1A. Сортировка

2 секунды, 256 мегабайт

Дан массив целых чисел. Ваша задача — отсортировать его в порядке неубывания.

В данной задаче запрещено пользоваться стандартной библиотекой языка (функциями `std::sort`, `std::stable_sort` в C++ и их аналогами в других языках программирования).

Входные данные
В первой строке входного файла содержится число N ($1 \le N \le 100\,000$) — количество элементов в массиве.

Во второй строке находятся N целых чисел, по модулю не превосходящих 10^9.

Выходные данные
В выходной файл надо вывести этот же массив, отсортированный по неубыванию.

входные данные
10 1 8 2 1 4 7 3 2 3 6
выходные данные
1 1 2 2 3 3 4 6 7 8

1B. K-я порядковая статистика

2 секунды, 256 мегабайт

Дан массив, содержащий n целых чисел. Вам нужно найти в этом массиве k-й по счету минимальный элемент ($k \in [0, n-1]$), то есть элемент, который после сортировки массива по неубыванию окажется на k-м месте от начала массива (индексация элементов начинается с нуля).

Элементы массива a_i задаются при помощи псевдослучайного генератора по формуле:
$a_i = (1\,103\,515\,245 \cdot a_{i-1} + 12\,345) \bmod 2^{31}$, то есть все элементы массива задаются одним начальным значением a_0.

В данной задаче запрещено пользоваться стандартной библиотекой языка (функцией `std::nth_element` в C++ и их аналогами в других языках программирования).

Входные данные
Программа получает на вход три целых числа n, a_0 и k ($1 \le n \le 2 \cdot 10^7$, $0 \le a_0 < 2^{31}$, $0 \le k < n$) — количество элементов в массиве, значение первого элемента массива и индекс искомого элемента, соответственно.

Выходные данные
Программа должна вывести одно целое число — k-й минимум в данной последовательности.

входные данные
5 123456789 2
выходные данные
850994577

В примере из условия сгенерированный массив имеет вид $[123\,456\,789,\ 231\,794\,730,\ 1\,126\,946\,331,\ 1\,757\,975\,480,\ 850\,994\,577]$.

1C. Анти-QuickSort

1 секунда, 256 мегабайт

Для сортировки последовательности чисел широко используется быстрая сортировка — QuickSort. Далее приведена программа, которая сортирует массив a, используя этот алгоритм.

```
void quick_sort(int left, int right) {
    int i = left;
    int j = right;
    int key = a[(left + right) / 2];
    while (i <= j) {
        while (a[i] < key) {
            i++;
        }
        while (key < a[j]) {
            j--;
        }
        if (i <= j) {
            swap(a[i], a[j]);
            i++;
            j--;
        }
    }
    if (left < j) {
        quick_sort(left, j);
    }
    if (i < right) {
        quick_sort(i, right);
    }
}

...

quick_sort(0, n - 1);
```

Хотя QuickSort является самой быстрой сортировкой в среднем, существуют тесты, на которых она работает очень долго. Оценивать время работы алгоритма будем количеством сравнений с элементами массива (то есть суммарным количеством сравнений в первом и втором `while`). Требуется написать программу, генерирующую тест, на котором быстрая сортировка сделает наибольшее число таких сравнений.

Входные данные
В первой строке находится единственное число N ($1 \le N \le 70\,000$).

Выходные данные
Вывести перестановку чисел от 1 до N, на которой быстрая сортировка выполнит максимальное число сравнений. Если таких перестановок несколько, вывести любую из них.

входные данные
1
выходные данные
1

входные данные
3
выходные данные
1 3 2

1D. Количество инверсий

1 секунда, 256 мегабайт

Напишите программу, которая для заданного массива $A = \langle a_1, a_2, \ldots, a_n \rangle$ находит количество пар (i, j) таких, что $i < j$ и $a_i > a_j$.

Входные данные
Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 100\,000$) — количество элементов массива.

Вторая строка содержит n попарно различных элементов массива A — целых неотрицательных чисел, не превосходящих 10^6.

Выходные данные
В выходной файл выведите одно число — ответ на задачу.

входные данные
5 11 6 31 28 18
выходные данные
4

1E. Отрезки с большой суммой

1 секунда, 256 мегабайт

Вам задан массив $a_1, a_2, \ldots, a_n$. Найдите количество его отрезков с суммой не меньше k, то есть число пар (l, r), таких что $l \leq r$ и $\left(\sum_{i=l}^{r} a_i\right) \geq k$.

Входные данные
На первой строке заданы числа n и k ($1 \leq n \leq 2 \cdot 10^5; -10^{15} \leq k \leq 10^{15}$)

На второй строке задан массив a ($-10^9 \leq a_i \leq 10^9$).

Выходные данные
Выведите одно число — количество искомых отрезков.

входные данные
4 7 -1 3 5 2
выходные данные
5

1F. Приоритетная очередь с удалением

1 секунда, 256 мегабайт

Обратите внимание, что в данной задаче элементы кучи нумеруются числами от 1 до N.

Требуется реализовать с помощью кучи приоритетную очередь, поддерживающую три операции: добавить элемент, извлечь максимальный элемент и удалить заданный произвольный элемент по его индексу в куче.

В этой задаче в структуре могут храниться **одинаковые элементы**. Введем дополнительные правила на процедуры `sift_up` и `sift_down` для однозначности.

1. Процедуры просеивания не должны перемещать элемент дальше, чем это действительно необходимо. Например, если $A[i] = A[2i]$, то вызов *sift_up(2i)* не должен менять местами эти два элемента (хотя их обмен и не испортит кучу, он бесполезен).
2. Если при просеивании вниз можно перемещать рассматриваемый элемент как влево вниз, так и вправо вниз (это бывает, когда он меньше двух равных дочерних), то следует выбирать направление **влево**.

В данной задаче запрещено пользоваться стандартной библиотекой языка (структурами данных `std::priority_queue` и `std::set` в C++ и их аналогами в других языках программирования).

Входные данные
В первой строке вводятся два числа — максимальный размер приоритетной очереди N и количество запросов M ($1 \leq N, M \leq 10^5$).

Далее идут M строк, в каждой строке — по одному запросу.

Первое число в запросе задаёт его тип, остальные числа (если есть) — параметры запроса:

- Запрос первого типа: извлечь максимальный элемент (без параметров);
- Запрос второго типа: добавить данный элемент в очередь. Запрос имеет один параметр p — число из диапазона $[-10^9, 10^9]$;
- Запрос третьего типа: удалить произвольный элемент. Запрос имеет один параметр i — индекс элемента в куче.

Выходные данные
В ответ на запрос первого типа следует вывести:

- Если извлекать было нечего (очередь пуста), вывести -1;
- Иначе вывести два числа: первое — индекс конечного положения элемента после его просеивания (если же удален был последний элемент и просеивать осталось нечего, вывести 0); второе — значение извлеченного элемента.

В ответ на запрос второго типа следует вывести:

- Если добавить нельзя (нет места, поскольку в очереди уже N элементов), вывести -1 (при этом куча не должна измениться);
- Иначе — индекс добавленного элемента.

В ответ на запрос третьего типа следует вывести:

- Если элемента с таким индексом нет и удаление невозможно, вывести -1 (при этом куча не должна измениться);
- Иначе — значение удаленного элемента.

Кроме того, после выполнения всех запросов требуется вывести кучу в её конечном состоянии.

входные данные
4 10 1 2 9 2 4 2 9 2 9 2 7 1 3 4 2 1 3 3
выходные данные
-1 1 2 3 2 -1 2 9 -1 4 9 9 4 1

1G. Мёд для Михаила

2 секунды, 256 мегабайт

Медведь Михаил очень любит мёд. Вот беда — мёд у него в берлоге закончился, поэтому Михаил приехал к любимой бабушке, владеющей медовой фабрикой.

Бабушка очень добрая, она разрешила Мише брать столько мёда, сколько он хочет. Всего у бабушки есть n бочек с мёдом. В i-й бочке содержится a_i литров мёда. Миша взял с собой m ведер, каждое из которых вмещает p литров мёда.

Михаил очень хорошо разбирается в мёде и понимает, что если смешивать мёд из разных бочек, то он будет невкусный. Поэтому Миша не будет наливать мёд из нескольких бочек в одно ведро. Теперь Миша хочет узнать, какое максимальное количество мёда он сможет увезти с собой.

Входные данные
Первая строка содержит три целых числа n, m и p ($1 \le n, m \le 10^5$, $1 \le p \le 10^9$) — количество бочек с мёдом, количество ведер у Михаила и вместимость ведер, соответственно.

Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_i$ ($1 \le a_i \le 10^9$) — количество литров мёда в каждой из бочек.

Выходные данные
Выведите одно число — максимальное количество литров мёда, которое сможет унести Михаил.

входные данные
3 2 4 2 3 4
выходные данные
7

входные данные
3 2 4 1 2 7
выходные данные
7

В первом примере Миша нальет 3 литра мёда из второй бочки в одно ведро и 4 литра из третьей бочки во второе ведро.

Во втором примере Мише следует налить 3 и 4 литра мёда в два ведра из третьей бочки.

1H. Цифровая сортировка

1 секунда, 256 мегабайт

Даны n строк, требуется вывести их порядок после k фаз цифровой сортировки.

Входные данные
Первая строка содержит три целых числа n, m и k ($1 \le n \le 1\,000$, $1 \le k \le m \le 1\,000$) — количество строк, длина строк и количество фаз цифровой сортировки.

Каждая из следующих n строк содержит строку, состоящую из m символов.

Выходные данные
Выведите строки в том порядке, в котором они будут находиться после k фаз цифровой сортировки.

входные данные
3 3 1 bbb aba baa
выходные данные
aba baa bbb

входные данные
3 3 2 bbb aba baa
выходные данные
baa aba bbb

входные данные
3 3 3 bbb aba baa
выходные данные
aba baa bbb

2A. Проверьте сортирующую сеть

2 секунды, 256 мегабайт

Проверьте, является ли данная сеть из n проводов сортирующей.

Входные данные
Первая строка содержит три целых числа n, m и k ($1 \le n \le 15$, $0 \le m, k \le 150$) — количество проводов, количество компараторов и количество слоев в сети, соответственно.

Каждая из следующих k строк содержит описание очередного слоя сети.

Описание слоя начинается с целого числа r_i — количества компараторов в слое. Далее следуют r пар целых чисел (x_{ij}, y_{ij}) ($1 \le x_{ij}, y_{ij} \le n$, $x_{ij} \ne y_{ij}$) — номера проводов, которые сравнивает очередной компаратор.

Гарантируется, что $\sum r_i = m$.

Выходные данные
Выведите слово «`Yes`» (без кавычек), если сеть является сортирующей, либо слово «`No`» (без кавычек) в противном случае.

входные данные
4 6 3 2 1 2 3 4 2 1 4 2 3 2 1 2 3 4
выходные данные
Yes

входные данные
3 2 2 1 2 1 1 3 1
выходные данные
No

2B. Постройте сортирующую сеть

2 секунды, 256 мегабайт

Постройте сортирующую сеть для n проводов.

Входные данные
Единственная строка содержит одно целое число n ($1 \le n \le 16$) — количество проводов в сети.

Выходные данные
В первой строке выведите три целых числа n, m и k — количество проводов, количество компараторов и количество слоев, соответственно.

В каждой из следующих k строк выведите описание очередного слоя сортирующей сети.

Описание слоя должно начинаться с целого числа r_i — количества компараторов в слое. Далее должны следовать r_i пар чисел (x_{ij}, y_{ij}) ($1 \le x_{ij}, y_{ij} \le n$, $x_{ij} \ne y_{ij}$) — номера проводов, соединенных компаратором. Внутри одного слоя номера всех проводов должны быть попарно различны. Количество слоев k не должно превышать 12.

входные данные
4
выходные данные
4 6 3 2 1 2 3 4 2 1 4 2 3 2 1 2 3 4

входные данные
1
выходные данные
1 0 0

3A. Двоичный поиск

1 секунда, 256 мегабайт

Реализуйте двоичный поиск в массиве.

В данной задаче запрещено пользоваться стандартной библиотекой языка (функциями `std::binary_search`, `std::lower_bound`, `std::upper_bound` в C++ и их аналогами в других языках программирования).

Входные данные

В первой строке входных данных содержатся натуральные числа N и K ($1 \le N, K \le 100\,000$).

Во второй строке задаются N элементов первого массива.

В в третьей строке — K элементов второго массива.

Элементы обоих массивов — целые числа, каждое из которых по модулю не превосходит 10^9.

Выходные данные

Требуется для каждого из K чисел вывести в отдельную строку «`YES`», если это число встречается в первом массиве, и «`NO`» в противном случае.

входные данные
10 10 1 61 126 217 2876 6127 39162 98126 712687 1000000000 100 6127 1 61 200 -10000 1 217 10000 1000000000
выходные данные
NO YES YES YES NO NO YES YES NO YES

3B. Левый и правый двоичный поиск

1 секунда, 256 мегабайт

Дано два списка чисел, числа в первом списке упорядочены по неубыванию. Для каждого числа из второго списка определите номер первого и последнего появления этого числа в первом списке.

В данной задаче запрещено пользоваться стандартной библиотекой языка (функциями `std::binary_search`, `std::lower_bound`, `std::upper_bound` в C++ и их аналогами в других языках программирования).

Входные данные

В первой строке входных данных записано два числа N и M ($1 \le N, M \le 20\,000$).

Во второй строке записано N упорядоченных по неубыванию целых чисел — элементы первого списка.

В третьей строке записаны M целых неотрицательных чисел — элементы второго списка. Все числа в списках — целые 32-битные знаковые.

Выходные данные

Программа должна вывести M строчек.

Для каждого числа из второго списка нужно вывести номер его первого и последнего вхождения в первый список.

Нумерация начинается с единицы. Если число не входит в первый список, нужно вывести одно число 0.

входные данные
10 5 1 1 3 3 5 7 9 18 18 57 57 3 9 1 179
выходные данные
10 10 3 4 7 7 1 2 0

3C. Гирлянда

1 секунда, 256 мегабайт

Гирлянда состоит из n лампочек на общем проводе. Один ее конец закреплен на заданной высоте A, то есть $h_1 = A$. Благодаря силе тяжести, гирлянда прогибается: высота каждой неконцевой лампы на 1 меньше, чем средняя высота ее соседей, то есть $h_i = \frac{h_{i-1}+h_{i+1}}{2} - 1$ для всех $1 < i < N$. Требуется найти минимальную высоту второго конца B ($h_N = B$) при условии, что ни одна из лампочек не должна лежать на земле, то есть все h_i должны быть строго положительными.

Входные данные

Единственная строка содержит два числа N и A ($3 \le N \le 1\,000$, $10 \le A \le 1\,000$), где N — целое, а A — вещественное.

Выходные данные

Выведите одно вещественное число B с двумя знаками после запятой.

входные данные
8 15
выходные данные
9.75

входные данные
692 532.81
выходные данные
446113.34

3D. K-Best

2 секунды, 256 мегабайт

У Демьяны есть n драгоценностей. Каждая из драгоценностей имеет ценность v_i и вес w_i. С тех пор, как её мужа Джонни уволили в связи с последним финансовым кризисом, Демьяна решила продать несколько драгоценностей. Для себя она решила оставить лишь k лучших. Лучших в смысле максимизации достаточно специфического выражения: пусть она оставила для себя драгоценности номер $i_1, i_2, \ldots, i_k$, тогда максимальной должна быть величина

$$\frac{\sum_{j=1}^{k} v_{i_j}}{\sum_{j=1}^{k} w_{i_j}}$$

Помогите Демьяне выбрать k драгоценностей требуемым образом.

Входные данные
На первой строке n и k ($1 \le k \le n \le 100\,000$).

Следующие n строк содержат пары целых чисел v_i, w_i ($0 \le v_i \le 10^6, 1 \le w_i \le 10^6$, сумма всех v_i не превосходит 10^7, сумма всех w_i также не превосходит 10^7).

Выходные данные
Выведите k различных чисел от 1 до n — номера драгоценностей. Драгоценности нумеруются в том порядке, в котором перечислены во входных данных. Если есть несколько оптимальных ответов, выведите любой.

входные данные
3 2 1 1 1 2 1 3
выходные данные
1 2

3E. All your base are belong to us

2 секунды, 512 мегабайт

В 2021 году нашей эры началась война. Враг захватил все наши базы. Чтобы снова завладеть нашими базами, было решено обосновать штаб-квартиру. При том нужно расположить штаб-квартиру в таком месте, чтобы все базы были не очень далеко от нее. Поэтому мы решили выбрать такое место, чтобы минимизировать суммарное расстояние от штаб-квартиры до K самых далеких от нее баз.

Все базы являются точками на плоскости. Мы можем расположить штаб-квартиру в любой точке плоскости (даже если у этой точки не целые координаты).

От вас требуется определить оптимальное расположение штаб-квартиры.

Входные данные
В первой строке записаны два числа N и K. Число N обозначает количество баз $(1 \le N \le 200)$. Число K отвечает за то, расстояние до скольки баз мы учитываем в сумме ($1 \le K \le N$).

В каждой из следующих N строк записаны целые числа x_i и y_i — координаты i-й базы ($-1000 \le x_i, y_i \le 1000$).

Выходные данные
Выведите минимальную сумму расстояний от штаб-квартиры до K самых далеких от нее баз. Ответ считается корректным, если абсолютная или относительная погрешность не превосходит 10^{-4}.

входные данные
3 1 0 1 1 0 1 1
выходные данные
0.70711

входные данные
6 3 1 1 2 1 3 2 5 3 8 5 13 8
выходные данные
17.50426

входные данные
9 3 573 -50 -256 158 -751 14 314 207 293 567 59 -340 -243 -22 -268 432 -91 -192
выходные данные
1841.20904

3F. Провода

1 секунда, 256 мегабайт

Дано N отрезков провода длиной $L_1, L_2, \ldots, L_N$ сантиметров. Требуется с помощью разрезания получить из них K равных отрезков как можно большей длины, выражающейся целым числом сантиметров. Если нельзя получить K отрезков длиной даже 1 см, вывести 0.

Входные данные
В первой строке находятся числа N и K ($1 \le N, K \le 10\,000$).

В следующих N строках — $L_1, L_2, \ldots, L_N$, по одному числу в строке ($100 \le L_i \le 10\,000\,000$). Все числа целые.

Выходные данные
Вывести одно число — полученную длину отрезков.

входные данные
4 11 802 743 457 539
выходные данные
200

3G. Среди Них

2 s., 256 MB

Сережа играет в популярную компьютерную игру.

Действие в ней происходит на космическом корабле. На нем есть n предателей и m членов экипажа (всего $n + m$ игроков). В каждом раунде каждый из присутствующих на корабле предателей убивает одного члена экипажа (конечно же, никакие два предателя не могут убить одного члена экипажа). Если в начале раунда членов экипажа меньше, чем предателей, то некоторые предатели могут никого и не убить. После этого, если на корабле останется хотя бы один член экипажа, происходит общее собрание членов экипажа, и они выбрасывают одного из предателей в открытый космос, в результате чего предатель умирает и не может в дальнейшем убивать. Если все предатели или все члены экипажа оказываются мертвы, игра заканчивается, иначе начинается новый раунд.

Сереже стало интересно, кто же останется в конце игры: предатели или члены экипажа? Также он задался вопросом: через сколько раундов закончится игра? Помогите ему узнать это!

Входные данные
В первой строке записано целое число t ($1 \le t \le 10^5$) — количество тестовых случаев. Далее следуют t строк, в каждой из которых содержится описание каждого тестового случая.

Описание тестового случая состоит из двух записанных через пробел целых чисел n и m ($1 \le n \le 2 \cdot 10^9, 1 \le m \le 10^{18}$) — количество предателей и членов экипажа, соответственно.

Выходные данные

Для каждого из тестовых случаев выведите ответ в описанном ниже формате.

В первой строке выведите «Impostors» (без кавычек), если в конце игры останутся предатели, и «Crewmates» (без кавычек), если останутся члены экипажа.

Во второй строке выведите одно число — количество раундов, произошедших до конца игры.

входные данные
2 2 10 2 3
выходные данные
Crewmates 2 Impostors 2

Рассмотрим первый пример. После первого раунда будут убиты двое членов экипажа, после чего один из двух предателей будет отправлен в открытый космос. Далее начнется второй раунд, в котором будет убит еще один член экипажа, а последний предатель отправится за борт. В итоге после двух раундов семеро членов экипажа останутся в живых.

Рассмотрим второй пример. После первого раунда на борту космического корабля останется один член экипажа и один предатель. Во втором раунде член экипажа будет убит, в результате чего общее собрание не состоится, а значит предатель останется жив.

3H. Поле для крикета

1 секунда, 256 мегабайт

Жил-был жадный Король. Он приказал своему главному Архитектору построить поле для королевского крикета в парке. Король был таким жадным, что не послушал предложение своего Архитектора построить поле прямо в центре парка и окружить его живописным бордюром деревьев, специально посаженных вокруг. Вместо этого он приказал не срубать деревья и не сажать новых, но построить самое большое поле для крикета, какое только можно. Если Король обнаружит, что Архитектор посмел тронуть даже единственное дерево в парке или спроектировал меньшее поле, чем было возможно, Архитектор лишится головы. Более того, он потребовал от Архитектора представить план поля, где указаны его точное положение и размер.

Ваша задача - помочь бедному Архитектору сохранить голову, написав программу, которая найдёт максимальный размер поля для крикета и его положение внутри парка, удовлетворяющие требованиям Короля.

Задача слегка упрощена тем, что парк Короля имеет прямоугольную форму и расположен на плоской поверхности. Более того, границы парка параллельны направлениям север — юг и восток — запад. В то же время игра в королевский крикет всегда происходит на квадратном поле, границы которого также параллельны направлениям север — юг и восток — запад. Архитектор уже сопоставил парку прямоугольную декартову систему координат и точно определил координаты каждого дерева. Оси этой системы координат, конечно, параллельны направлениям север — юг и восток — запад. Юго-западный угол парка имеет координаты $(0, 0)$, а северо-восточный — координаты (W, H), где W и H — длина и ширина парка соответственно.

В этой задаче вы можете пренебречь диаметром деревьев. Деревья не могут находиться внутри поля для крикета, но могут располагаться на его сторонах. Поле для крикета может также касаться границы парка, но не должно лежать вне парка.

Входные данные

Первая строка содержит три целых числа, N, W и H, разделённых пробелами: N — число деревьев в парке, W и H — длина и ширина парка соответственно.

Следующие N строк описывают координаты деревьев в парке. Каждая строка содержит два целых числа x_i и y_i, разделённых пробелом и представляющих собой координаты i-го дерева. Все деревья имеют различные координаты.

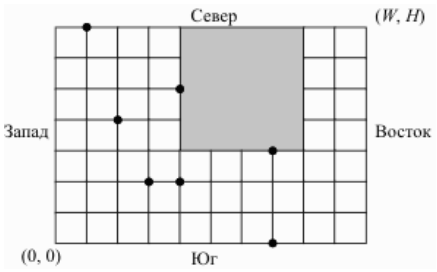
Ограничения:
$0 \le N \le 100, 1 \le W, H \le 10^4, 0 \le x_i \le W, 0 \le y_i \le H$.

Выходные данные

Вывести через пробел три целых числа, P, Q и L, где (P, Q) — координаты юго-западного угла поля для крикета, L — длина его сторон. Если существует несколько возможных положений поля максимального размера, вывести любое.

входные данные
7 10 7 3 2 4 2 7 0 7 3 4 5 2 4 1 7
выходные данные
4 3 4

Рисунок к первому примеру:



4A. Простой стек

1 секунда, 256 мегабайт

Реализуйте структуру данных "стек". Напишите программу, содержащую описание стека и моделирующую работу стека, реализовав все указанные здесь методы. Программа считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения каждой команды программа должна вывести одну строчку. Возможные команды для программы:

push n Добавить в стек число n (значение n задается после команды). Программа должна вывести ok.

pop Удалить из стека последний элемент. Программа должна вывести его значение.

back Программа должна вывести значение последнего элемента, не удаляя его из стека.

size Программа должна вывести количество элементов в стеке.

clear Программа должна очистить стек и вывести ok.

exit Программа должна вывести bye и завершить работу.

В данной задаче запрещено пользоваться стандартной библиотекой языка (классами `std::stack`, `std::vector`, `std::queue`, `std::deque` в C++ и их аналогами в других языках программирования).

Входные данные

Команды управления стеком вводятся в описанном ранее формате по 1 на строке.

Гарантируется, что набор входных команд удовлетворяет следующим требованиям: максимальное количество элементов в стеке в любой момент не превосходит 100, все команды pop и $back$ корректны, то есть при их исполнении в стеке содержится хотя бы один элемент.

Выходные данные

Требуется вывести протокол работы со стеком, по 1 сообщению в строке

входные данные
push 1 back exit
выходные данные
ok 1 bye

4B. Простая очередь

1 секунда, 256 мегабайт

Реализуйте структуру данных "очередь". Напишите программу, содержащую описание очереди и моделирующую работу очереди, реализовав все указанные здесь методы. Программа считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения каждой команды программа должна вывести одну строчку. Возможные команды для программы:

push n Добавить в очередь число n (значение n задается после команды). Программа должна вывести ok.

pop Удалить из очереди первый элемент. Программа должна вывести его значение.

front Программа должна вывести значение первого элемента, не удаляя его из очереди.

size Программа должна вывести количество элементов в очереди.

clear Программа должна очистить очередь и вывести ok.

exit Программа должна вывести bye и завершить работу.

Гарантируется, что набор входных команд удовлетворяет следующим требованиям: максимальное количество элементов в очереди в любой момент не превосходит 100, все команды pop и $front$ корректны, то есть при их исполнении в очереди содержится хотя бы один элемент.

В данной задаче запрещено пользоваться стандартной библиотекой языка (классами `std::stack`, `std::vector`, `std::queue`, `std::deque` в C++ и их аналогами в других языках программирования).

Входные данные

Вводятся команды управления очередью, по одной на строке

Выходные данные

Требуется вывести протокол работы с очередью, по одному сообщению на строке

входные данные
push 1 front exit
выходные данные
ok 1 bye

4C. Значение арифметического выражения

1 секунда, 256 мегабайт

Задано числовое выражение. Необходимо вычислить его значение или установить, что оно содержит ошибку. В выражении могут встречаться знаки сложения, вычитания, умножения, скобки и пробелы (пробелов внутри чисел быть не должно). Приоритет операций стандартный. Все числа в выражении целые и по модулю не превосходят $2 \cdot 10^9$. Также гарантируется, что все промежуточные вычисления умещаются в этот тип.

Входные данные

В первой строке вводится выражение. Его длина не превосходит 100 знаков. После выражения идет переход на новую строчку.

Выходные данные

Выведите значение этого выражения или слово «WRONG», если значение не определено.

входные данные
1+(2*2 - 3)
выходные данные
2

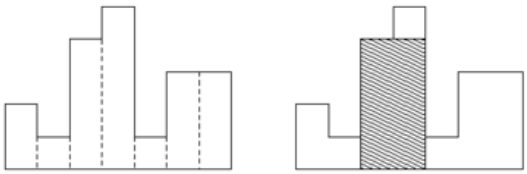
входные данные
1+a+1
выходные данные
WRONG

входные данные
1 1 + 2
выходные данные
WRONG

4D. Гистограмма

1 секунда, 256 мегабайт

Гистограмма является многоугольником, сформированным из последовательности прямоугольников, выровненных на общей базовой линии. Прямоугольники имеют равную ширину, но могут иметь различные высоты. Например, фигура слева показывает гистограмму, которая состоит из прямоугольников с высотами 2, 1, 4, 5, 1, 3, 3. Все прямоугольники на этом рисунке имеют ширину, равную 1.



Обычно гистограммы используются для представления дискретных распределений, например, частоты символов в текстах. Отметьте, что порядок прямоугольников очень важен. Вычислите область самого большого прямоугольника в гистограмме, который также находится на общей базовой линии. На рисунке справа заштрихованная фигура является самым большим выровненным прямоугольником на изображенной гистограмме.

Входные данные

В первой строке входного файла записано число N $(0 < N \leq 10^6)$ — количество прямоугольников гистограммы. Затем следует N целых чисел $h_1 \ldots h_n$, где $0 \leq h_i \leq 10^9$. Эти числа обозначают высоты прямоугольников гистограммы слева направо. Ширина каждого прямоугольника равна 1

входные данные
7 2 1 4 5 1 3 3
выходные данные
8

4E. Колонизаторы - 2

1 s., 256 MB

Недавно вышла новая версия *легендарной* настольной игры «Колонизаторы»! Вы со своими друзьями, конечно же, не упустите шанс поиграть в новую игру.

Одной из основных механик игры является событие «Грабёж». Пусть сейчас играет n человек, пронумерованных слева направо числами от 1 до n. У i-го игрока в руке находятся a_i карт. В момент, когда происходит событие «Грабёж», происходит следующее:

1. Сначала все игроки одновременно выполняют действие: найти ближайшего игрока справа, у которого в руке больше карт, и отдать ему одну карту.
2. Затем все игроки одновременно выполняют действие: найти ближайшего игрока слева, у которого в руке больше карт, и отдать ему одну карту.

Если у некоторого игрока нет необходимого соседа слева или справа, либо у него в руке нет ни одной карты, действие для данного игрока пропускается.

Вы с друзьями уже начали игру, и вот, внезапно, одно неаккуратное действие привело к событию «Грабёж». Выясните, какое максимальное количество карт в руке окажется у какого-то игрока после этого грабежа.

Входные данные
В первой строке записано целое число n ($1 \le n \le 100\,000$) — количество игроков.

Во второй строке через пробел записаны n целых чисел $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$) — количество карт в руках игроков.

Выходные данные
Выведите одно целое число — максимальное количество карт в руке после события «Грабёж».

входные данные
5 1 3 2 2 3
выходные данные
6

входные данные
1 1
выходные данные
1

Рассмотрим первый пример.

Количество карт у игроков после первой фазы «Грабежа»: $0, 4, 1, 1, 5$.

Количество карт у игроков после второй фазы «Грабежа»: $0, 6, 0, 0, 5$.

Максимальное количество карт после «Грабежа» — 6.

Во втором примере $n = 1$, поэтому единственный игрок никому не отдаст свои карты.

Тесты к этой задаче состоят из 2 групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов необходимых групп. Тесты из условия не оцениваются.

4F. Бутерброды

2 секунды, 512 мегабайт

Раньше Саша был олимпиадником, но из-за того, что он живет не в столице, а в небольшом городе К, он не смог на этом заработать денег, и теперь ему приходится работать поваром в студенческой столовой. Сейчас Саша занят составлением рецептов бутербродов, которые в будущем он планирует подавать на завтрак. Изначально перед ним лежит K кусков хлеба и он совершает с ними N последовательных операций, каждая из которых относится к одному из следующих типов:

1. Саша берет ингредиент x из холодильника и кладет сверху на i-й бутерброд;
2. Саша снимает с i-го бутерброда верхний ингредиент и возвращает его в холодильник;
3. Саше начинает казаться, что i-й и j-й бутерброды слишком сильно отличаются по размеру, поэтому он начинает перекладывать ингредиенты с большего бутерброда на меньший, пока разница их размеров станет не больше, чем 1.

Требуется написать программу, которая выведет количество различных бутербродов, которые получались у Саши в ходе составления рецептов. Пустой бутерброд не считается бутербродом. Бутерброды, которые получаются в процессе выполнения операции третьего типа, также следует принимать во внимание.

Два бутерброда считаются различными, если у них различается число ингредиентов или их порядок (то есть существует такое i что на i-м сверху месте расположены ингредиенты разного типа).

Входные данные
В первой строке указаны два числа K и N ($K \le 20$, $N \le 3 \cdot 10^5$) – количество кусков хлеба, лежащих перед Сашей и количество действий, соотвественно.

В последующих N строках описаны действия, совершенные Сашей. Каждая строка начинается с числа 1, 2 или 3, обознчающих тип операции. Если операция имеет тип 1, то после написаны два целых числа i и x ($1 \le i \le K$, $1 \le x \le 10^9$) – номер бутерброда и ингредиента, соотвественно. Для операции типа 2 записано одно целое число i ($1 \le i \le K$) – номер бутерброда. Гарантируется, что на этом бутерброде есть хотя бы один игредиент. Наконец, для операций типа 3 записана пара целых чисел i и j ($1 \le i, j \le K$, $i \ne j$) – номера бутербродов, которые надо «уравнять».

Выходные данные
Вывести требуется единственное число – ответ на задачу.

входные данные
3 5 1 1 3 1 1 3 3 2 1 1 2 7 1 3 7
выходные данные
4

4G. Игра в пьяницу

1 секунда, 256 мегабайт

В игре в пьяницу карточная колода раздается поровну двум игрокам. Далее они вскрывают по одной верхней карте, и тот, чья карта старше, забирает себе обе вскрытые карты, которые кладутся под низ его колоды. Тот, кто остается без карт — проигрывает.

Для простоты будем считать, что все карты различны по номиналу, а также, что самая младшая карта побеждает самую старшую карту ("шестерка берет туза").

Игрок, который забирает себе карты, сначала кладет под низ своей колоды карту первого игрока, затем карту второго игрока (то есть карта второго игрока оказывается внизу колоды).

Напишите программу, которая моделирует игру в пьяницу и определяет, кто выигрывает. В игре участвует 10 карт, имеющих значения от 0 до 9, большая карта побеждает меньшую, карта со значением 0 побеждает карту 9.

Входные данные

Программа получает на вход две строки: первая строка содержит 5 чисел, разделенных пробелами — номера карт первого игрока, вторая — аналогично 5 карт второго игрока. Карты перечислены сверху вниз, то есть каждая строка начинается с той карты, которая будет открыта первой.

Выходные данные

Программа должна определить, кто выигрывает при данной раздаче, и вывести слово **first** или **second**, после чего вывести количество ходов, сделанных до выигрыша. Если на протяжении 10^6 ходов игра не заканчивается, программа должна вывести слово **botva**.

входные данные
1 3 5 7 9 2 4 6 8 0
выходные данные
second 5

4H. Сложный выбор и обед

2 секунды, 64 мегабайта

Андрей играл в свою любимую игру под названием "Over часы". В одним момент у него набралось больше золота, чем он может позволить себе по своим личным соображениям, известным только ему. Поэтому он хочет срочно потратить их на что-то. У него есть выбор из n часов, однако правила покупки часов гласят: "Игрок должен выбрать подотрезок часов длины k, заплатить за них количество денег, равное минимуму из всех цен на этом отрезке и забрать себе все часы, принадлежащие ему". Число k — фиксированное. Андрей бережет всё своё золото. Поэтому он хочет заплатить минимальное число денег.

Помогите ему! Через 2 секунды мама позовет Андрея обедать, а в онлайн-играх нет паузы. Зная все стоимости часов, найдите для него на каждом отрезке длины k минимальное число.

Входные данные

В первой строке вводятся числа n и k. $1 \le k \le n \le 10^6$.

На следующей строке вводится n чисел a_i, задающих массив. $-10^6 \le a_i \le 10^6$.

Выходные данные

Для каждого подотрезка массива a длины k выведите минимум на нем. Каждый минимум выводите с новой строки. Всего должно быть выведено $n - k + 1$ чисел.

входные данные
7 3 1 3 2 4 5 3 1
выходные данные
1 2 2 3 1

входные данные
7 3 -5 3 1 8 0 5 3
выходные данные
-5 1 0 0 0

Первый пример: Минимум на первом отрезке [1; 2; 3] равен 1, на втором [3, 2, 4] — 2, на третьем — 2, на четвертом — 3, на пятом — 1.

4I. Очередь в магазине

1 s., 256 MB

В одном известном магазине случилась распродажа, однако администрация не учла одну проблему: в магазине всего одна касса! Сразу после начала распродажи возле кассы организовалась длинная очередь. Никто не любит очереди, поэтому у покупателей постепенно возрастает уровень агрессии. От вас требуется рассмотреть процесс продвижения очереди.

Могут происходить события трёх типов:

1. В конец очереди встал человек с уровнем агрессии a;
2. Первый человек в очереди начал ругаться с кассиром, в результате чего уровень его агрессии увеличился на x, а уровень агрессии каждого из **остальных** людей в очереди (если в очереди стоит не один человек) увеличился на y;
3. Первый человек в очереди оплатил покупку и ушёл из магазина.

От вас требуется обработать N событий. Будем считать, что изначально очередь пуста. Так как администрация магазина заботится о своей репутации, им важно знать, насколько агрессивными их покупатели уходят из магазина. Поэтому для каждого события третьего типа нужно определить уровень агрессии человека, который ушёл из магазина.

Входные данные

В первой строке записано одно число N — количество событий ($2 \le N \le 300000$).

В каждой из следующих N строк содержится описание очередного события:

- `1 a`, если произошло событие первого типа;
- `2 x y`, если произошло событие второго типа;
- `3`, если произошло событие третьего типа.

Для всех событий верно, что $1 \le a, x, y \le 10^9$. Гарантируется, что события второго и третьего типов происходят только в том случае, если в очереди есть хотя бы один человек. Также гарантируется, что после N событий в очереди не останется ни одного человека. Возможны случаи, когда первый человек в очереди несколько раз подряд ссорится с кассиром.

Выходные данные

Для каждого запроса третьего типа выведите одно число — уровень агрессии человека, который ушёл из магазина. Каждое число следует выводить на отдельной строке.

входные данные
8 1 4 1 2 2 6 1 3 2 10 20 1 1 3 3
выходные данные
10 13 1

Сначала в очередь встали два человека с уровнями агрессии 4 и 2 соответственно. Затем первый человек поссорился с кассиром, после чего уровни агрессии людей стали равны 10 и 3. После этого первый человек ушёл из очереди, а второй поссорился с кассиром. Теперь уровень его агрессии равен 13. Затем в очередь встал человек с уровнем агрессии 1, после чего оба человека ушли из магазина.

5A. Зайчик

2 секунды, 256 мегабайт

Зайчик прыгает по прямой просеке, для удобства разделённой на n клеток. Клетки пронумерованы по порядку натуральными числами от 1 до n. Некоторые клетки заболочены: если зайчик прыгнет на такую клетку, ему несдобровать. Некоторые другие клетки просеки поросли вкусной зелёной травой: прыгнув на такую клетку, зайчик сможет отдохнуть и подкрепиться.

Зайчик начинает свой путь из клетки с номером 1 и хочет попасть в клетку с номером n, по пути ни разу не провалившись в болото и скушав как можно больше вкусной зелёной травы. Конструктивные особенности зайчика таковы, что из клетки с номером k он может прыгнуть лишь в клетки с номерами $k+1$, $k+3$ и $k+5$.

Выясните, какое максимальное количество клеток с травой сможет посетить зайчик на своём пути.

Входные данные
В первой строке входного файла задано число n — количество клеток ($2 \le n \le 1000$). Вторая строка состоит из n символов; i-й символ соответствует i-й клетке просеки. Символ 'w' обозначает болото, символ '"'— зелёную траву, а символ '.' соответствует клетке без каких-либо особенностей. Гарантируется, что первая и последняя клетки не содержат болот и травы.

Выходные данные
В первой строке выходного файла выведите одно число — максимальное количество клеток с травой, которые зайчик сможет посетить на своём пути. Если зайчику не удастся оказаться в клетке с номером n, выведите «-1».

входные данные
4 ."".
выходные данные
2

входные данные
5 .w"..
выходные данные
0

5B. Черепаха и монеты

2 секунды, 256 мегабайт

Черепаха хочет переползти из левого верхнего угла поля размером N на M клеток ($2 \le N, M \le 1000$) в правый нижний. За один шаг она может переместиться на соседнюю клетку вправо или на соседнюю клетку вниз. Кроме того, проходя через каждую клетку, Черепаха получает (или теряет) несколько золотых монет (это число известно для каждой клетки).

Определите, какое максимальное количество монет может собрать Черепаха по пути и как ей нужно идти для этого.

Входные данные
В первой строке вводятся два натуральных числа: N и M ($2 \le N, M \le 1000$), разделённые пробелом. В каждой из следующих N строк записаны через пробел по M чисел a_{ij}($|a_{ij}| \le 10$), которые обозначают количество монет, получаемых Черепашкой при проходе через каждую клетку. Если это число отрицательное, Черепашка теряет монеты.

Выходные данные
В первой строке программа должна вывести наибольшее количество монет, которое может собрать Черепаха. Во второй строке без пробелов выводятся команды, которые нужно выполнить Черепахе: буква 'R' (от слова *right*) обозначает шаг вправо, а буква 'D' (от слова *down*) – шаг вниз.

входные данные
3 3 0 2 -3 2 -5 7 1 2 0
выходные данные
6 RRDD

5C. Наибольшая возрастающая подпоследовательность

2 секунды, 256 мегабайт

Дана последовательность, требуется найти её наибольшую возрастающую подпоследовательность.

Входные данные
В первой строке входных данных задано целое число n — длина последовательности ($1 \le n \le 1\,000$). Во второй строке задается сама последовательность. Числа разделяются пробелом. Элементы последовательности — целые числа, не превосходящие 10^9 по абсолютной величине.

Выходные данные
В первой строке выведите длину наибольшей возрастающей подпоследовательности, а во второй строке выведите через пробел саму наибольшую возрастающую подпоследовательность данной последовательности. Если ответов несколько — выведите любой.

входные данные
6 3 29 5 5 28 6
выходные данные
3 3 5 28

5D. Наибольшая общая подпоследовательность

3 секунды, 256 мегабайт

Даны две последовательности, требуется найти и вывести их наибольшую общую подпоследовательность.

Входные данные
В первой строке входных данных содержится целое число n — длина первой последовательности ($1 \le n \le 2\,000$). Во второй строке заданы члены первой последовательности (через пробел) — целые числа, не превосходящие 10^9 по модулю. В третьей строке записано целое число m — длина второй последовательности ($1 \le m \le 2\,000$). В четвертой строке задаются члены второй последовательности (через пробел) — целые числа, не превосходящие 10^9 по модулю.

Выходные данные
В первой строке выведите длину наибольшей общей подпоследовательности, а во второй строке выведите через пробел саму наибольшую общую подпоследовательность данных последовательностей. Если ответов несколько — выведите любой.

входные данные
3 1 2 3 4 2 3 1 5
выходные данные
2 2 3

5E. Расстояние по Левенштейну

1 секунда, 256 мегабайт

Дана текстовая строка. С ней можно выполнять следующие операции:

1. Заменить один символ строки на другой символ.
2. Удалить один произвольный символ.
3. Вставить произвольный символ в произвольное место строки.

Например, при помощи первой операции из строки «СОК» можно получить строку «СУК», при помощи второй операции — строку «ОК», при помощи третьей операции — строку «СТОК».

Минимальное количество таких операций, при помощи которых можно из одной строки получить другую, называется стоимостью редактирования или расстоянием Левенштейна.

Определите расстояние Левенштейна для двух данных строк.

Входные данные
Программа получает на вход две строки, длина каждой из которых не превосходит 1000 символов, строки состоят только из заглавных латинских букв.

Выходные данные
Требуется вывести одно число — расстояние Левенштейна для данных строк.

входные данные
ABCDEFGH ACDEXGIH
выходные данные
3

5F. Невозрастающая подпоследовательность

1 секунда, 256 мегабайт

Вам требуется написать программу, которая по заданной последовательности находит максимальную невозрастающую её подпоследовательность (т.е такую последовательность чисел $a_{i_1}, a_{i_2}, ..., a_{i_k}$ ($i_1 < i_2 < ... < i_k$), что $a_{i_1} \geq a_{i_2} \geq ... \geq a_{i_k}$ и не существует последовательности с теми же свойствами длиной $k + 1$).

Входные данные
В первой строке задано число n — количество элементов последовательности ($1 \leq n \leq 239\,017$). В последующих строках идут сами числа последовательности a_i, отделенные друг от друга произвольным количеством пробелов и переводов строки (все числа не превосходят по модулю 2^{31} - 2).

Выходные данные
Вам необходимо выдать в первой строке выходного файла число k — длину максимальной невозрастающей подпоследовательности. В последующих строках должны быть выведены (по одному числу в каждой строке) все номера элементов исходной последовательности i_j, образующих искомую подпоследовательность. Номера выводятся в порядке возрастания. Если оптимальных решений несколько, разрешается выводить любое.

входные данные
5 5 8 10 4 1
выходные данные
3 3 4 5

5G. Как переименовать папку

2 секунды, 256 мегабайт

Один известный видеоблогер под ником kriper2004 выпустил шуточный видеоролик, в котором он весьма подробно на протяжении четырех часов рассказывал, как можно создать папку на рабочем столе в операционной системе Windows. После невероятного успеха ролика в Интернете появилось большое количество ремейков данного видео. И вот настал тот день, когда Константин решил стать видеоблогером и снять свой ремейк на данное видео.

Константин решил поведать своей немногочисленной аудитории, как можно переименовать созданную ранее папку. В названии папки могут использоваться только строчные латинские буквы. При этом он не хочет, чтобы в названии папки встречались два подряд идущих последовательных символа. Например, следующие сочетания букв запрещены: «ab», «bc», «yz». А такие сочетания букв использовать можно: «ba», «za», «dx».

Константин уже придумал некоторую строку s, на которую будет начинаться название папки. Но он хочет, чтобы длина названия была равна n. Посчитайте количество допустимых названий для папки по модулю 998244353. Гарантируется, что $|s| \leq n$.

Входные данные
В первой строке задана строка s ($1 \leq |s| \leq 2 \cdot 10^5$). Строка s состоит из строчных латинских букв.

Во второй строке задано число n ($|s| \leq n \leq 2 \cdot 10^5$).

Выходные данные
Выведите одно число — количество допустимых названий папки длины n, начинающихся на строку s по модулю 998244353.

входные данные
yz 100500
выходные данные
0

входные данные
b 1
выходные данные
1

входные данные
e 2
выходные данные
25

5H. Игра на фортепиано

1 секунда, 256 мегабайт

Маленький Пол хочет научиться играть на фортепиано. Он уже нашёл мелодию, которую будет играть. Для простоты он выписал последовательность $a_1, a_2, \ldots, a_n$ целых чисел, которые означают номер клавиш: чем больше номер, тем правее клавиша на клавиатуре.

Пол очень умный и понимает, что самое важное — правильная аппликатура, то есть, правильно выбрать, каким пальцем какую ноту играть. Если выбрать неудобные пальцы, то потом можно потратить уйму времени на попытки научиться играть мелодию и всё равно в итоге не преуспеть.

Обозначим пальцы на руке числами от 1 до 5. Назовём *аппликатурой* любую последовательность $b_1, \ldots, b_n$ номеров пальцев. Назовём аппликатуру *удобной*, если для любого $1 \leq i \leq n - 1$ выполнено следующее:

- если $a_i < a_{i+1}$, то $b_i < b_{i+1}$, потому что иначе придётся оторвать руку от клавиатуры, чтобы сыграть $(i + 1)$-ю ноту;
- если $a_i > a_{i+1}$, то $b_i > b_{i+1}$ по той же причине;

- если $a_i = a_{i+1}$, то $b_i \neq b_{i+1}$, потому что довольно нелепо использовать один и тот же палец два раза подряд. **Обратите внимание, что между b_i и b_{i+1} стоит знак $\neq$, а не $=$.**

Найдите любую удобную аппликатуру или скажите, что таких нет.

Входные данные
Первая строка содержит одно целое число n ($1 \le n \le 10^5$), обозначающее число нот в мелодии.

Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$) — номера этих нот.

Выходные данные
Если удобных аппликатур к этой мелодии не существует, выведите -1. В противном случае выведите n целых чисел $b_1, b_2, \ldots, b_n$ от 1 до 5, задающих удобную аппликатуру.

входные данные
5 1 1 4 2 2
выходные данные
1 4 5 4 5

входные данные
7 1 5 7 8 10 3 1
выходные данные
1 2 3 4 5 4 3

входные данные
19 3 3 7 9 8 8 8 8 7 7 7 7 5 3 3 3 3 8 8
выходные данные
1 3 4 5 4 5 4 5 4 5 4 5 4 3 5 4 3 5 4

Третий тест из условия — что-то вроде песни "Нон стоп" группы Рефлекс.

6A. Магазин «Всё за O(1)»

1 секунда, 512 мегабайт

В магазине «Всё за $O(1)$» есть две кассы и много посетителей. Вам предстоит смоделировать очереди в эти кассы по записанной истории работы магазина.

Вам в хронологическом порядке даны события, закодированные следующими символами:

- `a` — в конец очереди в первую кассу встал очередной посетитель;
- `b` — в конец очереди во вторую кассу встал очередной посетитель;
- `A` — в первой кассе обслужили первого посетителя в очереди;
- `B` — во второй кассе обслужили первого посетителя в очереди;
- `>` — первая касса закрылась;
- `]` — вторая касса закрылась;
- `<` — первая касса открылась;
- `[` — вторая касса открылась.

Когда касса закрывается, все люди из очереди к этой кассе в обратном порядке, начиная с последнего, переходят в конец другой очереди. То есть первым переходит человек, стоявший последним, затем человек, стоявший предпоследним, и так далее. В итоге последним в получившейся очереди будет стоять тот, кто был первым в очереди к только что закрывшейся кассе.

Когда закрытая касса открывается, люди в очереди к другой кассе, начиная с последнего, переходят в нее, если их место в новой очереди окажется строго меньше текущего. Стоявший последним становится первым в новой очереди, стоявший предпоследним становится вторым и так далее.

Список событий корректен, то есть:

- Открываются только закрытые кассы;
- Закрываются только открытые кассы;
- Посетители не встают в очереди к закрытым кассам;
- Закрытые кассы не пытаются обслуживать посетителей;
- Кассы не обслуживают посетителей, если очереди к ним пустые;
- В каждый момент времени работает хотя бы одна касса.

Посетители нумеруются с единицы в порядке их появления в списке событий. В начальный момент обе кассы открыты и обе очереди пусты.

Входные данные
В первой строке входных данных содержится натуральное число n ($2 \le n \le 10\,000\,000$) — количество событий.

Во второй строке содержатся n символов, описывающих события согласно приведённым выше обозначениям.

Гарантируется, что во входных данных содержится хотя бы один запрос обслуживания посетителя

Выходные данные
В единственной строке выведите для каждой записи обслуживания последнюю цифру номера обслуженного посетителя. Ответы выводите в порядке выполнения запросов обслуживания, не используйте никаких разделителей.

Система оценки
Тесты к этой задаче состоят из трёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов предыдущих групп.

- Группа, в которой $n \le 1000$ оценивается в 26 баллов
- Группа, в которой $n \le 200\,000$ оценивается в 37 баллов
- Группа, в которой $n \le 10\,000\,000$ оценивается в 37 баллов

входные данные
15 aaabA>bBBb<BBAa
выходные данные
143256

входные данные
12 aaaaa><AABBB
выходные данные
12543

Пояснение к первому примеру:

№	Команда	Пояснение	1 очередь	2 очередь
1	a	В очередь 1 встал посетитель 1	1	–
2	a	В очередь 1 встал посетитель 2	1, 2	–
3	a	В очередь 1 встал посетитель 3	1, 2, 3	–
4	b	В очередь 2 встал посетитель 4	1, 2, 3	4
5	A	В очереди 1 обслужен посетитель 1	2, 3	4
6	>	Касса 1 закрылась	–	4, 3, 2
7	b	В очередь 2 встал посетитель 5	–	4, 3, 2, 5

8	B	В очереди 2 обслужен посетитель 4	–	3, 2, 5
9	B	В очереди 2 обслужен посетитель 3	–	2, 5
10	b	В очередь 2 встал посетитель 6	–	2, 5, 6
11	<	Касса 1 открылась	6	2, 5
12	B	В очереди 2 обслужен посетитель 2	6	5
13	B	В очереди 2 обслужен посетитель 5	6	–
14	A	В очереди 1 обслужен посетитель 6	–	–
15	a	В очередь 1 встал посетитель 7	7	–

7A. Подсчет опыта

2 секунды, 64 мегабайта

В очередной онлайн игре игроки, как обычно, сражаются с монстрами и набирают опыт. Для того, чтобы сражаться с монстрами, они объединяются в кланы. После победы над монстром, всем участникам клана, победившего его, добавляется одинаковое число единиц опыта. Особенностью этой игры является то, что кланы никогда не распадаются и из клана нельзя выйти. Единственная доступная операция — объединение двух кланов в один.

Поскольку игроков стало уже много, вам поручили написать систему учета текущего опыта игроков.

Входные данные

В первой строке входного файла содержатся числа n $(1 \le n \le 200000)$ и m $1 \le m \le 200000$ — число зарегистрированных игроков и число запросов.

В следующих m строках содержатся описания запросов. Запросы бывают трех типов:

- `join X Y` — объединить кланы, в которые входят игроки `X` и `Y` (если они уже в одном клане, то ничего не меняется).
- `add X V` — добавить `V` единиц опыта всем участникам клана, в который входит игрок `X` $(1 \le V \le 100)$.
- `get X` — вывести текущий опыт игрока `X`.

Изначально у всех игроков 0 опыта и каждый из них состоит в клане, состоящим из него одного.

Выходные данные

Для каждого запроса `get X` выведите текущий опыт игрока `X`.

входные данные
3 6 add 1 100 join 1 3 add 1 50 get 1 get 2 get 3
выходные данные
150 0 50

7B. Разрезание графа

2 секунды, 256 мегабайт

Дан неориентированный граф. Над ним в заданном порядке производят операции следующих двух типов:

- `cut` — разрезать граф, то есть удалить из него ребро;
- `ask` — проверить, лежат ли две вершины графа в одной компоненте связности.

Известно, что после выполнения всех операций типа `cut` рёбер в графе не осталось. Найдите результат выполнения каждой из операций типа `ask`.

Входные данные

Первая строка входного файла содержит три целых числа, разделённые пробелами — количество вершин графа n, количество рёбер m и количество операций k $(1 \le n \le 50\,000,\ 0 \le m \le 100\,000,\ m \le k \le 150\,000)$.

Следующие m строк задают рёбра графа; i-я из этих строк содержит два числа u_i и v_i $(1 \le u_i,\ v_i \le n)$, разделённые пробелами — номера концов i-го ребра. Вершины нумеруются с единицы; граф не содержит петель и кратных рёбер.

Далее следуют k строк, описывающих операции. Операция типа `cut` задаётся строкой "`cut` u v" $(1 \le u,\ v \le n)$, которая означает, что из графа удаляют ребро между вершинами u и v. Операция типа `ask` задаётся строкой "`ask` u v" $(1 \le u,\ v \le n)$, которая означает, что необходимо узнать, лежат ли в данный момент вершины u и v в одной компоненте связности. Гарантируется, что каждое ребро графа встретится в операциях типа `cut` ровно один раз.

Выходные данные

Для каждой операции `ask` во входном файле выведите на отдельной строке слово "`YES`", если две указанные вершины лежат в одной компоненте связности, и "`NO`" в противном случае. Порядок ответов должен соответствовать порядку операций `ask` во входном файле.

входные данные
3 3 7 1 2 2 3 3 1 ask 3 3 cut 1 2 ask 1 2 cut 1 3 ask 2 1 cut 2 3 ask 3 1
выходные данные
YES YES NO NO

7C. Реструктуризация компании

2 секунды, 256 мегабайт

В жизни даже самой успешной компании может наступить кризисный период, когда приходится принимать тяжёлое решение о реструктуризации, распускать и объединять отделы, увольнять работников и заниматься прочими неприятными делами. Рассмотрим следующую модель компании.

В Большой Софтверной Компании работают n человек. Каждый человек принадлежит какому-то *отделу*. Исходно каждый человек работает над своим проектом в своём собственном отделе (таким образом, в начале компания состоит из n отделов по одному человеку).

Однако, в жизни компании наступили тяжёлые времена, и руководство было вынуждено нанять кризисного менеджера, который начал переустраивать рабочий процесс для повышения эффективности производства. Обозначим за $team(person)$ команду, в которой работает человек $person$. Кризисный менеджер может принимать решения двух типов:

1. Объединить отделы $team(x)$ и $team(y)$, сформировав из них один большой отдел, содержащий всех сотрудников $team(x)$ и $team(y)$, где x и y ($1 \le x, y \le n$) — номера каких-то двух сотрудников компании. Если $team(x)$ совпадает с $team(y)$, ничего делать не требуется.
2. Объединить отделы $team(x)$, $team(x+1)$, ..., $team(y)$, где x и y ($1 \le x \le y \le n$) — номера каких-то двух сотрудников компании.

При этом кризисный менеджер иногда может интересоваться, работают ли в одном отделе сотрудники x и y ($1 \le x, y \le n$).

Помогите кризисному менеджеру, ответив на все его запросы.

Входные данные

Первая строка входных данных содержит два целых числа n и q ($1 \le n \le 200\,000$, $1 \le q \le 500\,000$) — количество сотрудников компании и количество запросов кризисного менеджера.

В последующих q строках находятся запросы кризисного менеджера. Каждый запрос имеет вид $type\ x\ y$, где $type \in \{1, 2, 3\}$. Если $type = 1$ или $type = 2$, то запрос представляет собой решение кризисного менеджера об объединении отделов соответственно первого или второго вида. Если $type = 3$, то требуется определить, работают ли в одном отделе сотрудники x и y. Обратите внимаие, что x может равняться y в запросе любого типа.

Выходные данные

На каждый запрос типа 3 выведите «YES» или «NO» (без кавычек), в зависимости от того, работают ли в одном отделе соответствующие люди.

входные данные
8 6 3 2 5 1 2 5 3 2 5 2 4 7 2 1 2 3 1 7
выходные данные
NO YES YES

8A. 0-1 рюкзак: наибольший вес

1 секунда, 256 мегабайт

Дано N золотых слитков массой $m_1, \ldots, m_N$. Ими наполняют рюкзак, который выдерживает вес не более M. Какую наибольшую массу золота можно унести в таком рюкзаке?

Входные данные

В первой строке вводится натуральное число N, не превышающее 100 и натуральное число M, не превышающее $10\,000$.

Во второй строке вводятся N натуральных чисел m_i, не превышающих 100.

Выходные данные

Выведите одно целое число — наибольшую возможную массу золота, которую можно унести в данном рюкзаке.

входные данные
2 3195 38 41
выходные данные
79

8B. Рюкзак

2 секунды, 256 мегабайт

Дано n предметов массой $m_1, \ldots, m_n$ и стоимостью $c_1, \ldots, c_n$ соответственно.

Ими наполняют рюкзак, который выдерживает вес не более m. Определите набор предметов, который можно унести в рюкзаке, имеющий наибольшую стоимость.

Входные данные

В первой строке вводится натуральное число n, не превышающее $1\,000$ и натуральное число m, не превышающее $1\,000$.

Во второй строке вводятся n натуральных чисел m_i, не превышающих 100.

Во третьей строке вводятся n натуральных чисел c_i, не превышающих 100.

Выходные данные

В первой строке выведите количество предметов, которые нужно взять. Во второй строке выведите номера предметов (числа от 1 до n), которые войдут в рюкзак наибольшей стоимости.

входные данные
4 6 2 4 1 2 7 2 5 1
выходные данные
3 1 3 4

8C. Банкомат

2 секунды, 64 мегабайта

В некотором государстве в обращении находятся банкноты определенных номиналов. Национальный банк хочет, чтобы банкомат выдавал любую запрошенную сумму при помощи минимального числа банкнот, считая, что запас банкнот каждого номинала неограничен. Помогите Национальному банку решить эту задачу.

Входные данные

Первая строка входных данных содержит натуральное число N не превосходящее 100 — количество номиналов банкнот в обращении. Вторая строка входных данных содержит N различных натуральных чисел $x_1, x_2, \ldots, x_N$, не превосходящих 10^6 — номиналы банкнот. Третья строчка содержит натуральное число S, не превосходящее 10^6 — сумму, которую необходимо выдать.

Выходные данные

В первую строку выходного файла выведите минимальное число слагаемых (или -1, если такого представления не существует). Во вторую строку выведите это представление в любом порядке.

входные данные
5 1 3 7 12 32 40
выходные данные
3 32 7 1

8D. Гирьки: кучки одного размера

1 секунда, 256 мегабайт

Дан набор гирек массой $m_1, \ldots, m_N$. Разделите этот набор на две кучки равной массы, содержащие равное число гирек.

Входные данные

Первая строка входных данных содержит натуральное число N, не превышающее 100.

В следующей строке через пробел записаны N натуральных чисел m_i, не превышающих 100.

Выходные данные

Необходимо вывести в первой строчке номера гирек (числа от 1 до N), входящие в первую кучку, во второй строчке — номера гирек во второй кучке.

Если задача не имеет решения, выведите одно число `-1`.

входные данные
4 4 2 3 1
выходные данные
1 4 2 3

8E. Гирьки: три кучки

1 секунда, 256 мегабайт

Дан набор гирек массой $m_1, \ldots, m_N$. Можно ли их разложить на три кучки равной массы?

Входные данные

Первая строка входных данных содержит натуральное число N, не превышающее 60.

Во второй строке через пробел записаны N натуральных чисел m_i, не превышающих 60.

Выходные данные

Выведите три строки, описывающие наборы гирек, либо число `-1`, если решение не существует.

В каждой строке сначала выведите количество гирек в соответствующем наборе, а затем через пробел номера гирек в наборе.

входные данные
5 4 2 3 1 5
выходные данные
2 4 1 2 3 2 1 5

8F. Черная пятница

1 s., 512 MB

Сегодня — черная пятница. В честь этого Миша решил закупить оборудование для своей майнинг фермы. В магазине действует удивительная акция, если купить видеокарту номер i в количестве от l_i до r_i, включительно ($1.4 \cdot l_i \le r_i$), то их стоимость будет сильно ниже рыночной. Миша может унести с собой не больше s видеокарт. Из-за такой щедрости магазина он решил купить как можно больше видеокарт по этой акции. Решить такую задачу Мише не представляется возможным, поэтому он просит вас о помощи. Для каждой видеокарты определите, сколько штук надо купить, либо не покупать вообще, чтобы все видеокарты были куплены по акции, при этом их количество было **максимально возможным** числом не большим s.

Входные данные

Первая строка входных данных содержит два целых числа n и s ($1 \le n \le 10^5, 1 \le s \le 10^{13}$) — количество различных видеокарт и максимальное число видеокарт, которое Миша может унести с собой.

Следующие n строк содержат по два целых числа l_i и r_i ($1 \le l_i \le 10^{13}, 1.4 \cdot l_i \le r_i \le 10^{13}$) — минимальное и максимальное количество видеокарт типа i, которое можно купить по скидке.

Выходные данные

В первой строке выведите единственное целое число w — максимальное количество видеокарт, которое можно купить по скидке ($0 \le w \le s$). Во второй строке выведите через пробел n целых чисел x_i. i-е число равно количеству видеокарт типа i, которое необходимо купить. Заметим, что либо $x_i = 0$, либо $l_i \le x_i \le r_i$, а также что сумма x_i равна w. Если существует несколько решений, максимизирующих w, выведите любое.

входные данные
3 20 1 2 10 17 11 16
выходные данные
19 2 17 0

s В примере из условия выгодно купить видеокарты первого и второго типа. Заметим, что нельзя одновременно купить видеокарты второго и третьего типа, так как тогда нужно будет купить хотя бы $10 + 11 = 21$ штуку, что больше 20.