

XORting

Временско ограничување: 2 секунди

Мемориско ограничување: 1024 MiB

Ова е интерактивна задача.

Ијан направил шега со Зоран: тој (во лифт) скрил пермутација $p_0, p_1, \dots, p_{N-1}$ на целите броеви од 0 до $N - 1$. Зоран очајнички сака да ја открие, а вие се согласивте да му помогнете.

Ијан нема директно да ја открие пермутацијата, но се согласил да одговара на прашања од следниов вид: можете да изберете два индекси i и j такви што $0 \leq i, j < N$, и тој ви кажува кој е битовскиот XOR (ексклузивно ИЛИ) на p_i и p_j .

Битовскиот XOR (ексклузивно ИЛИ) на два броја, означен со $\oplus$, ја врши логичката операција ексклузивно ИЛИ над секој пар соодветни битови. Резултатот на секоја позиција е 1 ако точно еден од битовите е 1, а е 0 ако двата бита се 0 или двата бита се 1. На пример, $75 \oplus 29 = 86$, бидејќи $1001011_{(2)} \oplus 0011101_{(2)} = 1010110_{(2)}$.

Во C++, XOR-от на a и b може да го пресметате користејќи го операторот $\wedge$.

Ијан е подготвен да одговори на **најмногу** Q прашања.

Меѓутоа, овие прашања не можат секогаш еднозначно да ја одредат скриената пермутација. Дефинираме дека пермутацијата $a_0, a_1, \dots, a_{N-1}$ е *неразличлива* од p ако $a_i \oplus a_j = p_i \oplus p_j$ за секој пар индекси i и j такви што $0 \leq i, j < N$. Секој одговор што го дава Ијан е ист без разлика дали скриената пермутација е p или a , па затоа никаква низа од прашања не може да да направи разлика меѓу двете пермутации. Забележете дека p е секогаш неразличлива од самата себе, и дека можеби нема друга таква пермутација.

Вашата задача е да ја најдете лексикографски најмалата пермутација неразличлива од p .

Низата $x_0, x_1, \dots, x_{N-1}$ е лексикографски помала од низата $y_0, y_1, \dots, y_{N-1}$ ако, на првата позиција k на која се разликуваат, важи $x_k < y_k$. На пример, $x = [2, 0, 1, 4, 3]$ е лексикографски помала од $y = [2, 0, 3, 1, 4]$, бидејќи првата позиција на која се разликуваат е $k = 2$, и $x_2 = 1 < 3 = y_2$.

Скриената пермутација е фиксирана пред да започне вашата програма и не се менува врз основа на вашите прашања.

Детали за имплементацијата

Вашата програма мора да ја имплементира функцијата `solve`:

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

- N : бројот на елементи на пермутацијата.

Оваа функција се повикува точно T пати, по еднаш за секоја скриена пермутација. Таа мора да ја врати лексикографски најмалата пермутација неразличлива од пермутацијата скриена за тој повик.

За да комуницирате со програмата на организаторите, ви е дадена функцијата:

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

Секој пат кога ќе се повика функцијата `get_xor`, таа враќа $p_i \oplus p_j$. Двата аргументи мора да бидат валидни индекси на пермутацијата. Ако овој услов не е исполнет, вашето решение ќе добие пресуда `Output isn't correct: Invalid argument`. Можете да претпоставите дека функцијата одговара за константно време $O(1)$.

Вредноста на Q е фиксна за секоја подзадача и не се предава кон вашата програма.

Ограничувања

- $1 \leq N_i \leq 2^{20}$ за секое $1 \leq i \leq T$, каде што N_i е вредноста на N во i -тиот повик на `solve`
- $1 \leq T \leq 2^{10}$
- $T \cdot \max(N_1, N_2, \dots, N_T) \leq 2^{25}$.
- За време на i -тиот повик на `solve` во еден тест може да поставите најмногу Q_i прашања, каде што Q_i е или N_i или N_i^2 , во зависност од подзадачата.

Пример

Разгледајте ја следнава интеракција, во која `solve` се повикува двапати:

Ваша програма	Судиска програма
	<code>solve(4)</code>
<code>get_xor(0, 1)</code>	2
<code>get_xor(0, 2)</code>	3

Ваша програма	Судиска програма
<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>	

Објаснување на првиот повик

Скриената пермутација е $[2, 0, 1, 3]$, но лексикографски најмалата неразличлива од неа е $[0, 2, 3, 1]$. Поставени се шест прашања за $N_1 = 4$, што е дозволено бидејќи $Q_1 = N_1^2 = 16$ во оваа подзадача.

Објаснување на вториот повик

Единствената можна пермутација со $N_2 = 1$ е $[0]$.

Подзадачи

Подзадача	Поени	N_i	T	Q_i	Дополнителни ограничувања
0	0	-	-	N_i^2	Примерот.
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	$N_i = 2^k$ за некој цел број k .
6	5	$\leq 2^{18}$	2^5	N_i	N_i е непарен број.
7	30	$\leq 2^{18}$	2^5	N_i	-
8	25	$\leq 2^{20}$	2^5	N_i	-

Пример-грејдер

Форматот на влезот е следниов:

- ред 1: два цели броја – вредностите на T и Q_{type} , каде што Q_{type} е или 1 или 2;
- следните $2T$ реда ги опишуваат тестовите:
 - ред $2i$: еден цел број – вредноста на N_i за i -тиот тест;
 - ред $2i + 1$: N_i цели броеви – пермутацијата $p_0, p_1, \dots, p_{N_i-1}$ за i -тиот тест.

Ако $Q_{type} = 1$, тогаш $Q_i = N_i$ за i -тиот повик на `solve`; ако $Q_{type} = 2$, тогаш $Q_i = N_i^2$.

Форматот на излезот е следниов:

- ред i : пермутацијата вратена од вашата програма на i -тиот тест.