1B. K-я порядковая статистика

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 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.

Примеры

входные данные	Скопировать
5 123456789 2	
выходные данные	Скопировать
850994577	

Примечание

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

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 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.

Примеры

входные данные	Скопировать
1	

выходные данные	Скопировать
1	

входные данные	Скопировать
3	

выходные данные	Скопировать
1 3 2	

1D. Количество инверсий

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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.

Пример

входные данные	Скопировать
5 11 6 31 28 18	
выходные данные	**Скопировать**
4	

1E. Отрезки с большой суммой

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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 $l \le r$ и $\left(\sum_{i=l}^{r} a_i\right) \ge k$.

Входные данные

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

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

Выходные данные

Выведите одно число — количество искомых отрезков.

Пример

входные данные	Скопировать
4 7 -1 3 5 2	

выходные данные	Скопировать
5	

1H. Цифровая сортировка

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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.

Примеры

входные данные Скопировать

```
3 3 1
bbb
aba
baa
```

выходные данные Скопировать

```
aba
baa
bbb
```

входные данные Скопировать

```
3 3 2
bbb
aba
baa
```

выходные данные Скопировать

```
baa
aba
bbb
```

входные данные Скопировать

```
3 3 3
bbb
aba
baa
```

выходные данные Скопировать

```
aba
baa
bbb
```

2A. Проверьте сортирующую сеть

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 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.

Примеры

входные данные	Скопировать
4 6 3 2 1 2 3 4 2 1 4 2 3 2 1 2 3 4	

выходные данные	Скопировать
Yes	

входные данные	Скопировать
3 2 2 1 2 1 1 3 1	

выходные данные	Скопировать
No	

3A. Двоичный поиск

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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.

Пример

входные данные Скопировать

```
10 10
1 61 126 217 2876 6127 39162 98126 712687 1000000000
100 6127 1 61 200 -10000 1 217 10000 1000000000
```

выходные данные Скопировать

```
NO
YES
YES
YES
NO
NO
YES
YES
NO
YES
```

1A. Сортировка

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 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.

Примеры

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

выходные данные	Скопировать
1 1 2 2 3 3 4 6 7 8	

3C. Гирлянда

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 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.

Примеры

входные данные	Скопировать
8 15	

выходные данные	Скопировать
9.75	

входные данные	Скопировать
692 532.81	

выходные данные	Скопировать
446113.34	

4A. Простой стек

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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

Пример

входные данные	Скопировать
push 1 back exit	

выходные данные	Скопировать
ok 1 bye	

4B. Простая очередь

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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

Пример

входные данные	Скопировать
push 1 front exit	
выходные данные	Скопировать
ok 1 bye	

5A. Зайчик

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 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».

Примеры

входные данные	Скопировать
4 ."".	

выходные данные	Скопировать
2	

входные данные	Скопировать
5 .w"..	

выходные данные	Скопировать
0	

5B. Черепаха и монеты

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Черепаха хочет переползти из левого верхнего угла поля размером N на M клеток ($2 \leq N, M \leq 1000$) 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 ($2 \leq N, M \leq 1000$), разделённые пробелом. В каждой из следующих N строк записаны через пробел по M чисел a_{ij}($|a_{ij}| \leq 10$), 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.

Пример

входные данные	Скопировать
3 3 0 2 -3 2 -5 7 1 2 0	

выходные данные	Скопировать
6 RRDD	

5D. Наибольшая общая подпоследовательность

ограничение по времени на тест: 3 секунды
ограничение по памяти на тест: 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.

Пример

входные данные	Скопировать
3 1 2 3 4 2 3 1 5	

выходные данные	Скопировать
2 2 3	

5C. Наибольшая возрастающая подпоследовательность

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Дана последовательность, требуется найти её наибольшую возрастающую подпоследовательность.

Входные данные

В первой строке входных данных задано целое число n — длина последовательности $(1 \leq n \leq 1\,000)$. 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.

Пример

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

5E. Расстояние по Левенштейну

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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.

Пример

входные данные	Скопировать
ABCDEFGH ACDEXGIH	
выходные данные	Скопировать
3	

7A. Подсчет опыта

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 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`.

Пример

входные данные Скопировать

```
3 6
add 1 100
join 1 3
add 1 50
get 1
get 2
get 3
```

выходные данные Скопировать

```
150
0
50
```

8B. Рюкзак

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 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.

Пример

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

8A. 0-1 рюкзак: наибольший вес

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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.

Пример

входные данные	Скопировать
2 3195 38 41	
выходные данные	**Скопировать**
79	

3B. Левый и правый двоичный поиск

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 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.

Пример

входные данные	Скопировать
10 5 1 1 3 3 5 7 9 18 18 57 57 3 9 1 179	

выходные данные	Скопировать
10 10 3 4 7 7 1 2 0	