**CHUYÊN ĐỀ DUYÊN HẢI MÔN TIN HỌC**

**GIẢI CÁC BÀI TOÁN HÌNH HỌC TRONG TIN HỌC**

**MỤC LỤC**

[I. Mở đầu 3](#_Toc81900365)

[II. Những công thức hình học cơ bản 3](#_Toc81900366)

[III. Một số bài tập áp dụng 5](#_Toc81900367)

[Bài 1: Vị trí tương đối giữa hai đoạn thẳng 5](#_Toc81900368)

[Bài 2: Loại hình tứ giác 7](#_Toc81900369)

[Bài 3: Loại đa giác 8](#_Toc81900370)

[Bài 4: Bao lồi 11](#_Toc81900371)

[Bài 5: Dãy tam giác bao nhau 14](#_Toc81900372)

[Bài 6. Diện tích đa giác tự cắt 16](#_Toc81900373)

[IV. Các bài tập tham khảo thêm 20](#_Toc81900375)

[Bài 1: Đếm hình chữ nhật 20](#_Toc81900376)

[Bài 2: Điểm và đa giác 20](#_Toc81900377)

[Bài 3: Đổi đất 20](#_Toc81900378)

[Bài 4: Dãy hình chữ nhật lồng nhau 21](#_Toc81900379)

[Bài 5: Chia đất 22](#_Toc81900380)

[V. Kết luận 23](#_Toc81900381)

**CHUYÊN ĐỀ GIẢI CÁC BÀI TOÁN HÌNH HỌC TRONG TIN HỌC**

# I. Mở đầu

Các bài toán hình học trong tin học rất đa dạng và phong phú. Mỗi bài có thể kết hợp nhiều thuật toán cơ bản khác nhau ở các chuyên đề như Quy hoạch động, Đồ thị,… kết hợp với các công thức hình học giải tích cơ bản cho ra kết quả của bài toán. Trong chuyên đề này tôi trình bày một số bài toán hình học có các cách kết hợp các thuật toán khác nhau.

# II. Những công thức hình học cơ bản

***1. Khoảng cách giữa 2 điểm A(x1,y1), B(x2,y2) trong mặt phẳng***



***2. Khoảng cách từ một điểm M(x0,y0) đến đường thẳng d có phương trình tổng quát ax + by + c = 0:***



***3. Cho 3 điểm phân biệt A(x1,y1), B(x2,y2), M(x3,y3) cùng nằm trên một đường thẳng, xét M có nằm giữa A, B hay không?***

Điều kiện là: (x1 - x3)(x3 - x2) >0 hoặc (y1 - y3)(y3 - y2) >0

***4. Phương trình đường thẳng đi qua hai điểm A(x1,y1), B(x2,y2):***



đưa về dạng tổng quát: (y1 - y2)x + (x2 - x1)y + (x1y2 - x2y1) = 0

***5. Vị trí tương đối giữa hai đường thẳng d1: a1x + b1y = c1,***

***d2: a2x + b2y = c2***

Ta tính các định thức:

D = a1b2 - a2b1; Dx = c1b2 - c2b1; Dy = a1c2 - a2c1.

Nếu (D = 0) và (Dx = 0) và (Dy = 0) thì d1 và d2 trùng nhau;

Nếu (D =0) và ((Dx <> 0) hoặc (Dy <> 0)) thì d1 song song với d2;

Nếu D <> 0 thì hai đường thẳng cắt nhau tại x=Dx/D; y=Dy/D.

***6. Hai đoạn thẳng giao nhau***

Cho hai đoạn thẳng được xác định bởi d1: A(x1,y1), B(x2,y2),

d2: (x3,y3), D(x4,y4).

*a, Cách 1*

Tìm giao điểm của 2 đường thẳng AB và CD, sau đó kiểm tra xem giao điểm có thuộc đồng thời cả hai đoạn thẳng AB và CD hay không.

*b, Cách 2*

Xây dựng hai phương trình đường thẳng đi qua AB (F) và CD (G). Nếu AB cắt CD thì C, D nằm về hai phía của đường thẳng AB tức là F(x3,y3)\*F(x4,y4)<0. Còn nếu CD cắt AB thì A, B nằm về hai phía của đường thẳng CD tức là G(x1,y1)\*G(x2,y2)<0.

Vậy để đoạn thẳng AB giao với đoạn thẳng CD ta cần có:

F(x3,y3)\*F(x4,y4)<0 và G(x1,y1)\*G(x2,y2)<0

***7. Công thức tính diện tích đa giác không tự cắt***

Cho đa giác không tự cắt A có n đỉnh: A1, A2, ..., An, công thức tính diện tích đa giác là:

 hoặc 

trong đó tọa độ đỉnh 1 và đỉnh n+1 trùng nhau.

# III. Một số bài tập áp dụng

***Các test tham khảo: https://www.mediafire.com/file/u7s4gag6p3brkh1/CD+HH+2021.rar/file***

Bài 1: Vị trí tương đối giữa hai đoạn thẳng

Trong hệ trục toạ độ ĐềCác, cho 2 đoạn thẳng AB và CD xác định bởi toạ độ các điểm A(x1,y1), B(x2,y2), C(x3,y3), D(x4,y4). Xét vị trí tương đối của hai đoạn thẳng trên xem chúng trùng nhau tại một điểm (1), trùng nhau nhiều điểm (2), cắt nhau (3), hay không có điểm chung nào (4)?

- Dữ liệu vào file **HAIDOANTHANG.INP**: Dòng đầu là tọa độ của A và B; dòng thứ hai là tọa độ của C và D.

- Kết quả ra file **HAIDOANTHANG.OUT** ghi ra số 1, hoặc 2, hoặc 3, hoặc 4 tương ứng các trường hợp trên.

**Ví dụ:**

|  |  |
| --- | --- |
| **HAIDOANTHANG.INP** | **HAIDOANTHANG.OUT** |
| 5 0 0 5  0 0 5 5 | 3 |

**Thuật toán:**

*- Tính hệ số ptđt đi qua AB và CD;*

*- Tính d=a1\*b2-a2\*b1; dx, dy;*

*- Nếu d==0 {*

*+ nếu dx!=0 hoặc dy!=0 =>4;*

*+ nếu dx==0 và dy==0 xét các trường hợp A với CD, B với CD, C với AB, D với AB nếu nằm trong =>2;*

*còn lại nếu trùng vào các mốc =>1;*

*- Còn lại Nếu d!=0, kiểm tra 2 đoạn thẳng cắt nhau, ta xét nếu tdt(A,B,C)\*tdt(A,B,D)<0 và tdt(C,D,A)\*tdt(C,D,B)<0 => 3;*

*còn lại =>4.*

**Chương trình:**

#include <bits/stdc++.h>

using namespace std;

fstream fi,fo;

struct diem

{

double x,y;

};

diem a,b,c,d;

double a1,b1,c1,a2,b2,c2,dd,dx,dy;

void dt(diem a,diem b,double &a1,double &b1,double &c1)

{

a1=b.y-a.y;

b1=a.x-b.x;

c1=-a.x\*b.y+b.x\*a.y;

}

double tdt(diem a,double a1,double b1,double c1)

{

return (a1\*a.x+b1\*a.y+c1);

}

int main()

{ fi.open("Haidoanthang.inp",ios::in);

fo.open("Haidoanthang.out",ios::out);

fi>>a.x>>a.y>>b.x>>b.y;

fi>>c.x>>c.y>>d.x>>d.y;

dt(a,b,a1,b1,c1);

dt(c,d,a2,b2,c2);

dd=a1\*b2-a2\*b1;

dx=-c1\*b2+c2\*b1;

dy=-a1\*c2+a2\*c1;

if (dd==0)

{

if (dx!=0||dy!=0) fi<<4;

else

{

if (((a.x-c.x)\*(c.x-b.x)>0||(a.y-c.y)\*(c.y-b.y)>0)||

((a.x-d.x)\*(d.x-b.x)>0||(a.y-d.y)\*(d.y-b.y)>0)||

((c.x-a.x)\*(a.x-d.x)>0||(c.y-a.y)\*(a.y-d.y)>0)||

((c.x-b.x)\*(b.x-d.x)>0||(c.y-b.y)\*(b.y-d.y)>0)) fo<<2;

else

if((a.x==c.x&&a.y==c.y)||(b.x==c.x&&b.y==c.y)||

(a.x==d.x&&a.y==d.y)||(b.x==d.x&&b.y==d.y))fo<<1;

else fo<<4;

} }

else

{if (tdt(c,a1,b1,c1)\*tdt(d,a1,b1,c1)<0&&tdt(a,a2,b2,c2)\*tdt(b,a2,b2,c2)<0) fo<<3;

else

if(tdt(c,a1,b1,c1)==0||tdt(d,a1,b1,c1)==0||tdt(a,a2,b2,c2)==0||tdt(b,a2,b2,c2)==0)fo<<1;

else fo<<4;

}

return 0;

}

## Bài 2: Loại hình tứ giác

Cho tứ giác lồi ABCD có toạ độ lần lượt là: (x1,y1), (x2,y2), (x3,y3), (x4,y4). Hãy xét xem tứ giác ABCD là hình gì? Trong các hình sau: Hình bình hành (1); Hình thoi (2); Hình chữ nhật (3); Hình vuông (4); Hình tứ giác thường (5) ?

- Dữ liệu vào file **TUGIACLOI.INP** gồm 4 dòng, mỗi dòng là số toạ độ của các điểm đã nêu ở trên.

- Kết quả ra file **TUGIACLOI.OUT** ghi ra số 1, hoặc 2, hoặc 3, hoặc 4, hoặc 5 tương ứng các trường hợp trên.

**Ví dụ:**

|  |  |
| --- | --- |
| **TUGIACLOI.INP** | **TUGIACLOI.OUT** |
| 0 0  0 5  5 5  5 0 | 4 |

**Thuật toán:**

*- ĐK HBH có 2 cặp cạnh đối song song và bằng nhau (xét hai cặp vector bằng nhau)*

*{ + ĐK Hình Vuông có 4 cạnh bằng nhau, đường chéo bằng nhau (4);*

*còn lại + ĐK Hình Chữ Nhật đường chéo bằng nhau (3);*

*còn lại + ĐK Hình Thoi có 4 cạnh bằng nhau (2);*

*còn lại HBT thường (1)*

*}*

*else còn lại là hình tứ giác thường.*

**Chương trình:**

#include <bits/stdc++.h>

using namespace std;

fstream fi,fo;

struct diem

{

int x,y;

};

diem a,b,c,d;

bool ss(diem a,diem b, diem c, diem d)

{

if (b.x-a.x==c.x-d.x&&b.y-a.y==c.y-d.y) return true;

else return false;

}

int kc(diem a, diem b)

{

return ((a.x-b.x)\*(a.x-b.x)+(a.y-b.y)\*(a.y-b.y));

}

int main()

{ fi.open("Tugiacloi.inp",ios::in);

fo.open("Tugiacloi.out",ios::out);

fi>>a.x>>a.y>>b.x>>b.y>>c.x>>c.y>>d.x>>d.y;

if(ss(a,b,c,d)&&ss(b,c,d,a))//đk HBH

{

if (kc(a,b)==kc(b,c)&&kc(b,c)==kc(c,d)&&kc(a,c)==kc(b,d)) fo<<4;

else if (kc(a,c)==kc(b,d)) fo<<3;

else if (kc(a,b)==kc(b,c)&&kc(b,c)==kc(c,d)) fo<<2;

else fo<<1;

}

else fo<<5;

return 0;

}

## Bài 3: Loại đa giác

Cho đa giác A gồm có n đỉnh A1, A2, …, An (3<n<=1000), tọa độ các đỉnh là số nguyên có giá trị tuyệt đối không vượt quá 30.000. Hãy xác định xem đa giác là đa giác gì trong các loại đa giác sau: Đa giác lồi (1); Đa giác lõm (2); Đa giác tự cắt (3).

Dữ liệu vào file **DAGIAC.INP**: Dòng đầu là số n; n dòng tiếp theo lần lượt là tọa độ các đỉnh của đa giác A.

Kết quả ra file **DAGIAC.OUT**: Dòng đầu là câu trả lời với các trường hợp đa giác như trên; dòng thứ 2 là nếu là trường hợp (1), (2) thì tính diện tích đa giác; còn lại trường hợp (3) ghi số 0. *(Lấy độ chính xác 3 chữ số sau dấu chấm thập phân)*

*Ví dụ:*

|  |  |
| --- | --- |
| **DAGIAC.INP** | **DAGIAC.OUT** |
| 4  0 0  0 5  5 5  5 0 | 1  25.000 |

**Thuật toán:**

*Xét các trường hợp:*

*- Đa giác lồi (xét cùng phía): Nếu thay tất cả các các đỉnh còn lại các phương trình đường thẳng (a1,a2), (a2,a3),… (an,an+1) an+1=a1 đều cùng dấu => đa giác lồi;*

*- Đa giác tự cắt: Duyệt tất cả các cặp cạnh (ai,ai+1), (aj,aj+1) với j=i+2..n, nếu có cặp cạnh nào đó cắt nhau => đa giác tự cắt.*

*Còn lại là đa giác lõm.*

**Chương trình:**

#include <bits/stdc++.h>

using namespace std;

fstream fi,fo;

const int nm=1002;

struct diem

{

double x,y;

};

diem a[nm];

bool kt;

int i,j,n;

double tdt(diem a,diem b,diem c)

{ double a1,b1,c1;

a1 = a.y-b.y;

b1 = b.x-a.x;

c1 = a.x\*b.y-a.y\*b.x;

return (a1\*c.x+b1\*c.y+c1);

}

void dtdg()

{ double dt=0;

for(int i=1;i<=n;i++)

dt=dt+(a[i].x-a[i+1].x)\*(a[i].y+a[i+1].y)/2;

dt=abs(dt);

fo<<setprecision(3)<<fixed<<dt;

}

bool ktloi(int i, int j)

{ int k,vt;

for(k=1;k<=n;k++)

if(tdt(a[i],a[j],a[k])\*tdt(a[i],a[j],a[k+1])<0) return 0;

return 1;

}

bool ktcat(int i, int j)

{

if(tdt(a[i],a[i+1],a[j])\*tdt(a[i],a[i+1],a[j+1])<0

&& tdt(a[j],a[j+1],a[i])\*tdt(a[j],a[j+1],a[i+1])<0) return 1;

return 0;

}

bool kttc()

{ int i,j;

for(i=1;i<n-1;i++)

for(j=i+2;j<=n;j++)

if(ktcat(i,j)) return 1;

return 0;

}

int main()

{ fi.open("DAGIAC.INP",ios::in);

fo.open("DAGIAC.OUT",ios::out);

fi>>n;

for(i=1;i<=n;i++)

fi>>a[i].x>>a[i].y;

a[n+1]=a[1];

kt=1;

for(i=1;i<=n;i++)

if(ktloi(i,i+1)==0){kt=0; break;}

if(kt)

{

fo<<1<<endl;

dtdg();

}

else

{

if(kttc()) fo<<3;

else

{

fo<<2<<endl;

dtdg();

}

}

return 0;

}

## Bài 4: Bao lồi

Cho N (N<=1000) điểm a1,a2,..., aN trên mặt phẳng. Các điểm đều có toạ độ nguyên và không có 3 điểm bất kỳ trong chúng thẳng hàng. Hãy viết chương trình xác định một đa giác không tự cắt có đỉnh là một số điểm trong các điểm đã cho và chứa tất cả các điểm còn lại. Hãy tính chu vi và diện tích đa giác này.

- Dữ liệu vào file BAOLOI.INP: Dòng đầu ghi số N, các dòng tiếp theo ghi tọa độ các điểm.

- Kết quả ra file BAOLOI.OUT: Gồm 3 số M, V, S, trong đó M là số đỉnh của đa giác, V là chu vi và S là diện tích của đa giác tìm được, lấy độ chính xác 3 chữ số sau dấu chấm thập phân.

**Ví dụ:**

|  |  |  |
| --- | --- | --- |
| **BAOLOI.INP** | **BAOLOI.OUT** | **Giải thích** |
| 8  7 1  16 4  2 3  13 7  11 12  5 11  2 8  7 5 | 6 39.631 105.000 | ***Bao lồi gồm các đỉnh:***  7 1  2 8  5 11  11 12  16 4 |

**Thuật toán:**

*- Sắp xếp các điểm theo tung độ tăng dần.*

*- Chọn điểm đầu tiên là điểm a[1], vt=1;*

*- Duyệt tất cả các điểm để chọn điểm tiếp theo (i), nếu ptdt(vt,i) mà tất cả các điểm còn lại đều cùng 1 phía thì điểm i là đỉnh tiếp theo của bao lồi, vt=i, lặp lại cho đến khi không chọn được điểm nào thêm.*

**Chương trình:**

#include <bits/stdc++.h>

using namespace std;

fstream fi,fo;

const int nm=1e4+1;

struct diem

{

double x,y;

};

diem a[nm];

vector<diem>b;

int i,j,n,d;

bool c[nm],kt;

double a1,b1,c1,p;

double kc(double x1,double y1,double x2,double y2)

{

return sqrt((x1-x2)\*(x1-x2)+(y1-y2)\*(y1-y2));

}

bool ss(diem a, diem b)

{

return (a.y<b.y);

}

bool cungphia(diem d1, diem d2)

{ int j=0;

double kq,a1,b1,c1;

a1=d1.y-d2.y; b1=d2.x-d1.x; c1=d1.x\*d2.y-d2.x\*d1.y;

do

{

j++;

kq=a1\*a[j].x+b1\*a[j].y+c1;

} while (kq==0&&j<n);

if(kq<0)

{

for(int k=j+1;k<=n;k++)

if(a1\*a[k].x+b1\*a[k].y+c1>0) return 0;

}

else

{

for(int k=j+1;k<=n;k++)

if(a1\*a[k].x+b1\*a[k].y+c1<0) return 0;

}

return 1;

}

void dtcvdg()

{ double dt=0,cv=0;

for(i=0;i<b.size()-1;i++)

{

dt=dt+(b[i].x-b[i+1].x)\*(b[i].y+b[i+1].y)/2;

cv=cv+kc(b[i].x,b[i].y,b[i+1].x,b[i+1].y);

}

dt=abs(dt);

fo<<setprecision(3)<<fixed<<cv<<" "<<dt;

}

int main()

{ fi.open("BAOLOI.INP",ios::in);

fo.open("BAOLOI.OUT",ios::out);

fi>>n;

for(i=1;i<=n;i++)

fi>>a[i].x>>a[i].y;

sort(a+1,a+1+n,ss);

b.push\_back(a[1]);c[1]=1;

do

{

kt=false;

for(i=2;i<=n;i++)

if(c[i]==0&&cungphia(b[b.size()-1],a[i]))

{

b.push\_back(a[i]);

c[i]=1;

kt=true;

}

}while(kt);

b.push\_back(a[1]);

fo<<b.size()-1<<" ";

dtcvdg();

return 0;

}

## Bài 5: Dãy tam giác bao nhau

Cho N tam giác được cho bởi toạ độ các đỉnh Ai(x,y), Bi(x,y), Ci(x,y), (i=1..N). Tam giác i được gọi là bao tam giác j nếu 3 đỉnh của tam giác j đều thuộc miền trong của tam giác i. Hãy tìm dãy các tam giác bao nhau có số lượng lớn nhất?

- Dữ liệu vào file DTAMGIAC.INP: Dòng đầu ghi số N (1<N<=1000); Trong N dòng tiếp theo dòng thứ i+1 ghi toạ độ 3 đỉnh của tam giác thứ i.

- Kết quả ra file TDAMGIAC.OUT: Số lượng dãy các tam giác bao nhau nhiều nhất nếu có; trường hợp không có dãy các tam giác bao nhau ghi số 0.

|  |  |  |
| --- | --- | --- |
| **DTAMGIAC.INP** | **DTAMGIAC.OUT** | **Giải thích** |
| 5  1 1 14 1 1 7  2 3 7 3 2 9  5 1 3 4 8 2  -4 2 2 5 8 3  7 6 9 8 9 2 | 2 | Có 2 tam giác bao nhau là: Tam giác số 1 bao tam giác số 3. |

**Ví dụ:**

**Thuật toán:**

*- Đọc dữ liệu, tính diện tích các tam giác;*

*- Sắp xếp diện tích các tam giác tăng dần;*

*- Chọn dãy tam giác theo thuật toán dãy con tăng dài nhất.*

*( Xét tam giác j nằm trong tam giác i=> xét 3 đỉnh của tam giác j nằm trong tam giác i=> xét từng đỉnh của j. Nếu thay các đỉnh vào các ptđt đi qua các đỉnh của i đều cùng dấu thì đỉnh này sẽ nằm trong.)*

**Chương trình:**

#include <bits/stdc++.h>

using namespace std;

fstream fi,fo;

const int nm=1e4+1;

struct tamgiac

{

double x1,y1,x2,y2,x3,y3,s;

};

tamgiac a[nm];

int i,j,n,b[nm],c[nm],maxb=0;

double a1,b1,c1,p;

double kc(double x1,double y1,double x2,double y2)

{

return sqrt((x1-x2)\*(x1-x2)+(y1-y2)\*(y1-y2));

}

bool ss(tamgiac x, tamgiac y)

{

return (x.s<y.s);

}

double tdt(double x1,double y1,double x2,double y2,double x3,double y3)

{

return ((y2-y1)\*x3+(x1-x2)\*y3-x1\*y2+x2\*y1);

}

bool ktdtg(tamgiac a,double x, double y)

{

if((tdt(a.x1,a.y1,a.x2,a.y2,x,y)>=0 &&

tdt(a.x2,a.y2,a.x3,a.y3,x,y)>=0 &&

tdt(a.x3,a.y3,a.x1,a.y1,x,y)>=0)||

(tdt(a.x1,a.y1,a.x2,a.y2,x,y)<=0 &&

tdt(a.x2,a.y2,a.x3,a.y3,x,y)<=0 &&

tdt(a.x3,a.y3,a.x1,a.y1,x,y)<=0)) return true;

return false;

}

bool kttgtg(tamgiac a, tamgiac b)

{

if(ktdtg(a,b.x1,b.y1)&&ktdtg(a,b.x2,b.y2)&&ktdtg(a,b.x3,b.y3)) return true;

return false;

}

int main()

{ fi.open("DTAMGIAC.INP",ios::in);

fo.open("DTAMGIAC.OUT",ios::out);

fi>>n;

for(i=1;i<=n;i++)

{

fi>>a[i].x1>>a[i].y1>>a[i].x2>>a[i].y2>>a[i].x3>>a[i].y3;

a1=kc(a[i].x1,a[i].y1,a[i].x2,a[i].y2);

b1=kc(a[i].x2,a[i].y2,a[i].x3,a[i].y3);

c1=kc(a[i].x1,a[i].y1,a[i].x3,a[i].y3);

p=(a1+b1+c1)/2;

a[i].s=sqrt(p\*(p-a1)\*(p-b1)\*(p-c1));

}

sort(a+1,a+1+n,ss);

b[1]=1;c[1]=0;

for(i=2;i<=n;i++)

{

for(j=i-1;j>=1;j--)

if(kttgtg(a[i],a[j])&&b[i]<b[j])

{

b[i]=b[j];

c[i]=j;

}

b[i]++;

maxb=max(maxb,b[i]);

}

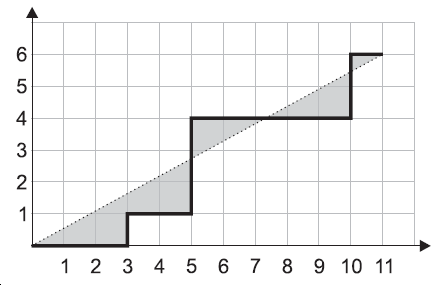
fo<<maxb;

return 0;

}

## Bài 6. Diện tích đa giác tự cắt

Trên mặt phẳng lưới toạ độ Đê các, xuất phát từ điểm (0, 0) người ta vẽ một đường gấp khúc có các cạnh song song với trục toạ độ theo quy tắc sau: bút vẽ được điều khiển bằng chương trình là một xâu các ký tự U, R. Gặp lệnh U bút vẽ sẽ chuyển lên trên một đơn vị, còn khi gặp lệnh R bút vẽ sẽ chuyển sang phải một đơn vị. Khi hết chương trình bút vẽ được kéo thẳng về gốc toạ độ. Hình bên tương ứng với chương trình vẽ là RRRURRUUURRRRRUUR.



***Yêu cầu***: Tìm diện tích bị giới hạn bởi đường gấp khúc và đường thẳng. Trên hình bên, miền cần tính diện tích được tô đậm.

***Dữ liệu***: Vào từ file văn bản **AREA.INP** gồm một dòng chứa một xâu các ký tự R, U, xác định một chương trình vẽ. Bút vẽ luôn chuyển động trong phạm vi lưới kích thước 1000\*1000.

***Kết quả***: Đưa ra file văn bản **AREA.OUT** diện tích tìm được, kết quả lấy độ chính xác 3 chữ số sau dấu chấm thập phân.

|  |  |
| --- | --- |
| **AREA.INP** | **AREA.OUT** |
| RRRURRUUURRRRRUUR | 8.515 |

**Ví dụ**:

**Thuật toán:**

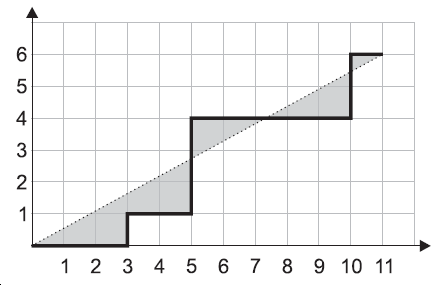
*- Đặt s=xx\*yy, diện tích hình chữ nhật có cạnh xx, yy; (xx là hoành độ max theo ox, yy là tung độ max theo oy)*

*- Đặt s1=diện tích hình màu đỏ; là các điểm nằm phía trên đường thẳng đi qua hai điểm (0,0) và (xx,yy).*

*- Đặt s2=diên tích hình màu xanh; là các điểm nằm phía dưới đường thẳng đi qua hai điểm (0,0) và (xx,yy).*

*- Diện tích cần tính bằng s-s1-s2;*

# 



**Chương trình:**

#include <bits/stdc++.h>

using namespace std;

const int nm=1e5+1;

struct diem

{

double x,y;

};

vector<diem> a,b,c;

int n,i,d;

double ss,s1,s2,a1,b1,c1,a2,b2,c2,xx,yy,dd,dx,dy,x,y;

fstream fi,fo;

string s;

double tdt(diem a,diem b,diem c)

{ double a1,b1,c1;

a1 = a.y-b.y;

b1 = b.x-a.x;

c1 = a.x\*b.y-a.y\*b.x;

return (a1\*c.x+b1\*c.y+c1);

}

bool ktcat(int i, int j)

{

if(tdt(a[0],a[n],a[i])\*tdt(a[0],a[n],a[j])<0

&&tdt(a[i],a[j],a[0])\*tdt(a[i],a[j],a[n])<0) return 1;

return 0;

}

double dientich(vector<diem> a)

{ double dt=0;

for(int i=0;i<a.size()-1;i++)

dt=dt+(a[i].x-a[i+1].x)\*(a[i].y+a[i+1].y)/2;

return abs(dt);

}

int main ()

{

fi.open("AREA.INP",ios::in);

fo.open("AREA.OUT",ios::out);

fi>>s;

s=s+'A';

a.push\_back({0,0});

d=1;

xx=0; yy=0;

for(i=0;i<s.length()-1;i++)

if(s[i]==s[i+1]) d++;

else

{

if(s[i]=='U') yy=yy+d;

else

if(s[i]=='R') xx=xx+d;

a.push\_back({xx,yy});

d=1;

}

n=a.size()-1;

a1=a[n].y-a[0].y; b1=a[0].x-a[n].x; c1=a[n].x\*a[0].y-a[0].x\*a[n].y;

b.push\_back(a[0]);

c.push\_back(a[0]);

for(i=1;i<n;i++)

{

if(tdt(a[0],a[n],a[i])>=0) b.push\_back(a[i]);

if(tdt(a[0],a[n],a[i])<=0) c.push\_back(a[i]);

if(ktcat(i,i+1))

{

a2=a[i+1].y-a[i].y; b2=a[i].x-a[i+1].x; c2=a[i+1].x\*a[i].y-a[i].x\*a[i+1].y;

dd=a1\*b2-a2\*b1; dx=-c1\*b2+c2\*b1; dy=-a1\*c2+a2\*c1;

x=dx/dd; y=dy/dd;

b.push\_back({x,y});

c.push\_back({x,y});

}

}

b.push\_back(a[n]);

b.push\_back({a[0].x,a[n].y});

b.push\_back(a[0]);

c.push\_back(a[n]);

c.push\_back({a[n].x,a[0].y});

c.push\_back(a[0]);

ss=xx\*yy;

s1=dientich(b);

s2=dientich(c);

fo<<setprecision(3)<<fixed<<ss-s1-s2;

}

# IV. Các bài tập tham khảo thêm

## Bài 1: Đếm hình chữ nhật

Trong hệ trục toạ độ ĐềCác, cho N (4<=N<=300) điểm phân biệt A1(x1,y1), A2(x2,y2), ...,An(xn,yn).

**Yêu cầu:** Đếm xem có bao nhiêu hình chữ nhật khác nhau được tạo ra bởi 4 điểm bất kỳ trong N điểm trên *(hai hình chữ nhật khác nhau khi chúng khác nhau ít nhất là một đỉnh)*.

Dữ liệu vào file **CCHUNHAT.INP**: Dòng đầu là số N, N dòng tiếp theo là toạ độ của N điểm trên.

Kết quả ra file **CCHUNHAT.OUT**: Ghi số lượng hình chữ nhật khác nhau đã đếm được.

**Ví dụ:**

|  |  |
| --- | --- |
| **CCHUNHAT.INP** | **CCHUNHAT.INP** |
| 7  0 0  5 0  0 5  5 5  10 10  10 0  0 10 | 2 |

## Bài 2: Điểm và đa giác

Cho đa giác A gồm n (n<=1000) đỉnh, mỗi đỉnh là tọa độ nguyên có giá trị tuyệt đối không quá 104 và điểm M(x,y). Hãy kiểm tra xem M nằm trong (1) hay trên cạnh (2) hay trùng vào đỉnh (3) hay nằm ngoài (4) đa giác A?

Dữ liệu vào file DAGIACM.INP: Dòng đầu là số n và tọa độ điểm M; n dòng tiếp theo là các tọa độ của đa giác.

Kết quả ra file DAGIACM.OUT: Ghi chỉ số tương ứng với các trường hợp đã nêu.

**Ví dụ:**

|  |  |
| --- | --- |
| **DAGIACM.INP** | **DAGIACM.OUT** |
| 5 3 3  0 0  10 0  10 10  7 11  0 9 | 1 |

## Bài 3: Đổi đất

Hai bộ lạc Anpha và Bêta sống rất hoà thuận với nhau. Một phần ranh giới của hai bộ lạc là một đường gấp khúc không tự cắt. Đường gấp khúc nhận được bằng cách lần lượt nối N điểm đôi một khác nhau A1, A2,......,AN. Điểm Ai được xác định bởi hoành độ xi và tung độ yi (xi là các số nguyên thoả mãn điều kiện xi <= xi+1). Phần đất của bộ lạc Anpha nằm ở phía trên đường gấp khúc.Nhân dịp năm mới, tù trưởng hai bộ lạc quyết định thay đổi đường ranh giới cũ bằng cách xây dựng một đường cao tốc là đường nối thẳng từ A1 tới AN và lấy đường cao tốc này làm ranh giới mới. Dĩ nhiên, sự thay đổi này sẽ chuyển một số phần đất của bộ lạc Anpha cho bộ lạc Bêta và ngược lại. Hai tù trưởng thoả thuận phần diện tích chênh lệch do việc thay đổi đường ranh giới sẽ được điều chỉnh trong tương lai bằng một cách khác.

**Yêu cầu :** Hãy tính diện tích phần đất SA của bộ lạc Anpha trở thành đất của bộ lạc Bêta và diện tích phần đất SB của bộ lạc Bêta trở thành đất của bộ lạc Anpha sau khi thay đổi đường ranh giới giữa hai bộ lạc.

**Dữ liệu :** Vào từ file văn bản DOIDAT.INP trong đó :

* Dòng đầu chứa số N (N<=10000).
* Dòng thứ i trong N dòng tiếp theo chứa hai số nguyên xi và yi đặt cách nhau ít nhất một dấu cách. (-32000 <=xi, yi <= 32000).

**Kết quả :** Đưa ra file văn bản DOIDAT.OUT trong đó dòng thứ nhất chứa SA, dòng thứ hai chứa SB. Kết quả được lấy chính xác với 4 chữ số sau chấm thập phân.

|  |  |
| --- | --- |
| **DOIDAT .INP** | **DOIDAT .OUT** |
| 6  0 0  2 4  5 1  7 11  10 8  11 11 | 8.0000  9.0000 |

**Ví dụ:**

## Bài 4: Dãy hình chữ nhật lồng nhau

Trên mặt phẳng tọa độ cho N hình chữ nhật với các cạnh song song với hệ trục tọa độ, các hình chữ nhật được đánh số từ 1 tới N. Hình chữ nhật thứ *i* được cho bởi 4 số nguyên dương *xi1*, *y­i1*, *xi2*, *yi2*, trong đó (*xi1*, *y­i1*) là tọa độ đỉnh trái dưới, còn (*xi2*, *y­i2*) là tọa độ đỉnh phải trên. Ta nói rằng hình chữ nhật thứ i nằm trong hình chữ nhật thứ j nếu trên mặt phẳng tọa độ, mọi điểm của hình chữ nhật i đều thuộc hình chữ nhật j.

**Yêu cầu**: Với N hình chữ nhật cho trước, hãy tìm K hình chữ nhật với chỉ số i1, i2, …, iK sao cho hình i1 nằm trong hình i2, hình i2 nằm trong hình i3, …, hình iK-1 nằm trong hình iK và K là lớn nhất.

**Dữ liệu**: Vào từ file văn bản **HCN.INP**:

* Dòng đầu tiên chứa số nguyên dương N (1N100).
* N dòng tiếp theo, dòng thứ i chứa 4 số nguyên dương *xi1*, *y­i1*, *xi2*, *yi2* có giá trị không vượt quá 30000.

**Kết quả**: Ghi ra file văn bản **HCN.OUT** số K tìm được.

**Ví dụ**:

|  |  |
| --- | --- |
| **HCN.INP** | **HCN.OUT** |
| 3  1 1 7 4  3 1 6 6  2 2 5 4 | 2 |

## Bài 5: Chia đất

Có một mảnh đất, trên biên của nó có cắm một số cột mốc để mảnh đất có thể xem như một đa giác có biên gồm các đoạn thẳng kề nhau nối các mốc này. Bắt đầu từ một cột mốc nào đó được đánh số 1, người ta đi vòng quanh mảnh đất theo một chiều xác định để đánh số các cột mốc kế tiếp (2, 3, ..., N) theo thứ tự được gặp.

*Yêu cầu:* Hãy phân chia mảnh đất thành hai phần bằng một đoạn thẳng nối hai cột nào đó để độ chênh lệch diện tích của hai phần được chia là nhỏ nhất. Đoạn thẳng chia không được phần nào nằm ngoài mảnh đất và không được chứa cột mốc nào ngoài hai cột mốc đầu mút của nó.

Dữ liệu vào file CHIADAT.INP: Dòng đầu là số N (4<=N<=500); N dòng tiếp theo, mỗi dòng ghi hai số nguyên x, y là các cặp toạ độ một cột mốc *( theo thứ tự các cột từ 1 đến N)*. Các số trên một dòng được ghi cách nhau ít nhất một dấu cách.

Kết quả ra file CHIADAT.OUT: Một dòng duy nhất gồm hai số ghi cách nhau ít nhất một dấu cách, là số hiệu của hai cột mốc tạo thành đoạn thẳng chia.

**Ví dụ:**

|  |  |
| --- | --- |
| **CHIADAT.INP** | **CHIADAT.OUT** |
| 10  2 1  2 2  0 0  -2 2  -3 2  -2 -2  -1 -3  0 -3  0 -2  1 -2 | 4 8 |

# V. Kết luận

Trên đây là những trình bày của tôi về một số công thức và phương pháp giải các bài toán hình học trong tin học. Do thời gian có hạn nên việc trình bày có thể chưa được đầy đủ và các thuật toán có thể chưa tối ưu. Rất mong nhận được sự đóng góp từ các bạn!