АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА
БИТОВА МАГИЯ

Условието **x & b = b** означава, че във всички битове, в които числото b има единица, числото x също **задължително** трябва да има единица. В битовете, в които b има нула, числото x може да съдържа както 0, така и 1.

За да намерим броя на подходящите числа в интервала [l, r], ще използваме стандартната формула:

$$ans(l,r)=count(r)-count(l-1),$$

където $count(N)$ е броят на числата x, за които: $0 \le x \le N$ и е изпълнено условието: $x \,\&\, b = b$.

Подзадачи 1– 4

В подзадачите 1– 4 ограниченията са достатъчно малки: r, b < 16^7**,** затова всички решения могат да се реализират с пълно изчерпване.

За всяка подзадача се обхождат всички числа x от l до r и се проверява условието: $(x \,\&\, b) = b$.

Подзадачи 5 – 6

В подзадачите 5 – 6 числата вече са достатъчно големи, но все още се побират в стандартния тип long long.

Тук вече се използва пълното решение**,** но без работа с дълга аритметика:

- изчислява се функцията $count(N)$ за едно число N**;**
- използва се побитово динамично програмиране по двоичното представяне на числото;
- минава се по битовете отляво надясно и се поддържа флаг, който показва дали текущото число вече е станало по-малко от N.

За всеки бит:

- ако битът $b_i = 1$, то в x може да се постави само 1;
- ако битът $b_i = 0$, може да се постави 0 или 1 (като се спазва ограничението $x \le N$).

Тъй като броят на битовете е ограничен (не повече от 60), решението работи за: $O(n)$ и напълно покрива ограниченията на подзадачи 5–6**.**

Подзадачи 7 – 8

В тези подзадачи числата вече не се побират в стандартните типове данни и могат да имат дължина до 16^{1000}**,** но решението все още се основава на динамично програмиране**.**

За да се изчисли $count(N)$:

- преобразуваме числото N и маската b в двоичен вид;
- разглеждаме позицията i, на която числото x за първи път става строго по-малко от N**;**
- проверяваме дали префиксът не противоречи на маската b;
- намираме броя на възможните продължения в суфикса.

Преброяването на допустимите варианти в суфикса се извършва чрез директно обхождане на битовете, затова за всяка позиция i са нужни: $O(n)$ операции.

Следователно общата сложност на това решение е $O(n^2)$. Това е достатъчно за подзадачи 7–8, но не е достатъчно за максималните ограничения.

Подзадача 10

Подзадачата по същество е аналогична на предишните, но с допълнително опростяване: $b = 0$.

Условието $x\ \&\ 0 = 0$ е изпълнено за всяко число $x \geq 0$, затова подходящи са всички числа в интервала $[l, r]$**.**

Отговорът е $(r - l + 1) \bmod (10^9 + 7)$, като всички изчисления се извършват с дълги числа**,** зададени в шестнадесетична бройна система**.**

Подзадачи 9 и 11

Дължината на числата достига 50 000 шестнадесетични символа, което съответства приблизително на 200 000 бита. Решение с квадратична сложност вече не е достатъчно тук.

Основната оптимизация се състои в ускоряване на изчисляването на броя възможни варианти в суфикса.

• Предварително се изчислява масива *suffix_zeros*[i], равен на броя нулеви битове в маската b на позиции $j \geqslant i$.

• Също така предварително се изчисляват степените на двойката $2^k \bmod (10^9 + 7)$.

Сега при разглеждане на позиция i, в която x става по-малко от N, броят на допустимите продължения в суфикса се изчислява за $O(1)$ като $2^{\text{suffix\_zeros}[i+1]}$.

Това позволява да се премахне вътрешния цикъл и да се сведе изчислението на *count(N)* до линейно време.

Пълно решение

Крайният алгоритъм:

1. Да се преобразуват числата l, r и b от шестнадесетичен в двоичен запис.
2. Предварително да се изчислят броя на нулите в маската b за всеки суфикс и степените на двойката.
3. Да се реализира линейно изчисление на *count(N).*
4. Да се получи отговорът като

$(\,count(r) - count(l) + check(l) + (10^9 + 7)\,) \bmod (10^9 + 7)$, където *check(l)* проверява дали числото l удовлетворява условието $l\ \&\ b = b$.

Времето за работа на крайното решение е $O(n)$, което позволява да се преминат подзадачи 9 и 11 и да се получи пълен брой точки.

Автор: Кинка Кирилова-Лупанова