题目描述

某长方形停车场，每个车位上方都有对应 监控器，当且仅当在当前车位或者前后左右四个方向任意一个车位范围停车时，监控器才需要打开；

给出某一时刻停车场的停车分布，请统计最少需要打开多少个监控器；

输入描述

第一行输入m，n表示长宽，满足1 < m,n <= 20；

后面输入m行，每行有n个0或1的整数，整数间使用一个空格隔开，表示该行已停车情况，其中0表示空位，1表示已停；

输出描述

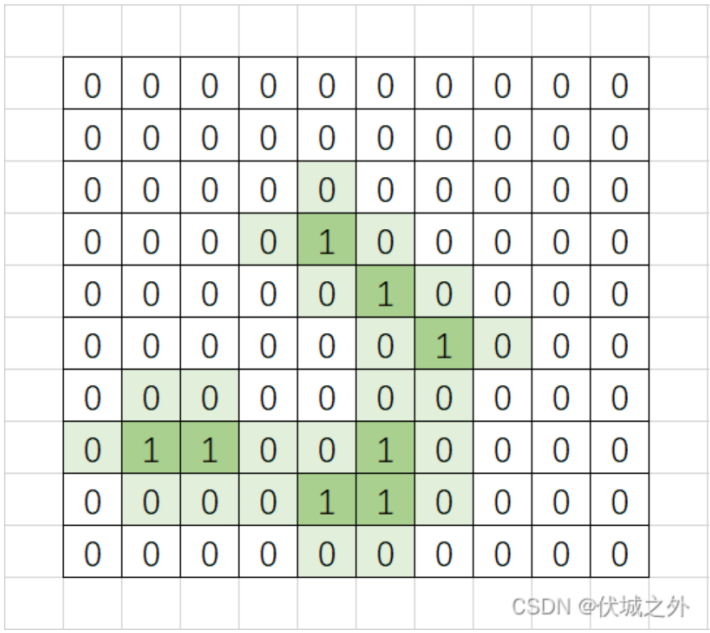
最少需要打开监控器的数量；

用例

输入	3 3 0 0 0 0 1 0 0 0 0
输出	5
说明	无

题目解析

本题题意比较难以理解，但是画一幅图给大家看下就懂了



停车（1）的位置必须要打开监控器，另外停车位置的上下左右位置（1或0）的监控器也要打开，问最终需要打开多少个监控器？

即，求解上面图示种有颜色的格子数量。

解题思路如下：

遍历矩阵每一个元素，

- 如果元素值为1，对应位置的监控器就要打开。
- 如果元素值为0，则需要进一步检查其上下左右四个位置，只要这四个位置有一个元素值为1，则当前位置的监控器需要打开

```java
import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    int m = sc.nextInt();
    int n = sc.nextInt();

    int[][] matrix = new int[m][n];
    for (int i = 0; i < m; i++) {
      for (int j = 0; j < n; j++) {
        matrix[i][j] = sc.nextInt();
      }
    }
```

```java

    System.out.println(getResult(m, n, matrix));
  }

  public static int getResult(int m, int n, int[][] matrix) {
    int count = 0;
    int[][] offsets = {{-1, 0}, {1, 0}, {0, 1}, {0, -1}};

    for (int x = 0; x < m; x++) {
      for (int y = 0; y < n; y++) {
        if (matrix[x][y] == 1) {
          count++;
          continue;
        }

        for (int[] offset : offsets) {
          int newX = x + offset[0];
          int newY = y + offset[1];

          if (newX >= 0 && newX < m && newY >= 0 && newY < n && matrix[newX][newY] == 1) {
            count++;
            break;
          }
        }
      }
    }

    return count;
  }
}
```