Problem 302: Smallest Rectangle Enclosing Black Pixels

Problem Information

Difficulty: Hard
Acceptance Rate: 60.75%
Paid Only: Yes
Tags: Array, Binary Search, Depth-First Search, Breadth-First Search, Matrix

Problem Description

You are given an `m x n` binary matrix `image` where `0` represents a white pixel and `1` represents a black pixel.

The black pixels are connected (i.e., there is only one black region). Pixels are connected horizontally and vertically.

Given two integers `x` and `y` that represents the location of one of the black pixels, return _the area of the smallest (axis-aligned) rectangle that encloses all black pixels_.

You must write an algorithm with less than `O(mn)` runtime complexity

Example 1:

Input: image = [["0","0","1","0"],["0","1","1","0"],["0","1","0","0"]], x = 0, y = 2 **Output:** 6

Example 2:

Input: image = [["1"]], x = 0, y = 0 **Output:** 1

Constraints:

* `m == image.length` * `n == image[i].length` * `1 <= m, n <= 100` * `image[i][j]` is either `'0'` or `'1'`. * `0 <= x < m` * `0 <= y < n` * `image[x][y] == '1'.` * The black pixels in the `image` only form **one component**.

Code Snippets

C++:

```
class Solution {
public:
int minArea(vector<vector<char>>& image, int x, int y) {

}
};
```

Java:

```
class Solution {
public int minArea(char[][] image, int x, int y) {

}
}
```

Python3:

```
class Solution:
def minArea(self, image: List[List[str]], x: int, y: int) -> int:
```