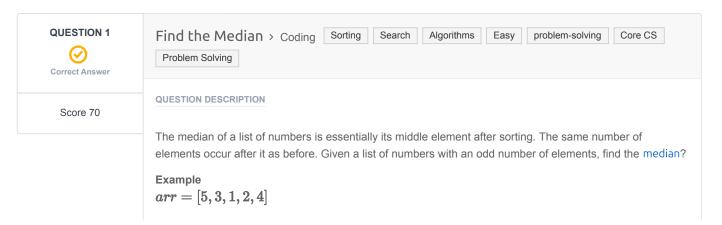
82.1% scored in Mock Test in 21 min 10 sec on 7 Aug 2025 19:04:34 IST

Recruiter/Team Comments:

No Comments.



problem-solving 160/195



The sorted array arr'=[1,2,3,4,5]. The middle element and the median is 3.

Function Description

Complete the *findMedian* function in the editor below.

findMedian has the following parameter(s):

• *int arr[n]:* an unsorted array of integers

Returns

• int: the median of the array

Input Format

The first line contains the integer n, the size of arr.

The second line contains n space-separated integers arr[i]

Constraints

- $1 \le n \le 1000001$
- **n** is odd
- $-10000 \le arr[i] \le 10000$

Sample Input 0

```
7
0 1 2 4 6 5 3
```

Sample Output 0

3

Explanation 0

The sorted arr = [0, 1, 2, 3, 4, 5, 6]. It's middle element is at arr[3] = 3.

CANDIDATE ANSWER

Language used: C

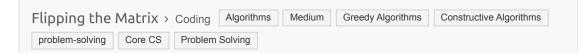
```
1 #include <stdio.h>
 void findMedian(int arr[], int n)
 3 {
 4
      for (int i=0; i < n; i++)
           for(int j=0; j< n; j++)
 8
                if(arr[j] < arr[j+1])</pre>
                    int temp=arr[j];
                    arr[j]=arr[j+1];
                    arr[j+1]=temp;
            }
       printf("%d",arr[n/2]);
18 }
19 int main()
20 {
      int n;
      scanf("%d",&n);
      int arr[n];
24
       for(int i=0;i<n;i++)
```

25	{						
26		scanf("%d'	',&arr[i]);				
27	}						
28	find	Median(arı	r,n);				
29	}						
	TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
	Testcase 1	Easy	Sample case	Success	0	0.0087 sec	7.25 KB
	Testcase 2	Easy	Hidden case	Success	35	0.0849 sec	7 KB
	Testcase 3	Easy	Hidden case	Success	35	0.1035 sec	7 KB
	Testcase 4	Easy	Hidden case	Terminated due timeout	to 0	2.0026 sec	7 KB
No	Comments						

QUESTION 2



Score 90



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

$$matrix = \left[[1,2],[3,4]\right]$$

- 1 2
- 3 4

It is 2×2 and we want to maximize the top left quadrant, a 1×1 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 ${\it flippingMatrix}$ 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, $oldsymbol{n}$.
- Each of the next 2n lines contains 2n space-separated integers matrix[i][j] in row i of the matrix.

Constraints

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

Sample Input

Sample Output

414

Explanation

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

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

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:

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

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

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

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

CANDIDATE ANSWER

```
1 #include <stdio.h>
2 int main()
3 {
 4
       int q;
       scanf("%d",&q);
      while(q--)
8
           int n;
          scanf("%d",&n);
          int size=2*n;
          int matrix[256][256];
           for(int i=0;i<size;i++)</pre>
14
               for(int j=0;j<size;j++)</pre>
                   scanf("%d", &matrix[i][j]);
           int sum=0;
           for (int i=0; i < n; i++)
               for(int j=0; j< n; j++)
                 int a=matrix[i][j];
                 int b=matrix[i][size-1-j];
                  int c=matrix[size-1-i][j];
                  int d=matrix[size-1-i][size-1-j];
                  int max=a;
                  if (b>max)
                   max=b;
                  if (c>max)
                   max=c;
                  if (d>max)
                   max=d;
                  }
41
                  sum+=max;
43
45
           printf("%d\n", sum);
47
       return 0;
48
49 }
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	Success	0	0.0078 sec	6.88 KB
Testcase 2	Easy	Hidden case	Success	15	0.0297 sec	7.13 KB
Testcase 3	Easy	Hidden case	Success	15	0.0497 sec	7.5 KB
Testcase 4	Easy	Hidden case	Success	15	0.0238 sec	7.5 KB
Testcase 5	Easy	Hidden case	Success	15	0.0314 sec	7.38 KB
Testcase 6	Easy	Hidden case	Success	15	0.0379 sec	7.5 KB

	Testcase 7	Easy	Hidden case	0	Success	15	0.04 sec	7.38 KB
	Testcase 8	Easy	Sample case	0	Success	0	0.0074 sec	7.38 KB
No	o Comments							

PDF generated at: 7 Aug 2025 13:58:34 UTC