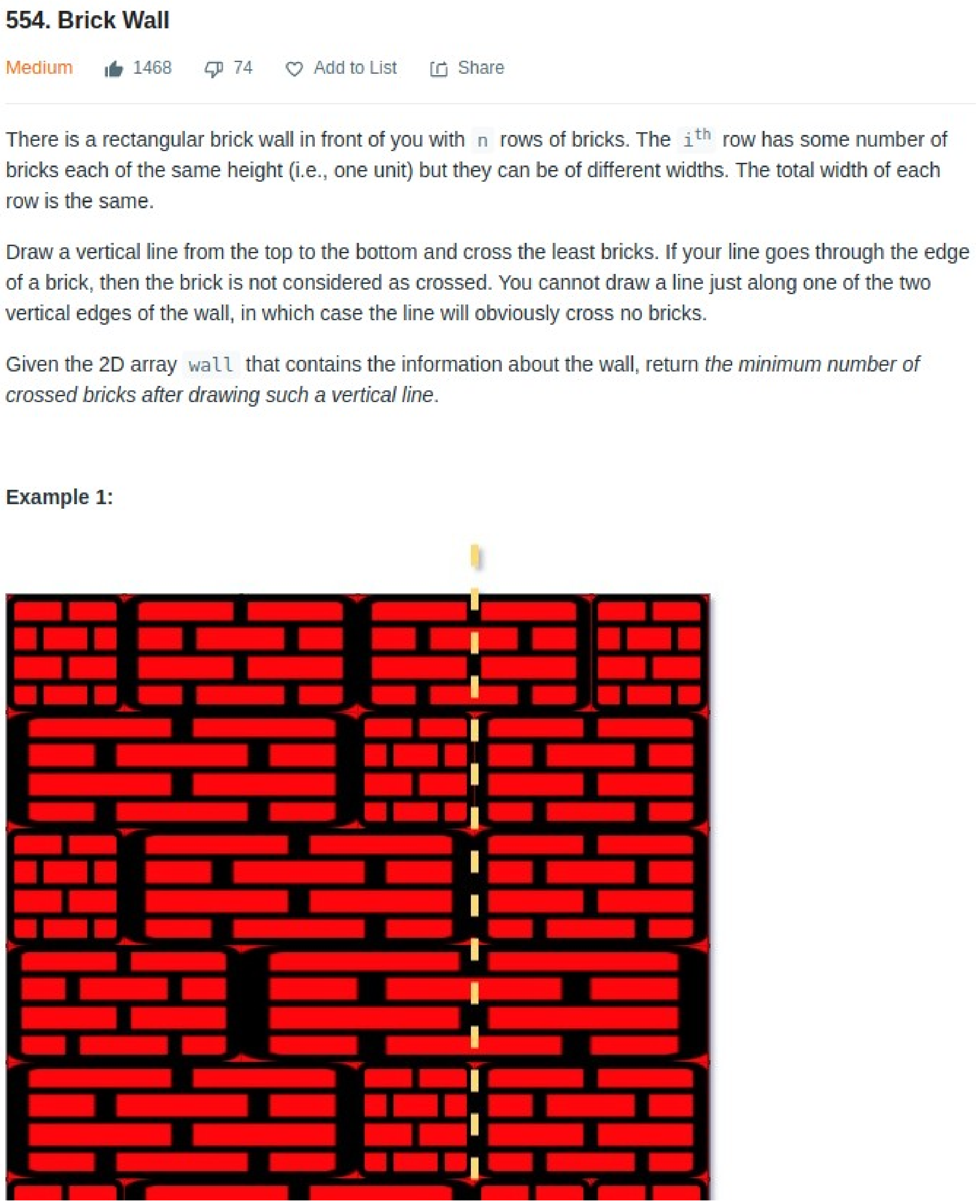
Brick wall problem

Problem description

https://leetcode.com/problems/brick-wall/



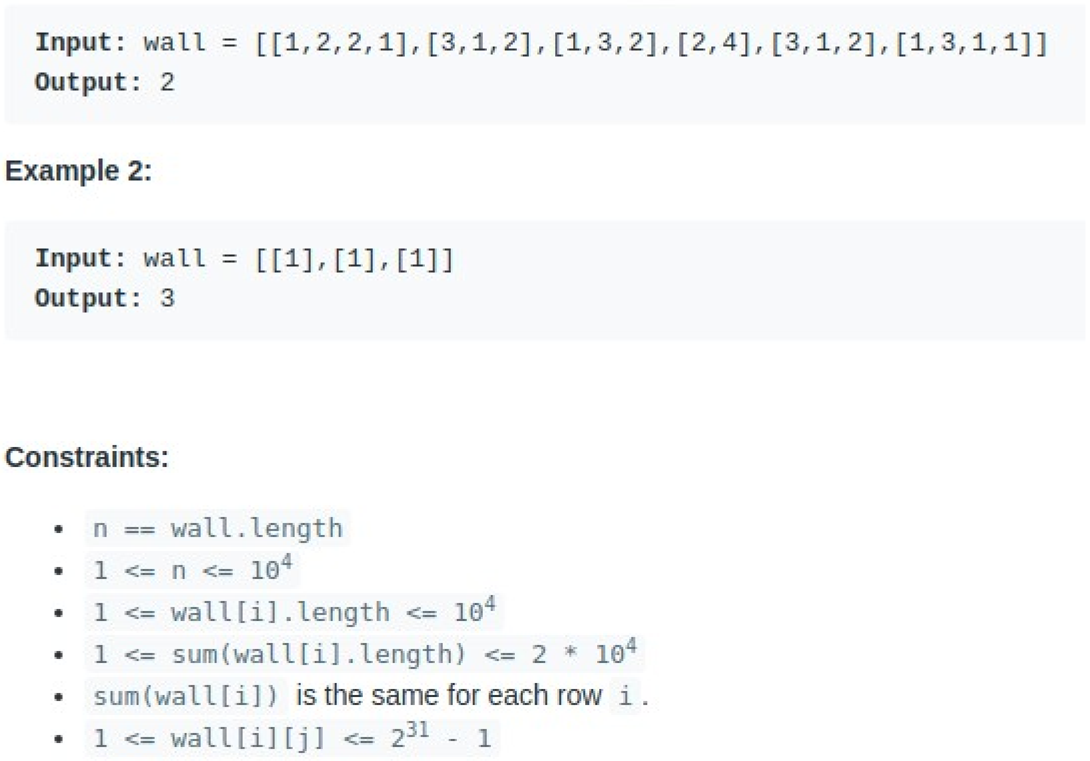
554. Brick Wall

There is a rectangular brick wall in front of you with `n` rows of bricks. The `i`th row has some number of bricks each of the same height (i.e., one unit) but they can be of different widths. The total width of each row is the same.

Draw a vertical line from the top to the bottom and cross the least bricks. If your line goes through the edge of a brick, then the brick is not considered as crossed. You cannot draw a line just along one of the two vertical edges of the wall, in which case the line will obviously cross no bricks.

Given the 2D array `wall` that contains the information about the wall, return *the minimum number of crossed bricks after drawing such a vertical line*.

Example 1:



```
Input: wall = [[1,2,2,1],[3,1,2],[1,3,2],[2,4],[3,1,2],[1,3,1,1]]
Output: 2
```

Example 2:

```
Input: wall = [[1],[1],[1]]
Output: 3
```

Constraints:

- `n == wall.length`
- `1 <= n <=` 10^4
- `1 <= wall[i].length <=` 10^4
- `1 <= sum(wall[i].length) <= 2 *` 10^4
- `sum(wall[i])` is the same for each row `i`.
- `1 <= wall[i][j] <=` 2^{31} `- 1`

Solution explanation

To figure out the solution, I represent the holes in the wall as a bunch of points in a Cartesian plan.

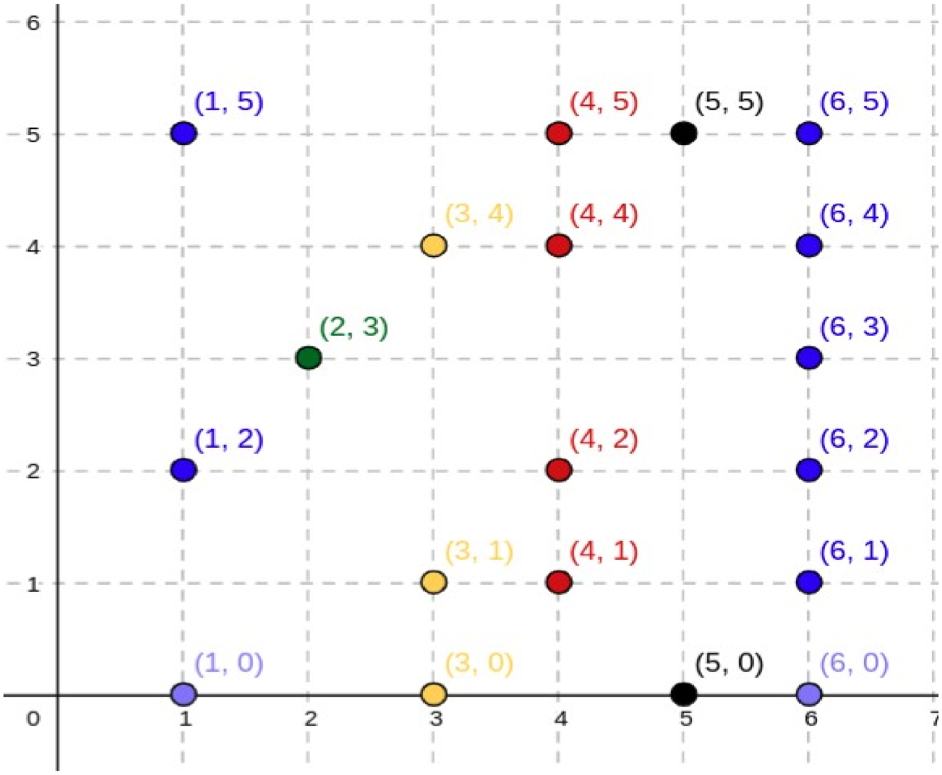
For example:

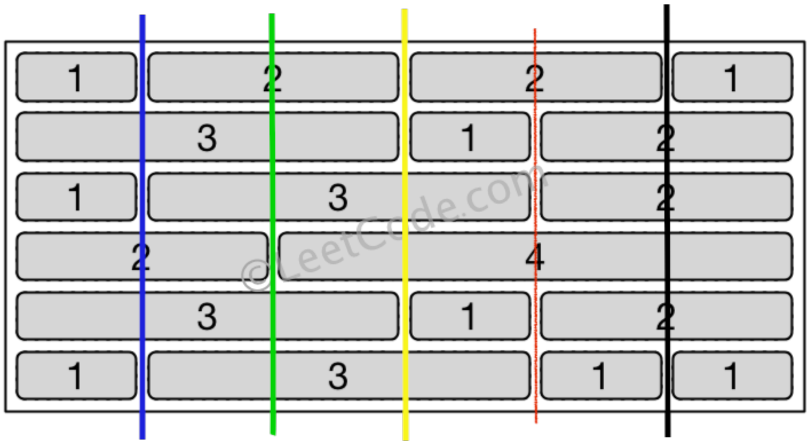
$[[1,2,2,1],[3,1,2],[1,3,2],[2,4],[3,1,2],[1,3,1,1]]$

at $y=0$, we have holes at $(1,0),(3,0),(5,0),(6,0)$

at $y=1$, we have $(3,1),(4,1),(6,1)$

and so on...





Then, we count the number of holes in each vertical line:
for this example,

- at $x=1$, we have 3 holes
- at $x=2$, we have 1 hole
- at $x=3$, we have 3 holes
- at $x=4$, we have 4 holes
- at $x=5$, we have 2 holes
- at $x=6$, we have 6 holes (**we gonna skip the last vertical, because it’s the edge**)

For this, I used a **map**. It have as key the $x-axis$ of each vertical line, and as value the number of holes in that line.

For this example, we have:
{1: 3, 2: 1, 3: 3, 4:4, 5:2, 6:6}

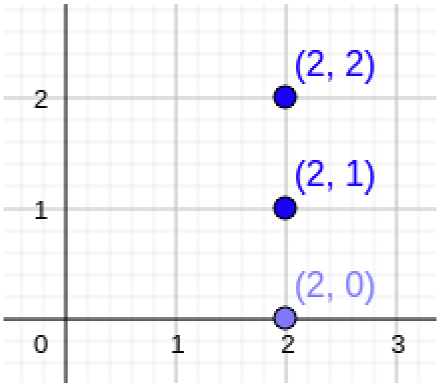
Iterate over the map, here we have two cases:

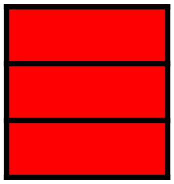
Case of one line of bricks

if the map is of size 1 , that mean that we have one vertical line without holes, an that is the edge of the wall, so we return the value.

Example #1:

$$wall=[[2],[2],[2]]$$

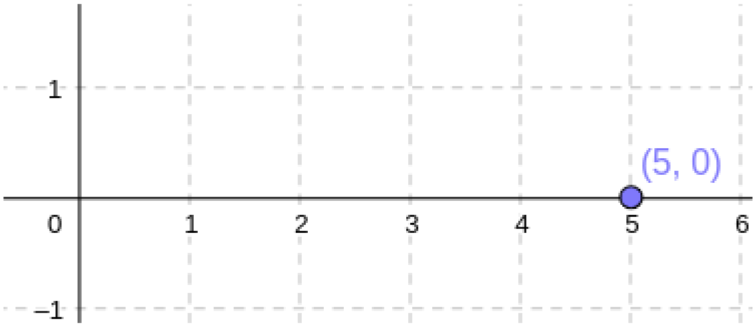




the map will be: $\{2:3\}$ (which means, we have 3 bricks on the vertical line $x=2$)
so, t**he minimum number of crossed bricks after drawing such a vertical line is equal to** 3

Example #2

$$wall=[[5]]$$

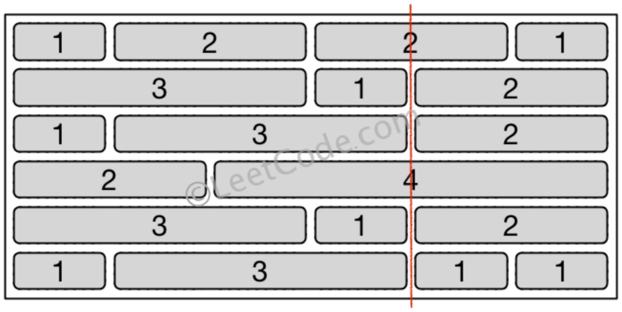


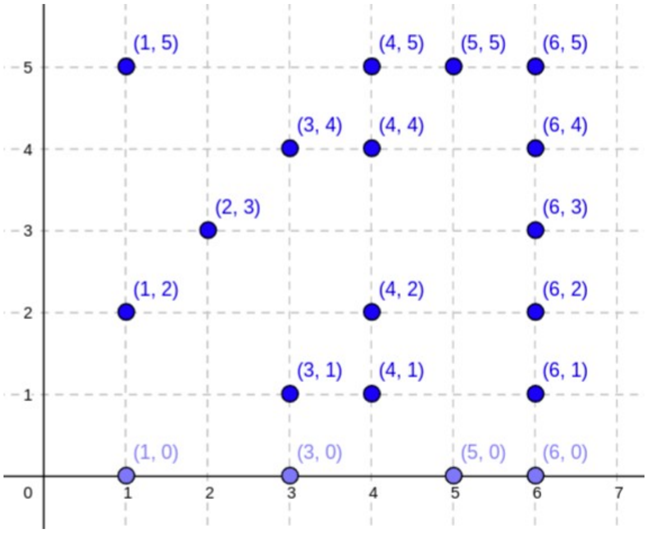
the map will be: $\{5:1\}$
so, t**he minimum number of crossed bricks after drawing such a vertical line is equal to** 1

Other cases

Except the last vertical line, compute the minimum number of crossed bricks by reducing from the wall's height the number of holes.

For example:





$map:\{1: 3, 2: 1, 3: 3, 4:4, 5:2, 6:6\}$

the 1st vertical line have 3 holes, so the number of crossed bricks is $6-3=3$

the 2nd vertical line have 1 holes, so the number of crossed bricks is $6-1=5$

the 3rd vertical line have 3 holes, so the number of crossed bricks is $6-3=3$

the 4th vertical line have 4 holes, so the number of crossed bricks is $6-4=2$

the 5th vertical line have 2 holes, so the number of crossed bricks is $6-2=4$

the 6th vertical is the edge, it is not counted.

the minimum number of crossed bricks after drawing such a vertical line is 2

Program flow

For each row of the wall

- increment the number of holes in the current vertical line of x-axis x (where x is the sum of the lengths of the bricks from 0 to $row[i]$, where $0 \leq i < row.size() - 1$. (*row.size()-1*, to avoid the last edge)
- update also, the max number of holes at that vertical line x .

example:

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

row	**i**	**x**	**Number holes at vertical line x-axis = x (map[x])**	**Max map[x]**
[1,2,2,1]	0	1	1	1
	1	3	1	1
	2	5	1	1
[3,1,2]	0	3	2	2
	1	4	1	
[1,3,2]	0	1	2	
	1	4	2	
[2,4]	0	2	1	
[3,1,2]	0	3	3	3
		4	3	
[1,3,1,1]	0	1	3	
		4	4	4
		5	2	

the maximum number of wholes is 4 , so t*he minimum number of crossed bricks after drawing such a vertical line at line* 4 is $6-4=2$

C++ code

```
int leastBricks(vector<vector<long>> wall) {
        long wall_height = wall.size();

        map<long, int> map;
        int max_num_holes = 0;
        for (auto row: wall) {
            long x = 0;
            for (int j = 0; j < row.size()-1; ++j) {
                x += row[j];
                map.insert({x, map[x]++});
                max_num_holes = max(max_num_holes, map[x]);
            }
        }


        return wall.size() - max_num_holes;
}
```