



Task Prison

 2 sec.  1024 MB

Alisa și Bob au fost condamnați pe nedrept la închisoare de maximă securitate. Acum trebuie să-și planifice evadarea. Pentru a face acest lucru, trebuie să poată comunica cât mai eficient posibil (în particular, Alisa trebuie să-i trimită zilnic informații lui Bob). Cu toate acestea, ei nu se pot întâlni și pot schimba informații doar prin notițele scrise pe șervețele. În fiecare zi, Alisa vrea să-i trimită lui Bob o nouă informație - un număr între 0 și $N - 1$. La fiecare prânz, Alisa primește trei șervețele și scrie un număr între 0 și $M - 1$ pe fiecare șervețel (pot exista repetări) și le lasă pe scaunul ei. Apoi, dușmanul lor, Charly, distruge unul dintre șervețele și le amestecă pe celelalte două. În cele din urmă, Bob găsește cele două șervețele rămase și citește numerele de pe ele. El trebuie să decodifice cu precizie numărul original pe care Alisa a vrut să i-l trimită. Spațiul pe șervețele este limitat, deci M este fix. Cu toate acestea, scopul lui Alisa și Bob este de a maximiza fluxul de informații, așa că sunt liberi să aleagă N cât mai mare posibil. Ajută-i pe Alisa și Bob implementând o strategie pentru fiecare dintre ei, încercând să maximizezi valoarea lui N .



Implementation details

Deoarece aceasta este o problemă de comunicare, programul dvs. va fi rulat în două execuții separate (una pentru Alisa și una pentru Bob) care nu pot partaja date sau comunica în alt mod decât cel descris aici. Trebuie să implementați trei funcții:

```
int setup(int m);
```

Aceasta va fi apelată o dată la începutul executării programului tău de către Alisa și o dată la începutul executării programului tău de către Bob. I se atribuie M și trebuie să returneze valoarea dorită N . Ambele apeluri către `setup` trebuie să returneze aceeași valoare N .

```
std::vector<int> encode(int a);
```

Aceasta implementează strategia lui Alisa. Va fi apelată cu numărul de codificat A ($0 \leq A < N$) și trebuie să returneze trei numere W_1, W_2, W_3 ($0 \leq W_i < M$) care codifică A . Această funcție va fi apelată de T ori - o dată pe zi (valorile lui A se pot repeta între zile).

```
int decode(int x, int y);
```

Aceasta implementează strategia lui Bob. Va fi apelată cu două dintre cele trei numere returnate de `encode` într-o anumită ordine. Trebuie să returneze aceeași valoare A pe care a primit-o `encode`. Această funcție va fi, de asemenea, apelată de T ori - corespunzând celor T apeluri către `encode`; acestea vor fi în aceeași ordine. Toate apelurile către



encode vor avea loc înaintea tuturor apelurilor către decode.



Constraints

- $M \leq 4300$
- $T = 5000$



Scoring

Pentru un anumit subtask, fracțiunea S din punctele obținute depinde de cel mai mic N returnat de `setup` la orice test din acel subtask. De asemenea, depinde de N^* , care este valoarea țintă a lui N de care aveți nevoie pentru a obține punctajul maxim pentru subtask:

- Dacă soluția eșuează la oricare test, atunci $S = 0$.
- Dacă $N \geq N^*$, atunci $S = 1.0$.
- Dacă $N < N^*$, atunci $S = \max\left(0, 35 \max\left(\frac{\log(N) - 0,985 \log(M)}{\log(N^*) - 0,985 \log(M)}, 0.0\right)^{0,3} + 0,65 \left(\frac{N}{N^*}\right)^{2,4}, 0.01\right)$.



Subtasks

Subtask	Points	M	N^*
1	10	700	82017
2	10	1100	202217
3	10	1500	375751
4	10	1900	602617
5	10	2300	882817
6	10	2700	1216351
7	10	3100	1603217
8	10	3500	2043417
9	10	3900	2536951
10	10	4300	3083817



Example

Luați în considerare următorul exemplu cu $T = 5$. Aici avem o schemă de codificare în care Alisa trimite trei numere egale pentru a codifica 0 sau trei numere distincte pentru a codifica 1. Observați că Bob poate decoda numărul original din oricare două dintre cele trei numere trimise de Alisa.



Execution	Function call	Return value
Alisa	setup(10)	2
Bob	setup(10)	2
Alisa	encode(0)	{5, 5, 5}
Alisa	encode(1)	{8, 3, 7}
Alisa	encode(1)	{0, 3, 1}
Alisa	encode(0)	{7, 7, 7}
Alisa	encode(1)	{6, 2, 0}
Bob	decode(5, 5)	0
Bob	decode(8, 7)	1
Bob	decode(3, 0)	1
Bob	decode(7, 7)	0
Bob	decode(2, 0)	1



Sample grader

Pentru sample grader, toate apelurile către `encode` și `decode` vor fi în aceeași execuție a programului. În plus, `setup` va fi apelat o singură dată (spre deosebire de două ori, o dată per execuție, ca în sistemul de evaluare).

Intrarea este doar un singur număr întreg – M . Apoi se va afișa N returnat de `setup`. Apoi se va apela `encode` și `decode` în aceasta ordine de T ori cu numere generate aleatoriu de la 0 la $N - 1$ și alegeri generate aleatoriu dintre cele două numere din `encode` care vor fi date către `decode` (și în ce ordine). Va afișa un mesaj de eroare dacă soluția a eșuat.