

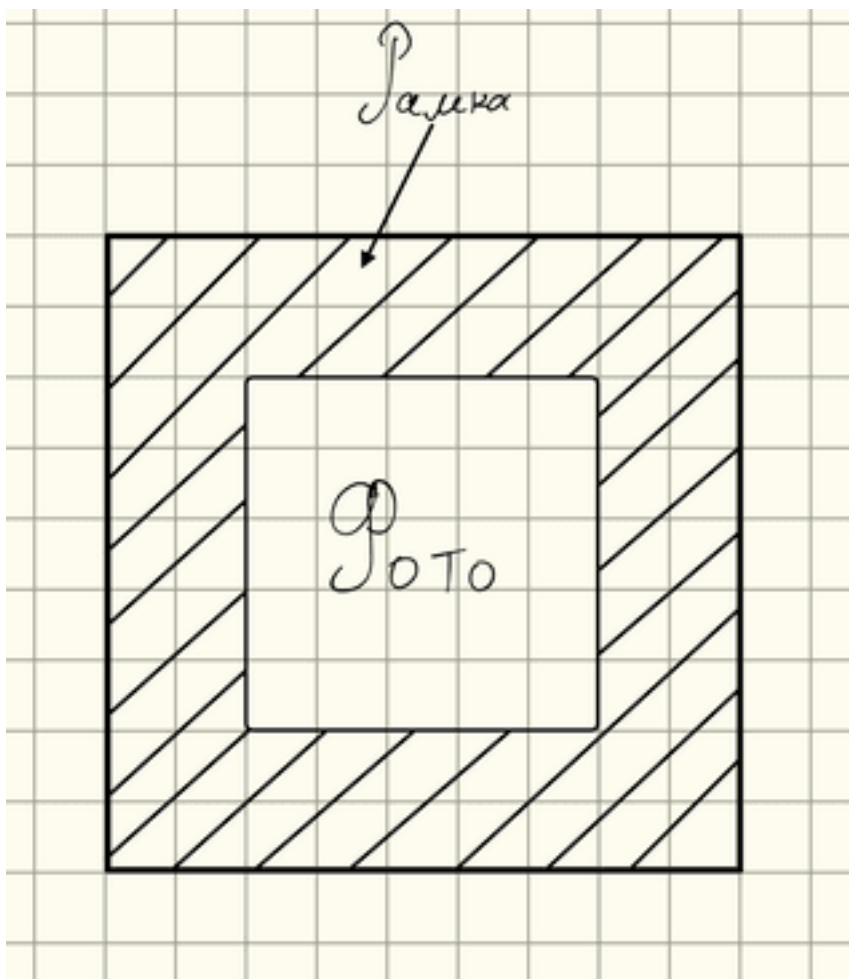
Задача А. Фотографія

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Зробивши фотографію, Василько захотів поставити її в рамку. Фотографія Василька має розмір $n \times n$, рамка повинна мати товщину t .

Порахуйте, яку площу буде мати рамка для фотографії Василька.

Наприклад, фотографія розміру 5 з шириною рамки 2 виглядає так:



Формат вхідних даних

Перший рядок містить два цілі числа n, t ($3 \leq n \leq 10^9$; $1 \leq t$; $2t < n$) — розмір фотографії, товщина рамки.

Формат вихідних даних

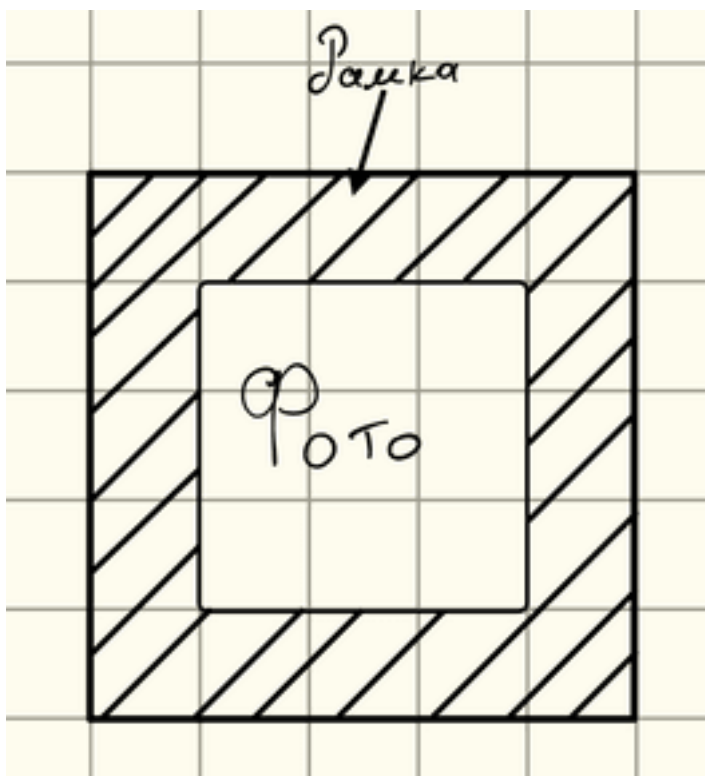
Одне ціле число, площа рамки для фотографії Василька.

Приклади

стандартний ввід	стандартний вивід
3 1	16
5 2	56

Примітка

Так виглядає рамка для першого прикладу:



Задача В. Пікнік

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Василько має пиріг, розрізаний на n шматочків. Вага кожного шматочка становить $a_1, a_2, \dots, a_n$. Він хоче поділити всі шматочки між собою та k друзями так, щоб кожен отримав порцію однакової ваги.

Пиріг потрібно ділити за такими правилами:

- Кожна людина отримує кілька шматочків (можливо, один), що йдуть у масиві один за одним (неперервний блок).
- Сумарна вага шматочків у кожній порції має бути однаковою.
- Всього пиріг ділиться на $k + 1$ рівну частину (тобто сам Василько та k друзів). Завдання задачі полягає в тому, щоб знайти максимальне число k (кількість друзів), при якому такий поділ можливий. Зверніть увагу, що кожен шматок має бути у якійсь порції.

Допоможіть Васильку порахувати, яку максимальну кількість друзів він може запросити на пікнік.

Формат входніх даних

Перший рядок містить одне число n ($1 \leq n \leq 5000$) — кількість кусочків пирога.

Другий рядок містить n цілих чисел $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq 10^4$) — вага кожного кусочка пирога.

Формат вихідних даних

Одне ціле число, максимальну кількість друзів, котрих може запросити Василько.

Приклади

стандартний ввід	стандартний вивід
11 1 3 4 2 2 3 1 4 1 1 2	5
5 2 3 2 1 3	0

Примітка

У першому прикладі він може розділити пиріг на 6 частин, де вага кожної частини буде 4. Це можна зробити, наприклад, так: $1 + 3$, 4 , $2 + 2$, $3 + 1$, 4 , $1 + 1 + 2$. Оскільки буде всього 6 частин, то він може запросити лише 5 друзів.

У другому прикладі він може лише розділити пиріг на 1 частину (весь пиріг). У такому випадку він не може запросити друзів.

Задача С. Гра

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Недавно граючись зі своїми друзями, Василько придумав нову гру. Сьогодні він вирішив поділитися нею з тобою.

В цій грі потрібно збудувати масив a довжини n , у якого такі властивості:

- кожне число має бути від $-m$ до m ;
- серед усіх можливих підвідрізків масиву є лише один, у якому сума чисел максимальна, і це підвідзок з l до r включно.

Вам потрібно побудувати такий масив, проте серед усіх таких можливих масивів вам потрібно побудувати саме той, у якому сума невід'ємних чисел максимальна.

Формат входніх даних

Перший рядок містить чотири цілі числа n, m, l, r ($1 \leq n \leq 10^5, 1 \leq m \leq 10^9, 1 \leq l \leq r \leq n$) — довжина масиву, обмеження на значення масиву, краї відрізка.

Формат вихідних даних

В одному рядку потрібно вписати побудований масив a довжини n .

Якщо відповідей декілька, то ви можете вивести будь-який масив.

Приклад

стандартний ввід	стандартний вивід
5 3 2 3	-1 3 3 -3 2

Примітка

Сума на кожному з відрізків з прикладу:

- В відрізку від 1 до 1 сума -1 .
- В відрізку від 1 до 2 сума 2 .
- В відрізку від 1 до 3 сума 5 .
- В відрізку від 1 до 4 сума 2 .
- В відрізку від 1 до 5 сума 4 .
- В відрізку від 2 до 2 сума 3 .
- В відрізку від 2 до 3 сума 6 .
- В відрізку від 2 до 4 сума 3 .
- В відрізку від 2 до 5 сума 5 .
- В відрізку від 3 до 3 сума 3 .

- В відрізку від 3 до 4 сума 0.
- В відрізку від 3 до 5 сума 2.
- В відрізку від 4 до 4 сума -3 .
- В відрізку від 4 до 5 сума -1 .
- В відрізку від 5 до 5 сума 2.

тому на відрізку від 2 до 3 буде максимальна сума, і він єдиний з такою сумою.

Задача D. Василько і комп'ютер

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Одного разу, блукаючи інтернетом, Василько натрапив на дивний комп'ютер, в котрому пам'ять мала вигляд стека, і вирішив його купити.

Коли той комп'ютер прийшов до нього, він помітив, що в подарунку до комп'ютера був масив a довжини n . Йому стало цікаво, на яку мінімальну кількість підмасивів потрібно поділити a , щоб кожен підмасив можна було відсортувати на цьому комп'ютері.

Масив можна відсортувати на тому комп'ютері, якщо йти по ньому зліва направо і додавати елементи на верх пам'яті в вигляді стека. У будь-який момент можна викласти один елемент з верху пам'яті у фінальну послідовність. Проте, не обов'язково додати лише одне число зі стека у масив між проходами. Масив можна відсортувати, якщо можна зробити фінальну послідовність відсортованою.

Наприклад, масив $[4, 2, 3, 7, 4, 5]$ можна відсортувати так:

1. додаємо число 4 у стек, зараз стек виглядає так $[4]$, а масив так $[]$;
2. додаємо число 2 у стек, зараз стек виглядає так $[4, 2]$, а масив так $[]$;
3. забираємо 2 зі стека та додаємо у масив, зараз стек виглядає так $[4]$, а масив так $[2]$;
4. додаємо число 3 у стек, зараз стек виглядає так $[4, 3]$, а масив так $[2]$;
5. забираємо 3 зі стека та додаємо у масив, зараз стек виглядає так $[4]$, а масив так $[2, 3]$;
6. забираємо 4 зі стека та додаємо у масив, зараз стек виглядає так $[]$, а масив так $[2, 3, 4]$;
7. додаємо число 7 у стек, зараз стек виглядає так $[7]$, а масив так $[2, 3, 4]$;
8. додаємо число 4 у стек, зараз стек виглядає так $[7, 4]$, а масив так $[2, 3, 4]$;
9. забираємо 4 зі стека та додаємо у масив, зараз стек виглядає так $[7]$, а масив так $[2, 3, 4, 4]$;
10. додаємо число 5 у стек, зараз стек виглядає так $[7, 5]$, а масив так $[2, 3, 4, 4]$;
11. забираємо 5 зі стека та додаємо у масив, зараз стек виглядає так $[7]$, а масив так $[2, 3, 4, 4, 5]$;
12. забираємо 7 зі стека та додаємо у масив, зараз стек виглядає так $[]$, а масив так $[2, 3, 4, 4, 5, 7]$.

Формат вхідних даних

Перший рядок містить одне ціле число n ($1 \leq n \leq 10^6$) — довжина масиву a .

Другий рядок містить n цілих чисел $a_1, a_2, \dots, a_n$ ($0 \leq a_i \leq 10^9$) — значення елементів масиву a .

Формат вихідних даних

Виведіть одне ціле число, відповідь на задачу, на яку мінімальну кількість підмасивів потрібно поділити a , щоб кожен можна було посортувати.

Система оцінювання

1. (8 балів): $n \leq 10^3$;

2. (12 балів): $a_i \leq 1$;
3. (15 балів): $a_i \leq 2$;
4. (18 балів): $a_i \leq 5$;
5. (19 балів): Відповідь не перевищує 20;
6. (28 балів): без додаткових обмежень.

Приклад

стандартний ввід	стандартний вивід
10 4 2 3 7 4 5 3 1 2 5	2

Примітка

Масив $[4, 2, 3, 7, 4, 5, 3, 1, 2, 5]$, можна поділити на $[4, 2, 3, 7, 4, 5]$ та $[3, 1, 2, 5]$, кожен з котрих можна посортувати на комп'ютері Василька.

Як можна відсортувати перший масив було пояснено раніше.

Другий масив $([3, 1, 2, 5])$ можна відсортувати так:

1. додаємо число 3 у стек, зараз стек виглядає так $[3]$, а масив так $[]$;
2. додаємо число 1 у стек, зараз стек виглядає так $[3, 1]$, а масив так $[]$;
3. забираємо 1 зі стека та додаємо у масив, зараз стек виглядає так $[3]$, а масив так $[1]$;
4. додаємо число 2 у стек, зараз стек виглядає так $[3, 2]$, а масив так $[1]$;
5. забираємо 2 зі стека та додаємо у масив, зараз стек виглядає так $[3]$, а масив так $[1, 2]$;
6. забираємо 3 зі стека та додаємо у масив, зараз стек виглядає так $[]$, а масив так $[1, 2, 3]$;
7. додаємо число 5 у стек, зараз стек виглядає так $[5]$, а масив так $[1, 2, 3]$;
8. забираємо 5 зі стека та додаємо у масив, зараз стек виглядає так $[]$, а масив так $[1, 2, 3, 5]$.

Задача Е. Подарунки Васильку

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Недавно Васильку на день святого Миколая отримав подарунки. Одним з тих подарунків був масив a довжини n . Також він отримав робота, котрий працює на масиві.

Якщо робот має оперативну пам'ять розміру m , то він може запам'ятати не більше ніж m чисел. Робота можна запустити на масиві, і він з елементів цього масиву сконструює новий масив b , який буде лексикографічно* найменший з усіх можливих, використовуючи такі операції в довільному порядку:

- Якщо кількість вільних місць в оперативній пам'яті не менша від одиниці, тоді помістити в неї число на позиції i і перейти на позицію $i + 1$.
- Якщо оперативна пам'ять не пуста, то записати довільне число в масив b та видалити його з оперативної пам'яті.

Васильку стало цікаво, яким буде k -те число в масиві b , якщо робот має оперативну пам'ять розміру m та запустить його на підмасиві a від l до r . Йому дуже подобається масив a , тому він не хоче запускати робота на ньому і просить вас відповісти на q запитань, що назбиралися в нього за цей тиждень.

*Масив a є лексикографічно меншим за масив b , якщо в першому моменті, коли a відмінний від b , в масиві a значення менше ніж в масиві b .

Формат вхідних даних

Перший рядок містить два цілі числа n та q ($1 \leq n \leq 10^5, 1 \leq q \leq 2 \cdot 10^5$) — довжина масиву a і кількість запитань.

Другий рядок містить n цілих чисел $a_1, a_2, \dots, a_n$ ($0 \leq a_i \leq 10^9$) — значення масиву a .

Кожен з наступних q рядків містить чотири цілі числа l, r, m та k ($1 \leq m \leq n, 1 \leq l \leq r \leq n, 1 \leq k \leq r - l + 1$) — два кінці підмасиву, розмір оперативної пам'яті та індекс числа, котре цікаве Васильку.

Формат вихідних даних

Для кожного запитання в окремому рядку виписіть відповідь на нього.

Система оцінювання

1. (5 балів): для всіх запитань $m = n, l = 1, r = n$;
2. (9 балів): сума довжин всіх запитань не перевищує 10^6 ;
3. (9 балів): для всіх запитань $k + m \leq 30$;
4. (6 балів): для всіх запитань $m = n$ і $n, q \leq 10^3$;
5. (9 балів): для всіх запитань $m = n$ і $q \leq 10^5$;
6. (7 балів): для всіх запитань $m = n$;
7. (12 балів): $a_i \leq 30$;

8. (11 балів): $n \leq 10^3$;
9. (12 балів): $q \leq 10^5$;
10. (20 балів): без додаткових обмежень.

Приклад

стандартний ввід	стандартний вивід
10 5	1
2 3 5 3 1 2 7 4 2 6	2
2 6 2 3	2
4 8 3 2	4
1 10 4 6	2
1 10 2 7	
5 8 3 2	

Примітка

В першому запитанні найменший лексикографічний масив має вигляд 3 3 1 2 5, на третьому місці стоїть 1. Найменші лексикографічні масиви в наступних запитах будуть такими:

- 1 2 3 4 7
- 2 1 2 3 3 2 4 5 6 7
- 2 3 3 1 2 5 4 2 6 7
- 1 2 3 7

Задача А. Фотографія

Автор: Андрій Холод
Розробник: Павло Цікалишин
Розбір підготовано: Костянтин Денисов

Фотографія має розмір $n \times n$. Рамка товщини t додає смугу шириною t з кожного боку, тому зовнішній розмір усієї конструкції дорівнює $(n + 2t) \times (n + 2t)$. Площа рамки дорівнює різниці площ зовнішнього квадрата та фотографії.

Площа фотографії: $S_{\text{in}} = n^2$.

Площа зовнішнього квадрата: $S_{\text{out}} = (n + 2t)^2$.

Отже, площа рамки: $S = S_{\text{out}} - S_{\text{in}} = (n + 2t)^2 - n^2$.

Спростимо вираз: $(n + 2t)^2 - n^2 = n^2 + 4nt + 4t^2 - n^2 = 4t(n + t)$.

Тому відповідь: $S = 4t(n + t)$.

Складність. Час: $O(1)$, пам'ять: $O(1)$.

Зауваження. За $n \leq 10^9$ значення площ може бути порядку 10^{18} , тому слід використовувати 64-бітний цілий тип (наприклад, `long long`).

Задача В. Пікнік

Автор: Андрій Холод
Розробник: Павло Цікалишин
Розбір підготовано: Костянтин Денисов

Потрібно розбити масив $a_1, \dots, a_n$ на $m = k + 1$ **послідовних** частин так, щоб сума в кожній частині була однаковою, і використати всі шматки. Нехай $S = \sum_{i=1}^n a_i$. Тоді кожна частина має вагу $T = S/m$, отже обов'язково $m \mid S$.

Через те, що $n \leq 5000$, можна зробити дуже просту перевірку: перебрати всі можливі m від найбільшого до 1 і перевірити, чи можна нарізати на m частин сумою T жадібно зліва направо. Оскільки ми знаємо наперед суму в кожній частині, то далі перший відрізок визначається однозначно, за ним другий і так далі.

Перевірка для фіксованого m . Обчислюємо $T = S/m$ і йдемо по масиву, накопичуючи *cur* та підраховуючи кількість друзів *cnt*:

- $cur := cur + a_i$.
- якщо $cur = T$, то "закриваємо" частину: $cnt := cnt + 1$, $cur := 0$;
- якщо $cur > T$, то таке m неможливе (бо всі $a_i \geq 1$ і ми вже перебрали суму).

В кінці потрібно, щоб $cur = 0$ і $cnt = m$.

Складність. Перебираємо m не більше ніж n разів, кожна перевірка за $O(n)$, отже загалом $O(n^2)$. За $n \leq 5000$ це проходить. Пам'ять $O(n)$.

Задача С. Гра

Автор: Костянтин Денисов
Розробник: Павло Цікалишин
Розбір підготовано: Костянтин Денисов

Очевидно, що на відрізку $[l, r]$ треба ставити якнайбільші числа, тобто $a_i = m$ для всіх $l \leq i \leq r$. Тоді цей відрізок має найбільший потенціал, і далі наша задача — зробити так, щоб жоден інший підвідрізок не міг з ним зрівнятися за сумою, та ще й серед усіх таких масивів максимізувати суму невід'ємних елементів.

Окремий випадок $l = r$. Тут максимальна сума має бути рівно m і досягатися **лише** на позиції l . Тому поза l не можна ставити m , інакше одиночний підвідрізок з цієї позиції теж матиме суму m . Отже поза l найкраще ставити $m - 1$. Але треба ще зіпсувати всі підвідрізки, які включають l і ще щось: для цього достатньо розставити $-m$ на кожній другій позиції, починаючи від сусідів $l - 1$ та $l + 1$. Тоді будь-яке розширення відрізка за межі $[l, l]$ неминуче підхопить хоча б один $-m$, і сума стане строго меншою за m .

Випадок $l < r$.

Поза відрізком $[l, r]$ оптимальна конфігурація має максимально багато m , але такі m доводиться ставити через один, чергуючи з $-m$; інакше з'являється занадто великий підвідрізок поза $[l, r]$ або підвідрізок, що "приклеюється" до $[l, r]$.

Тому для **префікса** довжини $s = l - 1$ оптимально ставити (зліва направо):

- якщо s **парна**, то $m, -m, m, -m, \dots, m, -m, m - 1, -m$; доводиться ставити одну $m - 1$, щоб сума елементів була від'ємна, бо інакше буде декілька елементів з однаковою максимальною сумою.
- якщо s **непарна**, то $m, -m, m, -m, \dots, m, -m, -m$.

Для **суфікса** довжини $t = n - r$ робимо аналогічну (дзеркальну) побудову за парністю t .

У будь-якому допустимому масиві поза $[l, r]$ не можна ставити багато великих додатних підряд: кожен додатний m змушений мати поруч достатню "компенсацію" $-m$, інакше з'являється підвідрізок з занадто великою сумою, який конкурує з $[l, r]$. Отже m можна ставити не частіше, ніж через один, тобто шаблон $m, -m, m, -m, \dots$ дає максимальну кількість m . У парному випадку рівно один раз доводиться замінити m на $m - 1$, щоб уникнути рівності максимумів.

Складність. Побудова займає $O(n)$ часу і $O(1)$ додаткової пам'яті (окрім виводу).

Задача D. Василько і комп'ютер

Автор: Костянтин Денисов
Розробник: Павло Цікалишин
Розбір підготовано: Костянтин Денисов

Можна розв'язувати жадібно: робити перший підмасив якомога довшим, потім другий якомога довшим, і т.д.

Чому так можна: припустимо, існує оптимальне розбиття, де перший підмасив закінчується раніше, ніж це можливо. Нехай другий підмасив починається з елемента x . Якщо додавання x в кінець першого підмасиву **не ламає** можливість відсортувати його стеком, то ми можемо просто "перекласти" x з початку другого підмасиву в кінець першого. Другий підмасив при цьому стає коротшим, але він все одно відсортовуваний (бо ми лише забрали з нього перший елемент). Отже завжди існує оптимальне розбиття, в якому перший підмасив максимально довгий. Те саме міркування працює для кожного наступного підмасиву.

Тому достатньо йти зліва направо і додавати елементи в поточний підмасив, доки це можливо; коли стає неможливо — робимо розріз і починаємо новий.

Ми будуємо один підмасив зліва направо і симулюємо, як би працювало сортування стеком. Коли ми кладемо елементи в стек, іноді ми **змушені** викидати деякі зі стеку у вихідну послідовність: якщо прийшов новий елемент x , який більший за верхівку стека, то цю верхівку вже не можна тримати далі, бо інакше x опиниться над меншим елементом і ми ніколи не зможемо отримати відсортований вихід.

Тому правило таке: поки $top < x$, ми викидаємо top зі стека. Серед усіх викинутих значень запам'ятовуємо найбільше m . Коли прийдемо до випадку $top > x$, то в цьому випадку вже не можна класти верхівку стеку до масиву, бо тоді новий елемент x точно не буде стояти на своїй позиції.

m — це фактично найбільше число, яке ми вже **викинули** зі стека у цьому підмасиві. Тобто у фінальній відсортованій послідовності воно вже стоїть десь зліва, і ми вже не можемо поставити перед ним щось менше.

Якщо в якийсь момент приходить елемент x та виконується $x < m$, то все: поточний підмасив вже неможливо відсортувати стеком. Чому: число m уже "пішло" у вихід і мусить стояти **до** x у фінальній послідовності, але $x < m$, отже порядок буде неправильний і це вже не виправити ніякими майбутніми діями зі стеком.

У такому випадку ми **зобов'язані** розрізати масив перед x і почати новий підмасив: очищаємо стек, скидаємо m і збільшуємо лічильник частин.

В інших випадках ($top \geq m$) нескладно збагнути, що можна просто перекинути весь стек у кінцевий масив, щоб він був відсортованим.

Складність. Кожен елемент один раз заходить у стек і не більше одного разу виходить зі стека, тому час $O(n)$, пам'ять $O(n)$.

Задача Е. Подарунки Васильку

Автор: Костянтин Денисов

Розробник: Павло Цікалишин

Розбір підготовано: Костянтин Денисов

1) **Жадібний алгоритм в лоб (для одного запиту).** Нехай робот запускається на підмасиві $a[l..r]$, пам'ять розміру m . Підтримуємо мультимножину S (вміст пам'яті) і покажчик i , який спочатку дорівнює l .

Поки будуємо b :

- поки $|S| < m$ і $i \leq r$, додаємо a_i в S і $i := i + 1$;
- інакше беремо мінімальний елемент з S , записуємо в b , і видаляємо його з S .

На кожній позиції b ми беремо найменше число, яке взагалі доступне зараз (тобто вже завантажено в пам'ять). Якщо на деякому кроці не взяти мінімум, то перша позиція, де ми відхилились, стане більшою — лексикографічно гірше.

2) Яка відповідь на k -ту позицію. Позначимо $len = r - l + 1$ і $p = l + k + m - 2$ (це $l + (k + m - 1) - 1$).

- Якщо $p > r$, тобто $k + m - 1 > len$, то робот встиг завантажити весь підмасив, і відповідь — просто k -та порядкова статистика на $[l, r]$: $b_k = k$ -те найменше на $a[l..r]$.
- Якщо $p \leq r$, тоді відповідь така: $b_k = \min(a_p, k\text{-те найменше на } a[l..p])$.

Доведення формули індукцією по k (випадок $p \leq r$). Позначимо $T(k) = l + k + m - 2$ (тобто $T(k) = p$) і $X(k)$ — k -те найменше на $a[l..T(k)]$. Твердження: $b_k = \min(a_{T(k)}, X(k))$.

База $k = 1$. До першого вибору робот може завантажити рівно m елементів, тобто $a[l..l + m - 1]$. Тому b_1 — мінімум серед цих m елементів, а це рівно $\min(a_{l+m-1}, 1\text{-ше найменше на } a[l..l + m - 1])$, тобто формула виконується.

Перехід $k \rightarrow k + 1$. Припустимо, що після k кроків робот видав $b_1, \dots, b_k$ за формулою, і розглянемо крок $k + 1$.

До моменту вибору $(k + 1)$ -го елемента робот може завантажити $m + k$ перших елементів підмасиву, тобто рівно $a[l..T(k + 1)]$. На перших k кроках жадібний алгоритм завжди забирав мінімально можливе, отже він забрав рівно k найменших елементів серед $a[l..T(k + 1)]$, окрім можливої заміни на $a_{T(k)}$ там, де це було менше — але це якраз і означає, що на кроці $k + 1$ серед кандидатів лишається:

- або $a_{T(k+1)}$, якщо він виявився "дуже малим" і пробиває статистику,
- або $X(k + 1)$ як найменший серед тих, що лишилися після вилучення k найменших.

Отже жадібний вибір на кроці $k + 1$ дає $b_{k+1} = \min(a_{T(k+1)}, X(k + 1))$.

Таким чином твердження доведене для всіх k .

Випадок $p > r$. Тут $k + m - 1 > r - l + 1$, робот встигає завантажити весь $a[l..r]$, тому b є просто відсортованим $a[l..r]$, і b_k — k -те найменше на $[l, r]$.

Як відповісти на q запитів швидко.

Нам треба багато разів знаходити k -те найменше число на відрізку $[L, R]$. Для цього будуємо персистентне дерево відрізків по "значеннях де в **лиستках** зберігаємо, скільки разів кожне значення зустрілося на префіксі.

1) Щоб були листки $0..n$. Значення a_i великі, тому спочатку впорядковуємо всі елементи. Сортуємо пари (a_i, i) за a_i і замінюємо кожен a_i на номер його позиції у відсортованому списку (число від 0 до n). Масив d зберігає цей відсортований список: якщо листок має номер pos , то справжнє значення дорівнює $d[pos].first$.

2) Персистентні версії для префіксів. Будуємо корені $c[0], c[1], \dots, c[n]$. $c[i]$ — це дерево для префікса $[1..i]$.

У листку з номером pos зберігаємо число cnt : скільки разів на префіксі $[1..i]$ зустрілося значення з номером pos .

У внутрішній вершині зберігаємо суму cnt по її дітях (тобто скільки елементів префікса попадає в її діапазон листків).

Перехід від $c[i - 1]$ до $c[i]$: ми збільшуємо на 1 значення в листку, що відповідає a_i . Персистентність означає: копіюємо тільки вершини на шляху від кореня до цього листка, і в них оновлюємо суми; всі інші вершини спільні з попередньою версією. Це $O(\log n)$ нових вершин (функція `add`).

3) Як отримати відрізок $[L, R]$ з двох префіксів. Нехай ми хочемо працювати з підмасивом $[L, R]$. Для будь-якого листка (тобто конкретного значення) кількість його появ на $[L, R]$ дорівнює: $cnt_{[L,R]} = cnt_{[1,R]} - cnt_{[1,L-1]}$, тобто різниця того самого листка в деревах $c[R]$ і $c[L - 1]$.

4) Пошук k -го найменшого. Маємо дві версії дерева: одна для префікса $[1..R]$, друга для $[1..L-1]$. Розглянемо будь-яку вершину дерева, яка відповідає деякому діапазону значень $[x, y]$. У цій вершині зберігається число елементів префікса, що попали в $[x, y]$. Тоді кількість елементів на відрізку $[L, R]$ у цьому діапазоні дорівнює різниці цих чисел у двох версіях.

Щоб знайти k -те найменше, спускаємося від кореня до листка. Нехай зараз ми в діапазоні $[x, y]$ і $mid = (x + y)/2$. Дивимось, скільки елементів з $[L, R]$ лежить у лівій половині $[x, mid]$: це різниця значень у **лівих дочірніх вершинах** двох версій. Позначимо це число як $left$.

- Якщо $left \geq k$, то шукане значення лежить у лівій половині, і ми переходимо в лівих дітей обох версій.
- Інакше воно лежить у правій половині, тоді переходимо в правих дітей, а k замінюємо на $k - left$.

Так ми за $O(\log n)$ кроків дійдемо до листка, який і відповідає шуканому числу (потім переводимо його назад у початкове значення через відсортований список).

Складність. Побудова: n додавань, кожне $O(\log n)$, разом $O(n \log n)$. Кожен запит: $O(\log n)$. Пам'ять: $O(n \log n)$ вершин.