Задача A. Снеговики

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мегабайта

Зима. 2012 год. На фоне грядущего Апокалипсиса и конца света незамеченной прошла новость об очередном прорыве в областях клонирования и снеговиков: клонирования снеговиков. Вы конечно знаете, но мы вам напомним, что снеговик состоит из нуля или более вертикально поставленных друг на друга шаров, а клонирование — это процесс создания идентичной копии (клона).

В местечке Местячково учитель Андрей Сергеевич Учитель купил через интернет-магазин «Интернет-магазин аппаратов клонирования» аппарат для клонирования снеговиков. Теперь дети могут играть и даже играют во дворе в следующую игру. Время от времени один из них выбирает понравившегося снеговика, клонирует его и:

- либо добавляет ему сверху один шар;
- либо удаляет из него верхний шар (если снеговик не пустой).

Учитель Андрей Сергеевич Учитель записал последовательность действий и теперь хочет узнать суммарную массу всех построенных снеговиков.

Формат входных данных

Первая строка содержит количество действий n ($1 \leqslant n \leqslant 200\,000$). В строке номер $i + 1$ содержится описание действия i:

- `t m` — клонировать снеговика номер t ($0 \leqslant t < i$) и добавить сверху шар массой m ($0 < m \leqslant 1000$);
- `t 0` — клонировать снеговика номер t ($0 \leqslant t < i$) и удалить верхний шар. Гарантируется, что снеговик t не пустой.

В результате действия i, описанного в строке $i + 1$ создается снеговик номер i. Изначально имеется пустой снеговик с номером ноль.

Все числа во входном файле целые.

Формат выходных данных

Выведите суммарную массу построенных снеговиков.

Пример

стандартный ввод	стандартный вывод
8 0 1 1 5 2 4 3 2 4 3 5 0 6 6 1 0	74

Задача C. Li-Chao Tree с добавлением на отрезке

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

Пусть S — множество прямых на плоскости. Изначально S пусто. Необходимо отвечать на запросы двух типов:

1. Добавить в множество S отрезок прямой, задаваемой уравнением $y = a_i \cdot x + b_i$, ограниченной по оси X отрезком $[l_i, r_i]$, где a_i, b_i, l_i и r_i — параметры запроса.
2. Найти $\min\limits_{f \in S} f(x_i)$, где x_i — параметр запроса. Иными словами, необходимо найти минимальное значение среди всех прямых из множества S в точке x_i.

Формат входных данных

Первая строка содержит одно целое число q ($2 \leqslant q \leqslant 200\,000$) — количество запросов.

Каждая из следующих q строк содержит описание очередного запроса. Описание каждого запроса начинается с целого числа t_i ($t_i \in \{1, 2\}$), обозначающего тип запроса.

Если $t_i = 1$, далее следуют четыре целых числа a_i, b_i, l_i и r_i ($-200\,000 \leqslant a_i, b_i, l_i, r_i \leqslant 200\,000$, $l_i \leqslant r_i$) — параметры добавляемой линейной функции. Это означает, что вы должны добавить в множество отрезок прямой, задаваемой уравнением $y = (a_i + p \bmod 200\,000) \cdot x + (b_i + p \bmod 200\,000)$, где p — ответ на последний поступивший запрос второго типа. Прямая ограничена по оси X отрезком $[l_i, r_i]$. Изначально $p = 0$. **Обратите внимание, что по модулю берется только число p, а не суммы $a_i + p$ и $b_i + p$.**

Если $t_i = 2$, далее следует одно целое число x_i ($-200\,000 \leqslant x_i \leqslant 200\,000$) — точка, в которой необходимо найти минимальное значение среди всех добавленных прямых.

Гарантируется, что есть хотя бы один запрос типа 2.

Формат выходных данных

Для каждого запроса типа 2 выведите одно целое число — ответ на запрос. **Если данная в запросе точка не покрыта не одним отрезком прямой, считайте, что ответ равен числу -1.**

Примеры

стандартный ввод	стандартный вывод
8 1 2 0 0 1 1 2 -4 2 3 1 0 1 0 3 2 0 2 1 2 2 2 3 2 4	0 1 0 1 -1
4 1 1 1 0 2 2 1 1 -2 -2 0 2 2 1	2 0

Задача D. K-я порядковая статистика на отрезке

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

Дан массив из N неотрицательных чисел, строго меньших 10^9. Вам необходимо ответить на несколько запросов о величине k-й порядковой статистики на отрезке $[l, r]$.

Формат входных данных

В первой строке дано единственное число N ($1 \leqslant N \leqslant 450\,000$) — размер массива.

Во второй строке даны N целых чисел $a_1,\ a_1,\ \ldots,\ a_N$ ($0 \leqslant a_i < 10^9$) — числа массива.

В третей строке дано единственное число M ($1 \leqslant M \leqslant 600\,000$) — число запросов.

В следующих строках описаны запросы. В i-й строке даны три целых числа l'_i, r'_i, k_i. Параметры i-го запроса строются по следующей формуле: $l_i = ((l'_i + ans_{i-1}) \bmod N) + 1$, $r_i = ((r'_i + ans_{i-1}) \bmod N) + 1$, где ans_{i-1} равно ответу на предыдущий запрос (считайте, что $ans_0 = 0$). Соответственно, в i-м запросе требуется найти значение k_i-й порядковой статистики на отрезке $[l_i, r_i]$. Гарантируется, что $1 \leqslant l_i \leqslant r_i \leqslant N$, $1 \leqslant k_i \leqslant (r_i - l_i + 1)$.

Формат выходных данных

Выведите единственное число — сумму ответов на запросы.

Пример

стандартный ввод	стандартный вывод
5 1 2 3 4 5 5 5 7 2 4 7 3 1 5 5 7 7 1 2 5 1	15

Замечание

В первом примере:

1. В первом запросе $l_1 = (5 \bmod 5) + 1 = 1$, $r_i = (7 \bmod 5) + 1 = 3$, на отрезке $[1, 3]$ 2 порядковая статистика равна 2.
2. Во втором запросе $l_2 = ((4 + 2) \bmod 5) + 1 = 2$, $r_2 = ((7 + 2) \bmod 5) + 1 = 5$, на отрезке $[2, 4]$ 3 порядковая статистика равна 4.
3. В третьем запросе $l_3 = ((1 + 4) \bmod 5) + 1 = 1$, $r_3 = ((5 + 4) \bmod 5) + 1 = 5$, на отрезке $[1, 5]$ 5 порядковая статистика равна 5.
4. В четвертом запросе $l_4 = ((7 + 5 \bmod 5) + 1 = 3$, $r_4 = ((7 + 5) \bmod 5) + 1 = 3$, на отрезке $[3, 3]$ 1 порядковая статистика равна 3.
5. В пятом запросе $l_5 = ((2 + 3) \bmod 5) + 1 = 1$, $r_t = ((5 + 3) \bmod 5) + 1 = 4$, на отрезке $[1, 4]$ 1 порядковая статистика равна 1.

Сумма ответов равна $2 + 4 + 5 + 3 + 1 = 15$.

Задача E. Соединение и разъединение

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Вы когда-нибудь слышали про обход в глубину? Например, используя этот алгоритм, вы можете проверить является ли граф связным за время $O(E)$. Вы можете даже посчитать количество компонент связности за то же время.

А вы когда нибудь слышали про систему непересекающихся множеств? Используя эту структуру, вы можете быстро обрабатывать запросы "Добавить ребро в граф" и "Посчитать количество компонент связности в графе".

А вы когда-нибудь слышали о *динамической* задаче связности? В этой задаче вам необходимо обрабатывать три типа запросов:

1. Добавить ребро в граф.
2. Удалить ребро из графа.
3. Посчитать количество компонент связности в графе.

Можно считать, что граф является неориентированным. Изначально граф является пустым.

Формат входных данных

В первой строке находятся два целых числа N и K — количество вершин и количество запросов, сответственно ($1 \leqslant N \leqslant 300\,000$, $0 \leqslant K \leqslant 300\,000$). Следующие K строк содержат запросы, по одному в строке. Каждый запрос имеет один из трех типов:

1. `+` u v: Добавить ребро между вершинами u и v. Гарантируется, что такого ребра нет.
2. `-` u v: Удалить ребро между u и v. Гарантируется, что такое ребро есть.
3. `?`: Посчитать количество компонент связности в графе.

Вершины пронумерованы целыми числами от 1 до N. Во всех запросах $u \neq v$.

Формат выходных данных

Для каждого запроса типа '`?`', Выведите количество компонент связности в момент запроса.

Пример

стандартный ввод	стандартный вывод
5 11 ? + 1 2 + 2 3 + 3 4 + 4 5 + 5 1 ? - 2 3 ? - 4 5 ?	5 1 1 2

Задача F. Комбокамень

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Одна большая компания разрабатывает компьютерную игру «Комбокамень», которая должна мигом перевернуть всю индустрию. Правила игры достаточно сложные, и на реализацию серверного движка, моделирующего ход игры, был объявлен конкурс. От вас требуется реализовать подобный серверный движок.

Суть игры — игроки умеют призывать на арену и усиливать существ, используя заклинания, а также заставлять их сражаться друг с другом. Каждое существо имеет два параметра — численное значение атаки a и численное значение оставшегося здоровья h. Для краткости будем обозначать параметры существа как (a, h). Исходно на арене нет существ.

Игроку доступны следующие заклинания:

- *Призыв существа*: Призвать новое существо с характеристиками $(1, 1)$. Если уже в игру было введено k существ, то новое существо получает номер $k + 1$.
- *Благословение силы*: Удвоить атаку выбранного существа. Если до применения этого заклинания оно имело характеристики (a, h), то после этого действия оно будет иметь характеристики $(2a, h)$.
- *Божественный дух*: Удвоить здоровье выбранного существа. Если до применения этого заклинания оно имело характеристики (a, h), то после этого действия оно будет иметь характеристики $(a, 2h)$.
- *Копия из лавы*: Призвать новое существо, которое будет иметь такие же характеристики, как и выбранное заклинанием существо. Если уже в игру было введено k существ, то новое существо получает номер $k + 1$.
- *Сражайся!*: Заставить двух различных существ сразиться. Во время сражения оба существа одновременно наносят друг другу по одному удару, уменьшая количество здоровья соперника на значение своей атаки. Так, если сражаются два существа с характеристиками (a_1, h_1) и (a_2, h_2), то после сражения они будут иметь характеристики $(a_1, h_1 - a_2)$ и $(a_2, h_2 - a_1)$, соответственно. Если после сражения у существа остается 0 или меньше единиц здоровья, оно умирает и больше не может участвовать в игре.

От серверного движка, на реализацию которого объявлен конкурс, требуется способность промоделировать все события и для каждого созданного во время игры существа вывести номер хода, на котором оно погибло, либо определить, что оно осталось живо к концу игры.

Кроме того, движок должен корректно обрабатывать случаи, когда игрок пытается взаимодействовать с мертвыми по мнению сервера существами: если заклинание *Благословение силы*, *Божественный дух* или *Сражайся!* обращено к уже мертвому существу, то не должно произойти ничего. Если заклинание *Копия из лавы* применено к мертвому существу, создается его мертвая копия с такими же характеристиками, но умершая на текущем ходу, в момент копирования.

Формат входных данных

В первой строке дано число n — количество совершенных ходов ($1 \leqslant n \leqslant 250\,000$).

В следующих n строках даны ходы, пришедшие к серверному движку, в следующем формате:

- 1 — применить заклинание *Призыв существа*;
- 2 i — применить заклинание *Благословение силы* к существу с номером i;
- 3 i — применить заклинание *Божественный дух* к существу с номером i;
- 4 i — применить заклинание *Копия из лавы* к существу с номером i;
- 5 i j — применить заклинание *Сражайся!* к существам с номерами i и j.

Гарантируется, что любые упомянутые в запросах существа к моменту запроса уже были призваны, но, возможно, могут уже быть мертвы.

Формат выходных данных

В первой строке выведите одно целое число k — количество существ, призванных за время игры.

В следующей строке выведите k целых чисел $t_1, t_2, \ldots, t_k$ — если существо с номером i осталось живо к концу игры, то t_i должно быть равно -1, иначе t_i должно быть равно номеру хода, на котором оно погибло.

Пример

стандартный ввод	стандартный вывод
16 1 2 1 3 1 1 5 1 2 3 1 1 3 3 3 3 4 1 5 1 3 3 3 5 1 3 5 4 3 5 4 3 4 1	5 13 5 14 -1 16

Замечание

В таблице можно увидеть, как изменялись характеристики существ в первом примере.

ход	1	2	3	4	5
0	-	-	-	-	-
1	(1, 1)	-	-	-	-
2	(2, 1)	-	-	-	-
3	(2, 2)	-	-	-	-
4	(2, 2)	(1, 1)	-	-	-
5	(2, 1)	мертво	-	-	-
6	(2, 2)	мертво	-	-	-
7	(2, 2)	мертво	(1, 1)	-	-
8	(2, 2)	мертво	(1, 2)	-	-
9	(2, 2)	мертво	(1, 4)	-	-
10	(2, 2)	мертво	(1, 4)	(2, 2)	-
11	(2, 1)	мертво	(1, 2)	(2, 2)	-
12	(2, 1)	мертво	(1, 4)	(2, 2)	-
13	мертво	мертво	(1, 2)	(2, 2)	-
14	мертво	мертво	мертво	(2, 1)	-
15	мертво	мертво	мертво	(2, 1)	-
16	мертво	мертво	мертво	(2, 1)	мертво

Задача G. Минимизируй!

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Дан массив целых чисел a длины n. Поступает q запросов двух типов:

- $1\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $\min(a_i, x)$.
- $2\ l\ r$. Необходимо вывести сумму элементов массива a на отрезке от l до r включительно.

Формат входных данных

В первой строке входных данных дано целое число n ($1 \leqslant n \leqslant 300\,000$) — количество элементов массива a.

Во второй строке даны n целых чисел a_i ($1 \leqslant a_i \leqslant 10^9$) — элементы массива a.

В третьей строке дано целое число q ($1 \leqslant q \leqslant 300\,000$) — количество запросов.

В последующих q строках даны запросы по одному в строке.

Запрос первого типа задается так: $1\ l\ r\ x$

Где $1 \leqslant l \leqslant r \leqslant n$, $1 \leqslant x \leqslant 10^9$ — целые числа. Это означает, что все элементы массива a на отрезке от l до r нужно заменить на минимум из текущего значения и x.

Запрос второго типа задается так: $2\ l\ r$

Где $1 \leqslant l \leqslant r \leqslant n$ — целые числа. Это означает, что нужно вывести сумму элементов массива a на отрезке от l до r.

Формат выходных данных

Для каждого запроса второго типа выведите в отдельной строке сумму элементов на соответствующем отрезке.

Примеры

стандартный ввод	стандартный вывод
3 1 4 2 5 2 1 3 1 1 3 3 2 1 3 1 1 3 1 2 1 3	7 6 3
7 1 7 2 4 8 4 100 7 1 3 6 3 2 2 7 1 2 3 5 2 1 7 1 1 7 3 2 1 4 2 2 7	118 117 9 17

Задача H. Минимизируй, прибавляй!

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Дан массив целых чисел a длины n. Поступает q запросов трех типов:

- $1\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $\min(a_i, x)$.
- $2\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $a_i + x$.
- $3\ l\ r$. Необходимо вывести сумму элементов массива a на отрезке от l до r включительно.

Формат входных данных

В первой строке входных данных дано целое число n ($1 \leqslant n \leqslant 300\,000$) — количество элементов массива a.

Во второй строке даны n целых чисел a_i ($-10^9 \leqslant a_i \leqslant 10^9$) — элементы массива a.

В третьей строке дано целое число q ($1 \leqslant q \leqslant 300\,000$) — количество запросов.

В последующих q строках даны запросы по одному в строке.

Запрос первого типа задается так: $1\ l\ r\ x$

Где $1 \leqslant l \leqslant r \leqslant n$, $-10^9 \leqslant x \leqslant 10^9$. Это означает, что все элементы массива a на отрезке от l до r нужно заменить на минимум из текущего значения и x.

Запрос второго типа задается так: $2\ l\ r\ x$

Где $1 \leqslant l \leqslant r \leqslant n$, $-10^7 \leqslant x \leqslant 10^7$. Это означает, что ко всем элементам массива a на отрезке от l до r нужно прибавить x.

Запрос третьего типа задается так: $3\ l\ r$

Где $1 \leqslant l \leqslant r \leqslant n$. Это означает, что нужно вывести сумму элементов массива a на отрезке от l до r.

Формат выходных данных

Для каждого запроса третьего типа выведите в отдельной строке сумму элементов на соответствующем отрезке.

Примеры

стандартный ввод	стандартный вывод
3 1 4 2 9 3 1 3 1 1 3 3 3 1 3 1 1 3 1 3 1 3 2 1 3 5 3 1 3 1 1 3 3 3 1 3	7 6 3 18 9
7 1 7 2 4 8 4 100 10 1 3 6 3 3 2 7 1 2 3 5 2 3 4 -10 3 1 7 1 1 7 3 3 1 4 3 2 7 2 1 7 5 3 1 7	118 97 -11 -3 33

Задача I. Kinetic segment tree

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мегабайт

Даны три массива k, b и t длины n. Поступает q запросов 4 типов:

- $1\ l\ r$. Требуется вывести $\min\limits_{i=l}^{r}(k_i t_i + b_i)$.
- $2\ l\ r\ d$. Требуется прибавить d к b_l, b_{l+1}, …, b_r.
- $3\ l\ r\ d$ ($d \geqslant 0$). Требуется увеличить t_l, t_{l+1}, …, t_r на d.
- $4\ i\ x\ y\ z$. Требуется изменить k_i на x, b_i на y и t_i на z.

Формат входных данных

Первая строка содержит n ($1 \leqslant n \leqslant 10^6$).
Вторая строка содержит k_1, k_2, …, k_n ($-10^6 \leqslant k_i \leqslant 10^6$).
Третья строка содержит b_1, b_2, …, b_n ($-10^6 \leqslant b_i \leqslant 10^6$).
Четвёртая строка содержит t_1, t_2, …, t_n ($-10^6 \leqslant t_i \leqslant 10^6$).
Пятая строка содержит q ($1 \leqslant q \leqslant 10^6$).
Следующие q строк содержат описание запросов в одном из следующих форматов:

- $1\ l\ r$ ($1 \leqslant l \leqslant r \leqslant n$)
- $2\ l\ r\ d$ ($1 \leqslant l \leqslant r \leqslant n$, $-10^6 \leqslant d \leqslant 10^6$)
- $3\ l\ r\ d$ ($1 \leqslant l \leqslant r \leqslant n$, $0 \leqslant d \leqslant 10^6$)
- $4\ i\ x\ y\ z$ ($1 \leqslant i \leqslant n$, $-10^6 \leqslant x, y, z \leqslant 10^6$)

Формат выходных данных

Для каждого запроса первого типа выведите искомый минимум.

Пример

стандартный ввод	стандартный вывод
5 1 0 -1 -2 -3 -1 0 1 2 10000 -10 -10 -10 -10 -10 8 1 1 5 2 2 2 -12 1 1 5 3 3 5 20 1 1 5 1 1 3 4 1 100 -100 -100 1 1 5	-11 -12 -18 -12 -10100

Задача J. Минимизируй, прибавляй, НОДируй и их друзья

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мегабайт

Дан массив целых чисел a длины n. Поступает q запросов восьми типов:

- $1\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $\min(a_i, x)$.
- $2\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $\max(a_i, x)$.
- $3\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на x.
- $4\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $a_i + x$.
- $5\ l\ r$. Необходимо вывести сумму элементов массива a на отрезке от l до r включительно.
- $6\ l\ r$. Необходимо вывести минимум элементов массива a на отрезке от l до r включительно.
- $7\ l\ r$. Необходимо вывести максимум элементов массива a на отрезке от l до r включительно.
- $8\ l\ r$. Необходимо вывести НОД (наибольший общий делитель) элементов массива a на отрезке от l до r включительно.

Формат входных данных

В первой строке входных данных дано целое число n ($1 \leqslant n \leqslant 200\,000$) — количество элементов массива a.

Во второй строке даны n целых чисел a_i ($1 \leqslant a_i \leqslant 10^9$) — элементы массива a.

В третьей строке дано целое число q ($1 \leqslant q \leqslant 200\,000$) — количество запросов.

В последующих q строках даны запросы по одному в строке.

Запрос первого типа задается так: $1\ l\ r\ x$

Где $1 \leqslant l \leqslant r \leqslant n$, $1 \leqslant x \leqslant 10^9$ — целые числа. Это означает, что все элементы массива a на отрезке от l до r нужно заменить на минимум из текущего значения и x.

Запрос второго типа задается так: $2\ l\ r\ x$

Где $1 \leqslant l \leqslant r \leqslant n$, $1 \leqslant x \leqslant 10^9$ — целые числа. Это означает, что все элементы массива a на отрезке от l до r нужно заменить на максимум из текущего значения и x.

Запрос третьего типа задается так: $3\ l\ r\ x$

Где $1 \leqslant l \leqslant r \leqslant n$, $1 \leqslant x \leqslant 10^9$ — целые числа. Это означает, что все элементы массива a на отрезке от l до r нужно заменить на x.

Запрос четвертого типа задается так: $4\ l\ r\ x$

Где $1 \leqslant l \leqslant r \leqslant n$, $1 \leqslant x \leqslant 10^7$ — целые числа. Это означает, что ко всем элементам массива a на отрезке от l до r нужно прибавить x.

Запрос пятого типа задается так: $5\ l\ r$

Где $1 \leqslant l \leqslant r \leqslant n$ — целые числа. Это означает, что нужно вывести сумму элементов массива a на отрезке от l до r.

Запрос шестого типа задается так: $6\ l\ r$

Где $1 \leqslant l \leqslant r \leqslant n$ — целые числа. Это означает, что нужно вывести минимум элементов массива a на отрезке от l до r.

Запрос седьмого типа задается так: $7\ l\ r$

Где $1 \leqslant l \leqslant r \leqslant n$ — целые числа. Это означает, что нужно вывести максимум элементов массива a на отрезке от l до r.

Запрос восьмого типа задается так: $8\ l\ r$

Где $1 \leqslant l \leqslant r \leqslant n$ — целые числа. Это означает, что нужно вывести наибольший общий делитель элементов массива a на отрезке от l до r.

Формат выходных данных

Для каждого запроса 5, 6, 7 и 8 типов выведите в отдельной строке ответ.

Пример

стандартный ввод	стандартный вывод
7 1 2 3 4 5 6 7 20 4 2 7 10 5 1 6 6 1 6 7 1 6 8 1 6 2 1 6 14 5 2 7 6 2 7 7 2 7 8 2 7 1 2 7 12 5 1 6 6 1 6 7 1 6 8 1 6 3 2 6 15 5 1 7 6 1 7 7 1 7 8 1 7	71 1 16 1 90 14 17 1 74 12 14 2 101 12 15 1

Задача K. Контрол Икс Контрол Вэ

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	960 мегабайт

Вам дана строка S состоящая из цифр 1, 2, 3. Длину строки обозначим за $|S|$.

У вас есть курсор. Будем обозначать его позицию за ℓ. Означать позиция будет следующее:

- Если $\ell = 0$, то курсор находится перед первым символом S.
- Если $\ell = |S|$, то курсор находится после последнего символа S.
- Если $0 < \ell < |S|$, то курсор находится между S_ℓ и $S_{\ell+1}$.

Будем называть S_{left} префикс строки S до позиции курсора, и S_{right} суффикс строки S после курсора. Иначе говоря, курсор «разрезает» строку S на S_{left} и S_{right}.

Еще у нас есть строка C, изначально пустая. С ней доступны операции трех видов:

- **Вырезать**. Присваивает $C \leftarrow S_{\text{right}}$, затем присваивает $S \leftarrow S_{\text{left}}$.
- **Вставить**. Приписывает строку C к строке S справа.
- **Подвинуть**. Сдвигает курсор на 1 вправо, увеличивает ℓ на единицу.

Изначально курсор находится в позиции $\ell = 0$. Затем мы делаем следующий алгоритм:

1. **Подвинуть**.
2. **Вырезать**.
3. **Вставить**, выполняется S_ℓ раз.
4. Если $\ell = |S|$, остановить работу. Иначе начать сначала с шага 1.

Собственно, что от вас требуется:

Вам дается строка S, и два натуральных числа x и n. Выполним наш алгоритм и дождемся момента, когда ℓ впервые достигнет значения x; это произойдет после очередного **Подвинуть**. Чему будут равны последние n символов у S_{left} в этот момент?

Боги копипасты утверждают, что в ходе работы алгоритма ℓ достигнет значения x.

Формат входных данных

В первой строке вводится число t — число тестов. Тесты перечисляются один за другим.

В первой строке для каждого теста вводятся числа x и n. В следующей строке для этого теста будет введена изначальная строка S.

- $1 \leqslant t \leqslant 250$
- $1 \leqslant |S| \leqslant 200$
- $n \leqslant x \leqslant 10^{16}$
- $1 \leqslant n \leqslant 200$

Формат выходных данных

Для каждого теста, выведите в новой строке n символов — ответ на тест.

Пример

стандартный ввод	стандартный вывод
3 7 3 2323 20 10 333 24 11 132133	323 3333333333 31332133213

Замечание

Разберем третий тест. Изначально у нас есть $S =$ 132133, $\ell = 0$ и $C = \varepsilon$ (пустая строка). Потомм происходит следующее:

- Step 1, *Подвинуть:* $\ell = 1$.
- Step 2, *Вырезать:* $S =$ 1 и $C =$ 32133.
- Step 3, *Вставить* $S_\ell =$ *1 раз:* получили $S =$ 132133.
- Step 4: $\ell = 1 \neq |S| = 6$, начнем заново.
- Step 1, *Подвинуть:* $\ell = 2$.
- Step 2, *Вырезать:* $S =$ 13 и $C =$ 2133.
- Step 3, *Вставить* $S_\ell =$ *3 раза:* $S =$ 13213321332133.
- Step 4: $\ell = 2 \neq |S| = 14$, начнем сначала.
- Step 1, *Подвинуть:* $\ell = 3$.
- Step 2, *Вырезать:* $S =$ 132 и $C =$ 13321332133.
- Step 3, *Вставить* $S_\ell =$ *2 раза:* $S =$ 132133213321333213321332133.
- Step 4: $\ell = 3 \neq |S| = 25$, начнем сначала.
- Step 1, *Подвинуть:* $\ell = 4$.
- Step 2, *Вырезать:* $S =$ 1321 и $C =$ 332133213313321332133.
- Step 3, *Вставить* $S_\ell =$ *1 раз:* $S =$ 132133213321333213321332133.
- Step 4: $\ell = 4 \neq |S| = 25$, Начнем сначала.
- Step 1, *Подвинуть:* $\ell = 5$.
- И так далее ...

В какой-то момент, ℓ достигнет $x = 24$, и S будет какой-то очень длинной строкой. Можно проверить, что в этот момент времени последние $n = 11$ символов S слева от курсора будут 31332133213.

Задача L. Урны и шары

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

Пусть у вас есть n урн, в каждой из которых лежит по одному шарику. Урна с номером i содержит шарик под номером i. У вас есть специальное устройство, которое позволяет перемещать шарики. Им чрезвычайно просто пользоваться: сначала вы выбираете некоторый отрезок последовательных урн. После этого вы выбираете некоторый другой отрезок последовательных урн такой же длины, как и исходный, и затем шарики из урн первого отрезка перемещаются в соответствующие урны второго отрезка.

Дана последовательность перемещений. Установите, в какой урне окажется каждый шарик.

Формат входных данных

Первая строка входных данных содержит два числа n и m — число урн и число перемещений, соответственно ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 50\,000$). Каждая из следующих m строк содержит три числа $count_i$, $from_i$ и to_i, которые означают одновременное перемещение всех шариков из урны $from_i$ в урну to_i, всех шариков из урны $from_i + 1$ в урну $to_i + 1$, ..., всех шариков из урны $from_i + count_i - 1$ в урну $to_i + count_i - 1$ ($1 \leqslant count_i, from_i, to_i \leqslant n$, $\max(from_i, to_i) + count_i \leqslant n + 1$).

Формат выходных данных

Выведите n чисел — итоговые позиции каждого шарика.

Пример

стандартный ввод	стандартный вывод
2 3 1 1 2 1 2 1 1 2 1	1 1

Задача M. Дели, прибавляй!

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

Дан массив целых чисел a длины n. Поступает q запросов четырех типов:

- $1\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $a_i + x$.
- $2\ l\ r\ x$. Для каждого i на отрезке от l до r включительно нужно заменить a_i на $\left\lfloor \frac{a_i}{x} \right\rfloor$ ($\lfloor \cdot \rfloor$ — это округление вниз).
- $3\ l\ r$. Необходимо вывести минимум элементов массива a на отрезке от l до r включительно.
- $4\ l\ r$. Необходимо вывести сумму элементов массива a на отрезке от l до r включительно.

Формат входных данных

В первой строке входных данных даны два целых числа n и q ($1 \leqslant n, q \leqslant 200\,000$) — количество элементов массива a и количество запросов.

Во второй строке даны n целых чисел a_i ($-10^9 \leqslant a_i \leqslant 10^9$) — элементы массива a.

В последующих q строках даны запросы по одному в строке.

Запрос первого типа задается так: $1\ l\ r\ x$

Где $0 \leqslant l \leqslant r < n$, $-10^7 \leqslant x \leqslant 10^7$ — целые числа. Это означает, что ко всем элементам массива a на отрезке от l до r нужно прибавить x.

Запрос второго типа задается так: $2\ l\ r\ x$

Где $0 \leqslant l \leqslant r < n$, $1 \leqslant x \leqslant 10^9$ — целые числа. Это означает, что все элементы массива a на отрезке от l до r нужно поделить на x и округлить вниз.

Запрос третьего типа задается так: $3\ l\ r$

Где $0 \leqslant l \leqslant r < n$ — целые числа. Это означает, что нужно вывести минимум элементов массива a на отрезке от l до r.

Запрос четвертого типа задается так: $4\ l\ r$

Где $0 \leqslant l \leqslant r < n$ — целые числа. Это означает, что нужно вывести сумму элементов массива a на отрезке от l до r.

Формат выходных данных

Для каждого запроса 3 и 4 типов выведите в отдельной строке ответ.

Пример

стандартный ввод	стандартный вывод
7 7 10 -6 4 7 12 1 0 2 1 4 4 1 0 5 -8 3 0 6 3 1 5 4 0 6 4 1 5 4 2 6	-10 -10 -34 -36 -26

Замечание

Обратите внимание на то, что элементы массива нумеруются с нуля.

Задача N. Исторический максимум

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Есть массив a из n элементов. Вам нужно написать структуру данных, которая обрабатывает два вида запросов:

- прибавить ко всем элементам на отрезке от l до r число x,
- узнать на отрезке от l до r исторический максимум, то есть максимальное значение, которое было в какой-то момент у какого-то элемента отрезка.

Формат входных данных

В первой строке дано число n ($1 \leqslant n \leqslant 300\,000$) — количество элементов массива a.

Во второй строке даны n чисел — элементы массива a ($-10^9 \leqslant a_i \leqslant 10^9$).

В третьей строке дано число q ($1 \leqslant q \leqslant 300\,000$) — количество запросов.

В последующих q строках даны запросы.

Запрос прибавления на отрезке задается следующим образом: 1 ql qr x. Это означает, что на отрезке от ql до qr включительно ($1 \leqslant ql \leqslant qr \leqslant n$) нужно прибавить ко всем числам x ($-10^9 \leqslant x \leqslant 10^9$).

Запрос поиска исторического максимума задается следующим образом: 2 ql qr. Это означает, что нужно найти максимум исторических максимумов на отрезке от ql до qr включительно ($1 \leqslant ql \leqslant qr \leqslant n$).

Формат выходных данных

Для каждого запроса второго типа выведите ответ в отдельной строке.

Пример

стандартный ввод	стандартный вывод
5 1 2 3 4 5 5 1 1 4 2 2 2 5 1 3 5 -5 2 4 5 2 1 5	6 6 6

Задача O. River Land

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	5 секунд
Ограничение по памяти:	512 мегабайт

Есть страна с большой речной системой. Города в этой стране пронумерованы от 1 до N; город № 1 является столицей. Речная система образует корневое дерево, так что столица является корнем. Каждая река соединяет два города и течет к корню. Для каждого некоренного города i есть река, которая течет из этого города в город p_i. В каждом городе также есть порт.

В стране есть только два способа путешествовать между городами: на лодке и на пароме. Лодка может двигаться только вниз по течению (то есть к столице), в то время как паром может двигаться в обоих направлениях. Есть прямые паромные маршруты из каждого порта в любой другой порт.

Изначально все порты открыты. Иногда некоторые порты могут закрываться или открываться снова (возможно, несколько раз). Порт в столице всегда открыт. Не разрешается путешествовать на пароме из закрытого порта, но допускается путешествие на пароме из открытого порта в закрытый порт. Путешествие на лодке всегда возможно, даже из закрытого порта.

Иногда вы хотите отправиться в путешествие. Вы уже выбрали город, в котором начнете путешествие. Однако у вас достаточно денег только для одной поездки на пароме. Поэтому вы хотите совершить путешествие следующим образом:

- Выбрать город b и отправиться в этот город на лодке. Должна быть возможность путешествовать от v до b на лодке (в частности, $v = b$ разрешено).
- Решить либо продолжить движение из города b на пароме, либо завершить поездку в этом городе.
- Если вы решаете продолжить путь на пароме, вы выбираете город f и отправляетесь на пароме от b до f. Также должна быть возможность путешествовать от b до f на пароме: не разрешается путешествовать вдоль любой реки дважды (независимо от направления), т.е., когда вы решаете продолжить путь на пароме, кратчайшие пути между (v, b) и (b, f) не должны иметь общих рек.
- Вы любите путешествовать на пароме, поэтому чем больше рек вы посещаете во время путешествия на пароме, тем счастливее вы становитесь. Также в каждом городе есть фиксированная красота; давайте обозначим красоту города i как a_i. Затем, для поездки со значением T (для каждой поездки задано своё значение) определим, что вы получаете счастье $a_b + D \cdot T$, где D — длина маршрута парома (количество рек, посещаемых во время путешествия на пароме; возможно, ноль, если продолжить путешествие на пароме из города b невозможно).

Вы должны обработать Q запросов. Есть три типа запросов:

1. - v: порт в городе v закрывается. Гарантируется, что порт в городе v был открыт до этого запроса.
2. + v: порт в городе v вновь открывается. Гарантируется, что порт в городе v был закрыт до этого запроса.
3. ? v T: вы хотите совершить поездку со значением T из города v.

Для каждого запроса третьего типа найдите максимальное счастье, которое вы можете получить от такой поездки.

Формат входных данных

Первая строка содержит два целых числа N и Q ($2 \leqslant N \leqslant 3 \cdot 10^5$, $1 \leqslant Q \leqslant 3 \cdot 10^5$).
Вторая строка содержит $N - 1$ целое число $p_2, p_3, \ldots, p_N$ ($1 \leqslant p_i < i$).

Третья строка содержит N целых чисел $a_1, a_2, \ldots, a_N$ ($1 \leqslant a_i \leqslant 10^9$).

Следующие Q строк описывают запросы в формате, описанном в условии. $1 \leqslant v \leqslant N$, $1 \leqslant T \leqslant 10^9$.

Формат выходных данных

Для каждого запроса третьего типа выведите ответ на него в отдельной строке.

Система оценки

Подзадача	**Доп. ограничения**	**Баллы**
1	$N, Q \leqslant 1000$	30
2	$N, Q \leqslant 10^5$	35
3	—	35

Пример

стандартный ввод	стандартный вывод
10 9 1 2 3 2 2 6 3 8 8 30 20 6 13 8 40 7 9 13 1 ? 4 11 - 4 ? 4 11 - 7 ? 10 6 + 7 - 6 - 2 ? 7 4	57 42 33 30

Замечание

В первом запросе вы путешествуеет на пароме из города 4 в город 7.

Во втором запросе вы не можете путешествовать на пароме из города 4, поэтому вы должны использовать второй по выгодности маршрут, который идет от города 2 к городу 7.

В третьем запросе вы едете в город 8 на лодке, а затем в город 7 на пароме.

В четвертом запросе вы едете в столицу (город № 1) на лодке и не путешествуете на пароме.

Задача P. Персистентная очередь

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Реализуйте персистентную очередь.

Формат входных данных

Первая строка содержит количество действий n ($1 \leqslant n \leqslant 200\,000$). В строке номер $i + 1$ содержится описание действия i:

- `1 t m` — добавить в конец очереди номер t ($0 \leqslant t < i$) число m;
- `-1 t` — удалить из очереди номер t ($0 \leqslant t < i$) первый элемент.

В результате действия i, описанного в строке $i + 1$ создается очередь номер i. Изначально имеется пустая очередь с номером ноль.

Все числа во входном файле целые, и помещаются в знаковый 32-битный тип.

Формат выходных данных

Для каждой операции удаления выведите удаленный элемент на отдельной строке.

Пример

стандартный ввод	стандартный вывод
10 1 0 1 1 1 2 1 2 3 1 2 4 -1 3 -1 5 -1 6 -1 4 -1 8 -1 9	1 2 3 1 2 4

Задача Q. Intercity Express

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1.5 секунд
Ограничение по памяти:	256 мегабайт

Андрей разрабатывает систему для продажи железнодорожных билетов. Он собирается протестировать ее на Междугородней Экспресс линии, которая соединяет два больших города и имеет $n - 2$ промежуточных станций, то есть в итоге есть n станций, пронумерованных от 1 до n.

В Междугороднем Экспресс поезде есть s мест, пронумерованных с 1 до s. В тестирующем режиме система имеет доступ к базе данных, содержащей проданные билеты в направлении от станции 1 до станции n и должна отвечать на вопросы, можно ли продать билет от станции a до станции b, и если да, нужно найти минимальный номер места, которое свободно на протяжении всего пути между a и b.

Изначально система имеет только доступ на чтение, то есть даже если есть свободное место, она должна сообщить об этом, но не должна изменять данные.

Помогите Андрею протестировать его систему написанием программы, которые будет находить ответы на вопросы.

Формат входных данных

Первая строка содержит число n — количество станций, s — количество мест и m — количество уже проданных билетов ($2 \leqslant n \leqslant 10^9$, $1 \leqslant s \leqslant 100\,000$, $0 \leqslant m \leqslant 100\,000$).

В следующих m строках описаны билеты, описание каждого билета состоит из трех чисел: c_i, a_i и b_i — номер места, которое занимает владелец билета, номер станции, с которой продан билет и номер станции, до которой продан билет ($1 \leqslant c_i \leqslant s$, $1 \leqslant a_i < b_i \leqslant n$).

Следующие строки содержат число q — количество запросов ($1 \leqslant q \leqslant 100\,000$). Специальное значение p должно поддерживаться в течение считывания запросов. Изначально $p = 0$.

Следующие $2q$ строк описывают запросы. Каждый запрос описывается двумя числами: x_i и y_i ($x_i \leqslant y_i$).

Чтобы получить города a и b между которыми нужно проверить наличие места, используется следующая формула:

$a = x_i + p$, $b = y_i + p$. Ответ на запрос — число 0, если нет места на каждом отрезке между a и b, или минимальный номер свободного места.

После ответа на запрос, надо приравнять число p полученному ответу на запрос.

Формат выходных данных

Для каждого запроса выведите ответ на него.

Пример

стандартный ввод	стандартный вывод
5 3 5 1 2 5 2 1 2 2 4 5 3 2 3 3 3 4 10 1 2 1 2 1 2 2 3 -2 0 2 4 1 3 1 4 2 5 1 5	1 2 2 3 0 2 0 0 0 0

Замечание

Обратите внимание, что запросы выглядят так: $(1, 2)$, $(2, 3)$, $(3, 4)$, $(4, 5)$, $(1, 3)$, $(2, 4)$, $(3, 5)$, $(1, 4)$, $(2, 5)$, $(1, 5)$.

Задача R. Сбежать через лист

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.3 секунд
Ограничение по памяти:	256 мегабайт

Дано дерево на n вершинах (пронумерованных от 1 до n) с корнем в вершине 1. В вершине i записаны два числа: a_i и b_i.

Вы можете прыгнуть из вершины в любую вершину в её поддереве. Стоимость такого прыжка из вершины x в вершину y равна произведению a_x и b_y. Суммарная стоимость пути между вершинами, состоящего из нескольких прыжков равна сумме стоимостей прыжков в нём. Для каждой вершины посчитайте минимальную стоимость пути от неё до какого-либо листа. Обратите внимание, что корень дерева не является листом, даже если имеет степень 1.

Учтите, что нельзя совершать прыжок из вершины в ту же вершину.

Формат входных данных

В первой строке содержится целое число n ($2 \leqslant n \leqslant 10^5$) — количество вершин в дереве.

Во второй строке через пробел заданы n целых чисел $a_1, a_2, \cdots, a_n$ ($-10^5 \leqslant a_i \leqslant 10^5$).

Во третьей строке через пробел заданы n целых чисел $b_1, b_2, \cdots, b_n$ ($-10^5 \leqslant b_i \leqslant 10^5$).

В следующих $n-1$ строках содержатся пары целых чисел u_i и v_i ($1 \leqslant u_i, v_i \leqslant n$), разделённых пробелом, обозначающие ребро между вершинами u_i и v_i в дереве.

Формат выходных данных

Выведите n целых чисел через пробел, i-е из которых обозначает минимальную стоимость, чтобы добраться от вершины с номером i до какого-либо листа.

Примеры

стандартный ввод	стандартный вывод
3 2 10 -1 7 -7 5 2 3 2 1	10 50 0
4 5 -10 5 7 -8 -80 -3 -10 2 1 2 4 1 3	-300 100 0 0

Замечание

В первом тестовом примере вершина 3 сама является листом, поэтому ответ равен 0. Для вершины 2 прыжок в вершину 3 стоит $a_2 \times b_3 = 50$. Для вершины 1 прыжок в вершину 3 стоит $a_1 \times b_3 = 10$.

Во втором тестовом примере вершины 3 и 4 являются листьями, поэтому ответ для них равен 0. Для вершины 2 прыжок в вершину 4 стоит $a_2 \times b_4 = 100$. Для вершины 1 необходимо сначала прыгнуть в вершину 2 прыжком стоимостью $a_1 \times b_2 = -400$, а затем прыгнуть из 2 в 4 за $a_2 \times b_4 = 100$.

Задача S. Поиск идеи light

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3.5 секунд
Ограничение по памяти:	1024 мегабайта

3019 год. При раскопках древнего города Иннополис археологи обнаружили артефакт — жёсткий диск, на котором находится файл, предположительно содержащий тексты всех задач всероссийских олимпиад по информатике.

После исследования файла было выяснено, что информация в нём закодирована таким образом, что записанный в файле текст представляет собой строку t из букв английского алфавита. Текст с задачами оказался довольно длинным и содержал много повторений, поэтому файл хранился на диске в сжатом виде. Для его распаковки используется следующий алгоритм.

В процессе распаковки формируется строка t из строчных букв английского алфавита. Исходно строка пуста. Сжатый файл состоит из n блоков, которые должны быть обработаны в порядке следования. Каждый блок имеет один из двух типов.

- Блок «`1` w», где w — строка. При обработке такого блока в конец строки t дописывается строка w.
- Блок «`2` pos len», где pos и len — положительные целые числа. Пусть символы строки t пронумерованы с 1. При обработке такого блока в конец строки t по очереди приписываются len подряд идущих символов строки t, начиная с позиции pos. При этом, если значение len достаточно велико, некоторые только что выписанные символы могут быть снова использованы при обработке того же блока.

Ученые решили выяснить, сколько раз некоторая идея встречалась в олимпиадах. Для этого они сформировали строку p из строчных букв английского алфавита и хотят найти количество вхождений строки p как подстроки в полученную после распаковки файла строку t.

Строка p длины m входит в строку t как подстрока с позиции i, если m следующих подряд символов строки t, начиная с i-го, представляют собой строку p. Например, строка «`aba`» входит как подстрока в строку «`abababaa`» три раза: с позиций 1, 3 и 6.

Требуется написать программу, которая определяет количество вхождений заданной строки p в полученную после распаковки файла строку t.

Формат входных данных

В первой строке находятся натуральные числа m и n — длина строки p и количество блоков в сжатом тексте ($1 \leqslant m \leqslant 2 \cdot 10^5$, $1 \leqslant n \leqslant 10^4$). Ограничения в реальности чуть ниже, и тут проще загнать неэффективное решение.

Во второй строке входных данных задана непустая строка p, состоящая из строчных букв английского алфавита.

В следующих n строках находятся описания блоков в описанном в условии формате. Для блоков первого типа приписываемая строка w непуста, сумма длин всех строк w в блоках первого типа не превышает $2 \cdot 10^5$. Для блоков второго типа в строке t к моменту обработки этого блока находится не менее pos символов. Длина распакованного текста не превышает 10^{10} символов.

Формат выходных данных

Выведите одно целое число — количество вхождений строки p в текст.

Пример

стандартный ввод	стандартный вывод
3 4 aba 1 ab 2 1 3 2 3 3 2 1 8	6

Задача T. Поиск идеи

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3.5 секунд
Ограничение по памяти:	1024 мегабайта

3019 год. При раскопках древнего города Иннополис археологи обнаружили артефакт — жёсткий диск, на котором находится файл, предположительно содержащий тексты всех задач всероссийских олимпиад по информатике.

После исследования файла было выяснено, что информация в нём закодирована таким образом, что записанный в файле текст представляет собой строку t из букв английского алфавита. Текст с задачами оказался довольно длинным и содержал много повторений, поэтому файл хранился на диске в сжатом виде. Для его распаковки используется следующий алгоритм.

В процессе распаковки формируется строка t из строчных букв английского алфавита. Исходно строка пуста. Сжатый файл состоит из n блоков, которые должны быть обработаны в порядке следования. Каждый блок имеет один из двух типов.

- Блок «`1` w», где w — строка. При обработке такого блока в конец строки t дописывается строка w.
- Блок «`2` pos len», где pos и len — положительные целые числа. Пусть символы строки t пронумерованы с 1. При обработке такого блока в конец строки t по очереди приписываются len подряд идущих символов строки t, начиная с позиции pos. При этом, если значение len достаточно велико, некоторые только что выписанные символы могут быть снова использованы при обработке того же блока.

Ученые решили выяснить, сколько раз некоторая идея встречалась в олимпиадах. Для этого они сформировали строку p из строчных букв английского алфавита и хотят найти количество вхождений строки p как подстроки в полученную после распаковки файла строку t.

Строка p длины m входит в строку t как подстрока с позиции i, если m следующих подряд символов строки t, начиная с i-го, представляют собой строку p. Например, строка «`aba`» входит как подстрока в строку «`ababaaba`» три раза: с позиций 1, 3 и 6.

Требуется написать программу, которая определяет количество вхождений заданной строки p в полученную после распаковки файла строку t.

Формат входных данных

В первой строке находятся натуральные числа m и n — длина строки p и количество блоков в сжатом тексте ($1 \leqslant m \leqslant 2 \cdot 10^5$, $1 \leqslant n \leqslant 10^4$).

Во второй строке входных данных задана непустая строка p, состоящая из строчных букв английского алфавита.

В следующих n строках находятся описания блоков в описанном в условии формате. Для блоков первого типа приписываемая строка w непуста, сумма длин всех строк w в блоках первого типа не превышает $2 \cdot 10^5$. Для блоков второго типа в строке t к моменту обработки этого блока находится не менее pos символов. Длина распакованного текста не превышает 10^{15} символов.

Формат выходных данных

Выведите одно целое число — количество вхождений строки p в текст.

Пример

стандартный ввод	стандартный вывод
3 4 aba 1 ab 2 1 3 2 3 3 2 1 8	6

Задача U. Копирование памяти

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Одним осенним дождливым днём программист Алексей пытался в пятый раз сдать эезамен по программированию на ЭВМ в Московском Государственном Университете. Его преподаватель Эндрю-Шесть-Метров настолько устал от этого повторяющегося мучения, что в этот раз он всего-лишь попросил Алексея оптимизировать очень простую программу, пока Андрей сходит купить себе *компот*.

Эта программа производит некоторые операции с массивом `X`, состоящим из n целых чисел. Эндрю сказал, что эта программа только копирует участки памяти в массиве и иногда заполняет некоторые отрезки массива последовательными целыми числами.

Вам дан псевдокод в C-подобном синтаксисе трёх процедур, которые Эндрю применяет к массиву.

Пусть `X` — 1-индексированный глобальный массив целых чисел размера n и `last` — глобальная целочисленная переменная, равная изначально 0.

```
void forwardCopy(int a, int b, int l)
{
  for (int i = 0; i < l; i++)
  {
    X[b] = X[a];
    a++;
    b++;
  }
}

void backwardCopy(int a, int b, int l)
{
  for (int i = 0; i < l; i++)
  {
    X[a] = X[b];
    a--;
    b--;
  }
}

void fill(int a, int b)
{
  for (int i = a; i <= b; i++)
  {
    last++;
    X[i] = last;
  }
}
```

Программа Эндрю начинается с вызова "`fill(1, n)`", за которым последуют применения трёх процедур выше в определённом порядке. Вам дан порядок вызова процедур и аргументы, с которыми они производятся.

Ваша задача — определить содержимое массива `X` после выполнения всех операций. Эндрю гарантирует, что переменная `last` не превзойдёт $5n$ ни в какой момент выполнения программы и что каждый вызов процедур корректен (т. е. запись и чтение не будут производиться за пределами массива X).

Формат входных данных

В первой строке даны два целых числа n и m ($1 \leqslant n, m \leqslant 200\,000$) — размер массива `x` и число операций.

Во следующих m строках описываются операции:

- `f a b l` — вызвать `forwardCopy(a, b, l)`
- `b a b l` — вызвать `backwardCopy(a, b, l)`
- `x a b` — вызвать `fill(a, b)`

Формат выходных данных

Примеры

stdin	stdout
10 3 b 2 10 2 x 7 9 f 2 5 4	9 10 3 4 10 3 4 10 13 10

Задача W. Период

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

Периодом строки s называется такая строка t минимальной длины, что s представима в виде конкатенации нескольких копий t; например, периодом строки `abaaba` является строка `aba`, периодом строки `aaaaa` является строка `a`, а периодом строки `abaabaa` является сама строка `abaabaa`.

Вам дана строка s. Требуется выполнять запросы трёх типов:

- вставить символ c в строку s;
- удалить символ из строки s;
- вывести длину периода строки s.

Формат входных данных

Первая строка ввода содержит целое число $n (1 \leqslant n \leqslant 500\,000)$ — длину строки s.
Вторая строка содержит строку s из n маленьких латинских букв.
Третья строка содержит целое число q $(1 \leqslant q \leqslant 500\,000)$ — количество запросов.
Следующие q строк содержат описания запросов.

- Строка вида «`+` i c» $(0 \leqslant i \leqslant |s|)$ соответствует запросу «вставить символ c после i-й позиции». Так, если $i = |s|$, символ добавляется в конец строки, иначе символ вставляется перед i-м символом (если нумеровать символы строки с нуля). Все символы являются маленькими латинскими буквами.
- Строка вида «`-` i» $(0 \leqslant i \leqslant |s| - 1)$ соответствует запросу «удалить i-й символ строки» (в нумерации с нуля).
- Строка вида «`?`» соответствует запросу «найти период текущей строки».

Гарантируется, что в каждом тесте не более 100 000 запросов третьего типа. Гарантируется, что в любой момент времени строка s непуста.

Формат выходных данных

Для каждого запроса «`?`» выведите одно число на отдельной строке — период текущей строки s.

Пример

стандартный ввод	стандартный вывод
6 abaaba 7 ? - 2 ? - 4 ? + 0 a ?	3 5 2 5

Задача X. Li-Chao Tree

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

Пусть S — множество прямых на плоскости. Изначально S пусто. Необходимо отвечать на запросы двух типов:

1. Добавить в множество S прямую $y = a_i \cdot x + b_i$, где a_i и b_i — параметры запроса.
2. Найти $\min\limits_{f \in S} f(x_i)$, где x_i — параметр запроса. Иными словами, необходимо найти минимальное значение среди всех прямых из множества S в точке x_i.

Формат входных данных

Первая строка содержит одно целое число q ($2 \leqslant q \leqslant 200\,000$) — количество запросов.

Каждая из следующих q строк содержит описание очередного запроса. Описание каждого запроса начинается с целого числа t_i ($t_i \in \{1, 2\}$), обозначающего тип запроса.

Если $t_i = 1$, далее следуют два целых числа a_i и b_i ($-200\,000 \leqslant a_i, b_i \leqslant 200\,000$) — параметры добавляемой линейной функции. Это означает, что вы должны добавить в множество прямую, задаваемую уравнением $y = (a_i + p \bmod 200\,000) \cdot x + (b_i + p \bmod 200\,000)$, где p — ответ на последний поступивший запрос второго типа. Изначально $p = 0$. **Обратите внимание, что по модулю берется только число p, а не суммы $a_i + p$ и $b_i + p$.**

Если $t_i = 2$, далее следует одно целое число x_i ($-200\,000 \leqslant x_i \leqslant 200\,000$) — точка, в которой необходимо найти минимальное значение среди всех добавленных прямых.

Гарантируется, что первый запрос имеет тип 1. Также гарантируется, что есть хотя бы один запрос типа 2.

Формат выходных данных

Для каждого запроса типа 2 выведите одно целое число — ответ на запрос.

Примеры

стандартный ввод	стандартный вывод
4 1 1 3 1 2 0 2 1 2 4	2 7
4 1 1 1 2 1 1 -2 -2 2 1	2 0