Групи (анализ)

A5 / 320

Тази задача се състои от две части:

Комбинаторика

Както повечето от вас знаят, броят на начини да се избере група от K човека измежду N човека е равен на $\frac{N!}{K!\,(N-K)!}$, където символът ! означава факториел(factorial) или с други думи, комбинации от N елемента, K-ти клас - C_K^N.

Тъй като тези числа могат да станат много големи (около 10^{4300} в средния случай или 10^{456000} в най-лошия), не могат директно да се сметнат. За щастие, ако m е просто число:

$$\frac{a}{b} \; mod \; m =$$
$$= (a * b^{-1}) \; mod \; m =$$
$$= [(a \; mod \; m) * (b^{-1} \; mod \; m)] \; mod \; m =$$
$$= [(a \; mod \; m) * (b^{m-2} \; mod \; m)] \; mod \; m$$

За да разберете защо предпоследният и последният ред са еквивалентни, вижте тази статия.

Очевидно за повдигането на числа на степен 18181-2 трябва да се използва алгоритмът за бързо повдигане на степен, който следва от това, че

$$a^n = a^{\frac{n}{2}} * a^{\frac{n}{2}}, \text{ ако n е четно, и}$$
$$a^n = a * a^{\frac{n-1}{2}} * a^{\frac{n-1}{2}}, \text{ ако n е нечетно}$$

Също така, нужно е да се дефинират два масива – int f[100001] и int p[18181], където:

$f_a = \; a! \; mod \; 18181$, съответстващ на factorials[] в авторското решение, и

$p_a = \; a^{18181-2} \; mod \; 18181$, съотвестващ на fastPowed[] в авторското решение.

Тогава комбинациите K-ти клас от N елемента, C_K^N , са равни на $\left(f_N * p_{(f_K * f_{N-K}) \; mod \; 18181}\right) mod \; 18181$, което се смята с константна сложност.

Индексно дърво (или интервално/сегментно дърво)

В общия случай индексното дърво е двоично дърво, състоящо се от 2*N+1 върха, на което всеки връх съдържа информация за определен интервал в масив с размер N. Коренът на дървото отговаря за интервала [0,N-1]. Всеки връх с интервал [a,b], за който е вярно, че a != b, има две деца, отговарящи за интервалите [a,(a+b)/2] и [(a+b)/2+1, b], където знакът / означава **целочислено** делене.

Тази структура от данни позволява бързото намиране на сумите на елементи в интервали, както и бързото обновяване на елементи. И двете операции са със сложност $O(\log_2 N)$.

В тази задача има проблем, обаче – трябва да се намери броят на елементите със стойности 0, 1, ..., K-1 отделно в даден интервал. Ако се използва индексно дърво, то всеки негов връх трябва да съдържа масив с размер K. Това, обаче, би заело $\frac{2*4*N*K}{1024^2} \approx$

48 мегабайта, което очевидно е повече от предвидените 8 за програмата. Ако вместо това се направят K на брой компресирани индексни дървета, всяко от които е с размер N, биха заели $\frac{4*K*N}{1024^2} \approx 24$ мегабайта, което също е твърде много.

Може да се направи наблюдение върху това какво става с върховете, които отговарят за интервали [a,b], по-малки от K (т.е. b-a+1 < K). Тогава, за да се вземе броят на елементите със стойности от 0 до K-1 в него са нужни повече операции, отколкото ако се обходи интервалът [a,b].

Тогава може да се направи индексно дърво, на което листата да съответстват не на интервали с дължина 1, а на такива с дължина около K=64. Това дърво не само заема $\frac{4*4*N*K}{K*1024^2} < 2$ мегабайта място в паметта, но и заявките в него са по-бързи, защото за интервали по-малки от K не се правят K операции, а елементите на масива се обхождат линейно.