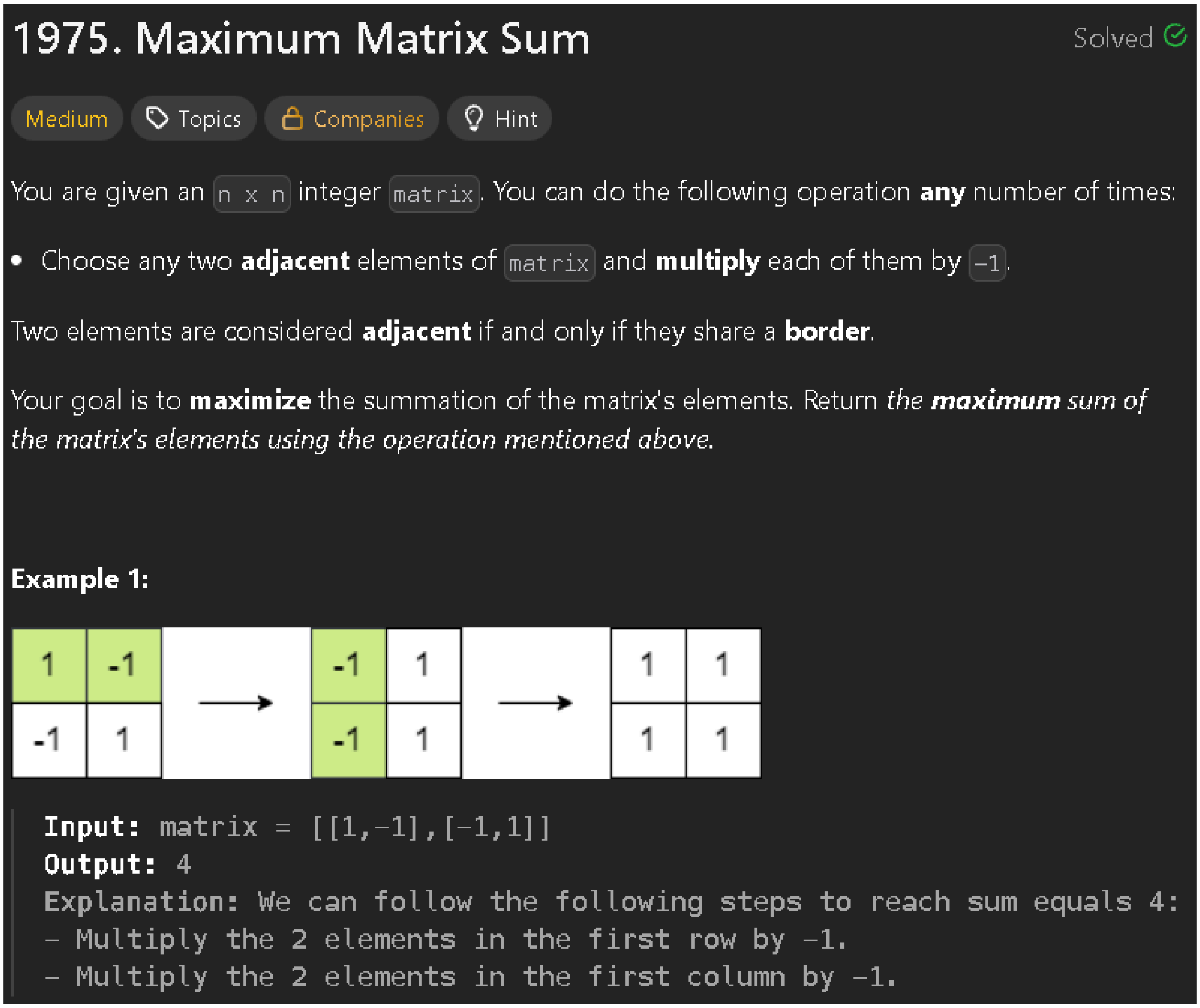


1975. Maximum Matrix Sum

You are given an `n x n` integer `matrix`. You can do the following operation **any** number of times:

- Choose any two **adjacent** elements of `matrix` and **multiply** each of them by `-1`.

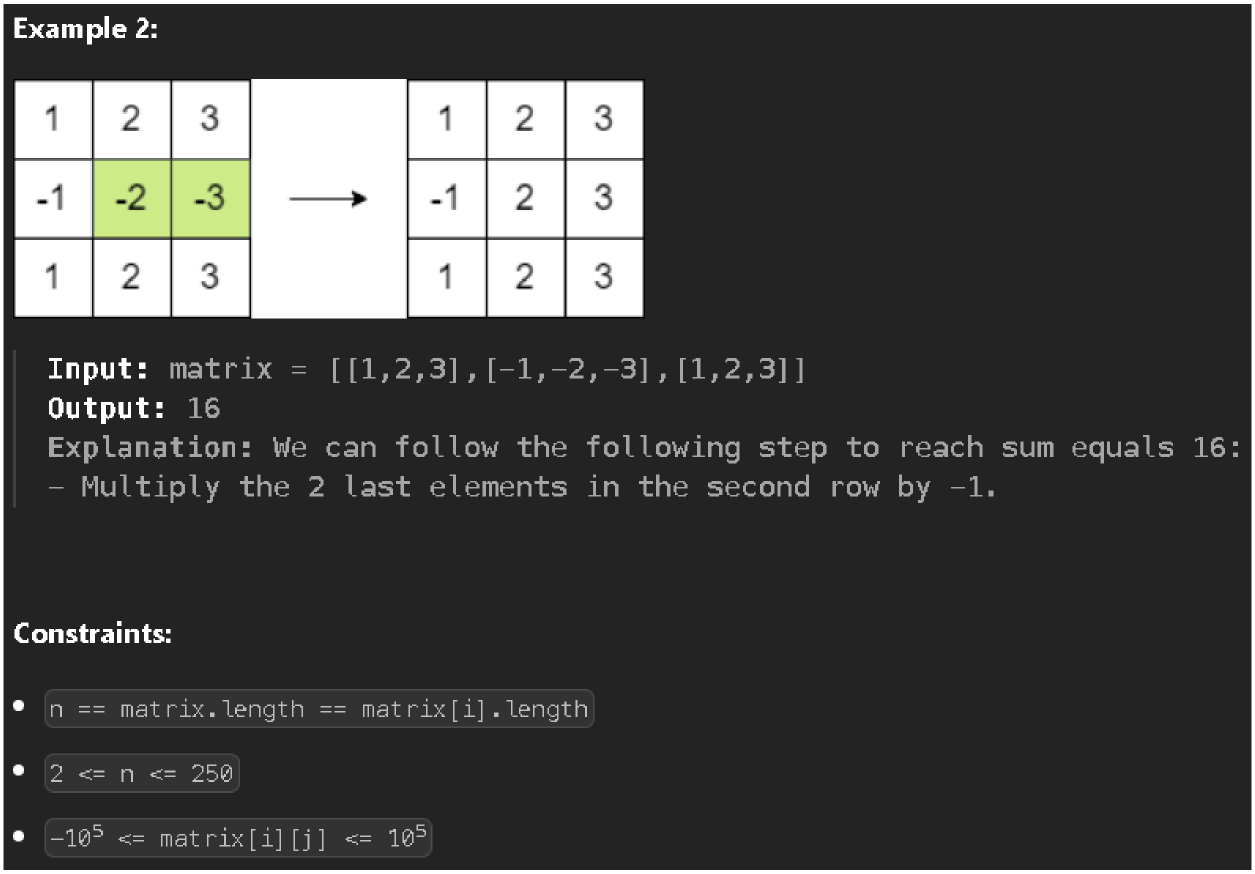
Two elements are considered **adjacent** if and only if they share a **border**.

Your goal is to **maximize** the summation of the matrix's elements. Return *the* ***maximum*** *sum of the matrix's elements using the operation mentioned above.*

Example 1:

1	-1	→	-1	1	→	1	1
-1	1		-1	1		1	1

```
Input: matrix = [[1,-1],[-1,1]]
Output: 4
Explanation: We can follow the following steps to reach sum equals 4:
- Multiply the 2 elements in the first row by -1.
- Multiply the 2 elements in the first column by -1.
```



Example 2:

1	2	3
-1	-2	-3
1	2	3

→

1	2	3
-1	2	3
1	2	3

```
Input: matrix = [[1,2,3],[-1,-2,-3],[1,2,3]]
Output: 16
Explanation: We can follow the following step to reach sum equals 16:
- Multiply the 2 last elements in the second row by -1.
```

Constraints:

- `n == matrix.length == matrix[i].length`
- `2 <= n <= 250`
- -10^5 `<= matrix[i][j] <=` 10^5

✅ CLEAN CODE 💯 ✅ ⚡ TIME: O(n^2) SPACE: O(1) ⚡

GREEDY + PARITY

Time Complexity: **O(n^2)**

Space Complexity: **O(1)**

What this code does?

1. Iterates over every element of the matrix.
2. Converts each value to its absolute value and adds it to totalSum.
3. Counts how many numbers in the matrix are negative.
4. Tracks the smallest absolute value in the entire matrix.
5. Uses the fact that sign flips affect negatives in pairs.
6. If the number of negative values is even, all numbers can be made positive.
7. In that case, returns the sum of all absolute values.
8. If the number of negative values is odd, one value must stay negative.
9. To minimize loss, subtracts 2 × (smallest absolute value) from the total.
10. Returns the maximum possible matrix sum after optimal sign flips.

Python:

```
class Solution:
    def maxMatrixSum(self, matrix: list[list[int]]) -> int:
        totalSum = 0
        neg = 0
        minAbs = float('inf')

        for row in matrix:
            for v in row:
                if v < 0:
                    neg += 1
                av = abs(v)
                totalSum += av
                minAbs = min(minAbs, av)

        return totalSum if neg % 2 == 0 else totalSum - 2 * minAbs
```

JavaScript:

```
var maxMatrixSum = function(matrix) {
    let totalSum = 0;
    let neg = 0;
    let minAbs = Infinity;

    for (const row of matrix) {
        for (const v of row) {
            if (v < 0) neg++;
            const av = Math.abs(v);
            totalSum += av;
            minAbs = Math.min(minAbs, av);
        }
    }

    return neg % 2 === 0 ? totalSum : totalSum - 2 * minAbs;
};
```

Python:

```
class Solution:
    def maxMatrixSum(self, matrix: list[list[int]]) -> int:
        totalSum = 0
        neg = 0
        minAbs = float('inf')

        for row in matrix:
            for v in row:
```

```
        if v < 0:
            neg += 1
        av = abs(v)
        totalSum += av
        minAbs = min(minAbs, av)

return totalSum if neg % 2 == 0 else totalSum - 2 * minAbs
```