I.S.M.B.
Olimpiada de Informatica – faza pe Municipiu
clasa a IX-a, 17 martie 2001

Planeta *cubică*

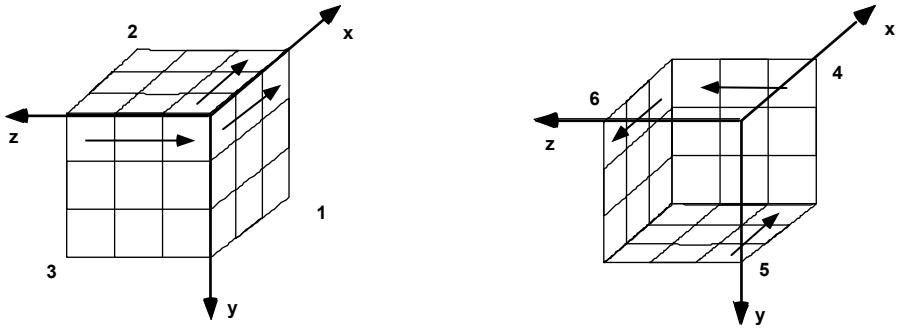
Punctaj: 50 puncte

Suntem în anul 3001 şi viteza superluminică a fost atinsă de mult. Pământenii au colonizat deja mai multe galaxii şi continuă explorarea spaţiului cosmic. Totuşi, una din problemele care n-au fost rezolvate eficient, este legislaţia interstelară. Fiindcă această nu propune o soluţie practică în problema autorităţii planetare, s-a făcut următoarea convenţie: la colonizarea unei noi planete, baza interstelară care are cea mai mare colonie pe planetă va deţine controlul pentru primii 400 de ani.

Cei de la Organizaţia Neutră a Informaticienilor v-au rugat să scrieţi un soft pentru a afla care este cea mai mare colonie de pe o planetă dată.

Ce este mai neobişnuit la ultimele planete descoperite este forma lor: sunt *cubice*.

Fiecare faţă a cubului este împărţită în NxN (N≤100) pătrate de arie egală. Pentru fiecare pătrat se cunoaşte baza interstelară de care aparţine reprezentată de un număr întreg între 1 şi 100. O colonie este considerată totalitatea pătratelor învecinate care aparţin aceleiaşi baze interstelare (o bază interstelară poate deţine mai multe colonii). Două pătrate sunt invecinate dacă au *exact* o muche comună.



Datele de intrare se vor citi din fişierul '**CUB.IN**' cu structura:
- pe prima linie n
- pe următoarele n linii, câte n valori întregi pe linie separate printr-un spaţiu, reprezentând bazele interstelare ce deţin pătratul i,j de pe faţa 1 a cubului (sistemul de coordonate şi sensurile de parcurgere a feţelor sunt cele din figură)
- analog, câte n linii pentru celelalte feţe ale cubului, în ordinea numerotării din figură

Datele de ieşire se vor scrie în fişierul '**CUB.OUT**' cu structura:
- prima linie va conţine două valori separate printr-un spaţiu reprezentând baza interstelară ce deţine cea mai mare colonie pe planetă şi respectiv numărul de pătrate ocupate de colonie. Dacă există două colonii cu aceeaşi dimensiune maximă atunci se va alege cea cu numărul de ordine mai mic.

Exemplu:
CUB.IN:
2
1 1
1 1
2 2
2 2
3 3
3 3

2 2
7 2
1 1
6 6
5 5
5 5

CUB.OUT:
2 7

Timp de execuţie: o secundă pe test.

Etapa

Punctaj: 50 puncte

Intr-o etapa in Europa se joaca n meciuri de fotbal (n<=100). Stiind rezultatele a n meciuri si doua numere x si y (0<=x,y<=100) sa se aleaga k meciuri (k<=n) din cele n astfel incat suma golurilor inscrise de gazde in cele k meciuri alese sa fie x si suma golurilor inscrise de oaspeti in cele k meciuri alese sa fie y. In cazul in care nu exista solutie se va afisa numarul 0 in fisierul de iesire.

Obs: un meci nu poate fi ales de doua ori;
daca exista mai multe solutii se va furniza una singura;

Datele de intrare se citesc din fisierul "input.txt" astfel:
n –pe prima linie numarul de meciuri;
x_1 y_1 –pe urmatoarele n linii rezultatele celor n meciuri;
x_2 y_2 –x_i , y_i numere pozitive (0<=x_i,y_i<=100);
……. (x_i – goluri inscrise de gazde; y_i – oaspeti)
x_n y_n
x y -numerele x si y;

Datele de iesire se scriu in fisierul "output.txt" astfel:
k –pe prima linie k (numarul de meciuri alese);
j_1 –pe urmatoarele k linii numarul de ordine al meciurilor alese;
…
j_k

Exemplu:

Input.txt:	Output.txt:
3	2
2 1	1
4 2	3
5 3	
7 4	

Input.txt:	Output.txt:
1	0
1 1	
2 2	

Timp de executie: o secunda pe test.

Nota: ambele subiecte sunt obligatorii; timp de lucru 3 ore.

Olimpiada de Informatică - Faza pe Municipiul Bucureşti

Clasa a X-a

Titlul problemei : Câinele nefericit

Primarul unui mare oraş iniţiaza o campanie de eradicare a câinilor vagabonzi. Unul dintre ei, sperând să scape, se gândeşte să-şi strângă proviziile (oase, evident) în câteva ascunzători săpate special in oraş. În orice ascunzătoare, câinele poate depozita oricâte oase.

Atât ascunzătorile, cât si oasele, se află în parcurile oraşului, care sunt în număr de N (N număr natural mai mic decât 200). În oricare din parcuri există fie o ascunzătoare, fie un os. Între unele dintre parcurile oraşului există străzi drepte, ce îndeplinesc următoarele condiţii:

- există exact n-1 străzi ce conectează cele n parcuri (o stradă conectează doua parcuri);
- o stradă poate fi parcursă în ambele sensuri;
- din orice parc se poate ajunge în orice alt parc, într-un mod unic determinat;
- parcurgerea unei străzi dureaza o unitate de timp.

Iniţial, câinele se afla într-una din ascunzătorile din oraş. El se poate deplasa pe străzi între parcuri, iar când se hotărăşte să ridice un os (poate ridica un os doar din parcul în care se află, dacă există un os în acel parc), este obligat să-l ducă într-una din ascunzători. Timpul în care câinele apucă osul sau timpul în care îl depozitează în ascunzătoare este neglijat. Nu se cronometrează decât unităţile de timp când câinele parcurge străzi (transportând sau cautând oase).

O asociaţie de protecţie a animalelor vă roagă să-l ajutaţi pe câine să-şi strângă toate oasele în cât mai puţine unităţi de timp.

Programul va citi din fişierul INPUT .TXT n+2 linii, cu următoarea semnificaţie:

- pe prima linie: numărul N, reprezentând numărul parcurilor oraşului;
- pe a doua linie, un şir $C_1C_2...C_n$ format din N caractere, carcaterul C_i reprezentând dacă în parcul i se găseşte o ascunzătoare sau un os. Dacă în parcul i se află un os, caracterul C_i este 'o', iar dacă în parcul i se află o ascunzătoare, caracterul C_i este 'a'.
- pe a treia linie, parcul unde se află iniţial câinele. (un număr întreg mai mic sau egal cu N);
- pe următoarele n-1 linii, se găsesc perechi de numere naturale distincte,

sub forma i_k j_k (cele două elemente ale perechii, numere naturale mai mici sau egale cu N fiind separate printr-un spaţiu). Perechea i_k j_k semnifică existenţa unei străzi între parcurile i_k si j_k.

Soluţia găsita de programul vostru va fi scrisă în fişierul OUTPUT.TXT, sub forma:

-pe prima linie, numărul minim de unităţi de timp necesare pentru depozitarea oaselor;

-pe urmatoarea linie, traseul parcurs de câine (indicii parcurilor în ordinea în care sunt vizitate de câine, fiecare indice fiind urmat de un caracter ce indica acţiunea efectuată de câine în parcul respectiv ('r' când câinele ridică un os; 'd' când câinele depozitează un os; 't' când câinele doar trece prin parcul respectiv)); Traseul începe cu parcul iniţial urmat de caracterul ' '(spaţiu), ca în exemplul următor.

Exemplu:
INPUT.TXT
5
oaoao
2
1 4
2 4
4 3
3 5

OUTPUT.TXT:
9
2 4t1r4d3r4d3t5r3t4d

Timp de execuţie : 3s/test.

Olimpiada de Informatica 2001
Clasa a XI-a si a XII-a - faza pe municipiu

Problema 1
Jocul de unul singur(50 puncte)

Cand Mars este cu adevarat plictisit , si simte nevoia de ceva nou si interesant , ia un pachet de carti special si se joaca . Prin ce e acest pachet special ? Prin faptul ca are numai carti negre si rosii , si anume N carti negre si R carti rosii (N si R cunoscute) .

Jocul consta in faptul ca dupa ce amesteca foarte bine cartile (adica orice configuratie are sanse egale sa apara) incepe si extrage o carte . Fara sa se uite la ea incearca sa ghiceasca ce carte este (rosie sau neagra) , dupa care se uita la ea . Daca a ghicit-o, o pune de-o parte si extrage alta , continuand procedeul pana nu mai nimereste sau se termina pachetul . Daca nu a ghicit-o jocul s-a terminat si isi numara cate carti a reusit sa ghiceasca (fara ultima, pe care n-ai ghicit-o) .

Dar Mars s-a plictisit sa joace la intamplare si si-a facut o strategie . El numara ce carti au iesit si astfel stie ce carti au ramas in pachet. Cand se pune problema sa ghiceasca, el alege culoarea prezenta cel mai des pe cartile ramase in pachet , iar in caz de egalitate alege rosu.

Jucand asa des, i-a venit in minte o intrebare . Care este numarul mediu de carti pe care reuseste sa le ghiceasca la un joc ? Intr-adevar el a jucat atat de mult incat si-a dat seama de rezultat , dar nu stie daca voi stiti ...

In fisierul de intrare JOC.IN se gasesc doua numere separate printr-un spatiu (N respectiv R) . Iar voi trebuie sa scrieti in fisierul JOC.OUT pe un singur rand un numar cu 4 zecimale exacte reprezentand numarul mediu de carti extrase la un joc .

Exemplu :
JOC.IN
1 1

JOC.OUT
1.0000

Explicatii :
Avem un pachet cu o carte rosie si una neagra . Mars zice rosu si are sanse 50% sa nimereasca . Daca nu a ghicit, are 0 carti ghicite, iar daca a nimerit sigur o va nimeri si pe a doua pentru ca stie ce carte a ramas si va avea 2 carti ghicite . Astfel numarul mediu este 1.

Observatii :
- 0<=N,R<=10000
- Daca N si R sunt 0 numarul de carti ghicite este 0
- Se recomanda folosirea tipului de date double pentru a evita pierderi de precizie

Timp de executie 1 secunda pe un 486 . Se asigura 200K memorie disponibila .

Problema 2
CUIE (50 puncte)

Pe o masa din lemn (considerata infinita) sunt amplasate N placi dreptunghiulare identice, de laturi L, respectiv H. Placile sunt plasate paralel cu axele, in pozitii de coordonate intregi, si se pot suprapune (partial si/sau total).

Sa se bata cuie in masa, astfel incat:

- fiecare cui sa intepe cel putin o placa;
- fiecare placa sa fie intepata de EXACT un cui.

Se considera ca un cui batut exact pe o margine a unei placi NU INTEAPA placa respectiva. Coordonatele cuielor pot fi intregi sau reale. L si H sunt numere naturale nenule, N<=1500.

Fisierul de intrare CUIE.IN are urmatoarea structura:

```
L H          - dimensiunile placilor
N            - numarul de placi
x1 y1
x2 y2
...
xN yN
```

Pozitia fiecarei placi K este specificata prin xK si yK. Placa este un dreptunghi delimitat de dreptele: x=xK, x=xK+L, y=yK, y=yK+H. Un cui de coordonate (XC,YC) inteapa placa K daca

xK < XC < xK+L si
yK < YC < yK+H

In fisierul de iesire CUIE.OUT se vor afisa coordonatele cuielor, cate un cui pe fiecare linie. Nu se impune o limita pentru numarul de cuie, dar solutia trebuie sa respecte restrictiile problemei.

Exemplu:

CUIE.IN
```
4 3
6
9 6
1 1
3 3
8 1
9 6
10 2
```

CUIE.OUT (exista si alte variante!!!)
```
2 2
6 5
11 3
10 8.2
```

Timp de executie: o secunda/test.