

**Full Name:** Dody Apriyanto S Email: dody.apriyanto@gmail.com Test Name: **Mock Test** 19 Jun 2022 18:52:06 IST Taken On: Time Taken: 22 min/ 24 min Invited by: Ankush 19 Jun 2022 18:51:53 IST Invited on: Skills Score: Tags Score: Algorithms 0/90 Constructive Algorithms 0/90 Core CS 0/90 Greedy Algorithms 0/90 Medium 0/90

Problem Solving

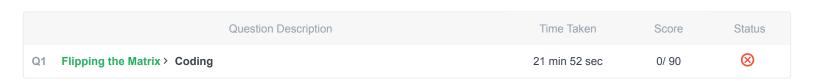
problem-solving

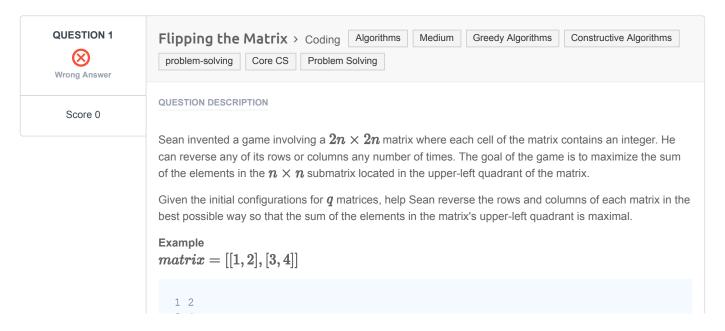
0/90



#### **Recruiter/Team Comments:**

No Comments.





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**

- $1 \le q \le 16$
- $1 \le n \le 128$
- $ullet 0 \leq matrix[i][j] \leq 4096$ , 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  $n \times n$  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:

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

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

### **CANDIDATE ANSWER**

Language used: Python 3

```
1 #
2 # Complete the 'flippingMatrix' function below.
4 # The function is expected to return an INTEGER.
5 # The function accepts 2D INTEGER ARRAY matrix as parameter.
6 #
7 maximumValue_recorded_before = 0
8 maximumValue recorded = 0
9 matrixIN = []
11 def flippingMatrix(matrix):
     # Write your code here
      global matrixIN
     matrixIN = matrix
     if ((int(len(matrixIN)/2) < 1) or (int(len(matrixIN)/2) > 128)):
          return
      currentCalculated = 0
      for counterxx in range(0, int(len(matrixIN))):
          for counterzz in range(0, int(len(matrixIN))):
               if ((matrixIN[counterxx][counterzz] > 4096) or
24 (matrixIN[counterxx][counterzz] < 0)):
                   return
      # flip it
      MaximumSum = False
      counterMax = 0
      while (MaximumSum == False):
           CalculateANDprintMaximum()
           JustFlip_row()
          JustFlip coloumn()
          if (maximumValue recorded == maximumValue recorded before):
               counterMax+=1
               if (counterMax > 500):
                  MaximumSum = True
          else:
               counterMax=0
```

```
return (maximumValue_recorded)
43 def CalculateANDprintMaximum():
      global maximumValue recorded
      global maximumValue recorded before
      maximumValue recorded before = maximumValue recorded
      currentCalculated = countMaximum()
       if (currentCalculated > maximumValue recorded):
           maximumValue recorded = currentCalculated
51 def countMaximum():
      currentCalculated = matrixIN[0][0]
      if (int(len(matrixIN)/2) > 1):
          currentCalculated = 0
          for counterxx in range(0, int(len(matrixIN)/2)):
              for counterzz in range(0, int(len(matrixIN)/2)):
                  currentCalculated = currentCalculated + matrixIN[counterxx]
58 [counterzz]
      return(currentCalculated)
61 def JustFlip row():
     global matrixIN
       matrixIN[random.randint(0, len(matrixIN)-1)].reverse()
65 def JustFlip_coloumn():
     global matrixIN
      i= random.randint(0, len(matrixIN)-1)
      # reverse it
     matrixtemp = []
     for counterii in range(0,len(matrixIN)):
           matrixtemp.append(matrixIN[counterii][i])
     matrixtemp.reverse()
      #copyback
      for counterii in range(0,len(matrixIN)):
           matrixIN[counterii][i] = matrixtemp[counterii]
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	Success	0	0.0794 sec	9.47 KB
Testcase 2	Easy	Hidden case	Wrong Answer	0	3.992 sec	12.5 KB
Testcase 3	Easy	Hidden case	Wrong Answer	0	7.4022 sec	14 KB
Testcase 4	Easy	Hidden case	Wrong Answer	0	2.8459 sec	12.7 KB
Testcase 5	Easy	Hidden case	Wrong Answer	0	3.2451 sec	13.5 KB
Testcase 6	Easy	Hidden case	Wrong Answer	0	5.457 sec	12.5 KB
Testcase 7	Easy	Hidden case	Wrong Answer	0	8.1315 sec	13.3 KB
Testcase 8	Easy	Sample case	Wrong Answer	0	0.0726 sec	9.58 KB

No Comments