

XORting

Vremensko ograničenje: 2 seconds

Memorijsko ograničenje: 1024 MiB

Ovo je interaktivni zadatak.

Ilija prenkuje Admira: ima tajnu permutaciju p (niz brojeva $p_0, p_1, \dots, p_{N-1}$) cijelih brojeva od 0 do $N - 1$. Admir je očajno želi otkriti, a vi ste pristali da mu pomognete.

Ilija neće otkriti permutaciju direktno, ali je pristao odgovoriti na pitanja sljedećeg tipa: možete odabrati dva indeksa i i j gdje $0 \leq i, j < N$, a Ilija će vam reći bitwise XOR (isključivo ILI) vrijednost brojeva p_i i p_j .

Bitwise XOR (isključivo ILI) dva broja, označeno sa $\oplus$, primijeni logičku isključivu ILI operaciju na svaki par odgovarajućih bita. Rezultat na svakoj poziciji je 1 ako je tačno jedan od bita 1, a rezultat će biti 0 ako su oba 0 ili ako su oba 1. Na primjer, $75 \oplus 29 = 86$, zato što je $1001011_{(2)} \oplus 0011101_{(2)} = 1010110_{(2)}$.

U C++ jeziku možete izračunati XOR brojeva a i b korištenjem $\wedge$ operatora.

Ilija će odgovoriti na **najviše** Q pitanja.

Ipak, ova pitanja ne mogu uvijek odrediti tajnu permutaciju jednoznačno. Za permutaciju a (niz brojeva $a_0, a_1, \dots, a_{N-1}$) kažemo da je *slična* permutaciji p (niz brojeva $p_0, p_1, \dots, p_{N-1}$) ako vrijedi $a_i \oplus a_j = p_i \oplus p_j$ za svaki par indeksa i i j , gdje $0 \leq i, j < N$. Svaki Ilijin odgovor je identičan bilo da je tajna permutaciju p ili a , tako da ne postoji niz pitanja na osnovu kojih se može odrediti koja je od te dvije. Primjetite da je p uvijek slična samoj sebi, te da možda nema više permutacija koje su joj slične.

Vaš zadatak je naći leksikografski najmanju permutaciju koja je slična p .

Permutacija $x_0, x_1, \dots, x_{N-1}$ je leksikografski manja od $y_0, y_1, \dots, y_{N-1}$ ako, na prvom indeksu k gdje se razlikuju vrijedi $x_k < y_k$. Na primjer, $x = [2, 0, 1, 4, 3]$ je leksikografski manja od $y = [2, 0, 3, 1, 4]$ zato što je prvi indeks gdje se razlikuju $k = 2$, te $x_2 = 1 < 3 = y_2$.

Tajna permutacija određena prije nego što se vaš program pokrene i ne mijenja se uslijed vaših pitanja.

Detalji implementacija

Vaš program mora implementirati funkciju `solve`:

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

- N : broj elemenata permutacije.

Ova funkcija će biti pozvana tačno T puta, jednom za svaku tajnu permutaciju. Ona mora vratiti leksikografski najmanju permutaciju koju je nemoguće razlikovati od tajne permutacije tog poziva vaše funkcije.

Kako biste komunicirali sa programom za testiranje dostupna vam je funkcija:

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

Svaki put kada se pozove funkcija `get_xor` ona vrati $p_i \oplus p_j$. Oba argumenta funkcije (i i j) moraju biti validni indeksi permutacije. Ako ovaj uslov nije ispunjen, vaše rješenje će dobiti odluku `Output isn't correct: Invalid argument`. Možete pretpostaviti da funkcija odgovara u konstantnom vremenu $O(1)$.

Vrijednost Q je unaprijed određena za svaki podzadatak i ne proslijeđuje se vašem programu.

Ograničenja

- $1 \leq N_i \leq 2^{20}$ za svaki $1 \leq i \leq T$, gdje je N_i vrijednost N u i -tom pozivu `solve`
- $1 \leq T \leq 2^{10}$
- $T \cdot \max(N_1, N_2, \dots, N_T) \leq 2^{25}$.
- U toku i -tog poziva `solve` u testnom primjeru možete postaviti najviše Q_i pitanja, gdje je Q_i ili N_i ili N_i^2 , ovisno od podzadatka.

Primjer

Razmotrite sljedeću interakciju, u kojoj se `solve` poziva dva puta:

Vaš program	Program za testiranje
	<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>	

Objašnjenje prvog poziva

Tajna permutacija je $[2, 0, 1, 3]$, ali leksikografski najmanja permutacija koja joj je slična je $[0, 2, 3, 1]$. Šest pitanja je postavljeno za $N_1 = 4$, što je dozvoljeno jer je $Q_1 = N_1^2 = 16$ u ovom podzadatku.

Objašnjenje drugog poziva

Jedina moguća permutacija sa $N_2 = 1$ je $[0]$.

Podzadaci

Podzadatak	Bodovi	N_i	T	Q_i	Dodatna ograničenja
0	0	-	-	N_i^2	Primjer iz teksta zadatka.
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	$N_i = 2^k$ za neki cijeli broj k .
6	5	$\leq 2^{18}$	2^5	N_i	N_i je neparno.
7	30	$\leq 2^{18}$	2^5	N_i	-
8	25	$\leq 2^{20}$	2^5	N_i	-

Sample grader

Format ulaza je:

- linija 1: dva cijela broja – vrijednosti T i Q_{tip} , gdje je Q_{tip} ili 1 ili 2;
- narednih $2T$ linija opisuje testne primjere:
 - linija $2i$: jedan cijeli broj – vrijednost N_i za i -ti testni primjer;
 - linija $2i + 1$: N_i cijelih brojeva – permutacija $p_0, p_1, \dots, p_{N_i-1}$ za i -ti testni primjer.

Ako je $Q_{type} = 1$, onda $Q_i = N_i$ za i -ti poziv `solve`; ako $Q_{type} = 2$, onda $Q_i = N_i^2$.

Format izlaza je:

- linija i : permutacija koju je vaš program vratio na i -tom testnom primjeru.