XORanges

Janez be galo mėgsta apelsinus! Todėl jis sukūrė apelsinams skirtą skenerį. Pasitelkęs kamerą ir Raspberry Pi 3b+ kompiuterį, jis pradėjo kurti trimačius apelsinų vaizdus. Jo vaizdų procesorius nėra labai geras, todėl vienintelė išvestis, kurią jis gauna, yra vienas 32 bitų sveikasis skaičius nusakantis žievėje esančias skylutes. 32 bitų sveikasis skaičius D yra apibūdinamas 32 skaitmenų (bitų) seka kurių kiekvienas yra lygus vienetui arba nuliui. Jei pradedame nuo 0, skaičių D galime gauti pridėdami 2^i kiekvienam i-ajam bitui, kuris yra lygus vienam. Kitaip tariant, jei $D = d_{31} \cdot 2^{31} + d_{30} \cdot 2^{30} + \ldots + d_1 \cdot 2^1 + d_0 \cdot 2^0$, tai skaičių D apibūdina seka $d_{31}, d_{30}, \ldots, d_0$. Pavyzdžiui, skaičių 13 apibūdina seka $0, ..., 0, 1, 1, 0, 1$.

Janez nuskenavo n apelsinų. Kartais vykdydamas programą jis nusprendžia pakartotinai nuskenuoti vieną iš apelsinų (t.y. i-ąjį apelsiną). Tai reiškia, kad nuo šio skenavimo jis naudos atnaujintą i-ojo apelsino reikšmę.

Janez nori išanalizuoti apelsinus. Jį itin domina griežtos disjunkcijos (XOR) operacija, tad jis nusprendė atlikti šiek tiek skaičiavimų. Jis pasirenka apelsinų intervalą nuo l iki u (kur $l \leq u$) ir nori atlikti XOR operaciją su visais intervale esančiais elementais, visomis intervale esančiomis gretimų elementų poromis, visais intervale esančiais gretimų elementų trejetais ir t.t. iki sekos, sudarytos iš $u - l + 1$ iš eilės einančių reikšmių (t.y. visų intervalo reikšmių).

Pavyzdžiui, jei $l = 2$ ir $u = 4$, o perskaitytos reikšmės įrašytos į masyvą A, programa turėtų apskaičiuoti reiškinio $a_2 \oplus a_3 \oplus a_4 \oplus (a_2 \oplus a_3) \oplus (a_3 \oplus a_4) \oplus (a_2 \oplus a_3 \oplus a_4)$ reikšmę. Čia $\oplus$ žymi XOR operaciją, o a_i žymi masyvo A i-ąjį elementą.

XOR operacija apibrėžiama taip:

Jei i-asis pirmosios reikšmės bitas lygus i-ajam antrosios reikšmės bitui, rezultato i-asis bitas yra lygus `0`. Jei i-asis pirmosios reikšmės bitas yra nelygus i-ajam antrosios reikšmės bitui, rezultato i-asis bitas yra lygus `1`.

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Pavyzdžiui, $13 \oplus 23 = 26$.

$13 =$	$0 \ldots 001101$
$23 =$	$0 \ldots 010111$
$13 \oplus 23 = 26 =$	$0 \ldots 011010$

Pradiniai duomenys

Pirmoje įvesties eilutėje pateikti 2 teigiami sveikieji skaičiai n ir q (skenavimų skaičius ir analizavimo užklausų skaičius).

Kitoje eilutėje pateikta n tarpais atskirtų neneigiamų sveikųjų skaičių, reiškiančių masyvo A reikšmes (apelsinų skenavimo rezultatus). a_i nurodo i-ojo apelsino reikšmę. Apelsinai numeruojami nuo 1.

Veiksmai pateikti tolesnėse q eilučių. Kiekvienoje šių eilučių pateikta po tris tarpais atskirtus teigiamus sveikuosius skaičius.

Jei veiksmo tipas yra 1 (pakartotinis skenavimas), pirmas skaičius lygus 1, o po jo pateiktas apelsino, kurį Janez nori pakartotinai nuskenuoti, indeksas i bei i-ojo apelsino pakartotinio skenavimo rezultatas j. Jei veiksmo tipas yra 2 (užklausa), pirmasis skaičius lygus 2, o po jo pateikti l ir u.

Rezultatai

Jūsų programa turi išvesti po vieną sveikąjį skaičių kiekvienai užklausai. Kiekviena reikšmė turi būti išvesta naujoje eilutėje. Atkreipkite dėmesį, kad i-oji išvesties eilutė turi būti i-osios užklausos rezultatas.

Ribojimai

- $a_i \leq 10^9$
- $0 < n, q \leq 2 \cdot 10^5$

Dalinės užduotys

1. **[12 taškų]**: $0 < n, q \leq 100$
2. **[18 taškų]**: $0 < n, q \leq 500$ ir nėra pakartotinių skenavimų (veiksmų, kurių tipas 1)
3. **[25 taškai]**: $0 < n, q \leq 5000$
4. **[20 taškų]**: $0 < n, q \leq 2 \cdot 10^5$ ir nėra pakartotinių skenavimų (veiksmų, kurių tipas 1)
5. **[25 taškai]**: Jokių papildomų ribojimų.

Pavyzdžiai

Pavyzdys nr. 1

Pradiniai duomenys

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

Rezultatai

```
2
0
```

Paaiškinimas

Pradiniu momentu $A = [1, 2, 3]$. Pirmoji užklausa apima visą masyvą. Užklausos rezultatas yra $1 \oplus 2 \oplus 3 \oplus (1 \oplus 2) \oplus (2 \oplus 3) \oplus (1 \oplus 2 \oplus 3) = 2$.

Po to pirmasis apelsinas perskanuojamas ir jo reikšmė yra pakeičiama į 3. Tai pakeičia tos pačios užklausos (intervalui $[1, 3]$) rezultatą į $3 \oplus 2 \oplus 3 \oplus (3 \oplus 2) \oplus (2 \oplus 3) \oplus (3 \oplus 2 \oplus 3) = 0$.

Pavyzdys nr. 2

Pradiniai duomenys

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

Rezultatai

```
2
5
4
4
```