Úloha č. 5
Cenzoři

Odpověz Sfinze! 10 b

Tato úloha je vyhodnocována automaticky. Je potřeba, aby výstup programu ***přesně*** *korespondoval se specifikací výstupu níže. Jak odevzdávat tento typ úloh se můžeš dočíst na webových stránkách FIKSu pod záložkou „Jak řešit FIKS“.*

Andrej, spokojen s dosavadní Eleanořinou prací, řekl „Poslední z úkolů je následující. Je dáno pole čísel o délce **N**, které jsou na začátku nastaveny na 0. Úkolem je zpracovat 3 typy dotazů:

- `0 b e` – vypiš minimum na intervalu od **b** do **e**,
- `1 b e A` – přičti hodnotu **A** všem prvkům na intervalu od **b** do **e**,
- `2 b e A` – nastav všechny prvky na intervalu **b** až **e** na hodnotu **A**.“

Eleanor zarazilo, že je zadání tak krátké. Hned věděla, že jedno z možných řešení spočívá v implementaci datové struktury, která pro ni bude výzvou [1]. Zasedla ke stroji a začala psát.

Vstup

Vstup bude zadán *nestandardním* způsobem: dostaneš pouze parametry pro jeho generování. Generování vstupu bude probíhat pomocí funkce nextInt, která je definovaná následovně:

$$\text{nextInt}: \ x = ((x \cdot a + b) \bmod 1000000007).$$

Na prvním řádku vstupu je číslo udávající počet testovacích sad T (v rozsahu $1 \leq T \leq 100$), následují jednotlivá zadání testovacích sad.

Každá testovací sada obsahuje právě jeden řádek s následujícími proměnnými: t, N, a, b, x_0, kde $1 \leq t \leq 10^6$ je počet operací, který bude proveden, $0 < N \leq 10^6$ je velikost pole čísel a $0 \leq a, b, x_0 < 10^9 + 7$ jsou parametry pro funkci *nextInt*. x_0 je hodnota x v prvním kroku.

Opravdové vstupy si pak vygeneruješ takto:

$$\begin{aligned} t &= \text{nextInt}() \bmod 3 \\ b &= \text{nextInt}() \bmod N \\ e &= \text{nextInt}() \bmod N \\ &\text{if}(b > e) : \text{swap}(b, e) \\ A &= \text{nextInt}() \bmod N \end{aligned}$$

Tento postup je nutné provést pro každou operaci na vstupu. Číslo A se sice pro vstup typu 0 nepoužije, ale i tak, je nutné funkci zavolat, aby se přepočítalo x. Význam výsledných čísel je následující:

- t: Druh dotazu který má být proveden
- b: Začátek intervalu
- e: Konec intervalu
- A: Přičítané/nastavované číslo

1 `https://en.wikipedia.org/wiki/Segment_tree`

Výstup

Za každou testovací sadu vypiš tři čísla na třech řádcích:

- Na prvním řádku bude xor všech **nejnižších** ID na intervalu pro všechny dotazy typu 0.
- Na druhém řádku bude xor všech **nejvyšších** ID na intervalu pro všechny dotazy typu 0.
- Na třetím řádku bude xor všech **sum** všech ID na intervalu pro všechny dotazy typu 0.

Tj. pokud bude pro jednu testovací sadu potřeba zpracovat 5 dotazů typu 0 a pro jejich intervaly bude součet ID roven $0, 0, 3, 9$ a 33, tak bude třetí řádek výstupu pro tuto testovací sadu 43, protože $\text{xor}(0, 0, 3, 9, 33) = 43$.

Pokud ti nepůjde některá z operací, můžeš místo ni tisknout 0 a můžeš tak získat parciální body.

Ukázkové vstupy

Následující dvojice vstupu a výstupu ti může přijít z testovacího systému. Vstup se dá do generátoru, a výstup je XOR opravdových odpovědí.

Vstup

```
2
3 3 4 5 5
10 10 10 5 4
```

Výstup

```
1
1
2
10
6
43
```

Následující dvojice vstupu a výstupu jsou data, se kterými budeš opravdu pracovat. Vstup vznikl za pomoci generátoru, a na výstupu jsme pro ilustraci vyznačili stav pole po každé operaci typu 1 a 2, a výstup pro každý dotaz typu 0.

Vstup

```
sady: 2
[formát] t: (b,e) [A]

1: (0,2) [1]
0: (1,2) [0]
2: (0,1) [2]

0: (5,5) [5]
2: (5,5) [5]
2: (0,0) [0]
0: (0,0) [0]
1: (3,4) [3]
0: (3,3) [3]
1: (2,9) [9]
0: (9,9) [9]
0: (1,4) [3]
1: (4,4) [4]
```

Výstup

```
pole: 0 0 0
pole: 1 1 1
výstup: 1 1 2
pole: 2 2 1
pole: 0 0 0 0 0 0 0 0 0 0
výstup: 0 0 0
pole: 0 0 0 0 0 5 0 0 0 0
pole: 0 0 0 0 0 5 0 0 0 0
výstup: 0 0 0
pole: 0 0 0 3 3 5 0 0 0 0
výstup: 3 3 3
pole: 0 0 9 12 12 14 9 9 9 9
výstup: 9 9 9
výstup: 0 12 33
pole: 0 0 9 12 16 14 9 9 9 9
```