Содержание

У вас не получается читать/выводить данные?
Воспользуйтесь примерами (c++) (python).

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

Обратите внимание на `GNU C++` компиляторы с суффиксом `inc`.
Подни можно пользоваться дополнительной библиотекой (`optimization.h`).
То есть, использовать быстрый ввод-вывод: **пример про числа и строки**.
И быструю аллокацию памяти (ускоряет vector-set-map-весь-STL): пример.

Для тех, кто хочет разобраться, как всё это работает.
Короткая версия быстрого ввода-вывода (тык) и короткая версия аллокатора (тык).

Must have

Задача 26A. Persistent Array Off [0.1, 256]

Дан массив (вернее, первая, начальная его версия).

Нужно уметь отвечать на два запроса:

∘ $a_i[j] = x$ — создать из i-й версии новую, в которой j-й элемент равен x, а остальные элементы такие же, как в i-й версии.

∘ `get` $a_i[j]$ — сказать, чему равен j-й элемент в i-й версии.

Формат входных данных

Количество чисел в массиве N ($1 \leqslant N \leqslant 10^5$) и N элементов массива. Далее количество запросов M ($1 \leqslant M \leqslant 10^5$) и M запросов. Формат описания запросов можно посмотреть в примере. Если уже существует K версий, новая версия получает номер $K + 1$. И исходные, и новые элементы массива — целые числа от 0 до 10^9. Элементы в массиве нумеруются числами от 1 до N.

Формат выходных данных

На каждый запрос типа `get` вывести соответствующий элемент нужного массива.

Пример

stdin	stdout
6 1 2 3 4 5 6 11 create 1 6 10 create 2 5 8 create 1 5 30 get 1 6 get 1 5 get 2 6 get 2 5 get 3 6 get 3 5 get 4 6 get 4 5	6 5 10 5 10 8 6 30

Задача 26B. Persistent Array On [0.7, 256]

Дан массив (вернее, первая, начальная его версия).

Нужно уметь в *online* отвечать на два запроса:

- `create i j x` — создать из i-й версии новую, в которой j-й элемент равен x, а остальные элементы такие же, как в i-й версии.
- `get i j` — сказать, чему равен j-й элемент в i-й версии.

Это интерактивная задача. Запросы нужно обрабатывать в online, результат каждого запроса `flush`-ить до чтения следующего.

Формат входных данных

Количество чисел в массиве N ($1 \leqslant N \leqslant 10^5$) и N элементов массива. Далее количество запросов M ($1 \leqslant M \leqslant 10^5$) и M запросов. Формат описания запросов можно посмотреть в примере. Если уже существует K версий, новая версия получает номер $K + 1$. И исходные, и новые элементы массива — целые числа от 0 до 10^9. Элементы в массиве нумеруются числами от 1 до N.

Формат выходных данных

На каждый запрос типа `get` вывести соответствующий элемент нужного массива.

Пример

stdin	stdout
6 1 2 3 4 5 6 11 create 1 6 10 create 2 5 8 create 1 5 30 get 1 6 get 1 5 get 2 6 get 2 5 get 3 6 get 3 5 get 4 6 get 4 5	6 5 10 5 10 8 6 30

Подсказка по решению

Сделайте ровное красивое дерево. Глубины $\log_2 n$.

Замечание

Буфферизированные способы ввода, например, `readInt`, не подойдут, так как пытаются читать сразу «до конца файла». Пример корректного кода:

```
string query;
for (int i = 0; i < queryCnt; i++) {
    getline(cin, query);
    process(query);
    cout << flush; // cout << endl; тоже делает flush
}
```

Задача 26C. Персистентная очередь [0.2, 256]

Реализуйте персистентную очередь.

Формат входных данных

Первая строка содержит количество действий n ($1 \leqslant n \leqslant 200\,000$).
В строке номер $i + 1$ содержится описание действия i:

- `1 t m` — добавить в конец очереди номер t ($0 \leqslant t < i$) число m;
- `-1 t` — удалить из очереди номер t ($0 \leqslant t < i$) первый элемент.

В результате действия i, описанного в строке $i + 1$ создается очередь номер i.
Изначально имеется пустая очередь с номером ноль.
Все числа во входном файле целые, и помещаются в знаковый 32-битный тип.
Вас никогда не попросят удалять из пустой очереди.

Формат выходных данных

Для каждой операции удаления выведите удаленный элемент на отдельной строке.

Примеры

stdin	stdout
10 1 0 1 1 1 2 1 2 3 1 2 4 -1 3 -1 5 -1 6 -1 4 -1 8 -1 9	1 2 3 1 2 4

Подсказка по решению

Все запросы в offline ⇒ вы простое и быстрое решение за $\mathcal{O}(n)$ с деревом версий.
Ещё можно сдать *pairing* идею с лекции.
Переиспользовать онлайн-персистентность из задачи A тоже можно попробовать, но может не пройти по времени.

Задачи здорового человека

Задача 26D. CHM [0.2, 256]

Ваша задача — реализовать **Persistent Disjoint-Set-Union**. Что это значит?
Про **Disjoint-Set-Union**:
Изначально у вас есть n элементов. Нужно научиться отвечать на 2 типа запросов.

- `+ a b` — объединить множества, в которых лежат вершины a и b
- `? a b` — сказать, лежат ли вершины a и b сейчас в одном множестве

Про **Persistent**:
Теперь у нас будет несколько копий (версий) структуры данных **Disjoint-Set-Union**.
Запросы будут выглядеть так:

- `+ i a b` — запрос к i-й структуре, объединить множества, в которых лежат вершины a и b. При этом i-я структура остается не изменной, создается новая версия, ей присваивается новый номер (какой? читайте дальше)
- `? i a b` — запрос к i-й структуре, сказать, лежат ли вершины a и b сейчас в одном множестве

Формат входных данных

На первой строке 2 числа N ($1 \leqslant N \leqslant 10^5$) и K ($0 \leqslant K \leqslant 10^5$) — число элементов и число запросов. Изначально все элементы находятся в различных множествах. Эта изначальная копия (версия) структуры имеет номер 0.

Далее следуют K строк, на каждой описание очередного запроса. Формат запросов описан выше. Запросы нумеруются целыми числами от 1 до K.

Пусть j-й из K запросов имеет вид «`+ i a b`». Тогда новая версия получит номер j. Запросы вида «`? i a b`» не порождают новых структур.

Формат выходных данных

Для каждого запроса вида `? i a b` на отдельной строке нужно вывести `YES` или `NO`.

Пример

stdin	stdout
4 7 + 0 1 2 ? 0 1 2 ? 1 1 2 + 1 2 3 ? 4 3 1 ? 0 4 4 ? 4 1 4	NO YES YES YES NO

Подсказка по решению

Эту задачу Вы умеете решать и в offline, и в online. Offline точно пройдёт по времени.

Задача 26E. Вперёд! [0.5, 256]

Капрал Дукар любит раздавать приказы своей роте. Самый любимый его приказ — "Вперёд!". Капрал строит солдат в ряд и отдаёт некоторое количество приказов, каждый из них звучит так: "Рядовые с l_i по l_j — вперёд!"

Перед тем, как Дукар отдал первый приказ, солдаты были пронумерованы от 1 до n, слева направо. Услышав приказ "Рядовые с l_i по l_j — вперёд!", солдаты, стоящие на местах с l_i по l_j включительно, продвигаются в начало ряда, в том же порядке, в котором были.

Например, если в какой-то момент солдаты стоят в порядке $1, 3, 6, 2, 5, 4$, то после приказа "Рядовые с 2 по 3 — вперёд!", порядок будет таким: $3, 6, 1, 2, 5, 4$. А если потом Капрал вышлет вперёд солдат с 3 по 4, то порядок будет уже таким: $1, 2, 3, 6, 5, 4$.

Вам дана последовательность из приказов Капрала. Найдите порядок, в котором будут стоять солдаты после исполнения всех приказов.

Формат входных данных

В первой строке входного файла указаны числа n и m ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$) — число солдат и число приказов. Следующие m строк содержат приказы в виде двух целых чисел: l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant n$).

Формат выходных данных

Выведите в выходной файл n целых чисел — порядок, в котором будут стоять солдаты после исполнения всех приказов.

Пример

stdin	stdout
6 3 2 4 3 5 2 2	1 4 5 2 3 6

Подсказка по решению

Берём уже написанное дерево по неявному ключу, используем `Split`/`Merge`.

Задача 26F. Переворачивания [1, 256]

Учитель физкультуры школы с углубленным изучением предметов уже давно научился считать суммарный рост всех учеников, находящихся в ряду на позициях от l до r. Но дети играют с ним злую шутку. В некоторый момент дети на позициях с l по r меняются местами. Учитель заметил, что у детей не очень богатая фантазия, поэтому они всегда «переворачивают» этот отрезок, т. е. l меняется с r, $l+1$ меняется с $r-1$ и так далее. Но учитель решил не ругать детей за их хулиганство, а все равно посчитать суммарный рост на всех запланированных отрезках.

Формат входных данных

В первой строке записано два числа n и m ($1 \leqslant n, m \leqslant 200\,000$) — количество детей в ряду и количество событий, произошедших за все время. Во второй строке задано n натуральных чисел — рост каждого школьника в порядке следования в ряду. Рост детей не превосходит $2 \cdot 10^5$. Далее в m строках задано описание событий: три числа q, l, r в каждой строке ($0 \leqslant q \leqslant 1$, $1 \leqslant l \leqslant r \leqslant n$). Число q показывает тип события: 0 показывает необходимость посчитать и вывести суммарный рост школьников на отрезке $[l, r]$; 1 показывает то, что дети на отрезке $[l, r]$ «перевернули» свой отрезок. Все числа во входном файле целые.

Формат выходных данных

Для каждого события типа 0 выведите единственное число на отдельной строке — ответ на этот запрос.

Пример

stdin	stdout
5 6 1 2 3 4 5 0 1 5 0 2 4 1 2 4 0 1 3 0 4 5 0 3 5	15 9 8 7 10

Подсказка по решению

Отложенные операции. Высплитить отрезок, вмерджить обратно.

Задача 26G. Машенька и её интерес [0.2, 256]

Есть n мальчиков и девочка Маша. Изначально каждый мальчик стоит сам по себе и i-й мальчик с точки зрения Маши имеет интересность i. Девочка Маша хочет провести некоторый эксперимент, в течение которого каждый мальчик стоит в некоторой шеренге. Мальчики несговорчивые, участвовать в эксперименте не хотят, поэтому Маша собирается прибегнуть к математическому моделированию. Для этого ей нужно научиться быстро обрабатывать следующие запросы:

- `link(a, b)` – взять мальчиков с номерами a и b, если они стоят в разных шеренгах, то объединить шеренгу в одну: в начале шеренга мальчика a, затем шеренга мальчика b.
- `split(a, k)` – взять шеренгу, в которой стоит мальчик с номером a и разбить её на две: первые k мальчиков и все остальные. Если размер шеренги не больше k, ничего делать не нужно.
- `interest(a, x)` – сделать интересность мальчика a равной x (целое от 0 до 10^9).
- `sum(a)` – суммарная интересность мальчиков в шеренге, в которой стоит мальчик a.

Формат входных данных

В первой строке n ($1 \leqslant n \leqslant 100\,000$) – количество мальчиков и m ($1 \leqslant m \leqslant 250\,000$) – количество запросов. Далее m строк. Для понимания формата смотри пример. Мальчики нумеруются числами от 1 до n.

Формат выходных данных

Для каждого запроса «`sum`» на отдельной строке одно число – суммарная интересность.

Примеры

stdin	stdout
5 12 sum 1 interest 5 10 sum 5 interest 3 7 link 3 1 link 3 5 sum 1 interest 1 20 sum 1 split 1 2 sum 3 sum 5	1 10 18 37 27 10

Подсказка по решению

Теперь нужно ещё хранить отца и уметь подниматься от вершины до корня.

Для искателей острых ощущений

Задача 26H. Поднимайся и вращай [0.2, 256]

Изначально у вас есть n чисел $1, 2, \dots n$. Каждое живёт само по себе. Далее числа будут объединяться в массивы. Вам нужно реализовать структуру, данных, умеющую отвечать на несколько запросов, за $\mathcal{O}(\log n)$ каждый.

- `+ i j` – взять массивы, в которых живут числа i и j, и объединить их в один массив именно в таком порядке.
- `! i k` – взять массив, в котором живёт число i, и повернуть его на k влево.
- `- i k` – взять массив, в котором живёт число i, и отрезать первые k элементов. Получится два новых массива.

Гарантируется, что запросы корректны. В первом i и j живут в разных массивах, во втором и третьем, длина массива, содержащего i, строго больше k.

Формат входных данных

На первой строке число элементов n ($2 \leqslant n \leqslant 100\,000$) и число запросов m ($1 \leqslant m \leqslant 100\,000$). На следующих m строках сами запросы.

Формат выходных данных

После всех запросов нужно вывести получившиеся массивы. На первой строке выведите число массивов. Далее k массивов в формате «число элементов и сами элементы». Отсортируйте массив массивов перед выводом (массивы сравниваются лексикографически).

Пример

stdin	stdout
5 5 + 1 2 + 1 3 + 1 4 ! 3 1 - 2 1	3 1 2 3 3 4 1 1 5

Подсказка по решению

Вам нужно каждому элементу сопоставить `node*` и научиться подниматься от неё до корня соответствующего декартова дерева.

Задача 26I. Вставка ключевых значений [0.5, 256]

Вас наняла на работу компания MacroHard, чтобы вы разработали новую структуру данных для хранения целых ключевых значений.

Эта структура выглядит как массив A бесконечной длины, ячейки которого нумеруются с единицы. Изначально все ячейки пусты. Единственная операция, которую необходимо поддерживать — это операция `Insert(L,K)`, где L — положение в массиве, а K — некоторое положительное целое ключевое значение. Операция выполняется следующим образом:

- Если ячейка `A[L]` пуста, то присвоить `A[L] := K`.
- Иначе выполнить `Insert(L+1,A[L])`, а затем присвоить `A[L] := K`.

По заданной последовательности из N целых чисел $L_1, L_2, \ldots, L_N$ вам необходимо вывести содержимое этого массива после выполнения следующей последовательности операций:

```
Insert(L_1, 1)
Insert(L_2, 2)
...
Insert(L_N, N)
```

Формат входных данных

В первой строке входного файла содержится N (число операций `Insert`) и M (максимальный номер позиции, которую можно использовать в операции `Insert`). ($1 \leqslant N, M \leqslant 131\,072$).

В следующей строке даны N целых чисел L_i ($1 \leqslant L_i \leqslant M$).

Формат выходных данных

Выведите содержимое массива после выполнения данной последовательности операций `Insert`. На первой строке выведите W — номер последней несвободной позиции в массиве. Далее выведите W целых чисел — $A[1], A[2], \ldots, A[W]$. Для пустых ячеек выводите нули.

Пример

stdin	stdout
5 4 3 3 4 1 3	6 4 0 5 2 3 1

Подсказка по решению

У жюри есть простое решение одним декартовым деревом.

Для мастеров AI

Правила.
Это блок задач про промтинг. Пользоваться можно только бесплатными версиями ИИ. В шапке исходника указывать последовательность промтов, и сайты, куда они были отправлены. Если в процессе использования ИИ вы получили какие-то важные идеи для решения, это тоже часть решения. Если с помощью ИИ вы написали генератор тестов или стресс-тест, это тоже следует указывать. Задокументируйте, пожалуйста, проделанную работу в шапке отосланного решения. Если это нужно, вы можете писать часть кода сами.

Задача 26J. Переворачивания [2, 4]

Учитель физкультуры школы с углубленным изучением предметов уже давно научился считать суммарный рост всех учеников, находящихся в ряду на позициях от l до r. Но дети играют с ним злую шутку. В некоторый момент дети на позициях с l по r меняются местами. Учитель заметил, что у детей не очень богатая фантазия, поэтому они всегда «переворачивают» этот отрезок, т. е. l меняется с r, $l+1$ меняется с $r-1$ и так далее. Но учитель решил не ругать детей за их хулиганство, а все равно посчитать суммарный рост на всех запланированных отрезках.

Формат входных данных

В первой строке записано два числа n и m ($1 \leqslant n, m \leqslant 200\,000$) — количество детей в ряду и количество событий, произошедших за все время. Во второй строке задано n натуральных чисел — рост каждого школьника в порядке следования в ряду. Рост детей не превосходит $2 \cdot 10^5$. Далее в m строках задано описание событий: три числа q, l, r в каждой строке ($0 \leqslant q \leqslant 1$, $1 \leqslant l \leqslant r \leqslant n$). Число q показывает тип события: 0 показывает необходимость посчитать и вывести суммарный рост школьников на отрезке $[l, r]$; 1 показывает то, что дети на отрезке $[l, r]$ «перевернули» свой отрезок. Все числа во входном файле целые.

Формат выходных данных

Для каждого события типа 0 выведите единственное число на отдельной строке — ответ на этот запрос.

Пример

stdin	stdout
5 6 1 2 3 4 5 0 1 5 0 2 4 1 2 4 0 1 3 0 4 5 0 3 5	15 9 8 7 10

Подсказка по решению

Это задача про «взять добротный квадрат» и попросить gpt пооптимизировать, используя mm256i и unroll-loops. Не забудьте включить AVX прагмами.

Если придумаете, как ещё улучшить, прекрасно. Авторский n^2 работает в 4.5 раз дольше авторского $n \log n$ на $n = 200\,000$.