Atena, septembrie 2004

Olimpiada INTERNAȚIONALĂ de Informatică 2004

Vă prezentăm în continuare enunțurile celor șase probleme propuse spre rezolvare la cea de-a șaisprezecea ediție a celui mai important concurs internațional de informatică destinat elevilor.

P040601: Artemis

Zeus i-a dăruit lui *Artemis*, zeița vânătorii, o zonă dreptunghiulară pentru a planta o pădure.

Zona este delimitată la stânga de un segment aflat pe semiaxa verticală pozitivă, iar la dreapta de un segment aflat pe semiaxa orizontală pozitivă.

Colțul din stânga-jos a hărții se află în punctul de coordonate (0, 0). *Zeus* i-a cerut zeiței să planteze copaci doar în puncte care au coordonate întregi.

Artemis a dorit ca pădurea să pară cât mai naturală, motiv pentru care a plantat pădurea astfel încât oricare linie care unește doi copaci să nu fie paralelă cu axele de coordonate.

Uneori *Zeus* dorește ca *Artemis* să taie copaci pentru el. Copacii trebuie tăiați respectându-se următoarele reguli:

- *Zeus* dorește să fie tăiați cel puțin *T* copaci.
- Pentru a obține o zonă în care se poate construi un teren de fotbal (*Zeus* dorește ca în viitor să obțină performanțe deosebite în acest domeniu), *Artemis* trebuie să taie toți copacii dintr-o zonă dreptunghiulară și să nu taie nici un copac din exteriorul zonei respective.
- Laturile zonei dreptunghiulare trebuie să fie paralele cu axele de coordonate.
- În două colțuri opuse ale zonei trebuie să se afle copaci și copacii din aceste colțuri trebuie și ei tăiați.

Datorită faptului că lui *Artemis* îi plac foarte mult copacii, ea dorește să îndeplinească aceste condiții tăind cât mai puțini copaci.

Va trebui să scrieți un program care, pe baza informațiilor disponibile despre pădure și a numărului minim *T* al copacilor care trebuie tăiați, determină zona dreptunghiulară din care *Artemis* va tăia copacii.

Date de intrare

Denumirea fișierului de intrare este **`artemis.in`**.

Prima linia a acestuia conține un număr întreg *N*, care reprezintă numărul copacilor din pădure.

Cea de-a doua linie conține un număr întreg *T* care reprezintă numărul minim al copacilor care trebuie tăiați.

Următoatele *N* linii descriu pozițiile celor *N* copaci. Fiecare dintre aceste linii conține două numere întregi *X* și *Y* care reprezintă coordonata orizontală, respectiv coordonata verticală, a unuia dintre copacii din pădurea plantată de către *Artemis*.

Date de ieșire

Denumirea fișierului de ieșire este **`artemis.out`**.

Acest fișier trebuie să conțină o singură linie pe care se vor afla două numere întregi *I* și *J*, separate printr-un spațiu.

Artemis ar trebui să folosească cel de-al *I*-lea copac (cel a cărui poziție este descrisă pe cea de-a (*I*+2)-a linie a fișierului de intrare) și cel de-al *J*-lea copac (cel a cărui poziție este descrisă pe cea de-a (*J*+2)-a linie a fișierului de intrare) pentru a identifica cele două colțuri opuse ale zonei dreptunghiulare din care vor fi tăiați copaci și unde va fi construit terenul de fotbal.

Ordinea celor două numere nu este importantă.

Dacă există mai multe soluții, trebuie determinată oricare dintre ele.

Se garantează existența cel puțin a unei soluții.

Restricții

Numărul copacilor din pădure este cuprins între 2 și 20.000.

Coordonatele copacilor din pădure sunt numere întregi cuprinse între 0 și 64.000.

Numărul copacilor care trebuie tăiați va fi întotdeauna cel mult egal cu numărul copacilor din pădurea plantată de *Artemis*.

Exemplu

artemis.in

```
3
2
1 1
2 3
5 6
```

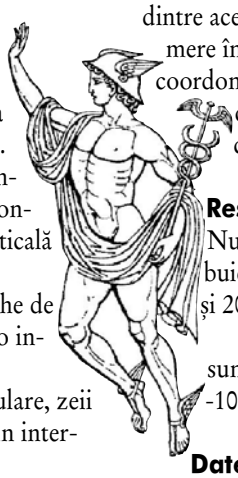
artemis.out

```
1 2
```

Timp de execuție: 1 secundă/test
Memorie disponibilă: 16 MB

P040602: Hermes

Într-un oraș modern al zeilor eleni, străzile sunt dispuse sub forma unui caroiaj cu coordonate întregi, fiecare stradă fiind paralelă cu una dintre axele de coordonate.



Pentru fiecare număr întreg Z, există o stradă orizontală la $y = Z$ și o stradă verticală la $x = Z$.

Ca urmare, orice pereche de numere întregi reprezintă o intersecție de străzi.

În timpul zilelor caniculare, zeii se odihnesc la cafenelele din intersecții.

Curierul *Hermes* trebuie să trimită mesaje fotonice zeilor din cafenele deplasându-se numai de-a lungul străzilor.

Fiecare mesaj este destinat unui anumit zeu, dar nu contează dacă și alți zei primesc mesajul în momentul în care acesta este transmis de către *Hermes*.

Mesajele trebuie transmise într-o anumită ordine și *Hermes* cunoaște coordonatele cafenelelor în ordinea respectivă. *Hermes* pornește din punctul de coordonate (0, 0).

Pentru a trimite un mesaj la o cafenea aflată în punctul de coordonate (X_i,Y_i), este suficient ca *Hermes* să ajungă într-un punct al străzii orizontale pe care se află cafeneaua (cea a cărei coordonată verticală este Y_i) sau într-un punct al străzii verticale pe care se află cafeneaua (cea a cărei coordonată orizontală este X_i). După trimiterea tuturor mesajelor, *Hermes* se oprește.

Trebuie să scrieți un program care, dându-se o secvență de cafenele, determină distanța minimă pe care trebuie să o parcurgă *Hermes* pentru a trimite toate mesajele.

Date de intrare

Denumirea fișierului de intrare este **hermes.in**.

Prima linia a acestuia conține un număr întreg *N*, care reprezintă numărul mesajelor care trebuie trimise.

Următoatele *N* linii descriu pozițiile celor *N* cafenele la care trebuie trimise mesajele, în ordinea în care trebuie trimise aceste mesaje. Fiecare dintre aceste linii conține două numere întregi *X* și *Y* care reprezintă coordonata orizontală, respectiv coordonata verticală, a unei cafenele.

Restricții

Numărul mesajelor care trebuie trimise este cuprins între 1 și 20.000.

Coordonatele cafenelelor sunt numere cuprinse între -1000 și 1000.

Date de ieșire

Denumirea fișierului de ieșire este **hermes.out**.

Acest fișier trebuie să conțină o singură linie pe care se va afla un singur număr, reprezentând distanța minimă pe care trebuie să o parcurgă *Hermes*.

Exemplu

hermes.in

```
5
8 3
7 -7
8 1
-2 1
-2 1
6 -5
```

hermes.out

```
11
```

Timp de execuție: 1 secundă/test
Memorie disponibilă: 16 MB

P040603: Poligon

Un poligon este format din toate punctele aflate în interiorul său sau pe laturile sale.

Un poligon convex are proprietatea că, oricum am alege două dintre punctele sale, toate punctele care se află pe segmentul care unește cele două puncte sunt puncte ale poligonului.

Pentru această problemă vom lua în considerare doar poligoane convexe cu cel puțin două vârfuri, iar coordonatele vârfurilor sunt numere întregi.

Impunem și o condiție suplimentară: oricare trei vărfuri ale poligonului nu sunt coliniare.

Dându-se două poligoane *A* și *B*, suma *Minkovski* a celor două poligoane este un poligon format din toate punctele de forma $(x_1 + x_2, y_1 + y_2)$, unde (x_1, y_1) este un punct care face parte din poligonul *A*, iar (x_2, y_2) este un punct care face parte din poligonul *B*.

Se poate demonstra că suma *Minkovski* a două poligoane este întotdeauna un poligon.

În figura 1 este prezentat un exemplu: două triunghiuri și suma *Minkovski* a acestora.

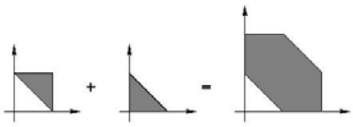


Figura 1

Ne interesează operația inversă sumei *Minkowski*. Pentru un poligon dat *P*, vom căuta două poligoane *A* și *B* astfel încât:

- Poligonul *P* reprezintă suma *Minkovski* a poligoanelor *A* și *B*.
- Poligonul *A* conține între două și patru vârfuri diferite. Cu alte cuvinte, este un segment (două vârfuri), un triunghi (trei vârfuri) sau un patrulater convex (patru vârfuri).
- Poligonul *A* trebuie să conțină cât mai multe vârfuri. Așadar, el trebuie să fie un patrulater dacă este posibil. Dacă nu poate fi patrulater, atunci trebuie să fie un triunghi, iar dacă nu poate fi nici triunghi, atunci trebuie să fie un segment (poligon degenerat cu două vârfuri).

Evident nici unul dintre poligoanele *A* și *B* nu poate fi identic cu *P*, deoarece celălalt poligon ar fi dat de un singur punct, ceea ce nu este permis.

Pentru această problemă sunt date mai multe fișiere de intrare, fiecare conținând o descriere a unui poligon *P*.

Pentru fiecare fișier de intrare trebuie determinate poligoanele *A* și *B* care respectă condițiile amintite anterior și trebuie creat un fișier de ieșire care conține descrierile poligoanelor *A* și *B*.

Se garantează existența poligoanelor *A* și *B* pentru toate poligoanele *P* descrise în fișierul de intrare.

Dacă există mai multe soluții posibile, poate fi aleasă oricare dintre ele.

Pentru această problemă nu trebuie scris nici un program, fiind suficientă generarea fișierelor de ieșire corecte.

Date de intrare

Sunt date zece fișiere de intrare, denumite **`poligon1.in`**, **`poligon2.in`** și așa mai departe, până la **`poligon10.in`**.

Fiecare fișier are structura descrisă în continuare.

Prima linie conține un număr întreg *N*, care reprezintă numărul vârfurilor poligonului *P*.

Următoarele *N* linii descriu vârfurile poligonului *P*, în ordinea inversă acelor de ceasornic. O astfel de linie descrie un vârf și conține două numere întregi *X* și *Y* care reprezintă coordonatele vârfului respectiv. Toate coordonatele sunt numere întregi nenegative.

Date de ieșire

Vor trebui generate zece fișiere de ieșire, denumite **`poligon1.out`**, **`poligon2.out`** și așa mai departe, până la **`poligon10.out`**.

Fiecare astfel de fișier va conține descrierea a două poligoane *A* și *B*, a căror sumă *Minkovski* este poligonul *P* din fișierul de intrare corespunzător.

Prima linie a unui astfel de fișier va trebui să conțină textul `#FILE polygon I`, unde *I* este un număr întreg cuprins între 1 și 10 și identifică numărul de ordine al fișierului respectiv. În continuare, structura fișierelor de ieșire este similară fișierelor de intrare.

Cea de-a doua linie trebuie să conțină un număr *NA* cuprins între 2 și 4, care reprezintă numărul vârfurilor poligonului *A*.

Următoarele *NA* linii trebuie să descrie vârfurile poligonului *A*, în ordinea inversă acelor de ceasornic. O astfel de linie va descrie un vârf și va conține două numere întregi *X* și *Y* care reprezintă coordonatele vârfului respectiv.

Linia următoare trebuie să conțină un număr *NB* mai mare sau egal cu 2, care reprezintă numărul vârfurilor poligonului *B*.

Următoarele *NB* linii trebuie să descrie vârfurile poligonului *B*, în ordinea inversă acelor de ceasornic. O astfel de linie va descrie un vârf și va conține două numere întregi *X* și *Y* care reprezintă coordonatele vârfului respectiv.

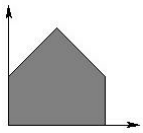
Exemplu

Vom considera că exemplu care urmează descrie fișierul cu numărul de ordine 0.

Datele de intrare sunt următoarele:

`polygon0.in`

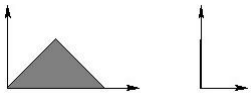
```
5
0 1
0 0
2 0
2 1
1 2
```



Pentru acest exemplu, ambele fișiere de ieșire prezentate în cele ce urmează sunt corecte. O primă soluție este:

`polygon0.out`

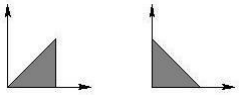
```
3
0 0
2 0
1 1
2
0 1
0 0
```



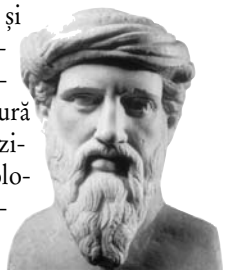
Cea de-a doua soluție este următoarea:

`polygon0.out`

```
3
0 0
1 0
1 1
3
0 1
0 0
1 0
```



P040604: Empodii



Matematicianul și filosoful *Pitagora* credea că realitatea are o natură matematică. În zilele noastre, biologii studiază proprietățile secvențelor biologice.

O secvență biologică este o succesiune de *M* numere întregi, astfel încât:

- Secvența conține toate numerele cuprinse între 0 și *M* - 1.
- Secvența începe cu 0 și se termină cu *M* - 1.
- Nu există două elemente de forma *E* și *E* + 1 aflate pe poziții consecutive în această ordine.

O subsecvență formată din elemente adiacente ale unei secvențe biologice este numită ***segment***.

Un segment al unei secvențe biologice este numit ***interval mărginit*** dacă include toate numerele întregi cuprinse între valoarea primului element al segmentului (aceasta trebuie să fie cea mai mică dintre valorile elementelor segmentului) și valoarea ultimului element al segmentului (aceasta trebuie să fie cea mai mare dintre valorile elementelor segmentului).

Un interval mărginit este numit ***empodiu*** dacă în cadrul său nu apare nici un alt interval mărginit mai scurt.

De exemplu, să considerăm secvența biologică (0, 3, 5, 4, 6, 2, 1, 7). Se observă că întreaga secvență biologică este un interval mărginit. Totuși, ea conține un alt interval mărginit, și anume (3, 5, 4,6), deci nu este un empodiu. Intervalul mărginit (3, 5, 4, 6) nu conține nici un alt interval mărginit, deci este un empodiu.

Mai mult, acesta este singurul empodiu din cadrul secvenței biologice considerate.

Trebuie să scrieți un program care, dându-se o secvență biologică, determină toate empodiile din cadrul secvenței biologice respective.

Date de intrare

Denumirea fișierului de intrare este **`empodia.in`**.

Prima linie a fișierului conține un singur număr întreg *M* care reprezintă numărul elementelor secvenței biologice.

Următoarele *M* linii conțin numerele întregi ale secvenței biologice, în ordinea în care apar acestea în secvență. Fiecare dintre aceste linii conține un singur număr întreg.

Date de ieșire

Denumirea fișierului de ieșire este **`empodia.out`**.

Prima linie a acestui fișier trebuie să conțină un număr întreg *H* care reprezintă numărul empodiilor din secvența biologică.

Următoarele *H* linii descriu toate empodiile din secvența biologică dată, în ordinea apariției acestora în secvența biologică. Fiecare dintre aceste linii trebuie să conțină două numere întregi *A* și *B* (în această ordine), separate printr-un spațiu, unde al *A*-lea element al secvenței biologice este primul element al empodiului, iar al *B*-lea elemenet al secvenței biologice este ultimul element al empodiului.

Restricții

Numărul elementelor din secvența biologică este cuprins între 2 și 1.100.000.

Exemplu

`empodia.in`

```
8
0
3
5
4
6
2
1
7
```

`empodia.out`

```
1
2 5
```

Timp de execuție: 1 secundă/test
Memorie disponibilă: 128 MB

P040605: Fermierul

Un fermier are mai multe terenuri, fiecare fiind înconjurat de chiparoși. De asemenea, fermierul are mai multe fâșii de pământ, fiecare conținând o linie de chiparoși.

Atât pe terenuri cât și pe fâșii, între oricare doi chiparoși consecutivi (învecinați) se află câte un măslin.

Toți chiparoșii fermierului înconjoară un teren sau se află pe o fâșie și toți măslinii fermierului se află între doi chiparoși consecutivi de pe un teren sau de pe o fâșie.

Într-o zi fermierul s-a îmbolnăvit și a simțit că i se apropie sfârșitul.

Când mai avea doar câteva zile de trăit l-a chemat pe fiul cel mare și i-a spus:

"*Îți voi da oricare Q chiparoși pe care îi dorești și toți măslinii care se află între doi chiparoși consecutivi dintre cei aleși.*"

De pe fiecare teren și de pe fiecare fâșie, fiul poate alege orice combinație de chiparoși.

Datorită faptului că fiului îi plac foarte mult măslinele, el dorește să aleagă cei Q chiparoși astfel încât să primească un număr cât mai mare de măslini.

Pentru exemplul din figura 2 considerăm că fiul primește 17 chiparoși. Pentru a maximiza numărul de măslini pe care îi va primi, fiul va trebui să aleagă toți chiparoșii de pe primele două terenuri. Astfel, fiul va reuși să obțină 17 măslini.

Trebuie să scrieți un program care, pe baza informațiilor referitoare la terenurile și la fâșiile fermierului și a numărului de chiparoși pe care îi poate alege fiul, determină cel mai mare număr de măslini pe care îi poate primi fiul.

Date de intrare

Denumirea fișierului de intrare este **`farmer.in`**.

Prima linie a fișierului conține trei numere întregi *Q*, *M* și *K*. *Q* reprezintă numărul chiparoșilor pe care îi poate alege fiul, *M* reprezintă numă-

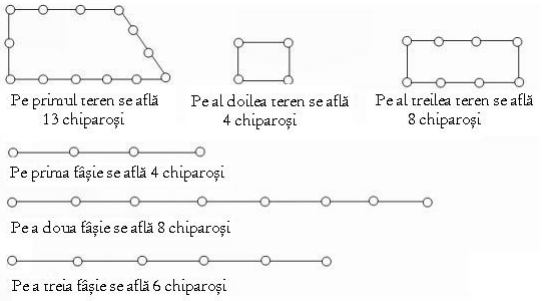


Figura 2

rul terenurilor, iar K reprezintă numărul fâșiilor.

Cea de-a doua linie conține M numere întregi, reprezentând numărul de chiparoși de pe fiecare dintre cele M terenuri.

Cea de-a treia linie conține K numere întregi, reprezentând numărul de chiparoși de pe fiecare dintre cele K fâșii.

Date de ieșire

Denumirea fișierului de ieșire este **farmer.out**.

Fișierul va conține o singură linie pe care se va afla un singur număr reprezentând numărul maxim al măslinilor pe care îi poate primi fiul fermierului.

Restricții și precizări

Numărul chiparoșilor care trebuie aleși este cuprins între 0 și 150.000.

Numărul terenurilor și numărul fâșiilor sunt cuprinse între 0 și 2000.

Pe fiecare teren se află între 3 și 150 de chiparoși.

Pe fiecare fâșie se află între 2 și 150 de chiparoși.

Exemplu

farmer.in

```
17 3 3
13 4 8
4 8 6
```

farmer.out

```
17
```

Timp de execuție: 1 secundă/test
Memorie disponibilă: 16 MB

P040606: Phidias

Faimosul sculptor elen *Phidias* se pregătește să construiască un nou monument și are nevoie de plăci dreptunghiulare de marmură de dimensiuni $W_1 \times H_1$, $W_2 \times H_2$, ..., $W_N \times H_N$.

Recent, *Phidias* a primit o dală mare din marmură având forma dreptunghiulară. El vrea să taie dala astfel încât să obțină plăcile de dimensiunile dorite.

Fiecare bucată de marmură (dala sau bucățile tăiate din aceasta) poate fi tăiată fie orizontal, fie vertical, obținându-se două plăci dreptunghiulare ale căror dimensiuni sunt numere întregi.

Fiecare tăietură trebuie să fie completă. Aceasta este singura modalitate de tăiere, iar două plăci tăiate nu pot fi lipite ulterior pentru a obține o nouă placă.

Datorită faptului că pe dala de marmură primită este desenat un model, plăcile tăiate nu pot fi rotite.

Dacă *Phidias* taie o placă de dimensiuni $A \times B$, ea nu poate fi folosită în locul unei plăci de dimensiuni $B \times A$ decât dacă $A = B$.

Pentru fiecare dimensiune specificată, *Phidias* poate tăia nici una, una, două sau mai multe plăci care au dimensiunea respectivă.

O placă este pierdută dacă nu are nici una dintre dimensiunile dorite după ce s-au efectuat toate tăieturile.

Phidias se întreabă cum poate tăia dala astfel încât să se piardă cât mai puțină marmură.

De exemplu, să presupunem că în figura 3 lungimea dalei este 21, iar lățimea este 11.

Phidias dorește să construiască dale de dimensiuni 10×4, 6×2, 7×5 și 15×10.

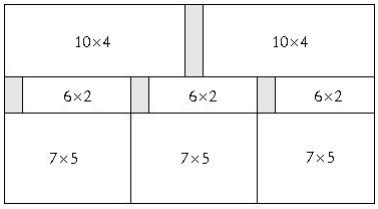


Figura 3

Suprafața minimă pierdută este 10, iar figura prezintă o succesiune de tăieturi care duce la obținerea acestei valori.

Trebuie să scrieți un program care, pe baza dimensiunilor dalei și ale pieselor care trebuie tăiate, determină suprafața minimă care este pierdută.

Date de intrare

Denumirea fișierului de intrare este **phidias.in**.

Prima linie a fișierului conține două numere întregi W și H. W reprezintă lungimea dalei, iar H reprezintă lățimea acesteia.

Cea de-a doua linie conține un număr întreg care reprezintă numărul dimensiunilor dorite pentru plăcile tăiate.

Următoarele N linii conțin dimensiunile dorite ale plăcilor. Fiecare dintre aceste linii conține câte două numere întregi. Primul dintre ele reprezintă lungimea dorită a piesei, iar al doilea reprezintă lățimea dorită.

Date de ieșire

Denumirea fișierului de ieșire este **phidias.out**.

Fișierul va conține o singură linie pe care se va afla un singur număr reprezentând suprafața minimă care este pierdută.

Restricții

Dimensiunile dalei sunt numere întregi cuprinse între 1 și 600.

Numărul dimensiunilor dorite este cuprins între 1 și 200.

Lățimile plăcilor care trebuie obținute sunt cuprinse între 1 și lățimea dalei, în timp ce lungimile acestor plăci sunt cuprinse între 1 și lungimea dalei.

Exemple

phidias.in

```
21 11
4
10 4
6 2
7 5
15 10
```

phidias.out

```
10
```

Timp de execuție: 1 secundă/test
Memorie disponibilă: 16 MB