

Задача А. Треугольники

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2.5 секунд
Ограничение по памяти: 256 мегабайт

Дан граф. Найдите количество циклов длины 3.

Формат входных данных

Первая строка содержит два целых числа n и m ($1 \leq n, m \leq 3 \cdot 10^5$) — количество вершин и рёбер, соответственно.

Каждая из следующих m строк содержит по два целых числа от 1 до n — вершины, которые соединяет соответствующее ребро.

Гарантируется, что в графе нет петель и кратных рёбер.

Формат выходных данных

Выведите одно число — ответ.

Пример

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

Задача В. Лунки

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

Маленький Петя очень любит играть. Больше всего он любит играть в игру «Лунки». Это игра для одного игрока со следующими правилами:

Есть N лунок, расположенных в ряд, пронумерованных слева направо числами от 1 до N . У каждой лунки изначально установлена своя сила выброса (у лунки с номером i она равна a_i). Если вбросить шарик в лунку i , то он тут же вылетит из нее и попадет в лунку $i + a_i$, после чего он опять вылетит и т.д.. Если же лунки с таким номером нету, то он просто вылетит за край ряда. На каждом из M ходов игрок выбирает одно из двух действий:

- Установить силу выброса лунки a равной b .
- Вбросить шарик в лунку a и посчитать количество прыжков шарика, прежде чем он вылетит за край ряда, а так же записать номер лунки, после выпрыгивания из которой шарик вылетел за край.

У Пети есть некоторые проблемы с математикой, поэтому, как Вы уже догадались, именно Вам предстоит произвести все подсчеты.

Формат входных данных

Первая строка содержит два числа N и M ($1 \leq N \leq 10^5$, $1 \leq M \leq 10^5$) — количество лунок в ряду и количество ходов. Следующая строка содержит N целых положительных чисел, не превышающих N — начальные силы выброса лунок. Следующие M строк задают ходы, сделанные Петей. Каждая строка может быть двух типов:

- 0 a b
- 1 a

Тут, первый тип означает что требуется установить силу выброса лунки a равной b , а второй означает что требуется вбросить мячик в лунку с номером a . Числа a и b — целые положительные и не превышают N .

Формат выходных данных

Для каждого хода второго типа (задающего вбрасывание шарика) в порядке следования во входном файле выведите два числа через пробел в отдельной строке — номер последней лунки перед вылетом шарика за край и количество прыжков.

Пример

стандартный ввод	стандартный вывод
8 5	8 7
1 1 1 1 1 2 8 2	8 5
1 1	7 3
0 1 3	
1 1	
0 3 4	
1 2	

Задача С. Частота строки

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1.5 секунд
Ограничение по памяти: 512 мегабайт

Дана строка s . Требуется ответить на n запросов. i -й запрос состоит из целого числа k_i и строки m_i , ответом является минимальная длина строки t такой, что t является подстрокой s и строка m_i входит в t как подстрока не менее k_i раз.

Подстрокой строки называется любая последовательность подряд идущих символов в этой строке.

Гарантируется, что для любых двух запросов строки m_i из этих запросов различны.

Формат входных данных

В первой строке содержится строка s ($1 \leq |s| \leq 10^5$).

Во второй строке содержится целое число n ($1 \leq n \leq 10^5$).

В каждой из следующих n строк содержатся целое число k_i ($1 \leq k_i \leq |s|$) и непустая строка m_i — параметры запроса с номером i .

Все строки во вводе состоят только из строчных букв латинского алфавита. Суммарная длина всех строк во вводе не превосходит 10^5 . Все m_i различны.

Формат выходных данных

Для каждого запроса выведите ответ на него в отдельной строке.

Если строка m_i встречается в s менее k_i раз, выведите -1 .

Примеры

стандартный ввод	стандартный вывод
aaaaa	3
5	4
3 a	4
3 aa	-1
2 aaa	5
3 aaaa	
1 aaaaa	
abbb	-1
7	2
4 b	-1
1 ab	3
3 bb	-1
1 abb	1
2 bbb	-1
1 a	
2 abbb	

Задача D. Фаброзавры-дизайнеры

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	15 секунд
Ограничение по памяти:	64 мегабайта

Фаброзавры известны своим тонким художественным вкусом и увлечением ландшафтным дизайном. Они живут около очень живописной реки и то и дело перестраивают тропинку, идущую вдоль реки: либо насыпают дополнительной земли, либо срывают то, что есть. Для того, чтобы упростить эти работы, они поделили всю тропинку на горизонтальные участки, пронумерованные от 1 до N , и их переделки устроены всегда одинаково: они выбирают часть дороги от L -ого до R -ого участка (включительно) и изменяют (увеличивают или уменьшают) высоту на всех этих участках на одну и ту же величину (если до начала переделки высоты были разными, то и после переделки они останутся разными).

Поскольку, как уже говорилось, у фаброзавров тонкий художественный вкус, каждый из них считает, что их река лучше всего выглядит с определенной высоты. Поэтому им хочется знать, есть ли поблизости от их дома место на тропинке, где высота на их взгляд оптимальна. Помогите им в этом разобраться.

Формат входных данных

Первая строка входного файла содержит два числа N и M — длину дороги и количество запросов соответственно ($1 \leq N, M \leq 5 \cdot 10^5$). На второй строке содержатся N чисел, разделенных пробелами — начальные высоты соответствующих частей дороги; высоты не превосходят 10^4 по модулю. В следующих M строках содержатся запросы по одному на строке.

Запрос $+ L R X$ означает, что высоту частей дороги от L -ой до R -ой (включительно) нужно изменить на X . При этом $1 \leq L \leq R \leq N$, а $|X| \leq 10^4$.

Запрос $? L R X$ означает, что нужно проверить, есть ли между L -ым и R -ым участками (включая эти участки) участок, где дорога проходит точно на высоте X . Гарантируется, что $1 \leq L \leq R \leq N$, а $|X| \leq 10^9$.

Формат выходных данных

На каждый запрос второго типа нужно вывести в выходной файл на отдельной строке одно слово «YES» (без кавычек), если нужный участок существует, и «NO» в противном случае.

Пример

стандартный ввод	стандартный вывод
10 5	NO
0 1 1 3 3 3 2 0 0 1	YES
? 3 5 2	YES
+ 1 4 1	
? 3 5 2	
+ 7 10 2	
? 9 10 3	

Задача Е. Мощные юнги

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2.5 секунд
Ограничение по памяти: 256 мегабайт

Имеется список из n юнг, для каждого из которых известен его рост a_1, a_2, \dots, a_n . Рассмотрим некоторый его подсписок a_l, a_{l+1}, \dots, a_r , где $1 \leq l \leq r \leq n$, и для каждого натурального числа s обозначим через K_s число юнг с ростом s в этом подсписке. Назовем *мощностью* подсписка сумму произведений $K_s \cdot K_s \cdot s$ по всем различным натуральным s . Так как количество различных чисел в массиве конечно, сумма содержит лишь конечное число ненулевых слагаемых.

Необходимо вычислить мощности каждого из t заданных подсписков.

Формат входных данных

Первая строка содержит два целых числа n и t ($1 \leq n, t \leq 200000$) — длина списка и количество запросов соответственно.

Вторая строка содержит n натуральных чисел a_i ($1 \leq a_i \leq 10^6$) — рост юнг.

Следующие t строк содержат по два натуральных числа l и r ($1 \leq l \leq r \leq n$) — индексы левого и правого концов соответствующего подсписка.

Формат выходных данных

Выведите t строк, где i -ая строка содержит единственное натуральное число — мощность подсписка i -го запроса.

Примеры

стандартный ввод	стандартный вывод
3 2 1 2 1 1 2 1 3	3 6
8 3 1 1 2 2 1 3 1 1 2 7 1 6 2 7	20 20 20

Задача F. Машинное обучение

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

На курсе машинного обучения вам выдали первое домашнее задание — вам предстоит проанализировать некоторый массив из n чисел.

В частности, вы интересуетесь так называемой *равномерностью* массива. Предположим, что в массиве число b_1 встречается k_1 раз, b_2 — k_2 раз, и т.д. Тогда *равномерностью* массива называется такое минимальное целое число $c \geq 1$, что $c \neq k_i$ для любого i .

В рамках вашего исследования вы хотите последовательно проделать q операций.

- Операция $t_i = 1, l_i, r_i$ задаёт запрос исследования. Необходимо вывести равномерность массива, состоящего из элементов на позициях от l_i до r_i включительно.
- Операция $t_i = 2, p_i, x_i$ задаёт запрос уточнения данных. Начиная с этого момента времени p_i -му элементу массива присваивается значения x_i .

Формат входных данных

Первая строка содержит n и q ($1 \leq n, q \leq 100\,000$) — размер массива и число запросов соответственно.

Во второй строке записаны ровно n чисел — a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$).

Каждая из оставшихся q строк задаёт очередной запрос.

Запрос первого типа задаётся тремя числами $t_i = 1, l_i, r_i$, где $1 \leq l_i \leq r_i \leq n$ — границы соответствующего отрезка.

Запрос второго типа задаётся тремя числами $t_i = 2, p_i, x_i$, где $1 \leq p_i \leq n$ — позиция в которой нужно заменить число, а $1 \leq x_i \leq 10^9$ — его новое значение

Формат выходных данных

Для каждого запроса первого типа выведите одно число — равномерность соответствующего отрезка массива.

Пример

стандартный ввод	стандартный вывод
10 4	2
1 2 3 1 1 2 2 2 9 9	3
1 1 1	2
1 2 8	
2 7 1	
1 2 8	

Замечание

Первый запрос состоит из ровно одного элемента — 1. Минимальное подходящее $c = 2$.

Отрезок второго запроса состоит из четырёх 2, одной 3 и двух 1. Минимальное подходящее $c = 3$.

Отрезок четвёртого запроса состоит из трёх 1, трёх 2 и одной 3. Минимальное подходящее $c = 2$.

Задача G. МЕХ на пути

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

Дано дерево, на каждом ребре которого написано неотрицательное целое число. Вам необходимо ответить на несколько запросов вида «для данных вершин u , v назовите наименьшее неотрицательное целое число, которое **не** встречается среди чисел, написанных на ребрах на пути от u до v ».

Формат входных данных

Первая строка входных данных содержит два числа n и q ($2 \leq n \leq 10^5$, $1 \leq q \leq 10^5$), количество вершин и количество запросов.

Следующие $n - 1$ строк содержат по три числа u_i, v_i, x_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$, $0 \leq x_i \leq 10^9$), которые описывают ребро дерева (u_i, v_i) , на котором написано число x_i .

Следующие q строк содержат по паре чисел a_j, b_j ($1 \leq a_j, b_j \leq n$), которая обозначает запрос на пути от a_j до b_j .

Формат выходных данных

Для каждого запроса выведите единственное число — минимальное число, которое не встречается на пути.

Пример

стандартный ввод	стандартный вывод
7 6	0
2 1 1	1
3 1 2	2
1 4 0	2
4 5 1	3
5 6 3	3
5 7 4	
1 3	
4 1	
2 4	
2 5	
3 5	
3 7	

Задача Н. Жесть

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 15 секунд
Ограничение по памяти: 256 мегабайт

Дан массив из N чисел. Нужно уметь обрабатывать 3 типа запросов:

- `get(L, R, x)` — сказать, сколько элементов отрезка массива $[L..R]$ не меньше x .
- `set(L, R, x)` — присвоить всем элементам массива на отрезке $[L..R]$ значение x .
- `reverse(L, R)` — перевернуть отрезок массива $[L..R]$.

Формат входных данных

Число N ($1 \leq N \leq 5 \cdot 10^5$) и массив из N чисел. Далее число запросов M ($1 \leq M \leq 5 \cdot 10^5$) и M запросов. Формат описания запросов предлагается понять из примера. Для всех отрезков верно $1 \leq L \leq R \leq N$. Исходные числа в массиве и числа x в запросах — целые от 0 до 10^9 .

Формат выходных данных

Для каждого запроса типа `get` нужно вывести ответ.

Пример

стандартный ввод	стандартный вывод
5	3
1 2 3 4 5	1
6	3
get 1 5 3	1
set 2 4 2	
get 1 5 3	
reverse 1 2	
get 2 5 2	
get 1 1 2	

Задача I. Brainfuck

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

Мы начинаем курс программирования на языке brainfuck <https://ru.wikipedia.org/wiki/Brainfuck>. Язык состоит всего из 8 команд, описание которых есть на странице википедии. Несмотря на ограниченное число команд, на этом языке можно написать любые по сложности программы. Отличием нашей версии от обычного языка будет только то, что количество ячеек памяти неограниченно. Обратите внимание, что при работе ваша программа не должна делать больше $5 \cdot 10^7$ действий, а так же длина исходной программы не должна превышать $5 \cdot 10^5$ символов.

В нашем первом уроке языка brainfuck мы научимся базовым вещам — вводу чисел, выводу чисел, сложению, вычитанию, умножению, делению и взятию по модулю.

От вас требуется написать программу, **генерирующую корректную программу на языке brainfuck**, и выводящую её в стандартный поток вывода. Формат ввода и вывода, с которым должна работать ваша сгенерированная программа, описан ниже. Обратите внимание, что ваша сгенерированная программа не должна содержать символов перевода строки.

Для удобства вот ссылка на тестировщик программ на языке brainfuck <http://algotcode.ru/docs/docs/bff.cpp>.

Формат входных данных

Сгенерированной вами программе на стандартный поток ввода через знак переноса строки будут передаваться 2 числа: a и b ($0 \leq a < 256$, $1 \leq b < 256$). Эти же числа, но в зашифрованном неизвестным вам образом, будут передаваться вашему генератору программы на brainfuck.

Формат выходных данных

Сгенерированная вами программа на стандартный поток вывода через пробел должна вывести 5 чисел: $a + b$, $a - b$, $a \cdot b$, a/b , $a\%b$. Операции сложения, вычитания и умножения следует выполнять по модулю 256. Деление и взятие по модулю надо выполнить как с обычными числами, при этом деление надо округлять вниз.

Система оценки

Всего 20 тестов, каждый стоит по 50 баллов, за каждое неверное число —10 баллов, дальше балл уменьшается в зависимости от числа итераций вашего решения. В первых 10 тестах все числа (в том числе в ответе) от 1 до 9

Пример

стандартный ввод	стандартный вывод
1	3 255 2 0 1
2	

Замечание

Конечно вы можете зауглеть, как реализовывать все возможные операции на brainfuck, но я рекомендую вам самим попробовать всё это придумать.

Чтобы не умереть во время написания генератора советую все повторяющиеся процедуры заводить в отдельные функции. Так например можно создать функцию *go* для перемещения, функцию *add* для прибавления текущего значения к какой-то ячейке памяти и так со всем. Вообще почти все части пишутся довольно просто, но вот деление может вызвать у вас вопросы. Подумайте, как писать *if else* конструкцию.