

*Комаров Андрей Валерьевич,  
Кротков Павел Андреевич,  
Кучеренко Демид Сергеевич,  
Ульянцев Владимир Игоревич*

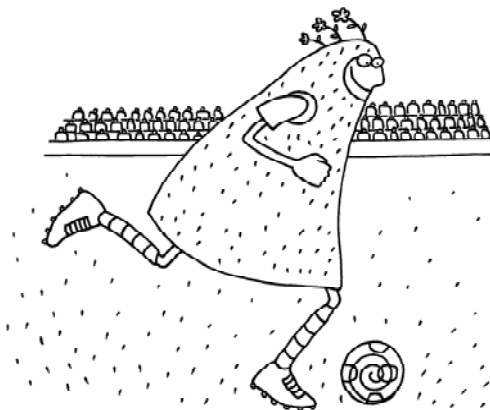
## ЗАДАЧА «ТЕЛЕСЪЕМКА»

Этой статьей мы продолжаем цикл публикаций олимпиадных задач по информатике для школьников. Решение таких задач и изучение разборов поможет повысить уровень практических навыков программирования и подготовиться к олимпиадам по информатике.

В этой статье рассматривается задача «Телесъемка», которая предлагалась на Второй командной интернет-олимпиаде по программированию в 2012–2013 учебном году. Материалы этой олимпиады можно найти на сайте <http://neerc.ifmo.ru/school/io/>.

### УСЛОВИЕ ЗАДАЧИ

Для съемок финального матча «Ротор – Закат» была нанята лучшая съемочная группа. Несмотря на то, что их работа была вы-



полнена на высочайшем уровне, после просмотра записи выяснилось, что один из игроков Заката ни разу не попал в кадр. Но тренеру интересны действия всех игроков, в том числе и того, которого не оказалось на записи – например, тренеру интересен маршрут, по которому такой игрок мог передвигаться на протяжении матча.

Для упрощения задачи будем считать, что игровое поле представляет собой клетчатый прямоугольник  $w \times h$  клеток. Каждую секунду каждый игрок обязательно перебегает в соседнюю по стороне клетку. Съемка же является последовательностью из  $n$  кадров, в каждом из которых виден некоторый подпрямоугольник поля, противоположные вершины которого расположены в клетках  $(x_{i1}, y_{i1})$  и  $(x_{i2}, y_{i2})$ . Поскольку известно, что этот игрок ни разу не попал в кадр, содержимое кадров не важно, важно лишь то, какой участок поля снимался в каждый момент времени. При этом съемка производится с частотой один кадр в секунду.

Ваша задача – помочь тренеру и восстановить какой-либо маршрут, по которому мог перемещаться игрок, не попавший в кадр на протяжении всего матча.

### Формат входного файла

В первой строке заданы натуральные числа  $w$  и  $h$  ( $1 \leq w, h \leq 300$ ) – ширина и дли-

на поля. Во второй строке задано число  $n$  ( $1 \leq n \leq 300$ ) – число кадров съемки.

В следующих строках задано по четыре числа  $x_{i1}, y_{i1}, x_{i2}$  и  $y_{i2}$  ( $1 \leq x_{i1} \leq x_{i2} \leq w$ ,  $1 \leq y_{i1} \leq y_{i2} \leq h$ ) – координаты левого нижнего и правого верхнего угла прямоугольника, соответствующего видимой на  $i$ -м кадре части поля.

Формат выходного файла

Выведите  $n$  строк, в каждой из которых должно быть по два натуральных числа  $x_i$  и  $y_i$  ( $1 \leq x_i \leq w$ ,  $1 \leq y_i \leq h$ ) – координаты клетки, в которой мог оказаться этот игрок на  $i$ -й секунде. Выведите 'Impossible', если не существует маршрута, перемещаясь по которому игрок мог ни разу не попасть в камеру.

Примеры входных и выходных данных

broadcast.in	broadcast.out
3 2	1 2
5	2 2
1 1 2 1	2 1
2 1 2 1	3 1
2 2 3 2	3 2
2 2 3 2	
2 1 2 2	

$$cell[t][x][y] = \begin{cases} -1, & \text{если в момент времени } t \text{ клетка } (x, y) \text{ попадает в кадр} \\ 1, & \text{если игрок может оказаться в клетке } (x, y) \text{ в момент времени } t \\ 0, & \text{если игрок не может оказаться в клетке } (x, y) \text{ в момент времени } t \end{cases} \quad (1)$$

**РАЗБОР ЗАДАЧИ**

Алгоритм, решающий данную задачу, является модификацией известного алгоритма поиска в ширину [1]. Сам поиск в ширину уже многократно описан в различной литературе, поэтому мы приведем только его упрощенный вариант, оптимально подходящий для решения этой задачи.

Определим *состояние* игрока как тройку чисел  $(t, x, y)$ , где  $x$  и  $y$  – координаты игрока на поле, а  $t$  – секунда, в которую он там находится. Чтобы решить задачу, посчитаем для каждого состояния значение  $cell[t][x][y]$  следующим образом (см. формулу (1) ниже).

В листинге 1 приведена реализация инициализации массива  $cell$ , в котором будут храниться эти значения.

Далее нам необходимо посчитать все остальные значения функции. Понятно, что в момент времени  $t = 1$  футболист может оказаться в любой клетке, не попадающей в первый кадр. Также можно заметить, что попасть в состояние  $(t, x, y)$  футболист мог только из состояния  $(t - 1, x', y')$ , где клетка

**Листинг 1.** Инициализация массива  $cell$

```

for i := 1 to MAXN do
  for j := 1 to MAXW do
    for k := 1 to MAXH do
      cell[i][j][k] := 0;
read(w, h);
read(n);
for i := 1 to n do begin
  read(ax, ay, bx, by);
  for x := ax to bx do
    for y := ay to by do
      cell[i][x][y] := -1;
end;
```

**Листинг 2.** Функция  $in\_bounds$

```

function in_bounds(w, h, x, y : longint) : boolean;
begin
  result := not ((x < 1) or (x > w) or (y < 1) or (y > h));
end;
```

$(x', y')$  соседствует с клеткой  $(x, y)$ .

В листинге 2 приведена реализация функции, проверяющей, лежит ли клетка с координатами  $(x, y)$  внутри поля размером  $w \times h$  клеток (в дальнейшем нам неоднократно понадобится это делать).

Кроме этого, в начале программы необходимо объявить два вспомогательных константных массива, которые позднее приго-

дятся нам для более удобного перебора всех соседей клетки  $(x, y)$ . Их объявление приведено в листинге 3.

Теперь у нас есть все необходимое для того, чтобы посчитать значения функции *cell* во всех состояниях так, как это определено выше (листинг 4).

Проверим, существует ли вообще путь, по которому футболист мог перемещаться,

### Листинг 3. Объявление массивов dx и dy

```
const
  dx : array [1..4] of longint = (-1, 1, 0, 0);
  dy : array [1..4] of longint = (0, 0, -1, 1);
```

### Листинг 4. Подсчет значений функции cell

```
for i := 1 to n do begin
  for x := 1 to w do begin
    for y := 1 to h do begin
      if (i = 1) then begin
        if (cell[i][x][y] = 0) then
          cell[i][x][y] := 1;
        continue;
      end;
      if (cell[i][x][y] = -1) then continue;
      for d := 1 to 4 do
        if ((in_bounds(w, h, x + dx[d], y + dy[d]))
          and (cell[i - 1][x + dx[d]][y + dy[d]] = 1)) then
          cell[i][x][y] := 1;
      end;
    end;
  end;
end;
```

### Листинг 5. Проверка существования ответа

```
fx := -1;
fy := -1;
possible := false;
for x := 1 to w do
  for y := 1 to h do
    if (cell[n][x][y] = 1) then begin
      possible := true;
      fx := x;
      fy := y;
    end;

if (not possible) then begin
  writeln('Impossible');
  close(input);
  close(output);
  halt;
end;
```

**Листинг 6.** Восстановление существующего ответа

```

for i := n downto 1 do begin
  ans_x[i] := fx;
  ans_y[i] := fy;
  if (i = 1) then break;
  for d := 1 to 4 do begin
    tx := fx + dx[d];
    ty := fy + dy[d];
    if ((in_bounds(w, h, tx, ty))
    and (cell[i - 1][tx][ty] = 1)) then begin
      fx := tx;
      fy := ty;
      break;
    end;
  end;
end;
end;
for i := 1 to n do
  writeln(ans_x[i], ' ', ans_y[i]);

```

не попадая ни в какой кадр. Заметим, что путь существует тогда и только тогда, когда футболист мог оказаться хотя бы в одном состоянии, которому соответствует время  $t = n$ . Реализация этой части программы приведена в листинге 5.

То, что после выполнения этой части программа не завершилась (`possible = true`), означает, что путь существует, и нам необходимо его восстановить и вывести. Последней клеткой подходящего пути могла ока-

заться клетка с координатами  $(fx, fy)$ , которую мы нашли на предыдущем этапе. Начнем восстановление пути с нее и  $n - 1$  раз повторим поиск той клетки, из которой мы могли прийти в текущую. Заодно будем записывать координаты клеток, которые мы проходим, в массивы `ans_x` и `ans_y`, которые затем будут выведены в качестве ответа. Реализация этой части программы приведена в листинге 6.

Заметим, что по приведенному коду не сложно оценить время его работы: суммарно программа выполнит  $O(n \times w \times h)$  операций, и время ее выполнения при данных в условии ограничениях однозначно уложится в ограничение на время работы программы.

**Комаров Андрей Валерьевич,**  
студент третьего курса кафедры  
«Компьютерные технологии»  
НИУ ИТМО, член жюри Интернет-  
олимпиад по информатике,

**Кротков Павел Андреевич,**  
студент третьего курса кафедры  
«Компьютерные технологии»  
НИУ ИТМО, член жюри Интернет-  
олимпиад по информатике,

**Кучеренко Демид Сергеевич,**  
студент второго курса кафедры  
«Компьютерные технологии»  
НИУ ИТМО, член жюри Интернет-  
олимпиад по информатике,

**Ульянцев Владимир Игоревич,**  
студент шестого курса кафедры  
«Компьютерные технологии»  
НИУ ИТМО, член жюри Интернет-  
олимпиад по информатике.

#### Литература

1. Скиена С. Алгоритмы. Руководство по разработке. 2-е изд.: Пер. с англ. СПб.: БХВ-Петербург, 2011.



Наши авторы, 2013.  
Our authors, 2013.