Школа современного программирования

ГОТОВИМСЯ К ОЛИМПИАДАМ ПО ИНФОРМАТИКЕ

Ахи Антон Андреевич,
Буздалов Максим Викторович,
Торопов Александр Владимирович,
Царёв Фёдор Николаевич

ЗАДАЧА «СИСТЕМА ГЛОБАЛЬНЕЙШЕГО ПОЗИЦИОНИРОВАНИЯ»

Этой статьей мы продолжаем цикл публикаций олимпиадных задач для школьников по информатике и программированию с разборами.

В этой статье рассматривается задача «Система глобальнейшего позиционирования», которая предлагалась в первой Интернет-олимпиаде сезона 2009–2010 (олимпиада состоялась 19 сентября 2009 года). Интернет-олимпиады по информатике проводятся Санкт-Петербургским государственным университетом информационных технологий, механики и оптики (сайт http://neerc.ifmo.ru/school/io/).

УСЛОВИЕ ЗАДАЧИ

Недавно во Флатландии было решено создать Систему Глобальнейшего Позиционирования. Поскольку страна занимает бесконечно большой участок плоскости, то вывод спутников очень затруднителен, поэтому было решено ограничиться наземным методом позиционирования.

Для этого во Флатландии было построено три радиовышки, не находящиеся на одной прямой. Объект, который хочет узнать свое местоположение, принимает сигнал от каждой из трех вышек (рис. 1). По силе сигнала, дошедшего от каждой из вышек, определяется расстояние между вышками и объектом.

Напишите программу, которая реализует последний компонент системы, который, получая координаты вышек и расстояния от объекта до каждой из них, находит координаты объекта.

Формат входного файла

В первой строке входного файла находятся три пары чисел x_1, y_1, x_2, y_2, x_3 и y_3 – координаты вышек. Во второй строке заданы три неотрицательных числа – расстояния до соответствующих вышек.

Все числа во входном файле целые и по абсолютной величине не превышают 50.

Формат выходного файла

Если не существует такого местоположения объекта, что расстояния до вышек соответствовали бы данным, то выведите в выходной файл единственное слово **Impossible**.

Иначе выведите два числа – координаты объекта. Ответ будет проверяться с точностью до шести знаков после запятой.

Примеры входных и выходных данных

gps.in	gps.out
0 4 2 6 5 0 2 2 5	2.000000 4.000000
0 0 0 3 1 -4 4 5 5	4.000000 0.000000
0 0 1 0 0 1 2 2 2	Impossible

РАЗБОР ЗАДАЧИ

Рассматриваемая задача относится к области аналитической геометрии, в ее разборе акцент делается на математической части задачи, а не на программной реализации.

В задаче требуется найти точку, находящуюся на расстоянии d_1 от точки $A = (x_1, y_1)$, на расстоянии d_2 от точки $B = (x_2, y_2)$ и на расстоянии d_3 от точки $C = (x_3, y_3)$.

Эта задача эквивалента тому, чтобы найти точку пересечения трех окружностей с центрами в точках A, B, C и радиусами соответственно d_1, d_2 и d_3.

Любые две окружности либо не имеют точек пересечения, либо пересекаются в одной или двух точках, либо совпадают. Последний случай не может быть реализован в рассматриваемой задаче, так как по условию точки A, B и C не лежат на одной прямой (если бы хотя бы две из них совпадали, то все три точки лежали бы на одной прямой).

Таким образом, для решения задачи достаточно выполнить следующие действия:

1. Выбрать любые две из трех данных окружностей.

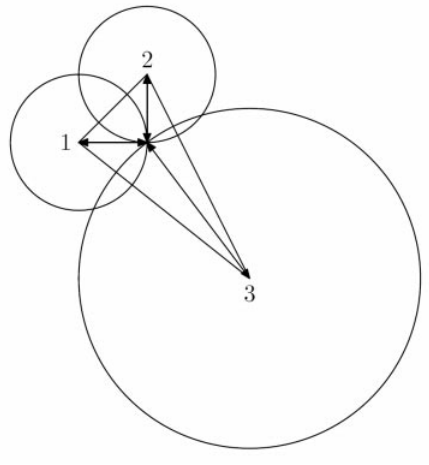

Рис. 1

2. Найти их точки пересечения (их окажется не более двух).
3. Проверить все найденные точки на то, являются ли они ответом.
4. Если ответ найден, то вывести его, иначе вывести **Impossible**.

Общие точки двух окружностей можно найти следующим образом. Пусть первая окружность имеет центр (x_1, y_1) и радиус d_1, а вторая окружность имеет центр (x_2, y_2) и радиус d_2. Тогда любая общая точка окружностей (x, y) является решением следующей системы уравнений:

$$\begin{cases}(x-x_1)^2+(y-y_1)^2=d_1^2\\(x-x_2)^2+(y-y_2)^2=d_2^2\end{cases}$$

Для решения этой системы уравнений вычтем из второго уравнения первое:

$$\begin{cases}(x-x_1)^2+(y-y_1)^2=d_1^2\\2x(x_1-x_2)+2y(y_1-y_2)=d_2^2-d_1^2+x_1^2+y_1^2-x_2^2-y_2^2\end{cases}$$

Заметим, что второе уравнение получилось линейным относительно x и y. Если верно неравенство $x_1 - x_2 \neq 0$, то из второго уравнения можно выразить x через y и подставить полученное выражение в первое уравнение. Получим квадратное уравнение относительно y, корни которого соответствуют общим точкам исходных окружностей.

Так же можно поступить при $y_1 - y_2 \neq 0$. Одновременно равенства $x_1 - x_2 = 0$ и $y_1 - y_2 = 0$ выполняться не могут, так как

это означало бы совпадение двух точек во входном файле (ранее показано, что такая ситуация невозможна).

Покажем, что при существовании ответа искомая точка является единственной. Пусть три окружности имеют две общие точки – D и E (больше двух общих точек у них не может быть, так как любые две из этих окружностей имеют не более двух общих точек).

Рассмотрим множество центров окружностей, на которых лежат точки D и E, то есть геометрическое место точек, равноудаленных от D и E. Известно, что это множество точек есть срединный перпендикуляр к отрезку DE. Таким образом, центры окружностей лежат на одной прямой, что противоречит условию задачи.

Отдельно стоит отметить, что хотя координаты центров окружностей, а также их радиусы целочисленны, координаты ответа могут быть нецелыми. Так, при $A = (-10; -10)$, $B = (-7; -8)$, $C = (-8; 0)$, $d_1 = 2$, $d_2 = 5$, $d_3 = 10$ ответом будет точка с координатами $(-\frac{154}{13}; -\frac{120}{13})$. Интересен тот факт, что координаты ответа в этой задаче всегда рациональны. Доказательство этого факта остается читателю в качестве упражнения.

В листинге 1 приведена программная реализация описанного алгоритма на языке программирования *Java*.

Листинг 1. Реализация алгоритма на языке программирования *Java*

```
import java.io.*;
import java.util.*;
public class gps implements Runnable {
  final String imp = "Impossible";
  final double eps = 1e-9;
    private void solve() throws IOException {
      int[][] c = new int[3][2];
      for (int i = 0; i < 3; ++i) {
        for (int j = 0; j < 2; ++j) {
             c[i][j] = in.nextInt();
        }
      }
      int[] r = new int[3];
      for (int i = 0; i < 3; ++i) {
           r[i] = in.nextInt();
      }

      double[][] cp = cross(c[0], r[0], c[1], r[1]);
      int cnt = 0;
      double[] ans = null;
      for (double[] p : cp) {
          if (Math.abs(Math.hypot(c[2][0]-p[0], c[2][1]-p[1])-r[2])< eps) {
              cnt++;
              ans = p;
           }
      }

      if (cnt != 1) {
          out.println("Impossible");
      } else {
          out.println(ans[0] + " " + ans[1]);
      }
    }
```

```
    double[][] cross(int[] c1, int r1, int[] c2, int r2) {
      long a = 2 * (c2[0] - c1[0]);
      long b = 2 * (c2[1] - c1[1]);
      long c = c1[0] * c1[0] + c1[1] * c1[1] - c2[0] * c2[0] -
               c2[1] * c2[1] - r1 * r1 + r2 * r2 + a * c1[0] + b * c1[1];
      double[][] ans = cross(r1, a, b, c);
      for (int i = 0; i < ans.length; ++i) {
          for (int j = 0; j < 2; ++j) {
              ans[i][j] += c1[j];
          }
      }
      return ans;
    }

    double[][] cross(long r, long a, long b, long c) {
      ArrayList<double[]> ans = new ArrayList<double[]>();
      long t2 = r * r * (a * a + b * b) - c * c;
      if (t2 < 0 || a == 0 && b == 0) {
           return ans.toArray(new double[ans.size()][]);
      }
      double t = Math.sqrt(t2);
      for (int s = -1; s <= 1; s += 2) {
           ans.add(new double[]{(- a * c + s * b * t) / (a * a + b * b),
                                (- b * c - s * a * t) / (a * a + b * b)});
           if (t2 == 0) break;
      }
      return ans.toArray(new double[ans.size()][]);
    }

    final String FILE_NAME = "gps";

    void myAssert(boolean e, String msg) {
      if (!e) {
         throw new Error(msg);
      }

    SimpleScanner in;
    PrintWriter out;

    public void run() {
      try {
          in = new SimpleScanner(new FileReader(FILE_NAME + ".in"));
          out = new PrintWriter(FILE_NAME + ".out");
          solve();
          out.close();
      } catch (Throwable e) {
          e.printStackTrace();
          System.exit(-1);
      }
    }

    public static void main(String[] args) {
      new Thread(new gps()).start();
    }
}

class SimpleScanner extends BufferedReader {

  private StringTokenizer st;
  private boolean eof = false;
```

```
  public SimpleScanner(Reader a) {
      super(a);
  }

  String next() {
    while (st == null || !st.hasMoreElements()) {
      try {
          st = new StringTokenizer(readLine());
      } catch (Exception e) {
          eof = true;
          return "";
      }
    }
    return st.nextToken();
  }

  private String cnv(String s) {
    if (s.length() == 0) {
        return "0";
    }
    return s;
  }

  int nextInt() {
    return Integer.parseInt(cnv(next()));
  }

  double nextDouble() {
    return Double.parseDouble(cnv(next()));
  }

  long nextLong() {
    return Long.parseLong(cnv(next()));
  }
}
```


Члены жюри Интернет-олимпиад по информатике базового уровня:

Ахи Антон Андреевич, студент кафедры «Компьютерные технологии» (КТ) СПбГУ ИТМО, финалист чемпионата мира по программированию среди студентов 2010 года,

Буздалов Максим Викторович, магистрант кафедры КТ СПбГУ ИТМО, чемпион мира по программированию среди студентов 2009 года,

Торопов Александр Владимирович, студент кафедры КТ СПбГУ ИТМО,

Царёв Фёдор Николаевич, аспирант кафедры КТ СПбГУ ИТМО, чемпион мира по программированию среди студентов 2008 года.