| Тагове | На пълното решение | На подзадачите |
|---|---|---|
| | Метод на показалките<br>Префиксни суми | Двоично търсене |

## Анализ

В анализа под подмасив ще имам в предвид подредица, образувана от редица, като от нея се изтрият няколко (възможно $0$) елемента от началото и края, като се запази реда на останалите. Интересен подмасив ще е такъв, в който ако двойка боксьори се подготвят, ще получат подготовка $\geq K$. Интересността на подмасив ще наричам подготовката при оптимално разпределение на боксовите круши. Централен елемент ще наричам този, при който боксьорите се „срещат“ при оптимална подготовка. Боксьорите ще ги означа като ляв и десен, спрямо това от кой край на подмасива започват да се подготвят. Със $sum(l, r)$ ще означа сбора на силите на боксовите круши с номера от $l$ до $r$. Подмасивите с дължина $1$ винаги имат подготовка равна на $0$, заради това те ще бъдат игнорирани в анализа.

### Подзадача №1

Както винаги, оставих подзадача с тестовите примери, за да има по-добра обратна връзка от системата.

### Подзадача №2

Тази подзадача беше с проучвателна цел – дали състезателите са успяли да прочетат условието напълно. Решението ѝ се състои от $2$ проверки – първо, ако $K = 0$, то всеки подмасив ще бъде интересен, заради това отговорът ще е $3$. В противен случай единствено подмасивът, съдържащ и двете боксови круши, би могъл да бъде интересен. Тогава очевидно боксьорите ще си поделят по една боксова круша и така трябва да се провери дали $\min(a_1, a_2) \geq K$, за да видим дали отговора е $1$ или $0$.

Постигната сложност: $O(1)$

Имплементация: `boxing_4p.cpp`

### Подзадача №3

Може да се имплементира подхода от условието, като се обходят всички подмасиви и за всеки от тях се приложи $O(N^2)$ проверка за това дали той е интересен. Фиксира се централен елемент и за линейно време се изчисляват подготовките на двамата боксьори. Намира се централния елемент, за който подготовката на по-непотготвения боксьор е възможно най-голяма.

Постигната сложност: $O(N^4)$

Имплементация: `boxing_12p.cpp`

### Подзадача №4

Тази подзадача е предвидена за overkill на решението на шеста подзадача, като сбор на числа в интервал се изчислява по наивен начин, вместо с префиксни суми.

Постигната сложност: $O(N^3 \log_2 N)$

Имплементация: `boxing_20p.cpp`

### Подзадача №5

Решението на подзадачата е оптимизация на решението на трета подзадача. Вместо всеки път наново да се смятат подготовките на двамата боксьори, ние може по много по-лесен начин да ги поддържаме в две променливи. За повече детайли вижте имплементацията.

Постигната сложност: $O(N^3)$

Имплементация: `boxing_25p.cpp`

### Подзадача №6

Време е за първата ни стъпка към решението. Къде всъщност се намира централната позицията на подмасив от $l$-та до $r$-та позиция? Нека сборът на силите на всички боксови круши в подмасива е $s$, а $p$ да е сборът на тези, с които тренира левия боксьор. Тогава подготовката на десния боксьор ще е равна на $s - p$. Кога левия боксьор ще е по-неподготвения? Когато $p \leq s - p$, тоест $2p \leq s$, левия боксьор ще е по-непоготвения, а иначе, когато $2p > s$, десния ще е по-непоготвения. Нека намерим последната позиция $pos$ в подмасива, за която $2p \leq s$. Нека централна позиция да бъде именно $pos$. Тъй като силите на боксовите круши са неотрицателни, за тази позиция $p$ ще е максимално, тоест подготовката на по-неподготвения боксьор ще е възможно най-голяма, следователно измежду всички централни позиции, за които $2p \leq s$, $pos$ ще е тази, при която ще се получи максимална подготовка. Нека централна позиция е $pos + 1$. Тогава $2p > s$, тоест $s - p$ ще е подоготовката на по-неподготвения боксьор. Тъй като измежду всички централни позиции, за които $2p > s$, $pos + 1$ ще е тази за която $p$ е минимално, или иначе казано $s - p$, равна на подготовката на по-неподготвения боксьор, ще бъде максимална. Заради това интересността на подмасива ще се явява по-голямото от $sum(l, pos)$ и $sum(pos + 2, r)$. Чрез двоично търсене може да намерим $pos$, а сборовете в интервал може да ги изчисляваме чрез префиксни суми.

Постигната сложност: $O(N^2 \log_2 N)$

Имплементация: `boxing_41p.cpp`

### Подзадача №7

Много често в подобни задачи двоичното търсене може да се замени с показалки. Нека чрез $central(l, r)$ означим позицията, за която $sum(l, pos) \leq sum(pos + 1, r)$ и $sum(l, pos + 1) > sum(pos + 2, r)$. Нека допуснем, че съществува двойка числа $(l, r)$, за които $1 \leq l < r < N$ и $central(l, r) > central(l, r + 1)$. Нека $pos_1 = central(l, r)$ и $pos_2 = central(l, r + 1)$, тогава $pos_2 < pos_1$ и $pos_2 + 1 \leq pos_1$. Два директни извода са:

1) $sum(l, pos_1) \leq sum(pos_1 + 1, r)$
2) $sum(l, pos_2 + 1) > sum(pos_2 + 2, r + 1)$

Следователно $sum(l, pos_2 + 1) > sum(pos_2 + 2, r)$, но $sum(l, pos_2 + 1) \leq sum(l, pos_1)$ и $sum(pos_2 + 2, r + 1) \geq sum(pos_1 + 1, r)$. От $sum(l, pos_2 + 1) > sum(pos_2 + 2, r)$ следва, че $sum(l, pos_1) > sum(pos_2 + 2, r)$, от което следва, че $sum(l, pos_1) > sum(pos_1 + 1, r)$, следователно достигаме противоречие.

Така доказахме, че $central(l, r) \leq central(l, r + 1)$. Аналогично $central(l, r) \leq central(l + 1, r)$. Нека обходим всички подмасиви с някое начало $l$. Поддържаме показалка, която винаги да се намира на $central(l, r)$. Тъй като тези централни позиции нарастват, показалката ще я преместим най-много $O(N)$ пъти. Така за всички $N$ начални позиции на подмасиви изпълняваме $O(N)$ операции.

Постигната сложност: $O(N^2)$

Имплементация: `boxing_55p.cpp`

### Подзадача №8

Нека с $cost(l, r)$ означим интересността на подмасива от $l$ до $r$. Нека $pos$ да е оптималната централна позиция за някой подмасив от $l$ до $r$, за който $l < r$. Тогава, $cost(l, r) = \min(sum(l, pos), sum(pos + 1, r))$. Подготовката в подмасива от $l + 1$ до $r$, ако изберем $pos$ за централна позиция, ще равна на $\min(sum(l + 1, pos), sum(pos + 1, r))$. Следователно $cost(l, r) \geq cost(l + 1, r)$. Аналогично $cost(l, r) \geq cost(l, r - 1)$.

Следователно, като изберем някоя позиция $r$ за крайна, всички подмасиви, които са интересни, ще са с $l \leq x$, където $x$ е най-дясната позиция, за която $cost(x, r) \geq K$. Ние може чрез двоично търсене да намерим $x$, като с още едно вложено двоично да изчисляваме $cost(l, r)$.

Постигната сложност: $O(N \log_2^2 N)$

Имплементация: `boxing_70p.cpp`

### Подзадача №9

Нека с $begin(r)$ означим именно това най-дясно $x$, за което $cost(x,r) \geq K$. Тогава $cost(begin(r),r) \geq K$, следователно $cost(begin(r),r+1) \geq K$, тоест $begin(r) \leq begin(r+1)$. От това следва, че може да броим подмасивите с метода на показалките, като ще трябва да намерим $cost(l,r)$ за $O(N)$ подмасива. За подзадачата е предвидено $cost(l,r)$ да се изчислява чрез двоично търсене.

Постигната сложност: $O(N \log_2 N)$

Имплементация: `boxing_79p.cpp`

### Подзадача №10

За пълното решение ще е нужно да се премахне и последното двоично търсене. Нека проследим как преместваме двете показалки, като броим подмасивите. Нека показалките са ни $l$ и $r$. Ние винаги местим $r$ с една позиция напред и впоследствие преместваме $l$ от $begin(r-1)$ до $begin(r)$. Тоест, за дадено $r$, ще трябва да проверим всички подмасиви с начало от $begin(r-1)+1$ до $begin(r)+1$ и край $r$. Подмасивът с начало $begin(r-1)$ няма да го проверяваме, защото знаем, $cost(begin(r-1),r) \geq K$, а подмасивът с начало $begin(r)+1$ ще трябва да го проверим, за да разберем, че $cost(begin(r)+1,r) < K$. Какво прави двоичното търсене, което използвахме в решението на деветата подзадача? То намира $central(l,r)$, а на нас ще ни трябва да намерим $central(l,r)$ за подмасивите в реда:

$$[1,2], \dots, [begin(2)+1,2], [begin(2)+1,3], \dots, [begin(3)+1,3], \dots,$$

$$[begin(N-1)+1,N], \dots, [begin(N)+1,N].$$

Тоест, ако към момента сме намерили $central(l,r)$, за някоя двойка $(l,r)$, то следващата двойка, за която ще трябва да го намерим ще бъде или $(l+1,r)$, или $(l,r+1)$. Ние забелязахме в 7-ма подзадача, че $central(l,r) \leq central(l+1,r)$ и $central(l,r) \leq central(l,r+1)$. Това означава, че ако поддържаме още една показалка, която да следи $central(l,r)$, тя ще се премести най-много $N$ пъти. Така може да заменим вложеното двоично търсене с още една показалка и да получим решение със сложност $O(N)$.

Постигната сложност: $O(N)$

Имплементация: `boxing_100p.cpp`

*Автор: Борис Михов*