G. Новый офис плюса

|  |  |
| --- | --- |
| Ограничение времени | 5 секунд |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Сервис Тындекс.Плюс так быстро растет, что для сотрудников и серверов потребовалось потребовалось построить новый офис.

Участок под застройку представляет из себя клетчатое поле n×m, часть клеток которого пригодна для строительства, а часть нет.

Новый офис должен выглядеть как знак "плюс"какого-то целого положительного размера k. Знак "плюс"размера k  — это такая клетчатая фигура, состоящая из пяти квадратов k×k клеток, при этом есть один центральный квадрат, а остальные четыре являются его соседями по стороне.

Новый офис должен быть как можно больше, поэтому необходимо найти максимальное k, такое что офис удастся разместить на участке под застройку.

Определите максимальное k. Гарантируется, что он можно построить офис хотя бы с k=1.

Формат ввода

В первой строке задано два целых числа n и m (1≤n,m≤2000) — длина и ширина участка под застройку.

В каждой из последующих n строк задана строка, состоящая из m символов, j-й символ в i-й строке равен #, если клетка с координатами (i,j) пригодна для строительства и . иначе.

Формат вывода

Выведите одно целое положительное число — максимально возможное k.

Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 9 12  ...##.###...  ...##.###...  .########...  .###########  ...#########  ...#########  ......###...  ......###...  ......###... | 3 |

Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 6 6  .##...  .##...  ######  ######  .##...  .##... | 1 |

Примечания

В первом тесте из примера можно выбрать плюс с k=3. Этот плюс выглядит следующим образом:

...###...

...###...

...###...

#########

#########

#########

...###...

...###...

...###...

using System;

using System.IO;

internal class Program

{

public static void Main()

{

StreamReader reader = new StreamReader("input.txt");

string[] input1 = reader.ReadLine().Split();

int n = Convert.ToInt32(input1[0]);

int m = Convert.ToInt32(input1[1]);

char[,] matrix = new char[n, m];

int[,] rows = new int[n,m];

int[,] columns = new int[n,m];

int count = 0;

for (int i = 0; i < n; i++)

{

int current = 1;

string input2 = reader.ReadLine();

for (int j = m-1; j >= 0; j--)

{

matrix[i, j] = input2[j];

if (input2[j] == '#')

{

rows[i, j] = current++;

count++;

}

else current = 1;

}

}

for (int j = 0; j < m; j++)

{

int current = 1;

for (int i = n-1; i >= 0; i--)

{

if (matrix[i, j] == '#')

{

columns[i, j] = current++;

count++;

}

else current = 1;

}

}

int left = 2;

int right = Math.Min(n, m)/3+1;

int reult = 1;

while (left < right)

{

int mid = (left + right) / 2;

bool not\_right = false;

bool find = false;

if (count >= mid\*mid\*5)

{

for (int i = 0; !find && i <= n - 3 \* mid; i++)

{

for (int j = mid; !find && j <= m - 2 \* mid; j++)

{

if (rows[i, j] >= mid && columns[i + mid, j - mid] >= mid)

{

bool flag = true;

for (int u = j; u < j + mid; u++)

{

if (columns[i, u] < 3 \* mid)

{

flag = false;

break;

}

}

if (!flag) continue;

for (int u = i; u < i + mid; u++)

{

if (rows[u + mid, j - mid] < 3 \* mid)

{

flag = false;

break;

}

}

if (!flag) continue;

if (flag)

{

find = true;

if (mid > reult) reult = mid;

}

}

}

}

}

if (find)

{

left = mid + 1;

}

else right = mid;

}

Console.WriteLine(reult);

}

}