Step one : Precomputation

This is our precomputing formula:

$$S[i][j] = \sum_{r=1}^{i} \sum_{c=1}^{j} \text{grid}[r-1][c-1]$$

This is a $(r+1, c+1)$ map, 0 is empty and 1 is obstacle, we just add a some 0 in $r(0)$ and $c(0)$:

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

And this is an empty $(r+1, c+1)$ map:

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

We start with a (r+1, c+1) matrix where we add +1 because the algorithm considers positions behind the current one. This ensures that indices r(0) and c(0) have the value 0.

We use the formula:

$$S[r][c] = S[r-1][c] + S[r][c-1] - S[r-1][c-1] + (1 \text{ or } 0)$$

Let's delve into the details:

- $S[r-1][c] = 6$
- $S[r][c-1] = 7$
- $S[r-1][c-1] = 5$

In our map, we have an obstacle at this position, so:

$$S[r][c] = 6 + 7 - 5 + 0 = 8$$

The general concept is to calculate how many obstacles we have already encountered behind our current position. We consider behind us as the leftward and upward directions, similar to a vector from the origin.

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  ,  0 ,  0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1  ,  2 ,  2, 2]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2  ,  3 ,  3, 3]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3  ,  4 ,  4, 4]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 4, 4, 4, [5], [6],  6, 6]
[0, 0, 0, 0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 6, 6, 6, [7], {8},  0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  ,  0 ,  0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  ,  0 ,  0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  ,  0 ,  0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  ,  0 ,  0, 0]
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  , 0  , 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, S, 0, 0, 0  , C  , 0, 0]
[0, 0, 0, 0, 0, S, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  , 0  , 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, S, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  , 0  , 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, S, 0, 0,[X] ,[0] , 0, 0]
[0, 0, 0, 0, 0, R, 0, 0, 0, R, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,[0] ,{0} , 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0  , 0  , 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  , 0  , 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0  , 0  , 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  , 0  , 0, 0]
```

So now we have our filled tab, and on each postion we can know how many obsactles we have behind us include ourself

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 , 0 , 0 , 0 , 0 , 0 ]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1 , 1 , 1 , 1 , 2 , 2 , 2 ]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2 , 2 , 2 , 2 , 3 , 3 , 3 ]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3 , 3 , 3 , 3 , 4 , 4 , 4 ]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 4 , 4 , 4 , 5 , 6 , 6 , 6 ]
[0, 0, 0, 0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 6 , 6 , 6 , 7 , 8 , 8 , 8 ]
[0, 0, 0, 0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[0, 0, 0, 0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[0, 0, 0, 0, 0, 2, 2, 3, 3, 4, 4, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 8 , 9 , 9 , 10, 11, 11, 11]
[0, 0, 0, 1, 1, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 7, 8, 8, 8, 8, 9, 10, 11, 11, 12, 13, 13, 13]
```

Step Two : Computation

Now let's take a look at the main algorithm. Initially, we consider that there is at least one 0*0 solution, so we start looking for a 1*1 solution.

We loop through rows and columns and examine the values in our table at positions c + k and r + k. Let's see how it works:

Now we will use the formula:

$$S[r+k][c+k] - S[r+k][c] - S[r][c+k] + S[r][c]$$

Here, $S[r][c]$ is our first case. We can easily compute:

$$0 - 0 - 0 + 0 = 0$$

Now we know that our first position in {0} is a 1 * 1 solution. The rule is to print only the topmost and leftmost solution. Therefore, if there is no 2 * 2 solution, the 1 * 1 solution at (0,0) is our final solution, and we stop.

So now we will look for the 2*2 square.

```
[ {0}, [0], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 , 0 , 0 , 0 , 0 , 0 , 0 ]
[ [0], [0], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1 , 1 , 1 , 1 , 2 , 2 , 2 ]
[  0 , 0,  0,  0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2 , 2 , 2 , 2 , 3 , 3 , 3 ]
[  0 , 0,  0,  0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3 , 3 , 3 , 3 , 4 , 4 , 4 ]
[  0 , 0,  0,  0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 4 , 4 , 4 , 5 , 6 , 6 , 6 ]
[  0 , 0,  0,  0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 6 , 6 , 6 , 7 , 8 , 8 , 8 ]
[  0 , 0,  0,  0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[  0 , 0,  0,  0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[  0 , 0,  0,  0, 0, 2, 2, 3, 3, 4, 4, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 8 , 9 , 9 , 10, 11, 11, 11]
[  0 , 0,  0,  1, 1, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 7, 8, 8, 8, 8, 9, 10, 11, 11, 12, 13, 13, 13]
```

This is the BSQ assuming the is not more than a 4*4 solution

```
.....HHHH..........o....o..
.....HHHH..................
.....HHHH...o...o..........
.....HHHH...........o..o...
....o...o..................
...............o...........
...........................
......o..............o.....
..o.......o................
```

More than one?

Again, k is the size of the square we are looking for. k is 1 plus our best solution.

To continue, let's take the step of looking for a 5x5 square. Suppose we assume we are at $k = 5$, but our best solution so far is a 4x4 square.

Now, we start again from the beginning, knowing that we are looking for a 5*5 square. We will use the formula:

$$S[r+k][c+k] - S[r+k][c] - S[r][c+k] + S[r][c]$$

Now we have:

$$S[5][5] - S[5][0] - S[0][5] + S[0][0]$$

$$2 - 0 - 0 + 0 = 2 \neq 0$$

This is not a solution.

```
[ {0}, 0, 0, 0, 0, [0], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 , 0 , 0 , 0 , 0 , 0 , 0 ]
[  0 , 0, 0, 0, 0,  0 , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1 , 1 , 1 , 1 , 2 , 2 , 2 ]
[  0 , 0, 0, 0, 0,  1 , 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2 , 2 , 2 , 2 , 3 , 3 , 3 ]
[  0 , 0, 0, 0, 0,  1 , 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3 , 3 , 3 , 3 , 4 , 4 , 4 ]
[  0 , 0, 0, 0, 0,  1 , 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 4 , 4 , 4 , 5 , 6 , 6 , 6 ]
[ [0], 0, 0, 0, 0, [2], 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 6 , 6 , 6 , 7 , 8 , 8 , 8 ]
[  0 , 0, 0, 0, 0,  2 , 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[  0 , 0, 0, 0, 0,  2 , 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[  0 , 0, 0, 0, 0,  2 , 2, 3, 3, 4, 4, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 8 , 9 , 9 , 10, 11, 11, 11]
[  0 , 0, 0, 1, 1,  3 , 3, 4, 4, 5, 5, 6, 6, 7, 7, 7, 8, 8, 8, 8, 9, 10, 11, 11, 12, 13, 13, 13]
```

Searching for a 5x5 Solution

Again, we try one step to the right, starting from the (0,1) position. If (0,0) is not a solution, we increment r by 1. If $r(0)$ is not a solution, we move to $c(1)$ and start from (0,1).

In this case, we found a solution at (0,13).

Let's calculate the function:

$$S[0+5][13+5] - S[0+5][13] - S[0][13+5] + S[0][13]$$

$$4 - 4 - 0 + 0 = 0$$

We found a solution for the 5*5 square.

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,{0}, 0, 0, 0, 0,[0], 0, 0, 0 , 0 , 0 , 0 , 0 , 0 , 0 ]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 , 0, 0, 0, 0, 0 , 0, 1, 1 , 1 , 1 , 1 , 2 , 2 , 2 ]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1 , 1, 1, 1, 1, 1 , 1, 2, 2 , 2 , 2 , 2 , 3 , 3 , 3 ]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2 , 2, 2, 2, 2, 2 , 2, 3, 3 , 3 , 3 , 3 , 4 , 4 , 4 ]
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2 , 2, 2, 2, 2, 2 , 2, 3, 4 , 4 , 4 , 5 , 6 , 6 , 6 ]
[0, 0, 0, 0, 0, 2, 2, 2, 2, 3, 3, 3, 3,[4], 4, 4, 4, 4,[4], 4, 5, 6 , 6 , 6 , 7 , 8 , 8 , 8 ]
[0, 0, 0, 0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4 , 4, 4, 5, 5, 5 , 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[0, 0, 0, 0, 0, 2, 2, 2, 2, 3, 3, 3, 3, 4 , 4, 4, 5, 5, 5 , 5, 6, 7 , 7 , 7 , 8 , 9 , 9 , 9 ]
[0, 0, 0, 0, 0, 2, 2, 3, 3, 4, 4, 4, 4, 5 , 5, 5, 6, 6, 6 , 6, 7, 8 , 9 , 9 , 10, 11, 11, 11]
[0, 0, 0, 1, 1, 3, 3, 4, 4, 5, 5, 6, 6, 7 , 7, 7, 8, 8, 8 , 8, 9, 10, 11, 11, 12, 13, 13, 13]
```

Searching for a 6x6 Solution

When we reach $(r+6, c+6) = (y, x)$ without finding any solution, we know there is no square larger than 5x5. This is because any larger square than 6x6 would contain a 6x6 square.

With this method, we can find the first square in the top-left corner by trying to find it row by row, starting from the left.

Note: This algorithm provides an $O(r, x)$ solution on the S_{tab}, but the S_{tab} is (r, c) sized, so we have:

$$\text{Stab size} = (y+1, x+1)$$

```
[S, X, X, X, X, X]
[X, H, H, H, H, H]
[X, H, H, H, H, H]
[X, H, H, H, H, H]
[X, H, H, H, H, H]
[X, H, H, H, H, H]
```

here S ithe coordonate in S_{tab}, but we have a $(-1, -1)$ vector translate to operate to come back in the map.
So the S postion is the right position of the topside/leftside BSQ in the map.

CQFD

Bonus

Generalized N-Dimensional Cumulative Sum Matrix

For an N-dimensional grid, the cumulative sum matrix S is defined as:

$$S[i_1][i_2][\ldots][i_N] = \sum_{a_1=1}^{i_1} \sum_{a_2=1}^{i_2} \ldots \sum_{a_N=1}^{i_N} \text{grid}[a_1-1][a_2-1][\ldots][a_N-1]$$

Where:

- $S[i_1][i_2][\ldots][i_N]$ represents the cumulative sum of elements in the sub-grid from $(1, 1, \ldots, 1)$ to $(i_1, i_2, \ldots, i_N)$.
- $\text{grid}[a_1-1][a_2-1][\ldots][a_N-1]$ is the value of the cell at position $(a_1-1, a_2-1, \ldots, a_N-1)$ in the original grid.

The recursive formula for calculating S is:

$$S[i_1][i_2][\ldots][i_N] = \text{grid}[i_1 - 1][i_2 - 1][\ldots][i_N - 1] + \sum_{j=1}^{N} S[\ldots, i_j - 1, \ldots] - \sum_{1 \le j < k \le N} S[\ldots, i_j - 1, \ldots, i_k - 1, \ldots] + \ldots + (-1)^{N-1} S[i_1 -$$

Obstacle Count in a Sub-Hypercube

For a sub-hypercube defined by its top-left corner $(i_1, i_2, \ldots, i_N)$ and side length d, the number of obstacles is:

$$\text{obstacle\_count} = S[i_1 + d][i_2 + d][\ldots][i_N + d] - \sum_{j=1}^{N} S[\ldots, i_j, \ldots, i_j + d, \ldots] + \sum_{1 \le j < k \le N} S[\ldots, i_j, \ldots, i_k, \ldots, i_j + d, i_k + d, \ldots] - \ldots + ($$

Where:

- $S[i_1 + d][i_2 + d][\ldots][i_N + d]$ is the cumulative sum up to the far corner of the sub-hypercube.
- The alternating sums and subtractions adjust for over-counted regions at the intersections of the sub-hypercube boundaries.