OJ 11343

Isolated Segments

Prof. Edson Alves

Faculdade UnB Gama

OJ 11343 – Isolated Segments

Problema

You're given n segments in the rectangular coordinate system. The segments are defined by start and end points (X_i, Y_i) and (X_j, Y_j) $(1 \leq i, j \leq n)$. Coordinates of these points are integer numbers with real value smaller then 1000. Length of each segment is positive.

When 2 segments don't have a common point then it is said that segments don't collide. In any other case segments collide. Be aware that segments collide even if they have only one point in common.

Segment is said to be isolated if it doesn't collide with all the other segments that are given, i.e. segment i is isolated when for each $1 \leq j \leq n$, $(i \neq j)$, segments i and j don't collide. You are asked to find number T – how many segments are isolated.

Entrada e saída

Input

First line of input contains number N ($N \leq 50$), then tests follow. First line of each test case contains number M ($M \leq 100$) — the number of segments for this test case to be considered. For this particular test case M lines follow each containing a description of one segment. Segment is described by 2 points: start point (X_{pi}, Y_{pi}) and end point (X_{ei}, Y_{ei}). They are given in such order: X_{pi} Y_{pi} X_{ei} Y_{ei}.

Entrada e saída

Output

For each test case output one line containing number T.

Exemplo de entradas e saídas

Sample Input

```
6
3
0 0 2 0
1 -1 1 1
2 2 3 3
2
0 0 1 1
1 0 0 1
2
0 0 0 1
0 2 0 3
2
0 0 1 0
1 0 2 0
2
0 0 2 2
1 0 1 1
2
1 3 1 5
1 0 1 6
```

Sample Output

```
1
0
2
0
0
0
```

Observações sobre o problema

- O problema consiste na identificação da interseção entre segmentos

Observações sobre o problema

- O problema consiste na identificação da interseção entre segmentos
- A verificação de interseção entre segmentos consiste em duas etapas

Observações sobre o problema

- O problema consiste na identificação da interseção entre segmentos
- A verificação de interseção entre segmentos consiste em duas etapas
- A primeira etapa é checar se as retas r e s que contém os segmentos se interceptam

Observações sobre o problema

- O problema consiste na identificação da interseção entre segmentos
- A verificação de interseção entre segmentos consiste em duas etapas
- A primeira etapa é checar se as retas r e s que contém os segmentos se interceptam
- Em caso positivo é preciso ainda ver se o ponto de interseção está contido nos segmentos ou não

Observações sobre o problema

- O problema consiste na identificação da interseção entre segmentos
- A verificação de interseção entre segmentos consiste em duas etapas
- A primeira etapa é checar se as retas r e s que contém os segmentos se interceptam
- Em caso positivo é preciso ainda ver se o ponto de interseção está contido nos segmentos ou não
- É preciso atentar aos *corner cases*

Observações sobre o problema

- O problema consiste na identificação da interseção entre segmentos
- A verificação de interseção entre segmentos consiste em duas etapas
- A primeira etapa é checar se as retas r e s que contém os segmentos se interceptam
- Em caso positivo é preciso ainda ver se o ponto de interseção está contido nos segmentos ou não
- É preciso atentar aos *corner cases*
- Como é preciso, para cada segmento i, testar todos os segmentos j, a solução tem complexidade $O(M^2)$

Observações sobre o problema

- O problema consiste na identificação da interseção entre segmentos
- A verificação de interseção entre segmentos consiste em duas etapas
- A primeira etapa é checar se as retas r e s que contém os segmentos se interceptam
- Em caso positivo é preciso ainda ver se o ponto de interseção está contido nos segmentos ou não
- É preciso atentar aos *corner cases*
- Como é preciso, para cada segmento i, testar todos os segmentos j, a solução tem complexidade $O(M^2)$
- Atenção ao produto dos discriminantes, que pode exceder a capacidade de um inteiro, de modo que é preciso usar o tipo **`long long`** em seu retorno

Solução AC com complexidade $O(N^2)$

```
using ll = long long;

struct Point { ll x, y; };

ll D(const Point& P, const Point& Q, const Point& R)
{
    return (P.x * Q.y + P.y * R.x + Q.x * R.y) - (R.x * Q.y + R.y * P.x + Q.x * P.y);
}

struct Segment
{
    Point A, B;

    bool contains(const Point& p) const
    {
        return (A.x == B.x) ? min(A.y, B.y) <= p.y && p.y <= max(A.y, B.y)
            : min(A.x, B.x) <= p.x && p.x <= max(A.x, B.x);
    }
```

Solução AC com complexidade $O(N^2)$

```
23      bool intersect(const Segment& s) const
24      {
25          auto d1 = D(A, B, s.A), d2 = D(A, B, s.B);
26
27          if ((d1 == 0 && contains(s.A)) || (d2 == 0 && contains(s.B)))
28              return true;
29
30          auto d3 = D(s.A, s.B, A), d4 = D(s.A, s.B, B);
31
32          if ((d3 == 0 && s.contains(A)) || (d4 == 0 && s.contains(B)))
33              return true;
34
35          return (d1 * d2 < 0) && (d3 * d4 < 0);
36      }
37  };
38
39  int solve(const vector<Segment>& vs, int M)
40  {
41      auto ans = 0;
```

Solução AC com complexidade $O(N^2)$

```
43      for (int i = 0; i < M; ++i)
44      {
45          int hits = 0;
46
47          for (int j = 0; j < M; ++j)
48          {
49              if (i == j) continue;
50
51              hits += vs[i].intersect(vs[j]) ? 1 : 0;
52          }
53
54          ans += (hits == 0) ? 1 : 0;
55      }
56
57      return ans;
58  }
```