문제 설명

1와 0로 채워진 표(board)가 있습니다. 표 1칸은 1 x 1 의 정사각형으로 이루어져 있습니다. 표에서 1로 이루어진 가장 큰 정사각형을 찾아 넓이를 return 하는 solution 함수를 완성해 주세요. (단, 정사각형이란 축에 평행한 정사각형을 말합니다.)

예를 들어

| **1** | **2** | **3** | **4** |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 |

가 있다면 가장 큰 정사각형은

| **1** | **2** | **3** | **4** |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 |

가 되며 넓이는 9가 되므로 9를 반환해 주면 됩니다.

제한사항

* 표(board)는 2차원 배열로 주어집니다.
* 표(board)의 행(row)의 크기 : 1000 이하의 자연수
* 표(board)의 열(column)의 크기 : 1000 이하의 자연수
* 표(board)의 값은 1또는 0으로만 이루어져 있습니다.

입출력 예

| **board** | **answer** |
| --- | --- |
| [[0,1,1,1],[1,1,1,1],[1,1,1,1],[0,0,1,0]] | 9 |
| [[0,0,1,1],[1,1,1,1]] | 4 |

입출력 예 설명

입출력 예 #1  
위의 예시와 같습니다.

입출력 예 #2  
| 0 | 0 | 1 | 1 |  
| 1 | 1 | 1 | 1 |   
로 가장 큰 정사각형의 넓이는 4가 되므로 4를 return합니다.

#include <iostream>

#include <vector>

#include<algorithm>

using namespace std;

int solution(vector<vector<int>> board)

{

int max = 0;

for (int i = 0; i < board.size(); i++)

{

for (int j = 0; j < board[i].size(); j++)

{

if (board[i][j] != 0)

{

if (i == 0 || j == 0) board[i][j] = 1;

else board[i][j] = min({ board[i][j - 1], board[i - 1][j - 1], board[i - 1][j] }) + 1;

}

if (max < board[i][j])

max = board[i][j];

}

}

return max \* max;

}

/\* 과정 설명

배열의 값이 1인 경우에 왼쪽, 왼쪽대각선위, 위 방향의 배열과 비교해서 minimum값을 구하고 + 1을 해준다

이렇게 max값을 구해서 서로 곱해주면 정사각형의 넓이가 나온다.

\*/