Problem 1139: Largest 1-Bordered Square

Problem Information

Difficulty: Medium
Acceptance Rate: 51.76%
Paid Only: No
Tags: Array, Dynamic Programming, Matrix

Problem Description

Given a 2D `grid` of `0`s and `1`s, return the number of elements in the largest **square** subgrid that has all `1`s on its **border** , or `0` if such a subgrid doesn't exist in the `grid`.

Example 1:

Input: grid = [[1,1,1],[1,0,1],[1,1,1]] **Output:** 9

Example 2:

Input: grid = [[1,1,0,0]] **Output:** 1

Constraints:

* `1 <= grid.length <= 100` * `1 <= grid[0].length <= 100` * `grid[i][j]` is `0` or `1`

Code Snippets

C++:

```
class Solution {
public:
int largest1BorderedSquare(vector<vector<int>>& grid) {

}
};
```

Java:

```
class Solution {
public int largest1BorderedSquare(int[][] grid) {

}
}
```

Python3:

```
class Solution:
def largest1BorderedSquare(self, grid: List[List[int]]) -> int:
```