Algoritmi geometrici

conf.univ.dr. Doru Popescu Anastasiu

1. Noţiuni generale

1.1 Drepte

Ecuaţia unei drepte în plan

Forma generală a ecuaţiei unei drepte d în sistemul de coordonate xOy este următoarea:

d: $a \cdot x + b \cdot y + c = 0$ (1).

Orice punct M(x,y) de pe dreapta d verifică ecuaţia (1).

Ecuaţia unei drepte care trece prin două puncte date A(xA,yA) şi B(xB,yB) este următoarea:

$\frac{x - xA}{y - yA} = \frac{xB - xA}{yB - yA}$ (2), sau echivalent cu aceasta: $(x - xA)(yB - yA) = (xB - xA)(y - yA)$, pentru a evita situaţiile când numitorul se anulează. O altă formă a acestei ecuaţii este

$$\begin{vmatrix} x & y & 1 \\ xA & yA & 1 \\ xB & yB & 1 \end{vmatrix} = 0 \quad (3).$$

După efectuarea calculelor în (2) sau (3) se obţine ecuaţia (1), în care

$$\begin{aligned} a &= yA - yB \\ b &= xB - xA \\ c &= -xB \cdot yA + yB \cdot xA \end{aligned} \qquad (4)$$

Ecuaţiile parametrice ale unei drepte

Considerăm două puncte distincte M1(x1,y1) şi M2(x2,y2). Un punct M(x,y) se găseşte pe dreapta M1M2, dacă

$$\begin{aligned} x &= x1 + (x2 - x1)t \\ y &= y1 + (y2 - y1)t \text{, t număr real.} \end{aligned} \qquad (5)$$

Ecuaţiile (5) se numesc ecuaţiile parametrice ale dreptei M1M2.

Dacă în (5) luăm t număr real din intervalul [0,1], atunci obţinem ecuaţiile parametrice ale segmentului [M1M2].

Panta unei drepte

Panta unei drepte reprezintă tangenta unghiului făcut de dreaptă cu axa Ox.

Panta unei drepte d dată prin ecuaţia (1) este $m = -\frac{a}{b}$, iar pentru dreapta dată de ecuaţia (2) sau (3) este $m = -\frac{xB - xA}{yB - yA}$.

Proprietăţi

1) Două drepte sunt paralele dacă şi numai dacă pantele lor sunt egale.
2) Două drepte sunt perpendiculare dacă şi numai dacă produsul pantelor este -1.

Ecuaţia unei drepte care trece printr-un punct şi are o pantă dată

Fie M1(x1,y1) şi un număr real m. Ecuaţia dreptei care trece prin punctul M1 şi are panta m este

$y - y1 = m \cdot (x - x1)$

Poziţia unui punct faţă de o dreaptă

Considerăm o dreaptă d dată prin ecuaţia $a \cdot x + b \cdot y + c = 0$ şi un punct M(xM,yM).

Ataşăm dreptei d funcţia $f : R \times R \to R, \quad f(x, y) = a \cdot x + b \cdot y + c$.

Dreapta d împarte planul în două semiplane, S+ şi S- definite astfel:

S+ este mulţimea punctelor M(xM,yM) care îndeplinesc condiţia $f(xM, yM) > 0$.

S- este mulţimea punctelor M(xM,yM) care îndeplinesc condiţia $f(xM, yM) < 0$.

Punctele M(xM,yM) de pe dreapta d îndeplinesc condiţia $f(xM, yM) = 0$.

Intersecţia a două drepte

Două drepte d1 şi d2, de ecuaţii

d1: $a1 \cdot x + b1 \cdot y + c1 = 0$

d2: $a2 \cdot x + b2 \cdot y + c2 = 0$

Găsirea punctului de intersecţie a dreptelor d1 şi d2 presupune rezolvarea sistemului:

$$\begin{cases} a1 \cdot x + b1 \cdot y + c1 = 0 \\ a2 \cdot x + b2 \cdot y + c2 = 0 \end{cases}$$

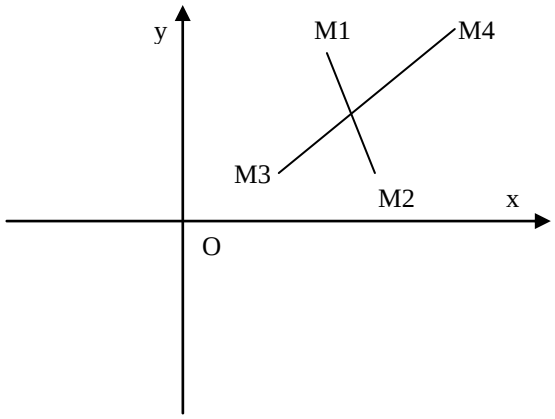
Acest sistem are soluţie unică, dacă pantele celor două drepte sunt diferite, adică dacă $a1 \cdot b2 \neq a2 \cdot b1$. Dacă această condiţie este îndeplinită atunci punctul de intersectie al dreptelor d1 şi d2 are coordonatele:

$$x = \frac{b1 \cdot c2 - b2 \cdot c1}{a1 \cdot b2 - a2 \cdot b1}$$

$$y = \frac{c1 \cdot a2 - c2 \cdot a1}{a1 \cdot b2 - a2 \cdot b1}.$$

Intersecţia a două segmente

Considerăm segmentele [M1M2], [M3M4], date prin coordonatele capatelor.
M1(x1,y1), M2(x2,y2), M(x3,y3), M4(x4,y4).



Pentru ca cele două segmente să se intersecteze trebuie ca:

- M1 şi M2 să se găsească în semiplane (închise) diferite delimitate de dreapta M3M4
- M3 şi M4 să se găsească în semiplane (închise) diferite delimitate de dreapta M1M3

Dacă ecuaţia dreptei M1M2 este $a1\cdot x+b1\cdot y+c1=0$, iar cea a dreptei M3M4 este $a2\cdot x+b2\cdot y+c2=0$, atunci cele două condiţii anterioare sunt echivalente cu
$(a2\cdot x1+b2\cdot y1+c2)\cdot(a2\cdot x2+b2\cdot y2+c2)<=0$ şi
$(a1\cdot x3+b1\cdot y3+c1)\cdot(a1\cdot x4+b1\cdot y4+c1)<=0$

1.2 Puncte. Distanţe

Distanţa dintre două puncte

Considerăm două puncte M1(x1,y1), M2(x2,y2).
Distanţa dintre M1 şi M2 este dată de relaţia
$M1M2=\sqrt{(x1-x2)^2+(y1-y2)^2}$

Puncte interior unui segment

Un punct M(x,y) este interior unui segment [M1M2], M1(x1,y1), M2(x2,y2) dacă M1M2=M1M+MM2, adică dacă
$\sqrt{(x1-x2)^2+(y1-y2)^2}=\sqrt{(x1-x)^2+(y1-y)^2}+\sqrt{(x-x2)^2+(y-y2)^2}$

Distanţa de la un punct la o dreaptă

Pentru o dreaptă d: $a\cdot x+b\cdot y+c=0$ şi un punct M0(x0,y0), distanţa de la M la d este egală cu
$dist(M0,d)=\frac{|a\cdot x0+b\cdot y0+c|}{\sqrt{a^2+b^2}}$

Distanţa de la un punct la un segment

Considerăm un segment [M1M2], cu M1(x1,y1), M2(x2,y2) şi un punct M(x,y). Prin distanţa de la M la segmentul [M1M2] înţelegem lungimea celui mai scurt segment cu un capăt în M şi celălalt pe [M1M2].

$$\text{dist(M, [M1M2])} = \begin{cases} \text{dist(M, M1M2), daca proiectia lui M pe M1M2 apartine lui[M1M2]} \\ \min\{MM1, MM2\}, \quad altfel \end{cases}$$

Centrul de greutate

Centrul de greutate al unui sistem omogen de n puncte: M1(x1,y1), M2(x2,y2), …, Mn(xn,yn) este un punct G(xG,yG), cu

$$xG = \frac{x1 + x2 + ... + xn}{n}$$
$$yG = \frac{y1 + y2 + ... + yn}{n}$$

Aria unui triunghi

1. Dacă triunghiul este dat prin lungimile laturilor: a, b, c, atunci pentru calculul ariei se foloseste formula lui Heron:

$S = \sqrt{p(p-a)(p-b)(p-c)}$, unde $p = \frac{a+b+c}{2}$

2. Dacă triunghiul este dat prin coordonatele vârfurilor sale A(xA,yA), B(xB,yB), C(xC,yC), atunci

$S = \frac{1}{2} abs(\begin{vmatrix} xA & yA & 1 \\ xB & yB & 1 \\ xC & yC & 1 \end{vmatrix})$, adică $S = \frac{1}{2} abs(xA \cdot yB + xB \cdot yC + yA \cdot xC - xC \cdot yB - xA \cdot yC - xB \cdot yA)$

Aria unui poligon convex

Pentru calculul ariei unui poligon convex A1A2…An, Ai(xi,yi), $i \in \{1,2,...,n\}$, se realizează o partiţie a suprafeţei poligonale în triunghiuri, de exemplu:

$$aria(A1A2...An) = aria(A1A2A3) + aria(A1A3A4) + ... + aria(A1An-1An)$$

O modalitate de calcul a ariei unui poligon (convex sau nu) este data de formula:

$$aria(A1A2...An) = \left| \sum_{i=1}^{n} x_i (y_{i+1} - y_{i-1}) \right| / 2$$ (vezi [5])

Punct interior unui poligon convex

Un punct M(x,y) se găseşte în interiorul sau pe frontiera unui poligon convex dacă

$$aria(A1A2...An) = aria(MA1A2) + aria(MA2A3) + ... + aria(MAnA1)$$

Rapoarte

Fie M1(x1,y1) şi M(x2,y2) distincte. M(x,y) un punct pe dreapta M1M2, diferit de M2. Numărul k definit astfel

$$k = \begin{cases} -\frac{MM1}{MM2}, dacă \quad M \in [M1M2] \\ \frac{MM}{MM2}, dacă \quad M \notin [M1M2] \end{cases}$$

se numeşte raportul în care M împarte segmentul [M1M2]. Fiind cunoscut numărul real $k \neq 1$, coordonatele lui M, care împarte segmentul [M1M2] în raportul k sunt

$$x = \frac{x1 - kx2}{1-k}$$

$$y = \frac{y1 - ky2}{1-k}.$$

Caz particular
M este mijlocul segmentului [M1M2], $k = -1$. În acest caz coordonatele lui M sunt

$$x = \frac{x1 + x2}{2}$$

$$y = \frac{y1 + y2}{2}$$

Probleme propuse

1. Se dau două segmente prin coordonatele capetelor lor. Se cere să se verifice dacă segmentele se intersectează şi în caz afirmativ să se determine coordonatele punctului de intersecţie.
2. Se dă un poligon prin coordonatele vârfurilor sale şi se cere să se verifice dacă este convex.
3. Se dă un punct M şi un poligon convex P prin coordonatele vârfurilor. Se cere să se determine distanţa de la punctul M la poligonul P (lungimea celui mai mic segment de la punctul M la un punct de pe poligonul P).
4. Se dă un poligon convex P prin coordonatele vârfurilor sale şi un punct M dat prin coordonate. Se cere să se determine poziţia punctului M faţă de poligonul P (interior, exterior sau pe poligon).

2. Înfăşurătoarea convexă

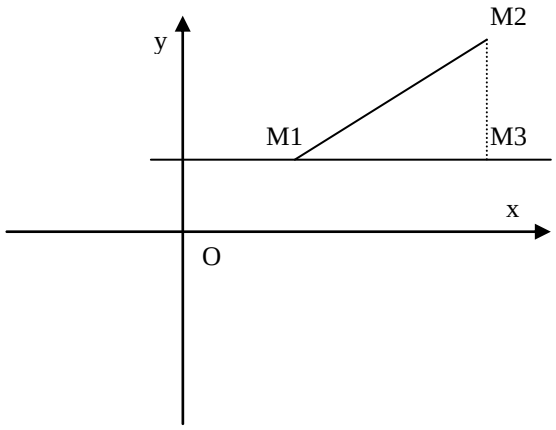
Se consideră n puncte în plan A1, A2, …, An, date prin coordonate: (x1,y1), (x2,y2), …, (xn,yn). Vrem să determinăm un poligon convex care aibă vârfurile printre punctele date şi să conţină pe laturi sau în interior celelalte puncte. Acest poligon se numeşte *înfăşurătoarea convexă* asociată punctelor A1, A2, …, An.

Un algorim cu complexitate O(nlogn) care poate fi implementat uşor este scanarea Graham.

Înainte de a descrie acest agoritm vom introduce două noţiuni utile: «unghi polar» şi «unghi care realizază o întoarcere».

1. *Unghi polar*

Fie M1 şi M2 două puncte distincte. Pentru punctul M1, unghiul polar asociat lui M2 este unghiul format de dreapta orizontală care trece prin M1 şi dreapta M1M2.

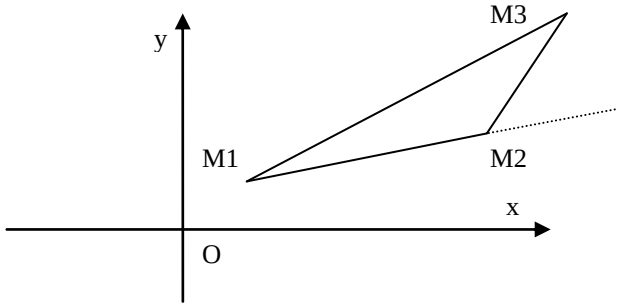


Dacă se cunosc coordonatele punctelor M1 şi M2 se poate determina uşor distanţa de la M1 la M2 şi distanţa de la M2 la orizontala care trece prin M1. Folosind acest două distanţe (M1M2 şi M1M3) se poate calcula cosinusul unghiului făcut de M1M2 cu axa Ox şi implicit unghiul făcut de M1M2 cu Ox (folosind funcţia arccos).

2. *Unghi care realizază o întoarcere*

Considerăm trei puncte M1(x1,y1), M2(x2,y2), M3(x3,y3).

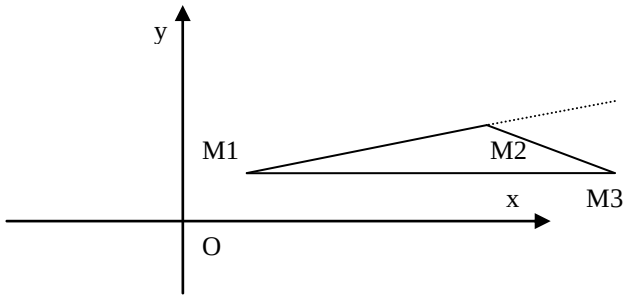
Spunem că unghiul M2M1M3 realizează o întoarcere spre stânga (a lui M3) faţă de M2M1, dacă ne aflăm în situaţia



Această situaţie este caracterizată de inegalitatea

$$\begin{vmatrix} x1 & y1 & 1 \\ x2 & y2 & 1 \\ x3 & y3 & 1 \end{vmatrix} > 0 \Leftrightarrow x1 \cdot y2 + x2 \cdot y3 + x3 \cdot y1 - x1 \cdot y3 - x2 \cdot y1 - x3 \cdot y2 > 0$$

Spunem că unghiul M2M1M3 realizează o întoarcere spre dreapta (a lui M3), faţă de M2M1, dacă ne aflăm în situaţia



Această situaţie este caracterizată de inegalitatea

$$\begin{vmatrix} x1 & y1 & 1 \\ x2 & y2 & 1 \\ x3 & y3 & 1 \end{vmatrix} < 0 \Leftrightarrow x1 \cdot y2 + x2 \cdot y3 + x3 \cdot y1 - x1 \cdot y3 - x2 \cdot y1 - x3 \cdot y2 < 0$$

Scanarea Graham rezolvă problema înfăşurătorii convexe prin păstrarea unei stive care conţine puncte. Fiecare vârf dat este inserat în stivă o singură dată, iar punctele care nu fac parte din înfăşurătoare sunt eliminate pe parcurs. Astfel în final stiva va conţine vârfurile înfăşurătorii convexe în ordine trigonometrică.

Descriere algoritm

Pas 1. Se determină vârful A0 care are ordonata cea mai mică şi în cazul în care există mai multe astfel de puncte se alege cel care are abscisa cea mai mică.

Pas 2. Se sortează punctele mulţimii {A1, A2, …, An}\{A0} crescător după unghiurile polare în jurul vârfului A0. Dacă există mai multe vârfuri cu acelaşi unghi polar, atunci se păstrează doar cel care află la distanţa maximă de A0. După această operaţie se obţin punctele A1, A2, …, Am.

Pas 3. Inserăm într-o stivă (iniţial vidă) punctele A0, A1, A2.

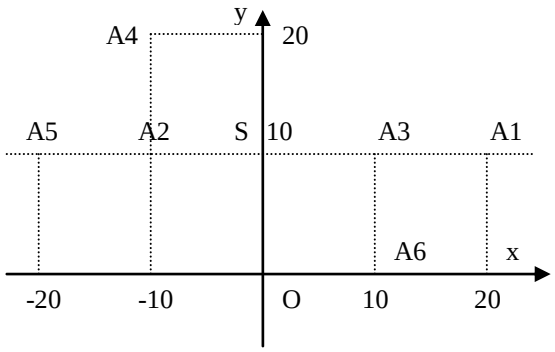
Pas 4.Vom considera pe rând nodurile Ai, $i \in \{3,4,..., m\}$. Vom nota cu Ai, nodul curent, cu Ai-1 nodul care face parte din vârful stivei şi cu Ai-2 nodul următor din stivă. Pentru fiecare nod Ai vom verifica dacă unghiul format de punctele Ai-2,Ai-1,Ai nu formaează un unghi care să anuleze convexitatea poligonului. Practic acest unghi trebuie să realizeze o întoarcere spre stânga. În cazul în care nu se păstrează convexitatea, vârful Ai-1 se elimină din stivă. Verificarea continuă (cu eliminări succesive dacă este cazul) până în momentul în care unghiul format nu indică pierderea convexităţii. În final stiva va conţine doar vârfurile înfăşurătorii convexe, în ordine trigonometrică.

3. Centrul de simetrie al unei mulţimi de puncte

Fie $A = \{A1, A2,..., An\}$ o mulţime de n puncte în plan.

Un punct S din plan se numeşte *centru de simetrie* dacă pentru orice punct Ai din mulţimea A, simetricul său faţă de S este tot un punct din mulţimea A.

Exemplu. Pentru punctele din figură, S(0,10) este centru de simetrie.



În continuare vom rearanja punctele din A astfel încât să fie ordonate crescător după abscisă şi la abscisă egală ordonate crescător după ordonată.

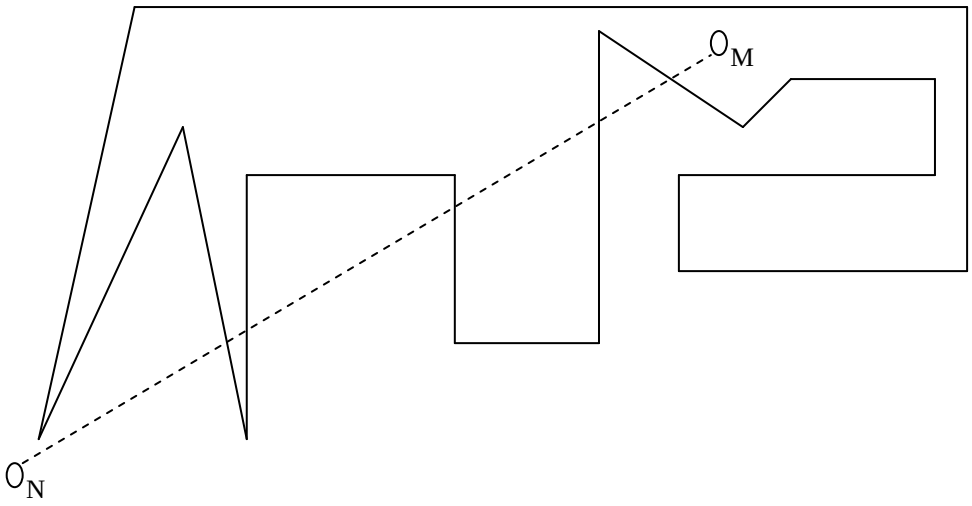
Proprietate
Mulţimea A are centru de simetrie dacă şi numai dacă:

- n este par
- Segmentele [A1An], [A2An-1], [A3An-2], … au acelaş mijloc. Acest punct este centrul de simetrie.

4. Poziţia unui punct faţă de un poligon

Prin poligon înţelegem o linie frântă care nu se autointersectează.
Considerăm în continuare un poligon P=A1A2…An şi un punct M.
A1(x1,y1), …, An(xn,yn) şi M(x0,y0).
Dorim să determinăm poziţia lui M faţă de poligonul P. Adică să verificăm dacă M este interior lui P, exterior lui P sau pe poligonul P.

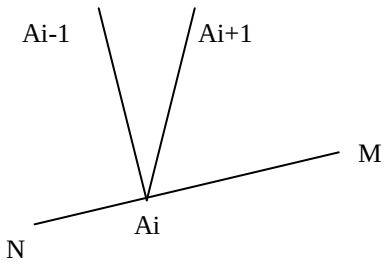
Pentru a verifica dacă M este interior lui P, folosim o observaţie imediată, şi anume faptul că, dacă intersecţia unui segment cu un capăt în M şi un capăt într-un punct N exterior lui P, cu laturile poligonului este formată dintr-un număr impar de puncte, atunci M este interior lui P.



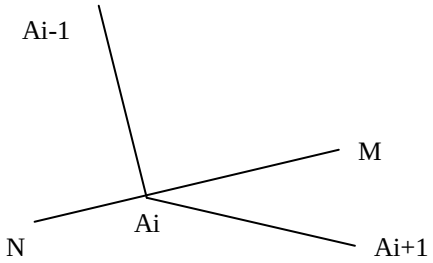
Există însă câteva situaţii când punctele de intersecţie dintre segmentul (MN) şi laturile lui P nu se numără. Aceste situaţii le prezentăm în continuare.

- Dacă dreapta MN coincide cu dreapta suport a unei laturi din P, atunci intersecţia dintre (MN) cu latura respectivă conţine o infinitate de puncte, care evident nu se numără.

- Dacă segmentul (MN) intersectează o latură ca în cazul următor



aceasta nu se numără. Însă intersecţia din cazul următor se numără



Un algoritm care determină poziţia unui punct faţă de un poligon poate fi următorul:

Pas 1. Determinăm un punct exterior lui P astfel: Dintre vârfurile poligonului P determinăm unul Ai, cu proprietatea că are abscisa minimă şi în cazul în care există mai multe astfel de puncte se ia cel cu ordonata minimă. Considerăm punctul exterior N(xi-1,yi).

Pas 2. Pentru orice segment (AiAi+1), dacă (AiAi+1) intersectat cu (MN) ne dă un punct, atunci mărim numărul de puncte de intersecţie cu 1.

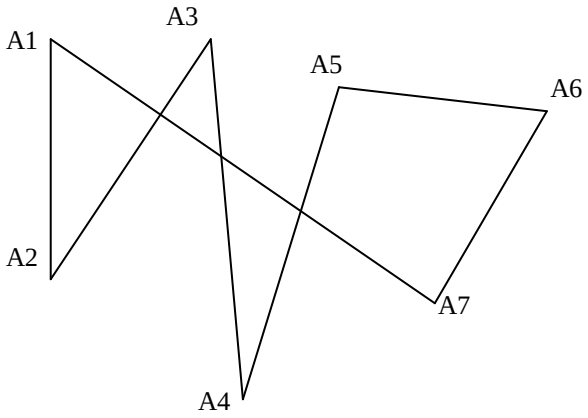
Pas 3. Pentru orice vârf Ai, al poligonului, dacă Ai-1 şi Ai+1 sunt de o parte şi de alta a dreptei MN, atunci numărul de puncte de intersecţie se măreşte cu 1.

Pas 4. Dacă numărul de puncte de intersecţie calculat la paşii anteriori este impar, atunci M este interior lui P. Altfel, verificăm dacă M se găseşte pe poligon (verificând dacă M se găseşte pe fiecare latură a lui P). Dacă nu se găseşte pe poligonul P, atunci M este exterior lui P.

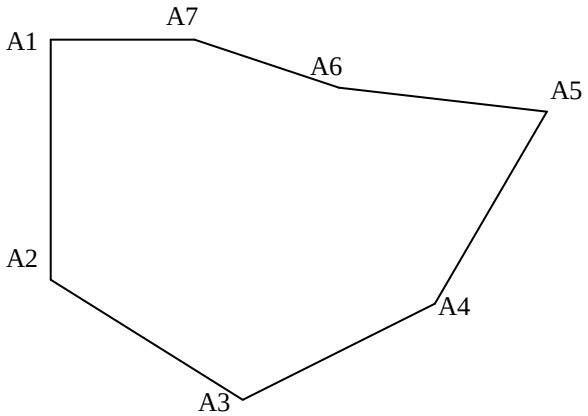
5. Determinarea unui poligon care trece printr-o mulţime de puncte

Considerăm în continuare o mulţime de puncte în plan A={A1,A2, …, An}.
Vrem să renumerotăm punctele din mulţime astfel încât A1A2…An să fie poligon.

Exemplu



Dacă considerăm punctele în ordinea de mai sus, A1A2…A7 nu este poligon, însă după renumerotarea de mai jos, A1A2…A7 este poligon.

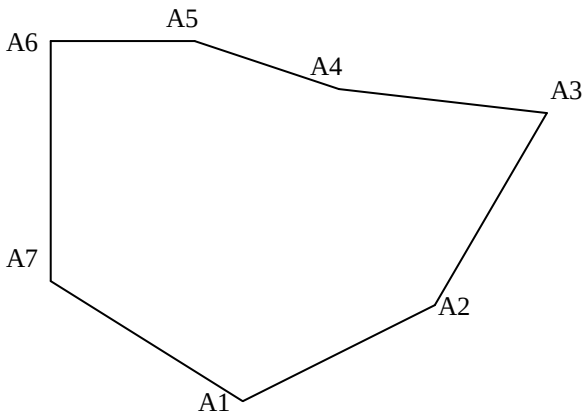


O modalitate de permutare a punctelor mulţimii A, pentru a obţine un poligon este inspirată de scanarea Graham. Mai precis:

Pas 1. Se determină vârful Ai care are ordonata cea mai mică şi în cazul în care există mai multe astfel de puncte se alege cel care are abscisa cea mai mică. Interschimbăm Ai cu A1.

Pas 2. Se sortează punctele mulţimii {A2, …, An} crescător după unghiurile polare în jurul vârfului A1. Dacă există mai multe vârfuri cu acelaşi unghi polar, atunci acestea se ordonează descrescător după distanţa de la A1. După această operaţie se obţin vârfurile poligonului dorit, în ordinea A1, A2, …, An.

Pentru exemplul anterior se obţine ordinea:



6. Poziţionarea unui cerc într-o mulţime de puncte

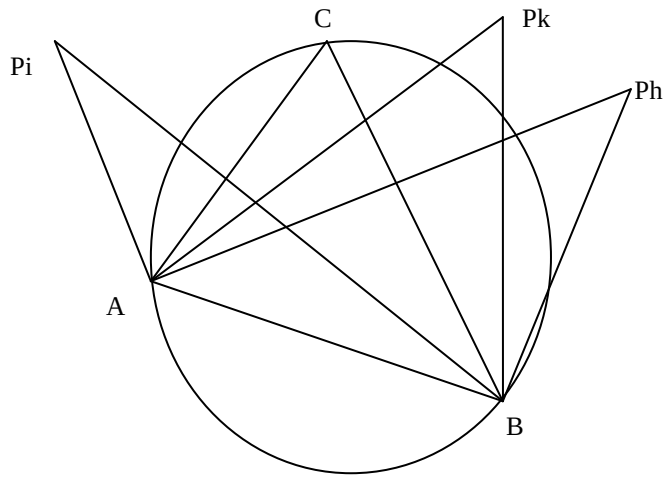
Considerăm n puncte în plan astfel încât oricare trei să fie necoliniare, date prin coordonatele lor carteziene P1(x_1,y_1); P2(x_2,y_2); ...; Pn(x_n,y_n), numere întregi. Vrem să determinăm un cerc (centrul şi raza sa) care să treacă prin cel puţin trei din cele n puncte şi să nu conţină nici un punct în interior.

Putem găsi acest cerc astfel:

1. Determinăm două puncte A şi B printre cele n puncte date cu proprietatea că toate celelalte n-2 puncte se află într-un semiplan delimitat de dreapta AB.

2. Se determină punctul C printre punctele diferite de A si B astfel încât unghiul ∠ACB să fie maxim.

3. A,B,C sunt punctele căutate, pentru care se determină cercul circumscris cu centrul în punctul de coordonate (a,b) şi rază r.



Pe aceeaşi idee putem rezolva problema găsirii unui cerc care să treacă prin trei puncte şi care să le conţină pe toate celelalte în interior. Mai precis în algoritmul anterior se consideră la pasul 2 în loc de unchiul cel mai mare, unghiul cel mai mic.

O altă problemă legată de găsirea unui cerc cu o anumită proprietate relativă la o mulţime de puncte este următoarea:

Se dau 2n+3 puncte în plan astfel încât oricare trei să fie necoliniare şi oricare 4 puncte nu se află pe acelaşi cerc, date prin coordonatele lor carteziene $P1(x_1,y_1)$; $P2(x_2,y_2)$; ...; $Pn(x_n,y_n)$, numere întregi. Se cere să se determine un cerc (centrul şi raza sa) care să treacă prin trei din cele 2n+3 puncte, să conţină în interior n puncte, iar în exterior n puncte.

Ideea de rezolvare a acestei probleme este dată de următorii paşi:

1. Printre cele 2n+3 puncte se caută două puncte B şi C cu proprietatea că toate celelalte puncte (notate cu A_1, A_2, …, A_{2n+1}) sunt în acelaşi semiplan.
2. Ordonăm punctele A_1, A_2, …, A_{2n+1} astfel încât $m(\angle BA_1C) < m(\angle BA_2C) < \ldots < m(\angle BA_{2n+1}C)$.
3. Cercul care trece prin punctele A_{n+1}, B, C va lăsa punctele A_1, …, A_n în afară şi pe A_{n+2}, …, A_{2n+1} în interiorul său.
4. Pentru cercul de la pasul anterior se determină coordonatele centrului şi raza sa.

Probleme propuse în concursuri

1.
Agentul **007** are de distrus o tabără de teroriști. Tabăra de teroriști este formată din mai multe obiective (depozite de muniţie, pavilioane pentru teroriști, etc.), considerate punctiforme în plan. Agentul **007** primeşte de la serviciul de informaţii o hartă cu **n** obiective din tabăra teroriştilor, date prin coordonatele carteziene. Pe lângă hartă, agentul **007** mai primeşte şi o armă specială (construită pentru această misiune). Arma primită are două ţevi şi permite tragerea simultană pe aceeaşi direcţie (rectilinie), dar în sens invers a două rachete cu aceeaşi viteză. După ce se trage cu arma, odată cu atingerea unei ţinte explodează şi cealaltă rachetă (chiar dacă aceasta din urma nu şi-a atins ţinta).

Cerinţă
Agentul **007** vrea să distrugă tabăra cât mai repede şi cu cât mai puţine rachete, pentru acest lucru el studiază posibilitatea să se aşeze **într-un punct** din tabără (diferit de obiective) care să permită trageri eficace, adică la fiecare tragere să distrugă câte două obiective simultan.
Determinaţi dacă este posibil să se găsească un astfel de punct.

Date de intrare
În fişierul **a007.in** pe prima linie se află numărul de teste **k**, după care urmează date pentru fiecare test. Pentru fiecare test pe o linie se află **n**, iar pe următoarele **n** linii sunt coordonatele obiectivelor din tabăra teroriştilor (separate printr-un spaţiu în ordinea `abscisă ordonată`).

Date de ieşire
În fişierul **a007.out** se vor scrie **k** linii, pe fiecare linie se va scrie **1**, dacă există soluţie şi **0** dacă nu există soluţie. În cazul în care există soluţie se va scrie în continuare pe aceeaşi linie separate, printr-un spaţiu coordonatele punctului cerut (numere reale trunchiate la 4 zecimale, în ordinea `abscisă ordonată`).

Restricţii
0 ≤ n ≤ 10000
1 ≤ k ≤ 3
coordonatele punctelor sunt întregi din intervalul [-10000, 10000]

Observaţie
Un obiectiv este distrus dacă racheta explodează exact în punctul corespunzător lui.

Exemplu

a007.in	a007.out

2 4 10 0 10 10 0 10 0 0 6 0 0 10 0 2 10 12 0 5 0 7 0	1 5.0000 5.0000 0

(ONI, 2004)

2.

În regatul *XLand* oraşele erau înconjurate de ziduri în formă de poligoane convexe. Împăratul a dispus construirea unui drum de legătură directă între capitală şi un alt oraş dat. Fiecare extremitate a drumului poate fi orice punct situat pe zidul oraşului, respectiv capitalei. Lungimea drumului este distanţa dintre extremităţile sale.

Cerinţă

Determinaţi cel mai scurt drum dintre capitală şi oraşul dat.

Date de intrare

Fişier de intrare: **DRUM.IN**

Linia 1: K1

- număr natural nenul, reprezentând numărul de colţuri ale zidurilor capitalei;

Linia 2: x_1 y_1 x_2 y_2 ... x_{K1} y_{K1}

- K1 perechi de numere întregi, separate prin câte un spaţiu, reprezentând coordonatele vârfurilor zidurilor capitalei;

Linia 3: K2

- număr natural nenul, reprezentând numărul de colţuri ale zidurilor oraşului dat;

Linia 4: x_1 y_1 x_2 y_2 ... x_{K2} y_{K2}

- K2 perechi de numere întregi, separate prin câte un spaţiu, reprezentând coordonatele vârfurilor zidurilor acestui oraş.

Date de ieşire

Fişier de ieşire: **DRUM.OUT**

Linia 1: x_1 y_1 x_2 y_2

- patru numere reale trunchiate la 4 zecimale, separate prin câte un spaţiu, reprezentând extremităţile drumului de legătură respectiv.

Restricţii

- $2 \le K1, K2 \le 20$
- Coordonatele vârfurilor zidurilor ce înconjoară oraşul, respectiv capitala sunt numere întregi aparţinând intervalului [-100,100] şi sunt date fie în ordinea deplasării acelor de ceasornic, fie în sens invers deplasării acelor de ceasornic.
- Capitala şi oraşul nu au nici un punct comun (nu au puncte interioare comune şi nu au puncte comune pe zidurile lor).

Exemplu

DRUM.IN	DRUM.OUT
4 3 4 3 2 5 2 5 4 4 8 3 8 6 11 6 11 3	5.0000 3.5000 8.0000 3.5000

Timp maxim de executare/test: 1 secundă

(ONI, 2001, Bacău)

3. Într-o zonă agricolă care conţine mai multe proprietăţi (de formă poligonală – convexe sau concave) s-a desfăşurat un concurs de paraşutism. La acest concurs au participat **N** paraşutişti codificaţi prin **1**, **2**, …, **N**. Concursul este câştigat de acei paraşutişti care aterizează pe una dintre proprietăţile cu aria cea mai mare.

Cerinţă
Scrieţi un program care determină paraşutiştii care câştigă concursul.

Date de intrare
Fişierul de intrare **concurs.in** conţine:

concurs.in	*Semnificaţie*
N a_1 b_1 a_2 b_2 ... a_N b_N **m** **c1** x_{11} y_{11} x_{12} y_{12} … x_{1c1} y_{1c1} **c2** x_{21} y_{21} x_{22} y_{22} … x_{2c1} y_{2c1} … **cm** x_{m1} y_{m1} x_{m2} y_{m2} … x_{mc1} y_{mc1}	**N** – numărul de paraşutişti a_i b_i – abscisa şi ordonata punctului în care aterizează paraşutistul **i** **m** – numărul de proprietăţi c_i reprezintă numărul de colţuri ale proprietăţii **i**, iar x_{ij} y_{ij} reprezintă abscisa, respectiv ordonata unui colţ al proprietăţii **i**. Colţurile unei proprietăţi sunt date într-o ordine astfel încât să se poată parcurge continuu frontiera ei.

Date de ieşire
Fişierul de ieşire **concurs.out** conţine pe prima linie codurile asociate paraşutiştilor care au câştigat concursul în ordine crescătoare (separate prin câte un spaţiu). Dacă nu există câştigători se va scrie cifra **0**.
Restricţii
- **N** număr natural, $1 \le N \le 50$;
- $0 \le a_i, b_i \le 100$, numere reale cu maxim 4 cifre zecimale;
- **m** număr natural, $1 \le m \le 20$;
- c_i numere naturale, $1 \le c_i \le 20$;
- x_{ij}, y_{ij} sunt numere naturale ≤ 100

Observaţii
- Frontiera unei proprietăţi nu face parte din proprietate;
- Proprietăţile sunt disjuncte (nu există două proprietăţi cu suprafeţe de teren comune).
- Două numere reale `x` şi `y` le considerăm egale dacă $|x-y|<10^{-4}$.

Exemplu

concurs.in	**concurs.out**
4 **3 5** **10 4** **7 1** **3.002 6** **2** **4 9 3 11 3 11 5 9 5** **5 5 3 4 5 5 7 3 7 2 5**	**1 4**

Timp maxim de execuţie: ***8 secunde/test***

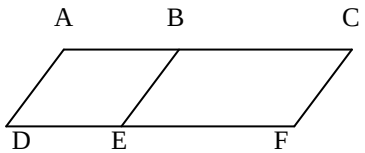
(Baraj, Bucuresti, 2002)

4. Se da un poligon convex cu **n** vârfuri ($50 \ge \mathbf{n} \ge 4$) prin coordonatele vârfurilor (numere întregi din intervalul [-2000, 2000]) date în sensul acelor de ceasornic.
a) Sa se verifice daca poligonul are centru de simetrie.
b) Sa se împarta poligonul (daca are centru de simetrie) în paralelograme.
Datele de intrare se citesc din fi[ierul text **POLIGON.IN** cu structura:
n
x1 y1
x2 y2
…
xn yn

Datele de iesire se vor introduce în fi[ierul text **POLIGON.OUT** cu structura:
mesaj
px11 py11 px21 py21 px31 py31 px41 py41
px12 py12 px22 py22 px32 py32 px42 py42
…
px1k py1k px2k py2k px3k py3k px4k py4k
unde **mesaj** poate fi "**DA**" sau "**NU**", în functie de existenta sau nu a centrului de simetrie. Daca mesajul este "**NU**", fisierul va contine doar prima linie, altfel va mai contine coordonatele vârfurilor (în sensul acelor de ceasornic) paralelogramelor, pe câte un rând fiecare.

Exemple:
Pentru fisierul **POLIGON.IN** cu continutul:
8
20 30
30 30
40 20
40 10
30 0
20 0
10 10
10 20
Fisierul de iesire **POLIGON.OUT** poate contine spre exemplu solutia:
DA
20 0 10 10 20 10 30 0
10 10 10 20 20 20 20 10
10 20 20 30 30 30 20 20
20 10 20 20 30 10 30 0
20 20 30 30 40 20 30 10
40 20 40 10 30 0 30 10
Pentru fisierul **POLIGON.IN** cu continutul:
4
10 20
40 30
40 10
20 10
Fisierul de iesire **POLIGON.OUT** va contine:
NU

Observatie: Sirul de paralelograme folosit la împartirea poligonului nu va contine paralelograme în configuratia din figura alaturata, pentru ca paralelogramele ABED si BCFE pot fi înlocuite cu ACFD.



(ONI, 2000)

5.
Generalul Donald are pe câmpul de luptă n soldaţi. Pentru a şti exact unde sunt dispuşi soldaţii a primit de la un colonel o hartă pe care este precizată poziţia lor. Pentru a obţine informaţii noi din tabăra inamicului a trimis o grupă de cercetare în teren. După câteva ore această grupă se întoarce cu o informaţie preţioasă, inamicul are o armă care poate distruge pe o direcţie dreaptă orice obstacol de pe hartă. Având această informaţie generalul Donald studiază harta şi observă că are cam mulţi soldaţi dispuşi pe aceeaşi linie dreaptă.

Cerinţă
Se cere să se determine numărul maxim de soldaţi care se găsesc poziţionaţi pe o aceeaşi linie dreaptă.

Date de intrare
Fişierul de intrare **sol.in** conţine pe prima linie numărul n, iar pe următoarele n linii coordonatele carteziene ale fiecărui soldat pe hartă. O astfel de linie are formatul:
abscisă ordonată (separate între ele printr-un singur spaţiu).

Date de ieşire
Fişierul de ieşire **sol.out** se va conţine numărul maxim de soldaţi care se găsesc poziţionaţi pe o aceeaşi linie dreaptă.

Restricţii
$1 \leq n \leq 300$
Toate coordonatele poziţiilor soldaţilor sunt numere întregi din intervalul [-1000,1000].
Nu există doi soldaţii aflaţi în aceeaşi poziţie.

Exemplu

sol.in	sol.out	Explicaţie
5 5 5 10 2 18 5 5 2 10 5	3	Există trei soldaţi care se găsesc poziţionaţi pe o linie dreaptă.

Timp maxim de execuţie: 0.1 secunde/test

(.campion, 2005-2006)

6.
O companie fomată din **n** soldaţi (codificaţi prin numerele 1, 2, …, **n**) are ca misiune să păzească un obiectiv militar de maximă importanţă. Soldaţii au fiecare la dispoziţie un anumit număr de cartuşe pe care le poate folosi pentru apărarea obiectivului. Fiecare soldat **i**, $\mathbf{i} \in \{1, 2, \ldots, \mathbf{n}\}$ se aşează într-un anumit loc caracterizat prin coordonate (abscisă şi ordonată) şi îşi sapă un adăpost pentru a putea rezista inamicului. Nu se pot aşeza doi soldaţi pe acelaşi loc.
Comandantul stabileşte zonele de apărare contrice. O zonă de apărare este delimitată de acei soldaţi care sunt în viaţă şi care dacă sunt uniţi prin tranşee (segmente de dreaptă) formează un poligon convex ce conţine în interiorul său toţi ceilalţi soldaţi aflaţi în viaţă. O zonă de apărare conţine cel puţin trei soldaţi şi se formează imediat după ce zona anterioară a cedat.
O zonă de apărare cedează când cel puţin un soldat din ea nu mai are cartuşe (în acel moment toţi soldaţii de pe zona de apărare respectivă sunt omorâţi de inamic).
Se ştie că soldaţii de pe aceeaşi zonă de apărare trag la un interval de o secundă, în acelaşi timp. După ce o zonă de apărare este distrusă, imediat în secunda următoare intră în acţiune zona de apărare următoare.

Cerinţă
Să se determine după câte secunde va cuceri inamicul obiectivul păzit.
Date de intrare
În fişierul **sold.in** se află pe prima linie **n**, iar pe următoarele **n** linii coordonatele poziţiei soldaţilor 1, 2, …, **n** împreună cu numărul de cartuşe pe care le au separate prin câte un spaţiu (ordinea pe linie este următoarea: abscisă, ordonată şi număr cartuşe).

Date de ieşire
În fişierul **sold.out** se va scrie numărul de secunde cerut.
Restricţii
n -număr natural nenul <3200
coordonatele poziţiei soldaţilor sunt numere întregi din intervalul [-32000, 32000].
numărul de cartuşe pentru fiecare soldat este natural nenul şi <1000000.

Exemplu

sold.in
```
11
5 0 23
0 2 100
4 3 20000
5 4 10
2 1 301
0 0 45
3 4 567
2 3 342
2 0 2123
3 2 90
4 1 20
```

sold.out
```
30
```

Timp maxim de execuţie/test: 1 secundă

(.campion, 2004)

7.
Ionică a primit de ziua lui de la tatăl său un joc format din piese de formă de triunghiulară de dimensiuni diferite şi o suprafaţă magnetică pe care acestea pot fi aşezate. Pe suprafaţa magnetică este desenat un triunghi dreptunghic cu lungimile catetelor `a`, respectiv `b` şi un sistem de coordonate xOy cu originea în unghiul drept al triunghiului, semiaxa [Ox pe cateta de lungime `a`, respectiv semiaxa [Oy pe cateta de lungime `b`. La un moment dat Ionică aşează pe tabla magnetică `n` piese, pentru care se cunosc coordonatele vârfurilor lor. Tatăl lui Ionică vrea să verifice dacă pe tablă piesele realizează o **partiţie** a triunghiului dreptunghic desenat, adică dacă sunt îndeplinite condiţiile:
nu există piese suprapuse;
piesele acoperă toată porţiunea desenată (în formă de triunghi dreptunghic);
nu există porţiuni din piese în afara triunghiului desenat.

Cerinţă
Se cere să se verifice dacă piesele plasate pe tabla magnetică formează o **partiţie** a triunghiului desenat pe tabla magnetică.

Date de intrare
Fişierul de intrare `part.in` conţine pe prima linie un număr natural `k`, reprezentând numărul de seturi de date din fişier. Urmează `k` grupe de linii, câte o grupă pentru fiecare set de date. Grupa de linii corespunzătoare unui set este formată dintr-o linie cu numerele `a, b, n` separate între ele prin câte un spaţiu şi `n` linii cu câte şase numere întregi separate prin spaţii reprezentând coordonatele vârfurilor (`abscisă ordonată`) celor `n` piese, câte o piesă pe o linie.

Date de ieşire
În fişierul `part.out` se vor scrie `k` linii, câte o linie pentru fiecare set de date. Pe linia `i` (`i=1, 2, …, k`) se va scrie `1` dacă triunghiurile din setul de date `i` formează o **partiţie** a triunghiului desenat pe tabla magnetică sau `0` în caz contrar.

Restricţii
`1 ≤ n ≤ 150`
`1 ≤ k ≤ 10`
`a, b` sunt numere întregi din intervalul [`0, 31000`]
Coordonatele vârfurilor pieselor sunt numere întregi din intervalul [`0, 31000`].

Exemplu

part.in	part.out
2 20 10 4 0 5 0 10 10 5 0 0 10 5 0 5 0 0 10 0 10 5 10 0 20 0 10 5 20 10 2 0 0 0 10 10 5 0 0 20 0 20 10	1 0

Timp maxim de execuţie: 0.3 secunde/test

(Baraj, 2006)

8.

Într-un document arheologic recent descoperit se face referire la un mare tezaur. Datorită faptului că documentul poate fi interpretat în mai multe moduri se face apel la **n** arheologi care vor studia independent documentul.

La terminarea studiului fiecare arheolog întocmeşte o hartă pe care marchează o zonă poligonală închisă şi convexă ce este considerată a conţine tezaurul.

Pentru că fondurile alocate pentru descoperirea tezaurului sunt reduse se ia hotararea să se începă cercetările pe teren doar în zona de pe hartă precizată de toţi arheologii.

Cunoscând **n** şi coordonatele vârfurilor zonelor determinate de arheologi se cere să se determine suprafaţa (aria) zonei de pe hartă precizată de toţi arheologii (intersecţia celor n zone

Date de intrare:

Fisierul de intrare **TEZAUR.IN** conţine:

n

m[1] -numărul de vârfuri pentru zona primului arheolog

x_{11} y_{11} ... x_1 $y_{1m[1]}$ -coordonatele varfurilor zonei primului arheolog

... (date in sens direct sau invers acelor de ceasornic)

m[n] -numarul de varfuri pentru zona ultimului arheolog

$x_{1m[n]}$ $y_{1m[n]}$ ... $x_{nm[n]}$ $y_{nm[n]}$ -coordonatele varfurilor zonei celui de-al n-lea arheolog (date in sens direct sau invers acelor de ceasornic)

Date de ieşire:

Ieşirea se va face în fişierul **TEZAUR.OUT** ce va conţine pe prima linie suprafaţa (aria) zonei de pe hartă precizată de toţi arheologii.

Observaţie: Dacă nu există suprafaţă comună zonelor celor **n** arheologi se va scrie în fişierul **TEZAUR.OUT** valoarea 0.

Exemplu:

TEZAUR.IN

```
3
4
-20 30 40 30 40 -20 -20 -20
5
10 -30 60 -50 100 60 70 80 10 80
5
40 0 70 0 70 60 20 60 20 30
```

TEZAUR.OUT

```
300.0000
```

Restricţii:

- $1 \le n \le 30$;
- $3 \le m_i \le 20$, $i \in \{1,2,\ldots,n\}$;
- Coordonatele vârfurilor zonelor sunt numere întregi din intervalul [-10000, 10000];
- Numărul din fişierul de ieşire se va scrie cu 4 zecimale.

Timp maxim de execuţie pe test: 1 secundă

(ONI, Constanţa, 2000)

9.
Se dau N pătrate situate în plan, având laturile paralele cu axele de coordonate. Coordonatele vârfurilor sunt numere întregi. Pătratele nu se ating şi nu se suprapun.
Se cere să se determine numărul de pătrate vizibile din originea O, O=(0,0).
Un pătrat este *vizibil* din originea O, dacă există două puncte distincte A şi B de coordonate întregi, situate pe o aceeaşi latură a pătratului, astfel încât interiorul triunghiului OAB nu are puncte comune cu nici unul dintre celelalte pătrate.

Date de intrare
Prima linie a fişierului de intrare SQUARES.IN conţine un număr întreg N, 1<=N<=1000, reprezentând numărul de pătrate.
Fiecare din următoarele N linii descrie un pătrat prin numerele întregi X,Y şi L separate prin câte un spaţiu 1<=X,Y,L<=10000. X şi Y sunt coordonatele colţului din stânga jos al pătratului, iar L este lungimea laturii.

Date de ieşire
Prima (şi singura) linie a fişierului de ieşire SQUARES.OUT trebuie să conţină numărul pătratelor vizibile din origine.

Exemple

SQUARES.IN	**SQUARES.OUT**
3 2 6 3 1 4 1 3 4 1	3
SQUARES.IN	**SQUARES.OUT**
4 1 2 1 3 1 1 2 4 2 3 7 1	2

(CEOI, Croaţia, 1998)

10.
N soldaţi din ţara *Caroiaj* sunt aşezaţi la întâmplare pe teritoriul ţării. O poziţie în ţara *Caroiaj* este dată de coordonatele (x,y), numere întregi. Soldaţii pot să-şi schimbe poziţia astfel: într-o mutare, un soldat poate să se deplaseze cu o unitate în sus, în jos, la stânga sau la dreapta (îşi schimbă fie coordonata x, fie coordonata y cu 1 sau -1).
Soldaţii vor să se alinieze pe orizontală, unul lângă altul (astfel încât poziţiile lor finale vor fi (x,y), (x+1,y), …, (x+N-1,y), pentru un anumit x şi y). Întregii x şi y sunt oarecare. Ordinea finală a soldaţilor pe linie, este de asemenea oarecare.
Să se determine numărul minim de interschimbări de poziţie a tuturor soldaţilor, necesare pentru aliniere.
Doi sau mai mulţi soldaţi nu pot ocupa în acelaşi timp aceeaşi poziţie.

Date de intrare
Prima linie a fişierului SOLDIERS.IN conţine un număr întreg N, 1<=N<=10000, reprezentând numărul de soldaţi.
Următoarele N linii conţin poziţiile iniţiale ale soldaţilor: pentru orice i, 1<=i<=N, linia i+1 conţine o pereche de numere întregi x[i] şi y[i] separate printr-un singur spaţiu, reprezentând coordonatele pătratului i, -10000<=x[i],y[i]<=10000.

Date de ieşire
Prima şi singura linie a fişierului de ieşire SOLDIERS.OUT va conţine numărul minim total de schimbări de poziţie a soldaţilor, necesar pentru aliniere.

Exemple

SOLDIERS.IN	SOLDIERS.OUT
3 1 0 2 4 3 2	4
SOLDIERS.IN	**SOLDIERS.OUT**
5 1 2 2 2 1 3 3 -2 3 3	8

(CEOI, Croaţia, 1998)

Bibliografie

1. Grigore Albeanu, *Algoritmi geometrici elementari*, Gazeta de Informatică, 4-5, 1998
2. T. Cormen, C.Leiserson, R. Rivest, *Introducere în algoritmi*, Ed. Agora, 2000
3. Victor Mitrana, *Provocarea algoritmilor*, Ed. Agni, 1994
4. Mihai Scorţaru, *Informatica pentru grupele de excelenţă*, Ed. Dacia, 2004
5. E. Cerchez, M. Şerban, *Programarea în limbajul C++*, III, Ed. Polirom, pg. 2258.