



## Problema Biatlon

Fișier de intrare      biatlon.in  
Fișier de ieșire      biatlon.out

Fiind încântat de proba de biatlon de la jocurile olimpice de iarnă, Rareș s-a gândit să inventeze propriul joc. El a așezat în linie un număr de ținte, apoi și-a propus să le doboare folosind mingea. Mingea fiind mai mare, la o lovitură se doboară două ținte aflate una lângă alta. Dacă o țintă este izolată (fără alta vecină) ea poate fi doborâtă la o lovitură cu mingea.

Fiind inventiv, el și-a stabilit câteva reguli speciale de joc:

- Dacă o țintă este vecină cu o alta, aceasta nu poate fi doborâtă singură la lovitură următoare;
- El dorește să execute un număr maxim de lovituri până reușește să doboare toate țintele, respectând regulile jocului;
- Rareș este un foarte bun executant de lovituri cu mingea, așa că el reușește să lovească mereu ținta sau țintele propuse.

De exemplu, să presupunem că avem 7 ținte dispuse în linie, ca în desen (4 aflate una lângă alta, apoi un loc liber, apoi una izolată, apoi alt loc liber și apoi alte două aflate una lângă alta). Mai departe în enunț, vom considera că ele formează 3 **grupuri** (primul cu patru ținte, al doilea cu o țintă și ultimul cu două ținte). Conform regulilor, iată câteva exemple de situații: Rareș nu poate lovi inițial doar ținta de pe poziția 8 din desen pentru că aceasta mai are una vecină și va trebui să aleagă să le lovească doar împreună. Rareș poate lovi inițial doar ținta de pe poziția 6, pentru că aceasta nu mai are alta vecină. Rareș nu poate lovi inițial doar ținta de pe poziția 3 pentru că aceasta are ținte vecine. Dacă dorește să lovească inițial ținta de pe poziția 3, el o poate face doar alături de cea de pe poziția 2 sau de cea de pe poziția 4.



### Cerințe

Problema are două cerințe, în funcție de valoarea lui  $C \in \{1, 2\}$ .

- C=1.** Să se determine numărul **maxim** de lovituri pe care le poate executa pentru șirul de valori din fișierul de intrare.
- C=2.** Rareș poate să modifice configurația țintelor în felul următor: formează  $N - 1$  grupuri prin alipirea țintelor din ultimul grup la unul dintre celelalte grupuri. Dorește să formeze un astfel de set cu  $N - 1$  grupuri asupra căruia să poată executa un număr de lovituri strict mai mare decât cel maxim posibil pe setul inițial de  $N$  grupuri. Programul trebuie să determine grupurile la care poate fi alipit ultimul astfel încât să se îndeplinească dorința lui.

### Date de intrare

Fișierul biatlon.in conține pe prima linie o valoare  $C$  reprezentând cerința și o valoare  $N$  reprezentând numărul de **grupuri** de ținte (toate țintele dintr-un grup se află una lângă alta dar cu separator față de țintele din alte grupuri). Pe linia a doua se află, valorile naturale nenule ce reprezintă numărul de ținte aflate inițial în fiecare grup (în ordinea grupurilor de la primul la ultimul). Valorile de pe aceeași linie sunt separate printr-un spațiu.

### Date de ieșire

Pentru  $C = 1$  fișierul de ieșire biatlon.out va conține o valoare naturală ce reprezintă numărul maxim de lovituri pe care le poate executa Rareș conform regulilor jocului. Dacă  $C = 2$ , fișierul va conține indicii grupurilor determinate, scriși în ordine crescătoare, separați prin câte un spațiu. Dacă nu există niciun grup care să satisfacă proprietatea din cerință, în fișierul de ieșire se va scrie valoarea 0.



## Restricții și precizări

- $1 \leq N \leq 50\,000$ ;
- $1 \leq$  numărul de ținte aflate inițial într-un grup  $\leq 50\,000$ ;
- Grupurile sunt numerotate începând cu 1;
- Pentru  $C = 2$ , se garantează că  $N \geq 2$ .

| # | Punctaj | Restricții     |
|---|---------|----------------|
| 1 | 13      | $C = 1, N = 1$ |
| 2 | 60      | $C = 1, N > 1$ |
| 3 | 27      | $C = 2$        |

## Exemple

| biatlon.in     | biatlon.out |
|----------------|-------------|
| 1 3<br>4 1 2   | 5           |
| 2 3<br>4 1 2   | 0           |
| 2 4<br>5 3 5 5 | 1 3         |

Explicație pentru exemplul 1

|                                             | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---------------------------------------------|---|---|---|---|---|---|---|---|---|
| Inițial:                                    |   |   |   |   |   |   |   |   |   |
| Execută o lovitură la pozițiile 2 și 3      |   |   |   |   |   |   |   |   |   |
| Execută o lovitură la pozițiile 8 și 9      |   |   |   |   |   |   |   |   |   |
| Execută o lovitură la poziția 1, de exemplu |   |   |   |   |   |   |   |   |   |
| Execută o lovitură la poziția 6, de exemplu |   |   |   |   |   |   |   |   |   |
| Execută o lovitură la poziția 4             |   |   |   |   |   |   |   |   |   |

Ar fi putut să elimine țintele din primul grup și din două lovituri (una la pozițiile 1, 2 și cealaltă la pozițiile 3, 4) însă nu l-ar fi avantajat pentru că planul său este acela de a executa totuși un număr maxim de lovituri, iar pe strategia descrisă mai sus în tabelul care explică exemplul, se observă că poate face asta din 3 lovituri.

Explicație pentru exemplul 2

Numărul maxim de lovituri ce se pot aplica asupra setului dat, 4 1 2, este 5. Dacă alegem să unim ultimul grup cu primul vom avea setul 6 1 și numărul maxim de lovituri ce se pot executa asupra acestuia este tot 5. Dacă alegem să unim ultimul grup la al doilea vom avea setul 4 3 și numărul maxim de lovituri ce se pot executa asupra acestuia este tot 5. Deci nu avem nicio modalitate de unire a ultimului grup care să producă o valoare maximă strict mai mare decât cea pentru setul inițial.

Explicație pentru exemplul 3

Numărul maxim de lovituri ce se pot aplica asupra setului dat, 5 3 5 5, este 11. Dacă unim ultimul grup cu primul sau cu al treilea, obținem seturile 10 3 5 respectiv 5 3 10, asupra cărora se pot aplica 12 lovituri. Dacă am fi unit ultimul grup cu al doilea, am fi obținut setul 5 8 5 asupra căruia se puteau aplica maxim 11 lovituri.