Задача ?. БАЛАНС (Анализ)

⏳ 0.2 сек. 💾 1024 MB

Нека за удобство индексирането на редицата започва от 1.

Първа подзадача

Достатъчно е да пробваме всички поредици от нула, един или два хода (ако няма такава поредица, подзадачата гарантира, че отговорът е 3). Понеже възможните ходове са точно $2 \times \frac{N(N+1)}{2} = N(N+1)$, броят редици от до два хода е сравнително малък.

Имплементация: `balance_kiril_11p.cpp` ⏳ $O(N^4 \times Q)$

Следващите три подзадачи са направени с целта състезателят да съобрази някои свойства на ходовете.

Втора подзадача

Нека с S_i означим сумата на всички числа в текущия интервал на редицата, тоест

$$S_i = \sum_{j=u_i}^{v_i} a_j.$$

Може да се види, че един ход може да увеличи или намали сумата на всички числа с най-много едно. Понеже искаме накрая сумата да е нула, броят ходове трябва да е поне S_i. От друга страна, извършването на единични ходове (такива, за които $l = r$) е достатъчно. Така отговорът на всеки въпрос е равен на S_i, което можем да изчислим за $O(1)$ чрез префиксни суми.

Имплементация: `balance_kiril_7p.cpp` ⏳ $O(N + Q)$

Трета подзадача

Интуитивно би било да увеличаваме само отрицателните числа и да намаляваме само положителните. Тук обаче ни е дадено, че абсолютните стойности са наредени в нарастващ ред. Можем да забележим, че следната алчна стратегия е оптимална: ходовете винаги да обхващат най-големия интервал от ненулеви числа. Това работи, защото ако разгледаме състоянието на редицата след извършването на даден ход, поради подредбата на числата всички числа, които се занулават, ще са в левия край на интервала - така най-оптималният ход винаги остава възможен. Оттук лесно се забелязва, че броят ходове е колкото най-голямото число в текущия интервал, тоест отговорът на всеки въпрос е равен на $|a_{v_i}|$.

Имплементация: `balance_kiril_9p.cpp` ⏳ $O(N + Q)$

Четвърта подзадача

Както видяхме в миналата задача, оптимално би било да правим колкото се може по-големи ходове, без да си пречим. От състезателят се очаква да се сети да раздели дадения интервал на алтернириращи подпоследователности, тоест да разбие интервала на няколко подмасива, за които е вярно, че знаците на числата във всеки подмасив се редуват. Ето пример за такова разбиване:

$$(2, -2, 2, 2, -2, -2, 2, -2, 2) \to (2, -2, 2), (2, -2), (-2, 2, -2, 2).$$

За всеки такъв подмасив можем да приложим точно по $|a_1|$ хода и сме готови.

Имплементация: `balance_kiril_14p.cpp` ⏳ $O(N)$

Пета подзадача

Тази подзадача е ключова за решението на задачата. За да е малко по-ясно обяснението, нека направим следната трансформация върху редицата: $a_i \to a_i \times (-1)^i$, тоест на всеки четен елемент обръщаме знака. Така всички числа са с един и същ знак и ходовете са се превърнали в "добави 1 към $a_l, \ldots, a_r$" и "извади 1 от $a_l, \ldots, a_r$".

Нека за удобство $u_1 = 1$ и $v_1 = N$. Можем да забележим, че в миналата подзадача винаги правихме ходове върху равни числа - това ни дава мотивация да разглеждаме какво се случва на границата между две числа. По-точно, нека да разглеждаме разликите между съседни числа. Нека

$$\Delta = |a_1| + |a_1 - a_2| + |a_2 - a_3| + \cdots + |a_{N-2} - a_{N-1}| + |a_{N-1} - a_N| + |a_N|.$$

Чисто интуитивно колкото по-близо тази стойност е до нула, толкова по-малко ходове трябва да направим, за да занулим редицата. Нека това го покажем - можем да направим подобни разсъждения на тези в подзадача 2. Всеки ход изменя Δ или с -2, или с 0, или с 2 (важно е да се отбележи, че Δ винаги е четно число). Понеже искаме $\Delta = 0$, броят ходове е поне $\frac{\Delta}{2}$.

А може ли да направим точно $\frac{\Delta}{2}$ хода? Да - след изпълнението на ход, който намалява Δ (такъв винаги съществува по очевидни причини), получаваме нова редица, но с по-малка стойност на Δ. Така можем рекурсивно да продължим, докато не достигнем нула. Така отговорът на единственият въпрос е $\frac{\Delta}{2}$, като стойността на Δ се отнася за зададения интервал ($[u_1, v_1]$).

Имплементация: `balance_kiril_43p.cpp` ⏳ $O(N)$

Шеста подзадача

Решението е същото като на миналата подзадача, стига да забележим, че фактът, че знаците не се редуват, не пречи по никакъв начин на разсъжденията ни. Остава да изчисляваме бързо стойността на Δ. Това може да стане чрез префиксни суми по абсолютните разлики на съседни елемтни.

Имплементация: `balance_kiril_63p.cpp` ⏳ $O(N + Q)$

Седма подзадача

Поддържането на промени може да стане бързо чрез сегментно дърво или дърво на фенуик.

Имплементация: `author.cpp` ⏳ $O((N + Q) \times log_2 N)$

Автор: Кирил Зулямски