,

Задача A. Хипуй!

Имя входного файла:	`heap.in`
Имя выходного файла:	`heap.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мегабайта

В этой задаче вам необходимо организовать структуру данных Heap для хранения целых чисел, над которой определены следующие операции:

- `Insert(X)` — добавить в Heap число X;
- `Extract` — достать из Heap наибольшее число (удалив его при этом).

Формат входных данных

Во входном файле записано количество команд N ($1 \leqslant N \leqslant 100\,000$), потом последовательность из N команд, каждая в своей строке.

Каждая команда имеет такой формат: "`0 <число>`"или "`1` что означает соответственно операции `Insert(<число>)` и `Extract`. Добавляемые числа находятся в интервале от 1 до 10^7 включительно.

Гарантируется, что при выполнении команды `Extract` в структуре находится по крайней мере один элемент.

Формат выходных данных

В выходной файл для каждой команды извлечения необходимо вывести число, полученное при выполнении команды `Extract`.

Примеры

heap.in	heap.out
7 0 100 0 10 1 0 5 0 30 0 50 1	100 50

,

Задача B. Простая куча

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	7.5 секунд
Ограничение по памяти:	256 мегабайт

Реализуйте любую сливаемую кучу.

Формат входных данных

В первой строке содержится два целых числа: N — общее количество куч и M — количество операций ($1 \leqslant N \leqslant 1000, 1 \leqslant M \leqslant 1\,000\,000$). Изначально все кучи пусты.

Требуется поддерживать следующие операции:

- `0 a v` — добавить элемент со значением v в кучу с номером a.
- `1 a b` — переложить все элементы из кучи с номером a в кучу с номером b. После этой операции куча a становится пустой.
- `2 a` — вывести на отдельной строке значение минимального элемента в куче с номером a. Гарантируется, что куча не пуста.
- `3 a` — удалить минимальный элемент из кучи с номером a. Если таковых несколько, то выбирается элемент с минимальным индексом. Гарантируется, что куча не пуста.

Формат выходных данных

Для каждой операции поиска минимального элемента выведите единственное число: значение искомого элемента.

Пример

стандартный ввод	стандартный вывод
3 13 0 1 10 2 1 0 2 5 0 2 7 2 2 1 2 1 2 1 3 1 2 1 0 1 9 1 1 3 0 3 8 2 3	10 5 5 7 7

Задача C. Биномиальная куча

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Реализуйте биномиальную кучу.

Формат входных данных

В первой строке содержится два целых числа: N — общее количество куч и M — количество операций ($1 \leqslant N \leqslant 1000, 1 \leqslant M \leqslant 1\,000\,000$). Изначально все кучи пусты.

Требуется поддерживать следующие операции:

- `0 a v` — добавить элемент со значением v в кучу с номером a. Вновь добавленный элемент имеет уникальный индекс равный порядковому номеру соответствующей операции добавления. Нумерация начинается с единицы.
- `1 a b` — переложить все элементы из кучи с номером a в кучу с номером b. После этой операции куча a становится пустой.
- `2 i` — удалить элемент с индексом i.
- `3 i v` — присвоить элементу с индексом i значение v. Гарантируется, что элемент существует.
- `4 a` — вывести на отдельной строке значение минимального элемента в куче с номером a. Гарантируется, что куча не пуста.
- `5 a` — удалить минимальный элемент из кучи с номером a. Если таковых несколько, то выбирается элемент с минимальным индексом. Гарантируется, что куча не пуста.

Формат выходных данных

Для каждой операции поиска минимального элемента выведите единственное число: значение искомого элемента.

Пример

стандартный ввод	стандартный вывод
3 19 0 1 10 4 1 0 2 5 0 2 7 4 2 3 2 20 4 2 1 2 1 4 1 5 1 4 1 3 2 3 4 1 2 2 4 1 0 1 9 1 1 3 0 3 8 4 3	10 5 7 7 10 3 10 8

Задача D. Дофига умная [практически бесполезная (но красивая)] куча

Имя входного файла:	`binomial.in`
Имя выходного файла:	`binomial.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Реализуйте **Фибоначчиеву** или **Quake** кучу.

Формат входных данных

В первой строке содержится два целых числа: N — общее количество куч и M — количество операций ($1 \leqslant N \leqslant 1000, 1 \leqslant M \leqslant 1\,000\,000$). Изначально все кучи пусты.

Требуется поддерживать следующие операции:

- `0 v a` — добавить элемент со значением v в кучу с номером a. Вновь добавленный элемент имеет уникальный индекс равный порядковому номеру соответствующей операции добавления. Нумерация начинается с единицы.
- `1 a b` — переложить все элементы из кучи с номером a в кучу с номером b. После этой операции куча a становится пустой.
- `2 i` — удалить элемент с индексом i.
- `3 i v` — присвоить элементу с индексом i значение v. Гарантируется, что элемент существует.
- `4 a` — вывести на отдельной строке значение минимального элемента в куче с номером a. Гарантируется, что куча не пуста.
- `5 a` — удалить минимальный элемент из кучи с номером a. Если таковых несколько, то выбирается элемент с минимальным индексом. Гарантируется, что куча не пуста.

Формат выходных данных

Для каждой операции поиска минимального элемента выведите единственное число: значение искомого элемента.

Примеры

binomial.in	binomial.out
3 19 0 1 10 4 1 0 2 5 0 2 7 4 2 3 2 20 4 2 1 2 1 4 1 5 1 4 1 3 2 3 4 1 2 2 4 1 0 1 9 1 1 3 0 3 8 4 3	10 5 7 7 10 3 10 8

,

Задача E. Advanced Succinct Rank

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	16 мегабайт

Дан массив длины $64 \cdot n$ из 0 и 1. Требуется написать безызбыточную структуру данных и отвечать на запрос «сколько единиц на префиксе длины k».

Формат входных данных

На первой строке ввода находится число n ($1 \leqslant n \leqslant 10^6$).

На второй строке находится n целых 64 битных числа, тогда если в i-м числе j-й младший бит равен единице, то бит в ячейке $64i + j + 1$ будет тоже равен единице.

На третьей строке задано число q — количество запросов ($1 \leqslant q \leqslant 10^6$). Следующая строка содержат q чисел r_i — запросы ($1 \leqslant r_i \leqslant 64n$).

Формат выходных данных

Для каждого запроса выведите количество единиц на префиксе длины r_i.

Пример

стандартный ввод	стандартный вывод
1 14 5 3 2 1 4 5	2 1 0 3 3

Замечание

Необходимо использовать Munro Rank.

В первом примере массив $[0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,$
$0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$.