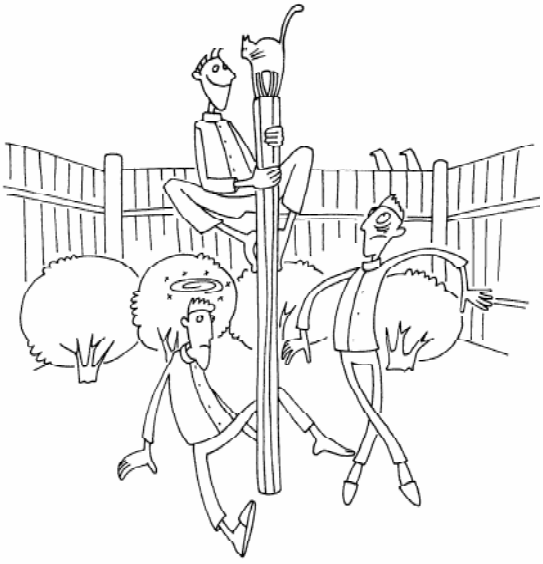
Кротков Павел Андреевич,
Ульянцев Владимир Игоревич,
Цыпленков Алексей Евгеньевич

ЗАДАЧА «ЗАБОР»

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

В этой статье рассматривается задача «Забор», которая предлагалась на первой интернет-олимпиаде базового уровня сезона 2012–2013. Интернет-олимпиады по информатике базового уровня проводятся Университетом ИТМО. Сайт этих олимпиад находится по адресу http://neerc.ifmo.ru/school/io.



УСЛОВИЕ ЗАДАЧИ

Забор вокруг больницы Принстон-Плейнсборо был построен задолго до появления там доктора Хауса, доктора Кадди и всех остальных ныне там работающих сотрудников. На плане больничной территории забор представляет собой выпуклый многоугольник. В вершинах этого многоугольника стоят столбы, а его ребрами являются прямые секции забора. Однако при строительстве забора по ошибке был построен лишний столб, который в итоге оказался внутри забора, то есть на территории больницы, и причинял персоналу и больным много неудобств.

Доктор Кадди решила исправить это досадное упущение, построив вокруг больницы новый забор, который должен удовлетворять трем следующим условиям:

– вершинами многоугольника, задающего новый забор, могут быть только уже существующие столбы (вершины старого забора и столб внутри него);

– внутри многоугольника, представляющего новый забор, не должно быть столбов, не являющихся вершинами многоугольника;

– площадь, огороженная новым забором, должна быть максимально возможной.

При этом многоугольник, представляющий новый забор, может быть невыпуклым, а некоторые столбы, находящиеся за его тер-

риторией, могут остаться неиспользованными. Помогите доктору Кадди проработать план нового забора.

Формат входного файла

Первая строка входного файла содержит одно целое число n ($3 \le n \le 10^5$) – количество вершин в многоугольнике, представляющем старый забор. Следующие n строк содержат по два целых числа x и y, не превосходящих по абсолютной величине 10^8, – координаты вершин этого многоугольника. Координаты даны в порядке обхода многоугольника против часовой стрелки, многоугольник выпуклый. Никакие три вершины многоугольника не лежат на одной прямой.

Последняя строка входного файла содержит два целых числа X и Y, не превосходящих по абсолютной величине 10^8, – координаты лишнего столба. Гарантируется, что точка (X, Y) лежит строго внутри многоугольника.

Формат выходного файла

Опишите многоугольник, представляющий новый забор, в том же формате, в котором описан старый. Внутри нового забора не должно быть столбов и его площадь должна быть максимальна.

Примеры входных и выходных данных

fence.in	fence.out
4 -1 -1 1 -1 1 1 -1 1 0 0	5 -1 -1 0 0 -1 1 1 1 1 -1

РАЗБОР ЗАДАЧИ

Докажем, что ответом задачи является многоугольник с $n + 1$ вершинами. Докажем от противного – пусть это не так и можно выбрать удовлетворяющий многоугольник с m вершинами. Выберем ближайшую к выбранному многоугольнику из не входящих в него точку A. Пусть BC – ближайшее к A ребро многоугольника (см. рис. 1). Заметим, что если добавить в выбранный многоугольник между точками B и C точку A, то площадь полученного многоугольника увеличится на площадь треугольника ABC, причем в полученном многоугольнике не будет столбов, не являющихся вершинами. Получено противоречие тому, что площадь выбранного многоугольника была максимальна.

Построим многоугольник, использующий все $n + 1$ столбов как вершины. Заметим, что подходящими многоугольниками являются те и только те, которые можно получить, заменив одно произвольное ребро BC исходного многоугольника на два ребра BX и CX, где X – заданная точка внутри старого многоугольника. Назовем эти многоугольники «хорошими». Любой другой многоугольник, построенный на $n + 1$ точке, будет иметь самопересечения, следовательно, один из хороших многоугольников будет его полностью покрывать, при этом его площадь будет гарантированно больше.

Выберем один из n хороших многоугольников. Заметим, что каждый такой многоугольник получается из исходного «отрезанием» треугольника BCX. Так как необходимо максимизировать площадь искомого многоугольника, то отрезать нужно треугольник с минимальной площадью. Для решения задачи переберем все такие треугольники, посчитаем площадь каждого и выберем минимальную. Чтобы восстановить ответ, запомним номер ребра, на котором был построен треугольник с минимальной площадью, и в обходе вставим в него вершину внутри старого многоугольника.

Для того чтобы не выделять в отдельный случай ребро, соединяющее первую и последнюю описанные во входном файле вершины многоугольника, можно добавить ко-

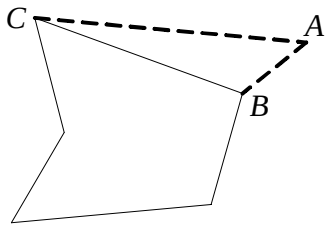


Рис. 1. Пример увеличения площади многоугольника

пию первой вершины в конец списка всех вершин исходного многоугольника.

При решении задачи также следует обратить внимание, что координаты точек по модулю не превосходят 10^8, соответственно площадь треугольников может достигать 10^{16}, а значит, использования типов **longint** или **int** для хранения значения площади невозможно, необходимо использовать тип **int64**. Более того, использование для вычисления площади треугольника формулы Герона невозможно, так как при промежуточных вычислениях может быть получено значение порядка 10^{32}.

Кроме того, при решении задачи могут возникнуть проблемы с точностью опреде-

Листинг 1. Реализация описанного алгоритма

```
var
  x, y : array [1..100001] of int64;
  pointX, pointY : int64;
  i, n, ansPos : longint;
  minSquare, curSquare : int64;

function calcSquare(x1, y1, x2, y2, x3, y3 : int64) : int64;
begin
  result := abs((x1 - x2) * (y3 - y2) - (x3 - x2) * (y1 - y2));
end;

begin
  reset(input, "fence.in");
  rewrite(output, "fence.out");
  readln(n);
  for i := 1 to n do begin
    readln(x[i], y[i]);
  end;
  readln(pointX, pointY);

  minSquare := high(int64);
  ansPos := -1;
  x[n + 1] := x[1];
  y[n + 1] := y[1];

  for i := 1 to n do begin
    curSquare := calcSquare(pointX, pointY, x[i], y[i],
                            x[i + 1], y[i + 1]);
    if (curSquare < minSquare) then begin
      minSquare := curSquare;
      ansPos := i;
    end;
  end;

  writeln(n + 1);
  for i := 1 to n do begin
    writeln(x[i], " ", y[i]);
    if (i = ansPos) then begin
      writeln(pointX, " ", pointY);
    end;
  end;
end.
```

ления площади. Чтобы от них избавиться, можно было считать площади треугольников с помощью псевдовекторного произведения. Так как все данные точки имели целые координаты, то площади треугольников будут полуцелыми. Если же считать удвоенные площади, то от вещественных чисел можно избавиться полностью. Таким образом, приведенное решение работает за время $O(n)$, что укладывается в ограничения задачи.

В листинге 1 приведена реализация такого решения на языке Pascal.

Кротков Павел Андреевич,
студент третьего 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,

Цыпленков Алексей Евгеньевич,
студент третьего курса кафедры «Компьютерные технологии» НИУ ИТМО, член жюри Интернет-олимпиад по информатике.