

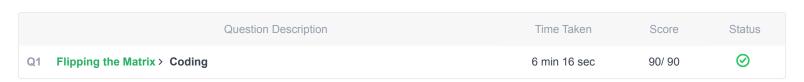
Mock Test > shaharas30@gmail.com

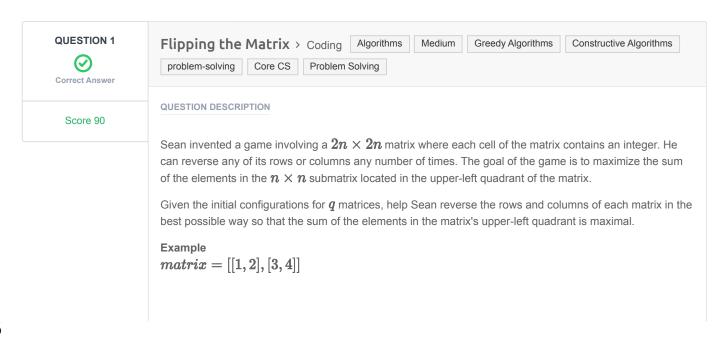
Full Name: Shahar Asher Email: shaharas30@gmail.com Test Name: **Mock Test** Taken On: 24 Nov 2024 16:14:16 IST Time Taken: 7 min 18 sec/ 24 min Invited by: Ankush Invited on: 24 Nov 2024 16:14:06 IST Skills Score: Tags Score: Algorithms 90/90 Constructive Algorithms 90/90 Core CS 90/90 Greedy Algorithms 90/90 Medium 90/90 90/90 Problem Solving 90/90 problem-solving

scored in **Mock Test** in 7 min 18 sec on 24 Nov 2024 16:14:16 IST

Recruiter/Team Comments:

No Comments.





```
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 *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

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

Sample Input

Sample Output

```
414
```

Explanation

Start out with the following 2n imes 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 imes n submatrix in the upper-left quadrant:

2. Reverse column 2 ([83, 56, 101, 114] ightarrow [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

Language used: C++14

```
1 #include <algorithm>
 2 #include <bits/stdc++.h>
4 using namespace std;
6 string ltrim(const string &);
7 string rtrim(const string &);
8 vector<string> split(const string &);
12 /*
   * Complete the 'flippingMatrix' function below.
   * The function is expected to return an INTEGER.
* The function accepts 2D INTEGER ARRAY matrix as parameter.
19 int flippingMatrix(vector<vector<int>> matrix) {
     int res = 0,
       c_r_len = matrix.size();
      for(size t r = 0; r<c r len/2; ++r)
           for(size t c = 0; c<c r len/2; ++c)
           {
               int temp sum = matrix[r][c];
               temp_sum = std::max(temp_sum, matrix[r][c_r_len-1-c]);
               temp_sum = std::max(temp_sum, matrix[c_r_len-1-r][c_r_len-1-c]);
              temp_sum = std::max(temp_sum, matrix[c_r_len-1-r][c]);
               res+=temp sum;
           }
       return res;
34 }
```

```
36 int main()
37 {
       ofstream fout(getenv("OUTPUT PATH"));
       string q_temp;
41
       getline(cin, q temp);
       int q = stoi(ltrim(rtrim(q temp)));
       for (int q itr = 0; q itr < q; q itr++) {
           string n temp;
47
           getline(cin, n temp);
          int n = stoi(ltrim(rtrim(n_temp)));
           vector<vector<int>> matrix(2 * n);
          for (int i = 0; i < 2 * n; i++) {
54
               matrix[i].resize(2 * n);
               string matrix row temp temp;
               getline(cin, matrix_row_temp_temp);
               vector<string> matrix row temp =
60 split(rtrim(matrix row temp temp));
               for (int j = 0; j < 2 * n; j++) {
                   int matrix_row_item = stoi(matrix_row_temp[j]);
                   matrix[i][j] = matrix row item;
               }
           }
           int result = flippingMatrix(matrix);
           fout << result << "\n";</pre>
       fout.close();
       return 0;
77 }
79 string ltrim(const string &str) {
       string s(str);
      s.erase(
           find if(s.begin(), s.end(), not1(ptr fun<int, int>(isspace)))
      );
      return s;
88 }
90 string rtrim(const string &str) {
      string s(str);
       s.erase(
           find if(s.rbegin(), s.rend(), not1(ptr fun<int, int>
95 (isspace))).base(),
           s.end()
       );
```

```
return s;
10 }
10 vector<string> split(const string &str) {
vector<string> tokens;
18
10
     string::size_type start = 0;
16
     string::size type end = 0;
16
10
     while ((end = str.find(" ", start)) != string::npos) {
10
          tokens.push_back(str.substr(start, end - start));
19
10
         start = end + 1;
     }
12
13
      tokens.push_back(str.substr(start));
14
15
     return tokens;
6 }
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	Success	0	0.0105 sec	8.9 KB
Testcase 2	Easy	Hidden case	Success	15	0.0554 sec	9.37 KB
Testcase 3	Easy	Hidden case	Success	15	0.0778 sec	9.27 KB
Testcase 4	Easy	Hidden case	Success	15	0.0377 sec	9.2 KB
Testcase 5	Easy	Hidden case	Success	15	0.0653 sec	9.34 KB
Testcase 6	Easy	Hidden case	Success	15	0.0725 sec	9.3 KB
Testcase 7	Easy	Hidden case	Success	15	0.087 sec	9.35 KB
Testcase 8	Easy	Sample case	Success	0	0.0085 sec	8.9 KB

PDF generated at: 24 Nov 2024 10:53:06 UTC