

Задача А. Биномиальная куча

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 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 ($1 \leq N \leq 1000, 1 \leq M \leq 1\,000\,000$). 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.

Пример

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

Задача В. Двоичное дерево поиска

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Реализуйте splay-дерево.

Формат входных данных

Входной файл содержит описание операций с деревом, их количество не превышает 100000. В каждой строке находится одна из следующих операций:

- **insert** x — добавить в дерево ключ x . Если ключ x в дереве уже есть, то ничего делать не надо.
- **delete** x — удалить из дерева ключ x . Если ключа x в дереве нет, то ничего делать не надо.
- **exists** x — если ключ x есть в дереве, выведите «**true**», иначе «**false**»
- **next** x — выведите минимальный элемент в дереве, строго больший x , или «**none**», если такого нет.
- **prev** x — выведите максимальный элемент в дереве, строго меньший x , или «**none**», если такого нет.

Все числа во входном файле целые и по модулю не превышают 10^9 .

Формат выходных данных

Выведите последовательно результат выполнения всех операций **exists**, **next**, **prev**. Следуйте формату выходного файла из примера.

Пример

стандартный ввод	стандартный вывод
insert 2	true
insert 5	false
insert 3	5
exists 2	3
exists 4	none
next 4	3
prev 4	
delete 5	
next 4	
prev 4	

Задача С. Переворот

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Дан массив. Надо научиться обрабатывать два типа запросов.

- 1 L R - перевернуть отрезок [L, R]
- 2 L R - найти минимум на отрезке [L, R]

Напишите сплей-дерево.

Формат входных данных

Первая строка файла содержит два числа n, m . ($1 \leq n, m \leq 10^5$) Во второй строке находится n чисел a_i ($1 \leq a_i \leq 10^9$)- исходный массив. Остальные m строк содержат запросы, в формате описанном в условии. Для чисел L,R выполняется ограничение ($1 \leq L \leq R \leq n$).

Формат выходных данных

На каждый запрос типа 2, во входной файл выведите ответ на него, в отдельной строке.

Пример

стандартный ввод	стандартный вывод
10 7	3
5 3 2 3 12 6 7 5 10 12	2
2 4 9	2
1 4 6	2
2 1 8	
1 1 8	
1 8 9	
2 1 7	
2 3 6	

Задача D. Персистентная приоритетная очередь

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Требуется реализовать структуру данных, которая хранит мультимножество и умеет изменять любую свою предыдущую версию, выполняя одну из этих операций:

1. Заданы v и x , требуется добавить в множество v элемент со значением x , после чего вывести минимальный элемент в получившемся множестве.
2. Заданы v и u , требуется объединить множества с номерами v и u , после чего вывести минимальный элемент в получившемся множестве.
3. Задано v , требуется вывести минимальный элемент в множестве v , после чего удалить минимальный элемент из множества v . Если множество пустое, то вывести, что множество пустое, и создать новое пустое множество.

Изначально есть одно пустое множество с номером 0. После операции с номером i множество, получаемое во время этой операции, получает номер i .

Формат входных данных

Первая строка содержит число n — количество операций для выполнения.

От вас потребуется отвечать на запросы в онлайн, при этом поддерживая переменную s . Она изначально равна нулю. После каждой операции, она пересчитывается следующим образом через предыдущее значение: если ответ на запрос равен x , то $s = (s_{old} + x) \bmod 239017$. Если же ответом на запрос является слово **empty**, то s не изменяется.

В следующих n строках заданы запросы.

Запросы первого типа описываются строкой **1 a b**, где a и b — неотрицательные целые числа, которые описывают v и x для соответствующего запроса, как $v = (a + s) \bmod i$ и $x = (b + 17s) \bmod (10^9 + 1)$, где i — номер соответствующего запроса.

Запросы второго типа описываются строкой **2 a b**, где a и b — неотрицательные целые числа, которые описывают v и u для соответствующего запроса, как $v = (a + s) \bmod i$ и $u = (b + 13s) \bmod i$, где i — номер соответствующего запроса.

Запросы третьего типа описываются строкой **3 a**, где a — неотрицательное целое число, которые описывает v для соответствующего запроса, как $v = (a + s) \bmod i$, где i — номер соответствующего запроса.

Число запросов не превышает 200 000. Гарантируется, что мощность любого созданного мультимножества не превышает 2^{63} .

Формат выходных данных

Требуется вывести ровно n строк, в каждой строке должно находиться неотрицательное целое число либо слово **empty**.

Для запросов первого и второго типа требуется вывести значение минимального элемента в только что созданном множестве, либо слово **empty**, если множество пустое.

Для запросов третьего типа требуется вывести минимальный элемент в множестве, либо слово **empty**, если множество пустое.

Пример

стандартный ввод	стандартный вывод
9	2
1 0 2	3
1 0 999999970	2
2 2 0	2
3 0	2
2 4 4	2
3 0	2
3 0	3
3 0	empty
3 8	

Задача Е. Асхат и дерево

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Это интерактивная задача.

Дано двоичное дерево поиска размера n , в каждой вершине есть значение от 1 до n . Петух по имени Асхат каждый день нумерует вершины числами от 1 до n , даёт вам номер корня r и просит вас найти номер вершины со значением x .

Вы можете совершать запросы — по номеру вершины узнать значение в ней и номера её детей. Пусть максимальное расстояние от корня до вершины в i -й день равно d . Тогда в i -й день вы можете совершить не более, чем d запросов. За все дни вы можете совершить не более, чем 70000 запросов.

Также вы можете менять детей у каждой вершины. Пусть в i -й день длина пути от корня до вершины с номером x равна s . Тогда вам в i -й день разрешено сделать не более, чем s запросов вида «установить у вершины новых детей». За все дни вы можете совершить не более, чем 70000 запросов этого типа.

Протокол взаимодействия

В первой строке ввода даны числа n и q ($1 \leq n \leq 2000$, $1 \leq q \leq 2000$) — размер дерева и число запросов, соответственно.

Для каждого запроса даны числа r и x ($1 \leq r, x \leq n$) — номер корня дерева и значение, вершину с которым требуется найти. Вы не сможете считать эти числа для следующего запроса, пока не дадите ответ на текущий.

Чтобы обратиться к вершине с номером i выведите «ask i » в отдельной строке. В ответ даются три числа val , L и R ($1 \leq val \leq n$, $0 \leq L, R \leq n$) — значение в этой вершине и номера левого и правого ребёнка, соответственно. В случае, если у вершины нет левого или правого ребёнка, $L = 0$ или $R = 0$, соответственно.

Чтобы поменять детей вершины с номером i на вершины с номером L и R выведите «change i L R » в отдельной строке. Чтобы у вершины с номером i не было левого или правого ребёнка, выведите 0 вместо L или R , соответственно. После выполнения этого запроса граф может перестать быть двоичным деревом поиска.

Если искомое значение находится в вершине с номером i и вы совершили все нужные изменения, выведите «confirm i » в отдельной строке. После этого вершины перенумеруются, а на ввод будет дан новый запрос. Если на момент выполнения этого запроса граф не является двоичным деревом поиска, вы получите вердикт «Wrong answer».

Задача F. Фибоначчиева куча

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Эта задача для тех, кто завалил регион (или его дисквалифицировали), отчаялся в олимпиадной проге и решил полностью посвятить себя теоретической информатике и продвинутым алгоритмам. Вам предстоит написать самую настоящую Фибоначчиеву кучу. Все запросы будут задаваться массивом a_i , где a_1 вам дано изначально, а для $i > 1$ $a_i = (a_{i-1} * b + c) \bmod 2^{32}$.

К вам будут поступать запросы двух типов:

1. В случае, если a_i чётно, то в i -м запросе от вас будет требоваться добавить число a_i в кучу.
2. В случае, если a_i нечётно от вас будет требоваться узнать значение верхнего (максимального) элемента в куче.

Формат входных данных

В первой строке вам дано единственное число n ($1 \leq n \leq 10^8$). В следующей строке вам дано 3 числа a_1 b и c ($0 \leq a_1, b, c \leq 2^{32}$). Гарантируется, что a_1 чётно.

Формат выходных данных

Для всех запросов второго типа выведите сумму ответов на них.

Пример

стандартный ввод	стандартный вывод
10 2 2 1	18

Замечание

Так как Фибоначчиева куча сложная и её код трудно читать, в этой задаче не будет ревью кода.

Задача G. Стёпа и Маша

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2.5 секунд
Ограничение по памяти:	64 мегабайта

Стёпе нравится Маша. А Маше нравится Стёпа. Именно поэтому тёплым весенним деньком Маша и Стёпа решили отправиться в парк на романтическую прогулку. Они беззаботно шли по тропинке, держась за руки, как вдруг подошли к странному спортивному снаряду. Он состоял из n подряд идущих столбиков, расположенных близко друг к другу.

Стёпа быстро оценил высоту каждого из них, и предположил, что высота столбика с номером i равняется a_i метров.

Чтобы произвести впечатление на Машу, Стёпа решил выполнить следующее упражнение: он прыгает на столбик с номером 1 и затем $k - 1$ раз повторяет следующую процедуру: пусть он стоит на столбике с номером i . Тогда прыгает на столбик j с минимальным номером, таким что $j > i$ и $a_j > a_i$. Проще говоря, он прыгает на ближайший столбик с большим номером и большей высотой. Если же такого столбика нет, Стёпа теряет надежду заполучить сердце Маши, плачет и уходит домой заниматься дифференциальной геометрией.

Стёпа уже выбрал число k и подошел к снаряду, как понял, что катастрофически ошибся. Слабое зрение Стёпы подвело его, и он неправильно оценил высоту некоторых столбиков.

— Ничего страшного - подумал Стёпа — и не такое случалось. Если высота этого столбика 16394 метра, то...

И вдруг Стёпа испугался. Он понял, на какую высоту ему придется залезть и понял, что это слишком опасно.

— Я еще так молод — бормотал под нос Стёпа — я впервые влюбился, я только полюбил эту жизнь... И терять её из-за этого снаряда я не готов!

Поэтому Стёпа решил немного уменьшить k , чтобы так не рисковать.

Но неудачи, казалось, преследовали Стёпу: он то обнаруживал, что высота столбика неверна, то число k ему казалось неподходящим. Действительно: если он залезет слишком низко, Маша не оценит его способности, а если слишком высоко, есть шанс упасть.

И каждое такое изменение заставляло Стёпу пересчитывать высоту, на которой он в итоге окажется. Казалось, что он будет вечность решать, что же делать, как вдруг Маша крикнула:

— Давай быстрее, милый! Я жду!

Больше откладывать выполнение упражнения было нельзя. Напишите программу, которая будет считать, на какой высоте окажется Стёпа после упражнения.

Формат входных данных

В первой строке входного файла находятся 2 целых числа ($1 \leq n, q \leq 500\,000$) — количество столбиков и запросов Стёпы, соответственно.

Во второй строке находится n целых чисел a_i ($1 \leq a_i \leq 10^9$) разделенные пробелами — начальные оценки высот столбиков Стёпой.

Следующие q строк содержат запросы Стёпы. Первое число в строке t_i ($1 \leq t_i \leq 2$) означает его тип.

Если $t_i = 1$, то далее следуют два числа p_i x_i ($1 \leq p_i \leq n, 1 \leq x_i \leq 10^9$) - теперь Степа считает высоту p_i столба равной x_i .

Если $t_i = 2$, то далее следуют одно число k_i ($1 \leq k_i \leq n$) — количество прыжков Стёпы.

Формат выходных данных

На каждый запрос второго типа выведите высоту на которую заберется Стёпа или '-1' без кавычек, если он не сможет выполнить заданное количество прыжков.

Пример

стандартный ввод	стандартный вывод
5 13	1
1 2 1 3 5	2
2 1	5
2 2	8
2 4	9
1 1 8	-1
1 3 6	28
1 2 9	-1
2 1	
2 2	
2 3	
1 5 28	
2 3	
1 1 333	
2 2	