Bursele AGORA

Vă prezentăm în continuare enunțurile problemelor propuse spre rezolvare la ultimele patru runde ale celei de-a patra ediții a concursului de programare Bursele Agora.

P040315: Heroes of Might & Magic

Magicianul *Hexis* controlează un număr de *N* castele ale căror coordonate se cunosc. Nu există trei castele ale căror coordonate sunt puncte coliniare. Pentru a se proteja de iminentul atac al adversarilor săi, *Hexis* decide să trimită câte un erou în fiecare dintre orașele aflate pe perimetrul teritoriului. Perimetrul celor *N* castele este poligonul de perimetru minim care conține în interior sau în vârfuri toate castelele. Datorită faptului că *Hexis* știe că, mai devreme sau mai târziu, acest perimetru va fi străpuns, el organizează apărarea în continuare astfel:

- castelele aflate pe perimetru (în vârfurile poligonului) sunt considerate a fi pierdute;
- pentru castelele rămase se determină un nou perimetru și în castelele aflate pe noul perimetru sunt trimiși câte doi eroi;
- evident, și acest perimetru va fi străpuns, deci este organizat un al treilea nivel de apărare care va fi dat de perimetrul castelelor rămase; în castelele de pe acest perimetru vor fi trimiși câte trei eroi;
- procedeul va continua (numărul eroilor crește cu 1 pentru fiecare nou perimetru) până în momentul în care nu va mai rămâne nici un castel.

Va trebui să determinați, pentru fiecare castel în parte, numărul eroilor care vor fi trimiși în castelul respectiv.

Date de intrare

Prima linie a fișierului de intrare **HOMM.IN** conține numărul *N* al castelelor de pe teritoriul lui *Hexis*. Fiecare dintre următoarele *N* linii va conține o pereche de numere reale reprezentând coordonatele unui castel, separate prin câte un spațiu.

Date de ieșire

Fișierul de ieșire **HOMM.OUT** trebuie să conțină *N* linii; fiecare linie corespunde unui castel. O astfel de linie va conține un singur număr care reprezintă numărul eroilor trimiși în castelul respectiv.

Ordinea castelelor din fișierul de ieșire trebuie să respecte ordinea acestora din fișierul de intrare.

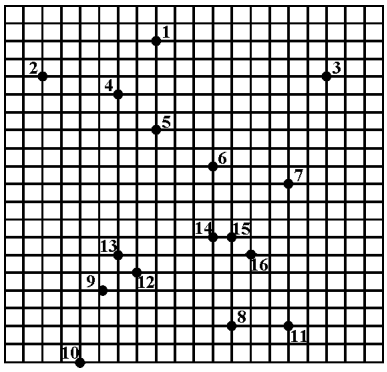
Restricții și precizări

- $1 \le N \le 1000$;
- coordonatele castelelor sunt numere reale cu cel mult trei zecimale exacte;
- ultimul perimetru poate conține unul sau două castele.

Exemplu

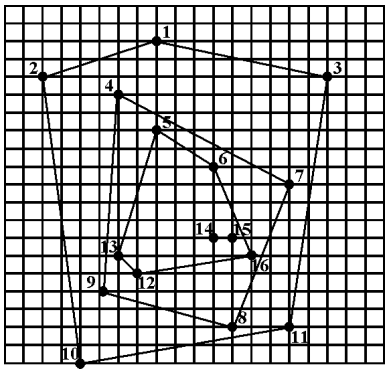
HOMM.IN

```
16
8 2
2 4
17 4
6 5
8 7
11 9
15 10
12 18
5.1 16
4 20
15 18
7 15
6 14
11 13
12 13
13 14
```



HOMM.OUT

```
1
1
1
2
3
3
2
2
2
1
1
3
3
4
4
3
```



Timp de execuție: 1 secundă/test

P040316: Fotbal

În *Liga de Fotbal Agora* participă N echipe, identificate prin numere cuprinse între 1 și N. Fiecare echipă joacă exact o partidă cu toate celelalte N - 1 echipe. Aveți la dispoziție un clasament final și va trebui să stabiliți rezultatele fiecărei partide, astfel încât să obțineți un clasament cât mai apropiat de cel dat. Pentru fiecare echipă se cunoaște numărul total al victoriilor, numărul total al partidelor terminate la egalitate și numărul total al înfrângerilor. Suma acestor trei numere va fi, pentru fiecare echipă, N - 1. De asemenea, pentru fiecare echipă se cunoaște numărul total al golurilor înscrise și numărul total al golurilor primite. Numărul total al golurilor înscrise de cele N echipe va fi egal cu numărul total al golurilor primite de cele N echipe.

Date de intrare

Prima linie a fișierului de intrare **FOOTBALL.IN** conține numărul N al echipelor participante. Fiecare dintre următoarele N linii conține câte cinci numere, separate prin câte un spațiu, reprezentând numărul victoriilor, al partidelor terminate la egalitate, al înfrângerilor, al golurilor marcate, respectiv al golurilor primite pentru o anumită echipă.

Date de ieșire

Datele de ieșire vor fi furnizate sub forma unei matrice pătratice de dimensiune N. Pentru fiecare pereche (i, j), $i < j$ elementul a_{ij} va conține numărul golurilor marcate de echipa i în partida dintre echipele i și j, iar elementul a_{ji} va conține numărul golurilor marcate de echipa j în partida dintre echipele i și j. Elementele de forma a_{ii} vor avea valoarea 0.

Fișierul de ieșire **FOOTBALL.OUT** va conține N linii pe care se vor afla câte N numere separate prin câte un spațiu. Aceste numere vor reprezenta elementele matricei descrise.

Restricții și precizări

- $2 \le N \le 1000$;
- ordinea echipelor din fișierul de ieșire trebuie să respecte ordinea din fișierul de intrare;
- într-o partidă o echipă poate marca cel mult zece goluri;
- întotdeauna va putea fi obținut un clasament identic cu cel dat.

Exemplu

FOOTBALL.IN

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

FOOTBALL.OUT

```
0 1 2 1
1 0 1 3
4 1 0 2
0 1 1 0
```

Modalitatea de acordare a punctajului

Vom considera că, pentru fiecare test, se vor putea obține cel mult X puncte. Pentru datele de ieșire furnizate de concurenți se va construi un clasament care va fi comparat cu cel dat. Pentru fiecare echipă se vor putea obține cel mult $P = X / N$ puncte.

Pentru o echipă, dacă numărul W' al victoriilor din clasamentul obținut este egal cu cel din clasamentul dat (W), se vor acorda $0.2 \cdot P$ puncte. În caz contrar, se vor acorda $\max(0, (1 - |W - W'| / (W + 1)) \cdot 0.2 \cdot P)$ puncte.

Dacă numărul D' al partidelor egale din clasamentul obținut este egal cu cel din clasamentul dat (D), se vor acorda $0.1 \cdot P$ puncte. În caz contrar, se vor acorda $\max(0, (1 - |D - D'| / (D + 1)) \cdot 0.1 \cdot P)$ puncte.

Dacă numărul L' al partidelor egale din clasamentul obținut este egal cu cel din clasamentul dat (L), se vor acorda $0.1 \cdot P$ puncte. În caz contrar, se vor acorda $\max(0, (1 - |L - L'| / (L + 1)) \cdot 0.1 \cdot P)$ puncte.

Dacă numărul S' al golurilor marcate din clasamentul obținut este egal cu cel din clasamentul dat (S), se vor acorda $0.3 \cdot P$ puncte. În caz contrar, se vor acorda $\max(0, (1 - |S - S'| / (S + 1)) \cdot 0.3 \cdot P)$ puncte.

Dacă numărul R' al golurilor primite din clasamentul obținut este egal cu cel din clasamentul dat (R), se vor acorda $0.3 \cdot P$ puncte. În caz contrar, se vor acorda $\max(0, (1 - |R - R'| / (R + 1)) \cdot 0.3 \cdot P)$ puncte.

Punctajul pentru un test va fi trunchiat la două zecimale exacte. Punctajul final va fi obținut prin adunarea punctajelor de la fiecare test și rotunjirea acestuia la cel mai apropiat număr întreg.

Să presupunem că, pentru exemplul dat, un concurent furnizează următorul fișier de ieșire:

FOOTBALL.OUT

```
0 1 1 1
1 0 1 1
1 1 0 1
1 1 1 0
```

Clasamentul obținut pe baza acestui fișier este:

```
0 3 0 3 3
0 3 0 3 3
0 3 0 3 3
0 3 0 3 3
```

Presupunem că pentru acest test pot fi obținute cel mult $X = 5$ puncte. Datorită faptului că avem patru echipe, pentru fiecare echipă vor fi acordate cel mult $P = 5 / 4 = 1.25$ puncte.

Pentru prima echipă avem:

- `W = 1,W' = 0: ((1-|1-0|/(1+1))*0.2*1.25` = **0.125** puncte
- `D = 1, D' = 3: ((1-|1-3|/(1+1))*0.1*1.25` = **0** puncte

- L = 1, L' = 0: ((1-|1-0|/(1+1))*0.1*1.25 = **0.0625** puncte
- S = 4, S' = 3: ((1-|4-3|/(4+1))*0.3*1.25 = **0.3** puncte
- R = 5, R' = 3: ((1-|5-3|/(5+1))*0.3*1.25 = **0.25** puncte.

Așadar, pentru prima echipă se vor acorda 0.125 + 0 + 0.0625 + 0.3 + 0.25 = **0.7375** puncte.

Pentru a doua echipă avem:

- W = 1, W' = 0: ((1-|1-0|/(1+1))*0.2*1.25 = **0.125** puncte
- D = 2, D' = 3: ((1-|2-3|/(2+1))*0.1*1.25 = **0.08333** puncte
- L = 0, L' = 0: 0.1*1.25 = **0.125** puncte
- S = 5, S' = 3: ((1-|5-3|/(5+1))*0.3*1.25 = **0.25** puncte
- R = 3, R' = 3: 0.3*1.25 = **0.375** puncte.

Așadar, pentru a doua echipă se vor acorda 0.125 + 0.08333 + 0.125 + 0.25 + 0.375 = **0.95833** puncte.

Pentru a treia echipă avem:

- W = 2, W' = 0: ((1-|2-0|/(2+1))*0.2*1.25 = **0.08333** puncte
- D = 1, D' = 3: ((1-|1-3|/(1+1))*0.1*1.25 = **0** puncte
- L = 0, L' = 0: 0.1*1.25 = **0.125** puncte
- S = 7, S' = 3: ((1-|7-3|/(7+1))*0.3*1.25 = **0.1875** puncte
- R = 4, R' = 3: ((1-|4-3|/(4+1))*0.3*1.25 = **0.3** puncte.

Așadar, pentru a treia echipă se vor acorda 0.08333 + 0 + 0.125 + 0.1875 + 0.3 = **0.69583** puncte.

Pentru a patra echipă avem:

- W = 0, W' = 0: 0.2*1.25 = **0.25** puncte.
- D = 0, D' = 3: ((1-|0-3|/(0+1))*0.1*1.25 < **0** puncte
- L = 3, L' = 0: ((1-|3-0|/(3+1))*0.1*1.25 = **0.03125** puncte
- S = 2, S' = 3: ((1-|2-3|/(2+1))*0.3*1.25 = **0.25** puncte
- R = 6, R' = 3: ((1-|6-3|/(6+1))*0.3*1.25 = **0.21429** puncte.

Așadar, pentru a patra echipă se vor acorda 0.25 + 0 + 0.03125 + 0.25 + 0.21429 = **0.74554** puncte.

În concluzie, pentru acest test se vor acorda 0.7375 + 0.95833 + 0.69583 + 0.74554 = **3.13** puncte (valoare obținută după trunchiere).

Timp de execuție: 1 secundă/test

P040317: Atlantis

Exploratorul *Iahim Uratrox* face o nouă vizită în capitala mitică a *Atlantidei*. Aici descoperă o piramidă ciudată a cărei intrare este, din nefericire, protejată printr-un nou tip de mecanism.

Datorită experienței acumulate, el observă deja că în fața piramidei se află mai multe comutatoare. Lângă fiecare comutator se află câte o plăcuță pe care este înscris un semn reprezentând un număr natural strict pozitiv. Pe o plăcuță mai mare este înscris un alt număr.

Iahim Uratrox a furnizat aceste informații cercetătorilor care au început să studieze secretul acestui nou tip de mecanism creat de atlanți cu mii de ani în urmă.

Cercetătorii au descoperit destul de repede (sunt și ei mai experimentați) că, pentru a debloca intrarea, trebuie activate mai multe comutatoare (cel puțin două), astfel încât suma numerelor de pe plăcuțele corespunzătoare acestora să fie egală cu valoarea înscrisă pe plăcuța mai mare.

Cu toate acestea, s-a dovedit că nu oricare astfel de alegere duce la deschiderea intrării în piramidă. Nu s-au descoperit alte detalii referitoare la mecanism, motiv pentru care *Iahim Uratrox* dorește să știe câte posibilități de a alege comutatoarele există pentru a ști câte șanse are să determine configurația, despre care se crede că este unică, ce va deschide intrarea.

Date de intrare

Fișierele de intrare sunt deschise și pot fi descărcate folosind *link*-urile de pe pagina dedicată problemei; adresa acesteia este http://www.ginfo.ro/concurs/runda19/index.shtml. Numele fișierelor au forma **ATLANTISXX.IN**, unde **XX** ia valori între **01** și **20** și reprezintă numărul testului.

Fiecare dintre cele 20 de fișiere de intrare conține pe prima linie numărul *N* al plăcuțelor "mici" și valoarea *S* inscripționată pe plăcuța mai mare.

Următoarele *N* linii vor conține numerele inscripționate pe cele *N* plăcuțe.

Date de ieșire

Pentru această problemă nu va trebui să scrieți un program, ci să generați doar cele 20 de fișiere de ieșire corecte. Acestea vor fi denumite **ATLANTISXX.OUT**, unde **XX** ia valori între 01 și 20 și reprezintă numărul testului.

Prima linie a unuia dintre fișierele de ieșire va conține textul ATLANTIS - TEST #XX, unde XX reprezintă numărul testului.

Cea de-a doua linie va conține un singur număr care va indica numărul posibilităților de a alege cel puțin două comutatoare astfel încât suma numerelor de pe plăcuțele corespunzătoare acestor comutatoare să fie egală cu valoarea de pe plăcuța mai mare.

Observație

Datorită faptului că denumirile fișierelor de ieșire trebuie să conțină 12 caractere, aceste fișiere nu vor putea fi create folosind sistemul de operare *MS-DOS*.

Exemplu
Vom considera că acest exemplu reprezintă testul **00**.

ATLANTIS00.IN
```
5 5
1
2
3
4
5
```

ATLANTIS00.OUT
```
2
```

P040318: Maxime
Se consideră un șir *a* cu *N* elemente ale căror valori sunt numere întregi distincte care nu sunt cunoscute.

Va trebui să determinați indicii celor mai mari trei numere din acest șir. Pentru determinarea acestora, puteți efectua un număr limitat de comparări între două elemente ale șirului *a*.

Pentru aceasta, aveți la dispoziție o bibliotecă externă, care vă pune la dispoziție rutine pentru:
- inițializare;
- determinarea numărului de elemente ale șirului;
- determinarea numărului maxim de comparări pe care le aveți la dispoziție;
- compararea a două elemente ale șirului *a*;
- furnizarea ca rezultat a celor trei numere;
- încheierea execuției programului.

Bibliotecile Pascal și C/C++
Pentru a putea folosi biblioteca, va trebui să includeți în programul dumneavoastră istrucțiunea **uses** `max`; pentru limbajul *Pascal* și `#include "max.h"` pentru limbajul *C/C++*.

În continuare sunt prezentate sintaxele funcțiilor și procedurilor incluse în biblioteci:

procedure `Init;`
void `Init()`
- trebuie apelată la începutul programului și este folosită de către bibliotecă pentru inițializare;
- întrerupe execuția programului dacă este apelată a doua oară.

function `GetN:Integer;`
int `GetN()`
- returnează numărul de elemente (*N*) ale șirului *a*.

function `GetNoComp:Integer;`
int `GetNoComp()`
- returnează numărul de comparări (apeluri ale funcției `IsGreater`) pe care le mai aveți la dispoziție (în momentul apelului funcției `GetNoComp`) pentru determinarea permutării *p*.

function `IsGreater(i,j:Integer):Boolean;`
int `IsGreater(int i,int j)`
- returnează `true` (valoare nenulă) dacă *a*[*i*] > a[*j*] și `false` (0) în caz contrar;
- întrerupe execuția programului dacă cel puțin una dintre valorile `i` și `j` nu este un număr întreg cuprins între 1 și *N* sau dacă numărul de apeluri ale acestei funcții depășește numărul maxim permis.

procedure `Max1(x:Integer);`
void `Max1(int x)`
- acceptă ca rezultat faptul că cel mai mare element al șirului se află pe poziția `x`;
- întrerupe execuția programului dacă este apelată a doua oară sau dacă valoarea `x` nu este cea corectă.

procedure `Max2(x:Integer);`
void `Max2(int x)`
- acceptă ca rezultat faptul că cel mai mare element al șirului obținut după eliminarea maximului se află pe poziția `x`;
- întrerupe execuția programului dacă este apelată a doua oară sau dacă valoarea `x` nu este cea corectă.

procedure `Max3(x:Integer);`
void `Max3(int x)`
- acceptă ca rezultat faptul că cel mai mare element al șirului obținut după eliminarea celor mai mari două numere se află pe poziția `x`;
- întrerupe execuția programului dacă este apelată a doua oară sau dacă valoarea `x` nu este cea corectă.

procedure `EndProgram;`
void `EndProgram(void)`
- întrerupe execuția programului.

Toate funcțiile și procedurile (excepție: `Init`) întrerup execuția programului dacă sunt apelate înaintea inițializării.

Restricții și precizări
- Programul vostru nu va citi date din nici un fișier de intrare și nu va scrie date în nici un fișier de ieșire.
- $3 \le N \le 255$.

Exemple de utilizare a bibliotecii

Subprogram apelat	*Valoare returnată*
`Init`	-
`GetN`	3
`GetNoComp`	3
`IsGreater(1,2)`	`true (1)`
`IsGreater(1,3)`	`false (0)`
`IsGreater(2,3)`	`false (0)`
`Max1(1)`	-
`Max2(3)`	-
`Max3(2)`	-
`EndProgram`	-

Timp de execuție: 1 secundă/test