Задача A. 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 регоин (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 $i > 1$ $a_i = (a_{i-1} * b + c) \mod 2^{32}$.

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 ($1 \leqslant n \leqslant 10^8$). В следующей строке вам дано 3 числа a_1 b и c ($0 \leqslant a_1, b, c \leqslant 2^{32}$). 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 B. Два-Три-Де... Дерево

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2.3 секунд
Ограничение по памяти:	256 мегабайт

В этой задаче от вас требуется реализовать добавление элемента в структуру данных «2-3 дерево». Для того, чтобы построение дерева было однозначным, будем считать, что элемент отправляется в поддерево с наибольшим максимумом (разумеется, среди тех, в которые его можно добавить).

После того, как вы построите 2-3 дерево, выведите его.

Формат входных данных

В первой строке вводится натуральное число n ($1 \leqslant n \leqslant 10^6$) — количество запросов на добавление в 2-3 дерево. Во второй строке вводится n различных натуральных чисел a_i ($1 \leqslant a_i \leqslant n$) — запросы добавления. Обратите внимание, что выполнять запросы нужно именно в этом порядке.

Формат выходных данных

Выведите все листья дерева в отсортированном порядке. Разделяйте два соседних числоа латинской буквой, которая будет соответствовать глубине LCA этих вершин. Будем считать, что корню соответствует символ 'A', вершинам на глубине 1 соответствует символ 'B', и так далее. В вашем выводе должно оказаться n чисел и $n - 1$ латинский символ.

Примеры

стандартный ввод	стандартный вывод
10 1 2 3 4 5 6 7 8 9 10	1 C 2 B 3 C 4 A 5 C 6 B 7 C 8 B 9 C 10
2 1 2	1 A 2

Задача C. 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 онлайне, 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 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 1 0 2 1 0 999999970 2 2 0 3 0 2 4 4 3 0 3 0 3 0 3 8	2 3 2 2 2 2 2 3 empty

Задача D. 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 ($1 \leqslant N \leqslant 1000, 1 \leqslant M \leqslant 1\,000\,000$). 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