PROBLEME propuse pentru REZOLVARE

Vă prezentăm enunțurile problemelor de la faza locală a Olimpiadei de Infromatică din județul Cluj. Au fost în total șase probleme, două pentru clasa a IX-a, două pentru a X-a și două pentru clasele a XI-a și a XII-a.

P030101: Tăierea cozii

Prin ***tăierea cozii*** se înțelege operația de transformare aplicată unui șir de caractere care constă în eliminarea ultimelor *k* elemente ale șirului (cu *k* mai mic sau egal cu jumătatea lungimii șirului). Tăierea cozii se poate face numai dacă primele *k* elemente sunt identice cu ultimele *k*, citite de la sfârșit la început.

Se citește de la tastatură un șir de lungime *n* ($1 < n < 256$), cu elemente din mulțimea `{0,1}`. Șirului i se pot aplica mai multe transformări succesive de tăiere a cozii.

Să se determine configurația finală minimă la care se poate ajunge după transformările posibile aplicate șirului, pentru *k* variind de la 1 la jumătatea lui *n*.

Exemple

Dacă șirul inițial este `1101011` (deci *k* ia valori între `1` și `3`), se obțin următoarele configurații minime:
- pentru *k* = 1 se obține `11010` prin două tăieri succesive.
- pentru *k* = 2 se obține `11010` printr-o singură tăiere.
- pentru *k* = 3 se obține `1101` printr-o singură tăiere.

Deci, în acest caz, configurația minimă obținută este `1101`.

Dacă șirul inițial este `1111` (deci *k* ia valorile `1` sau `2`), se obțin următoarele configurații minime:
- pentru *k* = 1 se obține `1`.
- pentru *k* = 2 se obține `11`.

Deci configurația minimă este `1`.

Dacă șirul inițial este `11011011` (deci *k* ia valori între `1` și `4`), se obțin următoarele configurații minime:
- pentru *k* = 1 se obține `110110` prin două tăieri succesive.
- pentru *k* = 2 se obține `110110` printr-o singură tăiere.
- pentru *k* = 3 se obține `11011` printr-o singură tăiere.
- pentru *k* = 4 se obține `1101` printr-o singură tăiere.

Deci, în acest caz, configurația minimă obținută este `1101`.

Notă

Citirea și tipărirea datelor se vor efectua exact ca în modelul de mai jos:

```
Date de intrare:
1101011
Date de iesire:
1101
```

P030102: Numere prime

Să se scrie un program care citește un număr natural *n* ($0 < n < 1.000.000.000$) corect introdus de la tastatură și tipărește toate numerele prime care se pot forma din secvențe de cifre alăturate, în ordinea apariției în număr, ale numărului dat. Într-o secvență poate intra orice număr de cifre, de la o cifră la toate. Fiecare număr va fi tipărit o singură dată, chiar dacă el se repetă ca secvență a numărului inițial.

Observație

`0` și `1` nu sunt considerate a fi numere prime.

Notă

Citirea și tipărirea datelor se vor efectua exact ca în modelele de mai jos:

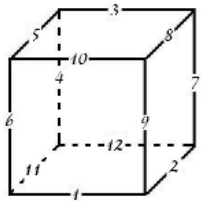
```
Date de intrare:
N=233
Date de iesire:
2 3 23 233

Date de intrare:
N=480
Date de iesire:
Nu exista solutii

Date de intrare:
N=2070
Date de iesire:
2 7
```

P030103: Cubul numerotat

Pe muchiile unui cub trebuie "plasate" numerele 1, 2, 3, ..., 12, astfel încât suma numerelor de pe cele patru muchii ale oricăror fețe să fie constantă. De asemenea, muchiile trebuie colorate cu un număr minim de culori astfel încât oricare două muchii incidente (care au un punct comun) să nu aibă aceeași culoare.

Exemplu

O soluție posibilă este



Rezultatele se vor afișa într-un fișier text **DATE.OUT** care va conține soluțiile numerotate astfel:

- pe prima linie numărul minim de culori folosite;
- pe următoarele linii numerele și culorile asociate muchiilor fiecărei fețe, sub forma (*număr - culoare*);
- fețele sunt numerotate așa cum se poate vedea mai jos.

Pentru exemplul considerat, o soluție va avea forma:

DATE.OUT

```
Nr. culori = 3
Sol. nr. 1
Fata 1.:(1 - 1) (6 - 2) (10 - 1) (9 - 3)
Fata 2.:(2 - 2) (7 - 3) (8 - 1) (9 - 3)
Fata 3.:(3 - 1) (4 - 2) (12 - 1) (7 - 3)
Fata 4.:(4 - 2) (5 - 3) (6 - 1) (11 - 3)
Fata 5.:(3 - 1) (5 - 3) (10 - 1) (8 - 2)
Fata 6.:(1 - 1) (2 - 2) (12 - 1) (11 - 3)
```

P030104: Eliminare palindroame

Se dau *n* numere naturale formate din oricâte cifre; acestea sunt conținute în fișierul text **DATE.IN**. Se cere curățarea șirului de elementele palindroame. Mai întâi se elimină numerele palindroame; apoi se concatenează două câte două numerele rămase. Se repetă procedeul (eliminare și concatenare) până când nu mai există numere palindroame.

Exemplu

Conținutul fișierului **DATE.IN** este următorul:

```
1 15 51 1 7 23 32 14 29 17
```

După eliminarea numerelor palindroame șirul devine `15 51 23 32 14 29 17`. După concatenare șirul este `1551 2332 1429 17`. După a doua eliminare șirul devine `1429 17`, iar după concatenare `142917`.

Fișierul de ieșire **DATA.OUT** va conține configurația finală a șirului. În cazul nostru, în final, șirul va conține un singur număr și anume `142917`.

P030105: Băcănie

Băcanul din orașul dumneavoastră încearcă o nouă metodă de afișare a prețurilor produselor pe care le are în stoc. În loc de a marca, pentru fiecare articol, prețul lui (în lei), etichetele băcanului au prețuri comparative față de alte produse. De exemplu, untul poate fi etichetat cu `unt=margarina+100`, iar margarina cu `margarina=cafea-111`. Unele articole sunt marcate chiar cu prețul corespunzător.

Datele de intrare se află în fișierul **BACANIE.IN** și sunt organizate astfel:

- toate articolele au denumiri formate din cel mult 10 litere;
- există cel puțin unul și cel mult 100 de articole;
- liniile de intrare indică fie direct prețul unui articol, fie indică faptul că acest preț este egal cu prețul unui alt produs `+/-` o sumă (în lei);
- toate prețurile care apar în fișier sunt numere naturale mai mici decât `1000`;
- toate liniile sunt corecte din punct de vedere sintactic;
- fiecare articol apare o singură dată în stânga semnului `'='`;
- articolele se află câte unul pe o linie din fișier;
- în fișierul de intrare nu există spații.

Fișierul de ieșire **BACANIE.OUT** va conține, pentru fiecare articol, câte o linie de forma `articol=preț`. Dacă prețul unui produs nu poate fi dedus pe baza datelor din fișierul de intrare, linia corespunzătoare va avea forma `articol=blank`. Ordinea articolelor din fișierul de ieșire nu are importanță.

Exemplu

BACANIE.IN

```
lapte=zahar-125
faina=225
zahar=faina+10
cafea=ceai
```

BACANIE.OUT

```
cafea=blank
faina=225
lapte=110
zahar=235
ceai=blank
```

P030106: Robot

Se dă fișierul de intrare **ROBOT.IN** (corect construit) în care:

- pe prima linie se află două numere naturale m și n ($0 < m, n < 10$).
- pe fiecare dintre următoarele m linii se află câte un șir de n numere din mulțimea `{0,1}`, despărțite prin câte un spațiu.
- pe linia următoare se află trei valori x, y, z, despărțite prin câte un spațiu ($0 < x \le m$, $0 < y \le n$, $z \in$ `{'N','S','E','V'}`).

Interpretarea valorilor este următoarea:

- m: numărul de linii ale unei matrice care codifică un labirint;
- n: numărul de coloane al matricei care codifică labirintul;
- valoare `1`: culoar (robotul poate ocupa această poziție);

- valoare `0`: zid (robotul nu poate ocupa această poziție);
- *x*: linia pe care se află inițial robotul;
- *y*: coloane pe care se află inițial robotul;
- *z*: - valoarea `'N'`: inițial robotul privește spre prima linie a matricei;
  - valoarea `'S'`: inițial robotul privește spre ultima linie a matricei;
  - valoarea `'E'`: inițial robotul privește spre ultima coloană a matricei;
  - valoarea `'S'`: inițial robotul privește spre prima coloană a matricei.

  În plus, (*x*, *y*) este o poziție în care se află un culoar.

Robotul are la dispoziție doar următoarele mișcări:

- `MOVE` - pentru deplasarea cu o poziție pe sensul de mers (mutarea este posibilă dacă în poziția în care se ajunge nu este zid);
- `ROTATE` - pentru rotirea cu `90°` în direcția acelor de ceasornic (robotul rămâne pe aceeași poziție dar își schimbă direcția de mers; mutarea este întotdeauna posibilă).

Programele trebuie să tipărească în fișierul **`ROBOT.OUT`** cea mai scurtă succesiune de mișcări pe care trebuie să le execute robotul pentru a ieși din labirint (pentru a ajunge pe prima sau ultima linie, sau pe prima sau ultima coloană). Dacă există mai multe secvențe egale, se va tipări una singură. Fiecare mișcare va fi scrisă pe câte o linie. Ultima linie va conține mesajul `Robotul a iesit din labirint in k miscari`, unde *k* reprezintă numărul mișcărilor. Dacă robotul nu poate ieși din labirint, în fișierul de ieșire se va scrie doar mesajul `Nu exista solutie`.

Exemple

Să considerăm următorul fișier de intrare:

`ROBOT.IN`
```
4 4
0 0 0 0
1 1 1 1
1 1 1 0
3 3 N
```

O soluție corectă este următoarea:

`ROBOT.OUT`
```
MOVE
ROTATE
MOVE
Robotul a iesit din labirint in 3 miscari
```

O altă soluție corectă este:

`ROBOT.OUT`
```
ROTATE
ROTATE
MOVE
Robotul a iesit din labirint in 3 miscari
```

Să considerăm acum următorul fișier:

`ROBOT.IN`
```
1 1
1
1 1 W
```

În acest caz singura soluție corectă este:

`ROBOT.OUT`
```
Robotul a iesit din labirint in 0 miscari
```

Iată și un exemplu pentru care nu există soluție:

`ROBOT.IN`
```
3 3
0 0 0
0 1 0
0 0 0
2 2 S
```

`ROBOT.OUT`
```
Nu exista solutie
```

Știați că...

- O mulțime de concurenți intră în criză de timp deoarece se prezintă la concurs fără a avea la ei un ceas.
- Multe programe nu mai funcționează corect datorită unor optimizări efectuate în ultimul moment și ale căror efecte nu au mai fost testate.
- Mulți concurenți încearcă să rezolve la început problema cea mai dificilă și apoi nu mai au timp să lucreze și la celelalte.
- Cazul cel mai defavorabil nu corespunde întotdeauna dimensiunilor maxime ale datelor de intrare.
- Multe programe funcționează corect pentru majoritatea datelor de intrare posibile, dar furnizează erori neașteptate pentru combinații particulare cum ar fi dimensiunile minime ale datelor de intrare.
- Cea mai enervantă eroare care poate apărea într-un program este scrierea incorectă a numelui fișierului de intrare sau a celui de ieșire.
- O rezolvare nu trebuie să fie neapărat corectă; ea trebuie să furnizeze rezultatul corect doar pentru seturile de date folosite de comisie la evaluare.