2125. Number of Laser Beams in a Bank

Solv

Medium | Topics | Companies | Hint

Anti-theft security devices are activated inside a bank. You are given a **0-indexed** binary string array `bank` representing the floor plan of the bank, which is an `m x n` 2D matrix. `bank[i]` represents the i^{th} row, consisting of `'0'`s and `'1'`s. `'0'` means the cell is empty, while `'1'` means the cell has a security device.

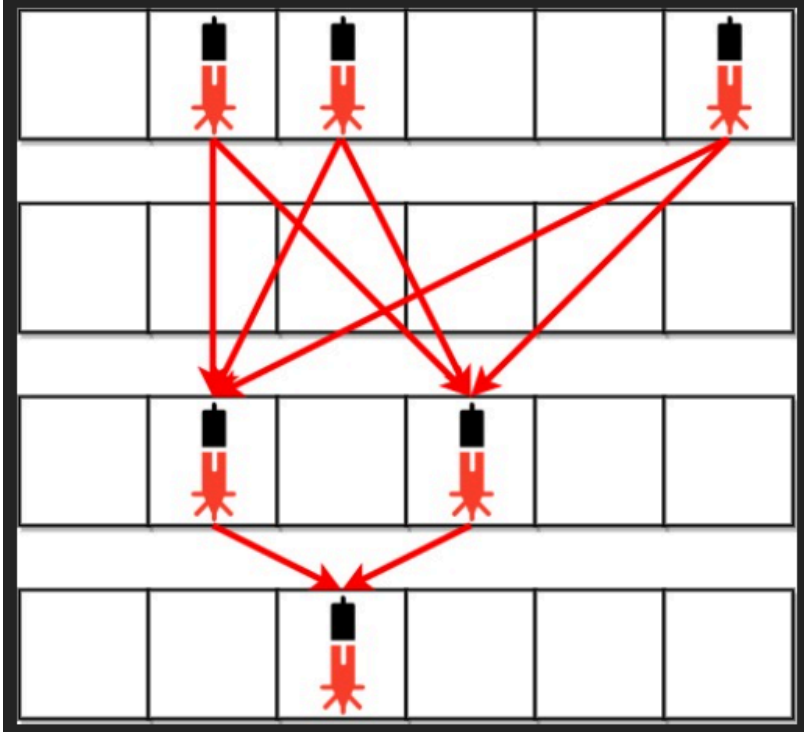
There is **one** laser beam between any **two** security devices **if both** conditions are met:

- The two devices are located on two **different rows**: r_1 and r_2, where $r_1 < r_2$.
- For **each** row i where $r_1 < i < r_2$, there are **no security devices** in the i^{th} row.

Laser beams are independent, i.e., one beam does not interfere nor join with another.

Return *the total number of laser beams in the bank*.

Example 1:



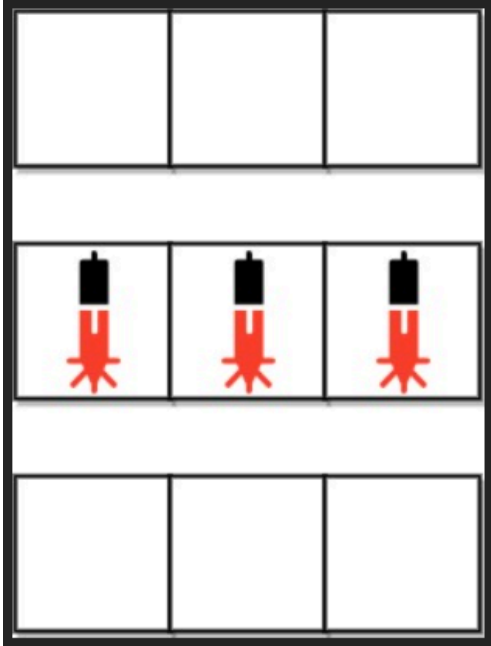
```
Input: bank = ["011001","000000","010100","001000"]
Output: 8
Explanation: Between each of the following device pairs, there is one
beam. In total, there are 8 beams:
```

Explanation: Between each of the following device pairs, there is one beam. In total, there are 8 beams:

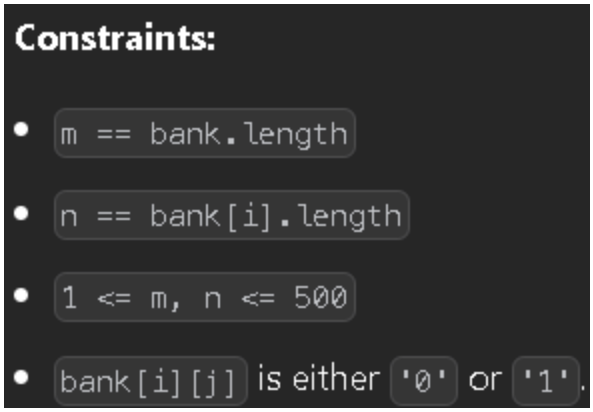
- bank[0][1] -- bank[2][1]
- bank[0][1] -- bank[2][3]
- bank[0][2] -- bank[2][1]
- bank[0][2] -- bank[2][3]
- bank[0][5] -- bank[2][1]
- bank[0][5] -- bank[2][3]
- bank[2][1] -- bank[3][2]
- bank[2][3] -- bank[3][2]

Note that there is no beam between any device on the 0^{th} row with any on the 3^{rd} row.
This is because the 2^{nd} row contains security devices, which breaks the second condition.

Example 2:



Input: bank = ["000","111","000"]
Output: 0
Explanation: There does not exist two devices located on two different rows.



Python:

```
class Solution(object):
    def numberOfBeams(self, bank):
        prev_row_count = 0
        total = 0

        for row in bank:
            cur_row_count = self.calc(row)
            if cur_row_count == 0:
                continue

            total += cur_row_count * prev_row_count
            prev_row_count = cur_row_count

        return total

    def calc(self, s):
        return sum(int(c) for c in s)
```

JavaScript:

```
/**
 * @param {string[]} bank
 * @return {number}
 */
var numberOfBeams = function(bank) {
    let prevRowCount = 0;
    let total = 0;

    const calc = (s) => {
        return s.split('').reduce((count, c) => count + parseInt(c), 0);
    };
```

```
    for (const row of bank) {
        const curRowCount = calc(row);
        if (curRowCount === 0)
            continue;

        total += curRowCount * prevRowCount;
        prevRowCount = curRowCount;
    }
    return total;
};
```

Java:

```
class Solution {
    public int numberOfBeams(String[] bank) {
        int prevRowCount = 0;
        int total=0;

        for(String row : bank) {
            int curRowCount = calc(row);
            if(curRowCount==0)
                continue;

            total += curRowCount * prevRowCount;
            prevRowCount = curRowCount;
        }
        return total;
    }

    private int calc(String s) {
        int count = 0;
        for(char c : s.toCharArray())
            count += c - '0';

        return count;
    }
}
```