Задача: Filler

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	10 секунд на 10 ходов
Ограничение по памяти:	64 мегабайта

Вам предлагается сыграть в достаточно известную игру “Filler”. Игра происходит на поле, в данном случае размера 20×20. Каждая клетка поля изначально окрашена в один из 10 цветов. Изначально первый игрок *владеет* клеткой, находящейся в левом верхнем углу, а также всеми клетками, до которых можно добраться из неё, переходя в соседние клетки такого же цвета (соседними считаются клетки, имеющие общую сторону). Аналогично второй игрок *владеет* симметричной клеткой в правом нижнем углу.

На своём ходу игрок может выбрать любой цвет, кроме того, которым окрашена исходная клетка противника (левая верхняя или правая нижняя соответственно) и перекрасить все клетки, которыми он *владеет* в этот цвет. При этом игрок начинает *владеть* всеми клетками, до которых можно добраться из его начальной клетки, переходя в соседние клетки, имеющие одинаковый цвет с изначальной. Игра заканчивается когда каждая клетка принадлежит одному из игроков, либо по прошествии 400 ходов (что наступит раньше).

Формат входного файла

Перед каждым ходом в стандартный поток ввода Вашей программе будет передано текущее состояние поля, 20 строк по 20 чисел в каждой. Завершаться передача состояния будет пустой строкой. Каждое из этих чисел a_{ij} обозначает цвет соответствующей клетки поля (цвета нумеруются от 1 до 10). Поле всегда будет повёрнуто таким образом, что Вашей исходной клеткой будет левая верхняя (с координатами $(0, 0)$).

Формат выходного файла

На каждом ходу Вам требуется вывести номер цвета, в который Вы хотите перекрасить свои клетки. После вывода хода выведите пустую строку (итого выводить требуется два символа перевода строки). Не забывайте сбрасывать буфер выходного потока после каждого хода. В C/C++ это можно сделать с помощью `fflush(stdout);`, в Pascal с помощью `flush(output);`

Также не забывайте, что Ваша программа не должна завершаться сама по себе, потому после последнего хода можно например сделать фиктивную операцию чтения.

В примере приведен вывод для одного хода.

Примеры

стандартный ввод
8 2 4 9 2 8 7 1 6 4 1 7 6 2 9 4 7 7 8 6 6 6 1 2 5 5 1 5 1 5 6 7 5 4 5 7 6 3 5 6 4 1 1 3 1 2 1 7 2 2 8 8 6 8 3 10 1 6 10 1 9 8 4 1 6 6 7 1 3 5 5 2 3 5 9 8 6 1 5 5 1 3 10 9 2 6 10 4 9 8 3 6 7 8 3 2 3 2 9 1 9 7 2 8 4 5 5 10 5 5 3 5 8 7 9 9 7 1 3 5 5 1 6 5 1 1 6 8 7 3 4 10 3 10 7 7 6 1 10 6 6 8 1 6 10 10 5 1 9 3 1 2 4 8 9 7 1 9 3 6 5 5 7 6 3 2 8 7 10 4 3 9 1 2 5 1 4 6 4 10 7 7 3 6 7 9 9 1 3 2 5 1 7 7 7 4 6 6 5 1 8 6 8 4 8 5 10 9 5 5 7 7 3 7 6 6 8 10 8 3 5 5 10 8 10 1 8 8 5 6 8 2 5 3 3 6 7 9 2 2 6 9 10 3 4 3 7 9 2 1 10 8 7 9 4 2 3 1 8 9 4 8 6 1 1 1 2 4 2 5 7 5 1 8 7 6 5 9 5 2 4 6 7 10 3 6 4 6 2 6 2 10 8 4 10 1 10 6 5 5 4 9 4 9 4 6 1 7 10 7 1 3 2 10 8 3 1 9 8 9 6 9 8 10 8 8 2 3 2 1 7 6 2 7 6 5 8 1 10 3 10 7 9 3 5 10 4 10 10 8 9 5 8 6 6 4 8 7 10 6 8 4 7 7 1 5 3 4 8 3 3 10 5 6 7 10 2 4 6 2 1 5 9 3 6 10 9 4 2 1 8 7 6 6 3 7 3 5 3 7
стандартный вывод
6

Пример реализованной стратегии, на каждом ходу выбирающей случайный цвет (C++):

```
#include <iostream>
#include <fstream>
#include <ctime>
#include <cstdlib>

using namespace std;

const int SZ = 20;
int field[SZ][SZ];

int main()
{
    srand((unsigned)time(NULL));
    int i, j;
    while (1)
    {
        for (i = 0; i < SZ; i++)
        {
            for (j = 0; j < SZ; j++)
            {
                scanf("%d", &field[i][j]);
            }
        }
        int newc = rand() % 10 + 1;
        while (newc == field[SZ - 1][SZ - 1]) newc = rand() % 10 + 1;
        printf("%d\n\n", newc);
        fflush(stdout);
    }
    // чтобы программа не завершилась раньше времени
    scanf("%d", &i);
    return 0;
}
```

Пример реализованной стратегии, на каждом ходу выбирающей случайный цвет (Pascal):

```
const SZ = 20;
var field : array[1..SZ, 1..SZ] of integer;
i, j : integer;
newc : integer;
begin
randomize;
while true do begin
        for i := 1 to SZ do begin
         for j := 1 to SZ do begin
         read(field[i, j]);
         end;
        end;
        newc := trunc(random() * 1000) mod 10 + 1;
        while (newc = field[SZ, SZ]) do newc := trunc(random() * 1000) mod 10 + 1;
        writeln(newc);
        writeln;
        flush(output);
    end;
    read(i);
end.
```