

XORtimine

Ajapiirang: 2 sekundit

Mälupiirang: 1024 MiB

See on interaktiivne ülesanne.

Iliyan on teinud Jonasele vembu: ta on peitnud permutatsiooni $p_0, p_1, \dots, p_{N-1}$ täisarvudest 0 kuni $N - 1$. Jonas tahab meeleheitlikult seda taastada ja sina oled nõustunud teda aitama.

Iliyan ei paljasta permutatsiooni otse, kuid on nõustunud vastama järgmist tüüpi küsimustele: sa võid valida kaks indeksit i ja j nii, et $0 \leq i, j < N$, ja ta ütleb sulle p_i ja p_j bitikaupa XOR-i (välistav VÕI).

Kahe arvu bitipõhine XOR (välistav VÕI), mida tähistatakse $\oplus$, teostab loogilise välistava VÕI tehte igal vastaval bitipaaril. Tulemus igas positsioonis on 1, kui täpselt üks bittidest on 1, aga on 0, kui mõlemad on 0 või mõlemad on 1. Näiteks $75 \oplus 29 = 86$, sest $1001011_{(2)} \oplus 0011101_{(2)} = 1010110_{(2)}$.

C++ keeles saad arvutada a ja b XOR-i, kasutades operaatorit $\wedge$.

Iliyan on nõus vastama **kõige rohkem** Q küsimusele.

Kuid need küsimused ei suuda alati üheselt määrata peidetud permutatsiooni. Ütleme, et permutatsioon $a_0, a_1, \dots, a_{N-1}$ on *eristamatu* permutatsioonist p , kui $a_i \oplus a_j = p_i \oplus p_j$ iga sellise indeksite paari i ja j korral, et $0 \leq i, j < N$. Iga vastus, mille Iliyan annab, on ühesugune, olenemata sellest, kas peidetud permutatsioon on p või a , seega ei suuda ükski küsimuste jada neid kahte eristada. Pane tähele, et p on alati eristamatu iseendast ning et võib ka juhtuda, et ühtegi teist sellist permutatsiooni ei ole.

Sinu ülesanne on leida leksikograafiliselt vähim permutatsioon, mis on eristamatu permutatsioonist p .

Permutatsioon $x_0, x_1, \dots, x_{N-1}$ on leksikograafiliselt väiksem kui $y_0, y_1, \dots, y_{N-1}$, kui esimesel positsioonil k , kus need erinevad, kehtib $x_k < y_k$. Näiteks $x = [2, 0, 1, 4, 3]$ on leksikograafiliselt väiksem kui $y = [2, 0, 3, 1, 4]$, sest esimene positsioon, kus need erinevad, on $k = 2$, ja $x_2 = 1 < 3 = y_2$.

Peidetud permutatsioon on kindlaks määratud enne teie programmi käivitumist ega muutu teie küsimustele vastamise ajal.

Teostuse üksikasjad

Sinu programm peab realiseerima funktsiooni `solve`:

```
std::vector<int> solve(int N)
```

- N : permutatsiooni elementide arv.

Seda funktsiooni kutsutakse täpselt T korda, ühe korra iga peidetud permutatsiooni jaoks. See peab tagastama leksikograafiliselt väikseima permutatsiooni, mida ei saa eristada selle kutse jaoks peidetud permutatsioonist.

Žürii programmiga suhtlemiseks on sulle antud funktsioon:

```
int get_xor(int i, int j)
```

Iga kord, kui funktsiooni `get_xor` kutsutakse, tagastab see $p_i \oplus p_j$. Mõlemad argumendid peavad olema permutatsiooni kehtivad indeksid. Kui see tingimus pole täidetud, saab sinu lahendus otsuse `Output isn't correct: Invalid argument`. Võid eeldada, et funktsioon vastab konstantse ajaga $O(1)$.

Q väärtus on iga alamülesande jaoks fikseeritud ja seda ei edastata sinu programmile.

Piirangud

- $1 \leq N_i \leq 2^{20}$ iga $1 \leq i \leq T$ korral, kus N_i on N väärtus i -ndal `solve` väljakutsel.
- $1 \leq T \leq 2^{10}$
- $T \cdot \max(N_1, N_2, \dots, N_T) \leq 2^{25}$.
- Ühes testis funktsiooni `solve` i -nda väljakutse ajal võid küsida kõige rohkem Q_i küsimust, kus Q_i on sõltuvalt alamülesandest kas N_i või N_i^2 .

Näide

Vaatleme järgmist suhtlust, mille käigus `solve` kutsutakse välja kaks korda:

Võistleja programm	Žürii programm
	<code>solve(4)</code>
<code>get_xor(0, 1)</code>	2
<code>get_xor(0, 2)</code>	3
<code>get_xor(0, 3)</code>	1
<code>get_xor(1, 2)</code>	1
<code>get_xor(1, 3)</code>	3
<code>get_xor(2, 3)</code>	2
<code>return {0, 2, 3, 1}</code>	
	<code>solve(1)</code>
<code>return {0}</code>	

Esimese väljakutse selgitus

Peidetud permutatsioon on $[2, 0, 1, 3]$, kuid leksikograafiliselt väikseim sellest eristamatu permutatsioon on $[0, 2, 3, 1]$. Väärtuse $N_1 = 4$ jaoks esitati kuus küsimust, mis on lubatud, sest selles alamülesandes $Q_1 = N_1^2 = 16$.

Teise väljakutse selgitus

Ainus võimalik permutatsioon, mille korral $N_2 = 1$, on $[0]$.

Alamülesanded

Alamülesanne	Punktid	N_i	T	Q_i	Täiendavad piirangud
0	0	-	-	N_i^2	Näide.
1	7	$\leq 2^3$	2^{10}	N_i^2	-
2	18	$\leq 2^7$	2^5	N_i^2	-
3	5	$\leq 2^{11}$	2^5	N_i^2	-
4	5	$\leq 2^{11}$	2^5	N_i	-
5	5	$\leq 2^{18}$	2^5	N_i	$N_i = 2^k$ mingi täisarvu k korral.
6	5	$\leq 2^{18}$	2^5	N_i	N_i on paaritu.
7	30	$\leq 2^{18}$	2^5	N_i	-
8	25	$\leq 2^{20}$	2^5	N_i	-

Näidishindaja

Sisendi vorming on järgmine:

- rida 1: kaks täisarvu – väärtused T ja Q_{type} , kus Q_{type} on kas 1 või 2;
- järgnevad $2T$ rida kirjeldavad teste:
 - rida $2i$: üks täisarv – väärtus N_i i -nda testi jaoks;
 - rida $2i + 1$: N_i täisarvu – permutatsioon $p_0, p_1, \dots, p_{N_i-1}$ i -nda testi jaoks.

Kui $Q_{type} = 1$, siis $Q_i = N_i$ funktsiooni `solve` i -nda väljakutse jaoks; kui $Q_{type} = 2$, siis $Q_i = N_i^2$.

Väljundi vorming on järgmine:

- rida i : permutatsioon, mille sinu programm tagastas i -ndal testil.