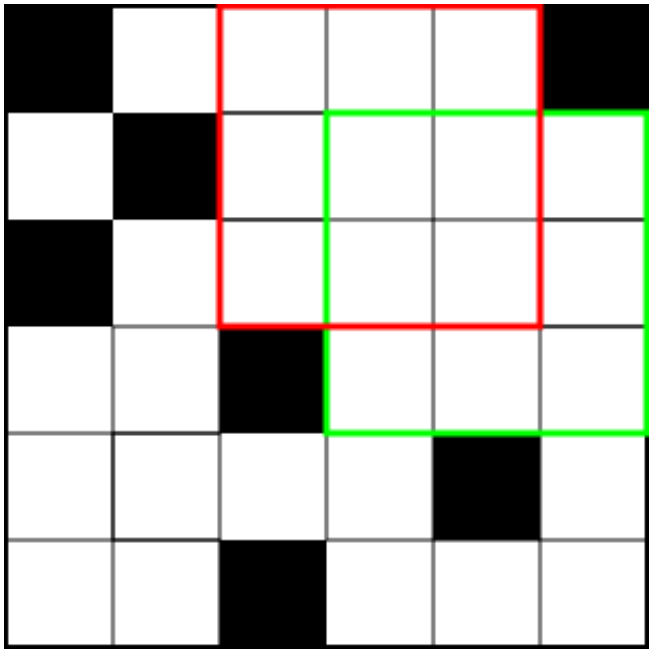
Square

Cutting a square from a material

Jian-Jia has a piece of metal material and he wants to cut a square out of it. The material consists of `n` by `n` unit grids and Jian-Jia can only cut the material along grid boundary. Each grid is either usable or defective, and Jian-Jia wants to cut the largest possible square from the material without any defective grids. After determining the maximum size of the square, Jian-Jia also wants to know how many ways he can cut the largest square from this material. Finally Jian-Jia will report the product of the maximum size and the number of possible ways.

Example

Consider the 6 by 6 material in the following figure. The black grids are defective. The largest square Jian-Jia can cut from the material is 3 by 3, and there are two ways to cut it -- the red square and the green square. Jian-Jan will report the product of 3 and 2, which is 6.



Statement

Your task is to find the size of largest squares in the material, count the number of ways to cut them, and report the product of the size and the number. You only need to implement one function `findMaxSquare`.

- `findMaxSquare`(material, materialSize)
  - The array `material` has the status of the grid; 1 is usable and 0 is defective. `materialSize` is the actual size used in this subtask.
  - The return value is the product of the size of largest square in the material, and the number possible locations in the material.

Subtask 1 [10 points]

- $1 \leq n \leq 100$
  - In any 2 by 2 material grids section there will be at least one defective grid.

Subtask 2 [20 points]

- $1 \leq n \leq 500$

Subtask 3 [70 points]

- $1 \leq n \leq 1000$

Implementation details

You have to submit exactly one file, called `square.c`, `square.cpp` or `square.pas`. This is file implements the subprograms described above using the following signatures. You also need to include a header file `square.h`.

C/C++ program

```
#include "square.h"
int findMaxSquare(int material[SIZE][SIZE], int materialSize);
```

Pascal program

```
const
   SIZE = 1000;
type
   materialType = array[0..SIZE-1, 0..SIZE-1] of longint;
function findMaxSquare(material: materialType; materialSize: longint):
   longint;
```

This subprogram must behave as described above. Of course you are free to implement other subprograms for their internal use. Your submissions must not interact in any way with standard input/output, nor with any other file.

Sample grader

The sample grader reads the input in the following format:

- line 1: the size of the material n;
- lines 2, ..., n + 1:
    - Each line has n integers. A 1 means the grid is useful and a 0 means the grid is defective.