

Задача А. Увеличение приоритета

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Запрос задаётся двумя целыми числами i и x . Требуется увеличить значение i -го элемента кучи на x и выполнить *SiftUp* для восстановления кучи.

Гарантируется, что $i \in [1; N]$, $x \geq 0$, новое значение $A[i] + x$ не превышает 10^9 и отличается от текущих значений всех остальных элементов кучи.

Формат входных данных

В первой строке задан размер кучи $N \in [1; 10^5]$.

Во второй строке вводится сама куча — N различных целых чисел, каждое из диапазона $[-10^9; 10^9]$. (Гарантируется, что эти числа составляют корректную максимальную кучу, все элементы уникальны).

В третьей строке вводится число M — количество запросов, $M \in [0; 10^5]$. Вследующих M строках вводятся сами запросы — по одному в строке.

Формат выходных данных

В качестве ответа на запрос требуется вывести одно число: сообщить, на каком месте массива оказался изменённый элемент после выполнения *SiftUp*. (Вывести в отдельной строке одно число — соответствующий индекс).

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

Пример

стандартный ввод	стандартный вывод
6	1
12 6 8 3 4 7	3
2	15 12 14 3 6 7
5 11	
3 6	

Задача В. Уменьшение приоритета

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Запрос задаётся двумя целыми числами i и x . Требуется уменьшить значение i -го элемента кучи на x и выполнить *SiftDown* для восстановления кучи.

Гарантируется, что $i \in [1; N]$, $x \geq 0$, новое значение $A[i] - x$ не превышает по модулю 10^9 и отличается от текущих значений всех остальных элементов кучи.

Формат входных данных

В первой строке задан размер кучи $N \in [1; 10^5]$.

Во второй строке вводится сама куча — N различных целых чисел, каждое из диапазона $[-10^9; 10^9]$. (Гарантируется, что эти числа составляют корректную максимальную кучу, все элементы уникальны).

В третьей строке вводится число M — количество запросов, $M \in [0; 10^5]$. В следующих M строках вводятся сами запросы — по одному в строке.

Формат выходных данных

В качестве ответа на запрос требуется вывести одно число: сообщить, на каком месте массива оказался изменённый элемент после выполнения *SiftDown*. (Вывести в отдельной строке одно число — соответствующий индекс).

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

Пример

стандартный ввод	стандартный вывод
6	5
12 6 8 3 4 7	1
2	10 4 8 3 1 7
2 5	
1 2	

Задача С. Извлечение максимального

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Дана куча размера $N > 1$. Требуется $N - 1$ раз выполнить извлечение максимального элемента. В процессе выполнения процедуры *ExtractMax* последний элемент кучи помещается в её корень, а затем просеивается вниз вызовом *SiftDown*. После каждого выполнения процедуры *ExtractMax* нужно будет вывести индекс конечного положения этого элемента после просеивания, а также значение извлечённого максимального элемента.

Формат входных данных

В первой строке задан размер кучи $N \in [2; 10^5]$.

Во второй строке вводится сама куча — N различных целых чисел, каждое из диапазона $[-10^9; 10^9]$. (Гарантируется, что эти числа составляют корректную максимальную кучу).

Формат выходных данных

Требуется вывести $N - 1$ строку, в каждой — два числа. Первое — индекс конечного положения элемента после его просеивания; второе — значение извлечённого элемента.

Пример

стандартный ввод	стандартный вывод
6	3 12
12 6 8 3 4 7	3 8
	2 7
	1 6
	1 4

Задача D. Приоритетная очередь

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Требуется реализовать с помощью кучи приоритетную очередь, поддерживающую две операции: добавить элемент и извлечь максимальный элемент.

В этой задаче в структуре могут храниться **одинаковые элементы**. Введем дополнительные правила на процедуры *SiftUp* и *SiftDown* для однозначности.

1. Процедуры просеивания не должны перемещать элемент дальше, чем это действительно необходимо. Например, если $A[i] = A[2i]$, то вызов *SiftUp*($2i$) 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**.

Формат входных данных

В первой строке вводятся два числа — максимальный размер приоритетной очереди N и количество запросов M . ($1 \leq M, N \leq 10^5$). Далее идут M строк, в каждой строке — по одному запросу.

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

Запрос 1-го типа: извлечь максимальный элемент (без параметров);

Запрос 2-го типа: добавить данный элемент в очередь. Запрос имеет один параметр — число из диапазона $[-10^9; 10^9]$.

Формат выходных данных

В ответ на запрос типа 1 следует вывести:

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

В ответ на запрос типа 2 следует вывести:

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

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

Пример

стандартный ввод	стандартный вывод
4 7	-1
1	1
2 9	2
2 4	3
2 9	2
2 9	-1
2 7	2 9
1	9 4 9

Задача Е. Приоритетная очередь с удалением

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

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 *SiftUp* и *SiftDown* для однозначности.

1. Процедуры просеивания не должны перемещать элемент дальше, чем это действительно необходимо. Например, если $A[i] = A[2i]$, то вызов *SiftUp*($2i$) 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**.

Формат входных данных

В первой строке вводятся два числа — максимальный размер приоритетной очереди N и количество запросов M . ($1 \leq M, N \leq 10^5$). Далее идут M строк, в каждой строке — по одному запросу.

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

Запрос 1-го типа: извлечь максимальный элемент (без параметров);

Запрос 2-го типа: добавить данный элемент в очередь. Запрос имеет один параметр — число из диапазона $[-10^9; 10^9]$.

Запрос 3-го типа: удалить произвольный элемент. Запрос имеет один параметр i — индекс элемента в куче.

Формат выходных данных

В ответ на запрос типа 1 следует вывести:

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

В ответ на запрос типа 2 следует вывести:

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

В ответ на запрос типа 3 следует вывести:

- -1 , если элемента с таким индексом нет и удаление невозможно. (При этом куча не должна измениться);
- Иначе — значение удалённого элемента.

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

Пример

стандартный ввод	стандартный вывод
4 10	-1
1	1
2 9	2
2 4	3
2 9	2
2 9	-1
2 7	2 9
1	-1
3 4	4
2 1	9
3 3	9 4 1

Задача F. Пирамидальная сортировка

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Дан массив, состоящий из N элементов. Отсортируйте его, используя пирамидальную сортировку. $1 \leq N \leq 10^5$. Числа в массиве не превосходят 10^9 по модулю.

Формат входных данных

Первая строка содержит количество чисел в массиве.
Вторая строка содержит сам массив — N целых чисел.

Формат выходных данных

Выведите массив, отсортированный в порядке неубывания.

Пример

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