Hlavná myšlienka

V každom kroku sa číslo c rozpadne na zvyšok a dve rovnaké čísla x_1 a x_2. Keďže postupnosť vytvorená z čísla x_1 a x_2 je taká istá, stačí si pamätať vzniknutý zvyšok a len jedno zo vzniknutých čísel ktoré budeme ďalej týmto spôsobom rozkladať, až pokým sa vzniknuté x_1 a x_2 nebudú rovnať nule.

Popis algoritmu

Vznik postupnosti si môžeme predstaviť ako strom, kde výsledná postupnosť sú zvyšky (prostredné deti každého vrcholu, ktorý nie je konečný, teda nie je 0). Ukážka stromu, kde $c = 13$: (rovnakou farbou sú zvýraznené periodicky sa opakujúce zvyšky)

```
                            13
              6               1               6
      3       0       3               3       0       3
  1   1   1       1   1   1       1   1   1       1   1   1
0 1 0   0 1 0   0 1 0   0 1 0   0 1 0   0 1 0   0 1 0   0 1 0
```

Podľa spomenutej hlavnej myšlienky vieme, že si stačí pamätať stále len jedno j. Budeme tak vytvárať len jednu vetvu stromu, počas čoho budeme počítať výskyty aktuálneho zvyšku v danom intervale pomocou jeho periódy opakovania sa vo výslednej postupnosti.

Dĺžka postupnosti vytvorenej z čísla x

Postupnosť, ktorá vznikne z čísla x je jednoznačne daná. Dĺžka tejto postupnosti je počet zvyškov ktoré vzniknú v strome vzniku tejto postupnosti. Môžeme si všimnúť, že v každej vrstve stromu sa počet vrcholov čo nie sú zvyšky ani 0, teda vrcholov v ktorých nekončí vetva, zdvojnásobí. Z každého z takýchto vrcholov vznikne práve 1 zvyšok. Keďže v každom kroku delíme x dvomi až pokým x nie je 0, strom má $\lfloor log_2 x \rfloor + 2$ vrstiev.

Z predchádzajúcich tvrdení tak vyplíva, že počet zvyškov v strome vzniku postupnosti je $2^{\lfloor log_2 x \rfloor + 2}$. To je tak aj dĺžka postupnosti vytvorenej z čísla x. Na výpočet dĺžky tejto postupnosti z x si môžeme definovať funkciu $f : y = 2^{\lfloor log_2 x \rfloor + 2}$. Pre účely algoritmu si tiež definujme $f(0) = 0$, pretože ak $x = 0$, vetvu, alebo celý strom končíme. V špeciálnom prípade, keď dostaneme na vstupe $c = 0$, to platiť nebude, keďže výsledná postupnosť bude 0 a teda má dĺžku 1. To nás však nezaujíma, pretože počet jednotiek v tejto postupnosti je 0, tak isto ako počet jednotiek v prázdnej postupnosti.

Perióda opakovania zvyšku vo výslednej postupnosti

Zvyšky v strome majú ako stredné dieťa svojho predchodcu, ľavého a pravého súrodenca. Medzi zvyškom i a najbližším zvyškom z rovnakej vrstvy j je tak vždy:

- Podstrom ľavého alebo pravého súrodenca zvyšku i.
- Zvyšok najbližšieho spoločného predchodcu týchto zvyškov. Keďže najbližší spoločný predchodca je vždy len jeden, aj tento zvyšok je vždy jeden.
- Podstrom ľavého alebo pravého súrodenca zvyšku j.

Keďže veľkosť obidvoch podpostupností vytvorených z podstromov medzi i a j je $f(x)$, kde x je súrodenec i, respektíve j, medzi i a j je $2 * f(x) + 1$ zvyškov. Z toho vyplíva, že perióda zvyšku so súrodencami x je $2 * f(x) + 2$. Pre jednoduchosť označme $p : y = 2 * f(x) + 2$.

Prvá pozícia zvyšku i vo výslednej postupnosti je tak $f(x)$, teda veľkosť podstromu ľavého súrodenca. Vo vzťahu na periódu to tak môžeme zapísať ako $p(x)/2 - 1$. Perióda je deliteľná dvomi, pretože $2 * f(x) + 2 = 2 * (f(x) + 1)$. Zvyšky z rovnakej vrstvy so súrodencom x sa tak nachádzajú vo výslednej postupnosti na indexoch $(p(x)/2 - 1) + k * p(x)$.

Spočítanie aktuálnych zvyškov nachádzajúcich sa v danom intervale

Vieme, že ak sa na indexe l nachádza zvyšok s aktuálnou periódou i, počet týchto zvyškov v danom intervale bude $\lfloor (r-l)/i \rfloor + 1$. Pre zvyšok je tak potretba vypočítať jeho prvý výskyt v danom intervale $\langle l, r \rangle$ a dosadiť ho do spomenutého vzorca ako l.

Ak $(l - i/2) \mod i \neq 0$, teda na indexe l sa nenachádza aktuálny zvyšok, musíme vypočítať l_2, ktoré bude značiť výskyt prvého aktuálneho zvyšku v danom intervale. Vieme, že číslo x môžeme zaokrúhliť na číslo y nahor ako $(x+y) - (x+y) \mod y$. Všetky výskyty aktuálneho zvyšku sú ale posunuté o $i/2$. Z toho vyplíva, že výsledný vzorec na posunutie l bude $(l+i) - (l+i-i/2) \mod i = (l+i) - (l+i/2) \mod i$. Takto vieme vypočítať l_2, ktoré potom môžeme dosadiť do vzorca $\lfloor (r-l)/i \rfloor + 1$ ako l, ak pôvodné l nie je na pozícii aktuálneho zvyšku.

V krajných prípadoch, kedy platí $l_2 > r$, nám stále výjde počet jednotiek 0, pretože $l_2 - r \leq i$, a keďže periódou i aj delíme keď chceme zistiť počet zvyškov na intervale výjde výsledok podielu -1 a $-1+1=0$.

Celý algoritmus

Počet jednotiek na intervale $\langle l, r \rangle$ tak môžeme vypočítať nasledovným algoritmom: (na začiatku nastavíme počet jednotkových zvyškov $s = 0$)

- Ak $c \mod 2 = 0$, môžeme aktuálny zvyšok preskočiť, keďže je 0 a tak počet jednotiek vo výslednej postupnosti na intervale $\langle l, r \rangle$ nijak neovplivní.
- Vypočítame periódu aktuálneho zvyšku ako $p(\lfloor c/2 \rfloor)$.
- Spočítame aktuálne zvyšky nachádzajúce sa na danom intervale a ich počet pripočítame k celkovému počtu s.
- Zmeníme hondotu $c := \lfloor c/2 \rfloor$ (tento krok vykonávame aj keď $c \mod 2 = 0$, teda aj keď periódu a súčet zvyškov v danom intervale nepočítame).
- Opakujeme pokým $c > 0$.

Výsledok, ktorý na konci vypíšeme je tak celkový počet jednotkových zvyškov s.

Časová a priestorová zložitosť

V algoritme delíme c dvomi pokým $c > 0$. To znamená, že vždy vykonáme rádovo $log_2 c$ krokov. Časová zložitosť je tak $O(log_2 c)$. Priestorová zložitosť je $O(1)$.

Python program

```python
# vypocita dlzku postupnosti vytvorenej z cisla x
def size(x):
    if x == 0:
        return 0
    return 2 ** (floor(log2(x)) + 1) - 1

# vrati pocet jednotiek v postupnosti vytvorenej z c na intervale <l, r>
def solve(c, l, r):
    total_interval_sum = 0
    while c > 0:
        if c % 2 == 1:
            period = 2 * size(c // 2) + 2
            shifted_l = l
            if (l - period // 2) % period != 0:
                shifted_l = (l + period) - (l + period // 2) % period
            current_inteval_sum = (r - shifted_l) // period + 1
            total_interval_sum += current_inteval_sum
        c //= 2
    return total_interval_sum

input_line = [int(x) for x in input().split()]
answer = solve(*input_line)
print(answer)
```