1.1 - ACS Encyclopedia - Ușoară

ACS Encyclopedia este formată din **N** pagini numerotate de la 1 la **N**. Fiecare pagină **i** conține un link catre o altă pagină L_i.

Un drum de învățare (learning pathway) este reprezentat de alegerea unei pagini și accesarea succesivă a link-urilor până când utilizatorul dorește să se oprească.

Știind că puteți să vă alegeți orice pagină drept punct de pornire ([1, **N**]), determinați numărul maxim de pagini al unui drum de învățare.

Input Format

De pe prima linie se citeste un număr natural **N**, cu semnificația din enunț. Pe a doua linie se găsesc, separate prin spațiu, **N** numere naturale. Al **i**-lea număr semnifică faptul că pagina **i** are un link către pagina L_i.

Constraints

- $2 \leq N \leq 10^4$
- $1 \leq L_i \leq N$
- $L_i \neq i$, pentru oricare **i**
- Limită de timp C++: 1 secundă
- Limită de timp Java: 2 secunde

Output Format

Se va afișa la ieșirea standard un singur număr: numărul maxim de pagini al unui drum de învățare.

Sample Input 0

```
5
4 3 5 1 2
```

Sample Output 0

```
3
```

Explanation 0

Pornind de la pagina 3, se poate crea următorul drum de învățare: 3 -> 5 -> 2.

Sample Input 1

```
5
2 4 2 2 3
```

Sample Output 1

```
4
```

Contest ends in an hour

Submissions: 136
Max Score: 50
Difficulty: Easy

Rate This Challenge:

More

C++

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

1.2 - Bigger Product - Ușoară

Se dă un șir de **N** numere întregi strict pozitive. Găsiți cardinalul **minim** al unei submulțimi de elemente astfel încat produsul elementelor din acea submulțime sa fie **mai mare sau egal** decât produsul elementelor rămase. Întrucât se poate produce overflow atunci când se calculează produsul, veți compara **logaritmul produsului** din submulțimea de cardinal minim cu **logaritmul produsului** elementelor rămase, folosind formula $log(x * y) = log(x) + log(y)$.

Input Format

Se va citi de la intrarea standard. Pe prima linie se află un număr natural nenul **N**, numărul de numere din șir. Pe următoarea linie se află cele **N** numere întregi strict pozitive x_i din șir, separate prin câte un spațiu.

Constraints

- **N** ∈ [2, 150000]
- x_i ∈ [1, 10^6]
- Limită de timp C++: 2 secunde
- Limită de timp Java: 2 secunde

Output Format

Se va afișa la ieșirea standard o singură linie care conține cardinalul minim al unei submulțimi de elemente astfel încât produsul elementelor din acea submulțime să fie mai mare sau egal decât produsul elementelor rămase.

Sample Input 0

```
3
9 3 1
```

Sample Output 0

```
1
```

Explanation 0

Selectăm elementul 9, care este mai mare sau egal decât produsul rămas, care este 3.

Sample Input 1

```
6
6 3 2 4 3 9
```

Sample Output 1

3

Explanation 1

Selectăm elementele 6, 9 și 4 care au produsul 216, mai mare sau egal decât produsul elementelor rămase, care este 18.

Contest ends in an hour

Submissions: 191
Max Score: 50
Difficulty: Easy

Rate This Challenge:

More

C++

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}

```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

2.1 - Temele de la ACS - Medie

Un student de la ACS are ca și criteriu principal pentru a trece materia soluționarea unui număr X de teme obligatorii din cele N propuse (temele numerotate de la 1 la N). Totuși, dintre aceste teme unele au legătură cu celelalte (fac parte dintr-un proiect mai mare) - practic, pentru fiecare temă există o listă de teme ce sunt postate mai repede și trebuie rezolvate până a putea rezolva problema respectivă.

Pentru ca studentul să promoveze materia are nevoie de ajutorul vostru să poată găsi ordinea corectă de a rezolva temele astfel încat să rezolve toate temele de care are nevoie.

Observație: după ce studentul rezolvă o temă trece la cealaltă.

Input Format

Se va citi de la intrarea standard. Pe prima linie se află, separate prin spațiu, două numere naturale N si X cu semnificația din enunț. Pe a doua linie se află, separate prin spațiu, X numere ce reprezintă temele obligatorii. Pe următoarele N linii corespunzătoare fiecărei teme *i* avem: numărul de teme A ce trebuie rezolvate înainte de tema *i* și o listă de A teme (A poate să fie și zero, lista fiind vidă).

Constraints

- $1 \leq N, X \leq 10^5$
- Limită de timp C++: 1 secundă
- Limită de timp Java: 2 secunde

Output Format

În cazul în care se poate găsi o ordonare validă, se va afișa la ieșirea standard, pe prima linie numărul minim de teme ce trebuie soluționate, iar pe a doua linie se vor afișa temele, separate prin spațiu și **sortate crescător**. Dacă nu se poate găsi o ordine validă, se va afișa la ieșirea standard -1.

Sample Input 0

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

Sample Output 0

```
5
1 2 3 4 5
```

Explanation 0

Tema 3 se poate rezolva direct; rezolvarea temei 5 este condiționată de rezolvarea temei 4, care la rândul ei depinde de temele 2 și 1. Temele 2 și 1 se pot rezolva direct.

Sample Input 1

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

Sample Output 1

```
-1
```

Contest ends in an hour

Submissions: 120
Max Score: 50
Difficulty: Medium

Rate This Challenge:

More

C++

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

2.2 - Factory Network - Medie

Se consideră o rețea de N fabrici (numerotate de la 1 la N), dintre care unele produc materii prime (raw), iar altele au nevoie de materii prime pentru a produce materiale rafinate. Fabricile sunt legate prin M drumuri bidirecționale și putem desemna o fabrică să fie ori producator (P) de materiale raw, ori consumator (C) de materiale raw. Un producător poate alimenta oricâți consumatori. Trebuie sa existe cel putin un producator.

Se cere să marcăm fiecare fabrică ori ca producător, ori consumator de materiale raw, astfel încât să se respecte urmatoarele condiții:

- fiecare producător sa fie conectat la cel puțin un consumator;
- să nu existe 2 producători conectați direct (printr-o singură muchie) între ei;
- alocarea trebuie sa inceapa din nodul 1;
- numărul de producători să fie minim.

Se garantează că rețeaua dată inițial este conexă. Va trebui afișat numărul minim de producători, astfel încât toate cerințele anterioare să fie îndeplinite.

Input Format

De pe prima linie de la intrare se citesc două numere naturale separate prin spațiu, N si M, semnificând numărul de fabrici, respectiv drumuri bidirecționale.

Pe următoarele M linii se găsesc câte 2 numere naturale x_i si y_i, semnificând câte un drum bidirecțional, care conectează fabrica x_i cu y_i.

Constraints

- $2 \le N \le 5,000$
- $1 \le M \le 20,000$
- $1 \le x_i, y_i \le N$, oricare i
- $x_i \ne y_i$, oricare i
- Limită de timp C++: 2 secunde
- Limită de timp Java: 4 secunde

Output Format

Se va afișa la ieșirea standard un singur număr natural: numărul minim de producători astfel încat toate cele patru cerințe să fie respectate.

Sample Input 0

```
5 4
1 2
2 3
1 5
4 5
```

Sample Output 0

```
2
```

Explanation 0

Alocând fabricile în următorul mod, se observă că sunt respectate toate cele patru condiții: 1 - Consumator, 2 - Producător, 3 - Consumator, 4 - Consumator, 5 - Producător. În total sunt 2 producători.

Sample Input 1

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

Sample Output 1

```
2
```

Contest ends in an hour

Submissions: 19
Max Score: 50
Difficulty: Medium

Rate This Challenge:

More

C++

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}

```

Line: 1 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code

3.1 - Minimum Work Time - Grea

Se consideră N liste de sarcini de lucru. Fiecare listă totalizează un număr de ore de lucru, notat cu L_i. Sarcinile de lucru trebuie distribuite catre K angajați, cu următoarele constrângeri:

- fiecare angajat să lucreze la task-uri cât mai similare ca durată, de aceea este obligatoriu sa aleagă un bloc contiguu de task-uri din ordonarea lor crescător în funcție de timp;
- numărul maxim de ore lucrate de un angajat pentru rezolvarea tuturor task-urilor atribuite lui să fie cat mai mic posibil.

Fiecare angajat trebuie să primească cel puțin un task. Oricărui angajat i se pot atribui mai multe task-uri; numărul total de ore lucrate de angajat este suma duratelor individuale pentru fiecare task.

Input Format

De pe prima linie de la intrarea standard se citesc 2 numere naturale N si K, separate prin spațiu, reprezentând numărul de sarcini de lucru, respectiv numărul de angajați disponibili.

Pe urmatoarea linie se găsesc N numere naturale separate prin spațiu, reprezentând duratele task-urilor (lista L).

Constraints

- $2 \leq N < 120{,}000$
- $2 \leq K \leq N$
- $1 \leq \text{sum}(L) < 2^{31}$
- Limită de timp C++: 1 secundă
- Limită de timp Java: 2 secunde

Output Format

Se va afișa la ieșirea standard un singur număr natural, reprezentând cel mai mic număr maxim de ore lucrate de un angajat, în condițiile date.

Sample Input 0

```
4 2
72 14 30 20
```

Sample Output 0

```
72
```

Explanation 0

Task-urile se pot atribui celor doi angajați astfel:

- angajatul 1 lucrează la task-urile de durată 14, 30, 20 (cu durata totală 64);
- angajatul 2 lucrează la task-ul de durată 72.

Maximul obținut este 72.

O altă alocare ar fi putut fi:

- angajatul 1 lucrează la task-urile de durată 14 și 72 (cu durata totală 86);
- angajatul 2 lucrează la task-urile de durată 20 și 30 (cu durata totală 50).

În cazul acesta, maximul este 86, care este mai mare decât cel anterior (72).

Sample Input 1

```
4 3
4685 18503 6038 1188
```

Sample Output 1

```
18503
```

Contest ends in 44 minutes

Submissions: 31
Max Score: 50
Difficulty: Hard

Rate This Challenge:

More

C++

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

3.2 - Coloana oficială - Grea

Președintele unui stat Z este plecat într-o misiune diplomatică și este însoțit de o coloana formata din X mașini de poliție și Y mașini de armată (masinile sunt pozitionate consecutiv) ce respecta urmatoarele condiții:

- nu pot fi mai mult de x_1 mașini de poliție una dupa alta(consecutive);
- nu pot fi mai mult de y_1 mașini de armată una dupa alta(consecutive).

Să se găsească câte astfel de așezări sunt valide pentru coloana oficială a președintelui știindu-se numărul total de mașini de poliție si armată. Deoarece rezultatul poate fi prea mare, se cere restul împărțirii la 100000000 (10^8).

Input Format

Se va citi de la intrarea standard. Pe prima linie se găsesc numerele X (numărul total de mașini de poliție), Y (numărul total de mașini de armată), separate prin spațiu. Pe a doua linie se găsesc, separate prin spațiu, x_1 și y_1, cu semnificațiile din enunț.

Constraints

- $1 \leq X, Y \leq 100$
- $1 \leq x_1, y_1 \leq 10$
- Limită de timp C++: 1 secundă
- Limită de timp Java: 2 secunde

Output Format

Se va afișa la ieșirea standard un singur număr natural ce reprezintă numărul de modalități cerut în enunț.

Sample Input 0

```
2 3
1 2
```

Sample Output 0

```
5
```

Explanation 0

Configurațiile posibile sunt: *PAPAA*, *PAAPA*, *APAPA*, *AAPAP*, *APAAP* cu semnificațiile:

- A - masină de armată;
- P - masină de poliție.

Sample Input 1

```
3 2
1 2
```

Sample Output 1

```
1
```

Contest ends in 44 minutes

Submissions: 8
Max Score: 50
Difficulty: Hard

Rate This Challenge:

More

C++

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}
```

Line: 1 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code