H. Наилучший запрет

|  |  |
| --- | --- |
| Ограничение времени | 3 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Константин и Михаил играют в настольную игру «Ярость Эльфов». В игре есть *n* рас и *m* классов персонажей. Каждый персонаж характеризуется своими расой и классом. Для каждой расы и каждого класса существует ровно один персонаж такой расы и такого класса. Сила персонажа *i*-й расы и *j*-го класса равна *ai j*, и обоим игрокам это прекрасно известно.

Сейчас Константин будет выбирать себе персонажа. Перед этим Михаил может запретить одну расу **и** один класс, чтобы Константин не мог выбирать персонажей, у которых такая раса **или** такой класс. Конечно же, Михаил старается, чтобы Константину достался как можно более слабый персонаж, а Константин, напротив, выбирает персонажа посильнее. Какие расу и класс следует запретить Михаилу?

Формат ввода

Первая строка содержит два целых числа *n* и *m* (*2 ≤ n,m ≤ 1000*) через пробел — количество рас и классов в игре «Ярость Эльфов», соответственно.

В следующих *n* строках содержится по *m* целых чисел через пробел. *j*-е число *i*-й из этих строк — это *ai j* (*1 ≤ ai j ≤ 109*).

Формат вывода

В единственной строке выведите два целых числа через пробел — номер расы и номер класса, которые следует запретить Михаилу. Расы и классы нумеруются с единицы. Если есть несколько возможных ответов, выведите любой из них.

Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2 2  1 2  3 4 | 2 2 |

Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3 4  1 3 5 7  9 11 2 4  6 8 10 12 | 3 2 |

using System;

public struct Point

{

public int X;

public int Y;

public int number;

}

class Program

{

static void Main()

{

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

int n = int.Parse(input1[0]);

int m = int.Parse(input1[1]);

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

Point max = new Point();

max.number = 0;

Point max2 = new Point();

max2.number = -1;

int resN = 0, resM = 0;

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

{

string[] input = Console.ReadLine().Split();

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

{

power[i, j] = int.Parse(input[j]);

if (power[i, j] > max.number)

{

max2 = max;

max.number = power[i, j];

max.X = i;

max.Y = j;

}

else if (power[i, j] > max2.number)

{

max2.number = power[i, j];

max2.X = i;

max2.Y = j;

}

}

}

if (max.X == max2.X || max.Y == max2.Y)

{

bool Xs = max.X == max2.X;

Point max3 = new Point();

max3.number = 0;

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

{

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

{

if (power[i, j] > max3.number && ((Xs && i != max.X) || (!Xs && j != max.Y)))

{

max3.number = power[i, j];

max3.X = i;

max3.Y = j;

}

}

}

if (Xs)

{

resN = max.X;

resM = max3.Y;

}

else

{

resN = max3.X;

resM = max.Y;

}

}

else

{

resN = max.X;

resM = max2.Y;

Point max3 = new Point();

max3.number = 0;

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

{

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

{

if (power[i, j] > max3.number && ((i != max.X && i != max2.X) || (j != max.Y && j != max2.Y)))

{

max3.number = power[i, j];

max3.X = i;

max3.Y = j;

}

}

}

if (max3.X == max.X || max3.Y == max2.Y)

{

resN = max.X;

resM = max2.Y;

}

else if (max3.X == max2.X || max3.Y == max.Y)

{

resN = max2.X;

resM = max.Y;

}

}

resM++;

resN++;

Console.WriteLine(resN + " " + resM);

}

}