Задача A. Мосты

Имя входного файла:	`bridges.in`
Имя выходного файла:	`bridges.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан неориентированный граф. Требуется найти все мосты в нем.

Формат входного файла

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($n \le 20\,000$, $m \le 200\,000$).

Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается двумя натуральными числами b_i, e_i — номерами концов ребра ($1 \le b_i, e_i \le n$).

Формат выходного файла

Первая строка выходного файла должна содержать одно натуральное число b — количество мостов в заданном графе. На следующей строке выведите b целых чисел — номера ребер, которые являются мостами, в возрастающем порядке. Ребра нумеруются с единицы в том порядке, в котором они заданы во входном файле.

Пример

bridges.in	bridges.out
6 7 1 2 2 3 3 4 1 3 4 5 4 6 5 6	1 3

Задача B. Конденсация графа

Имя входного файла:	`condense2.in`
Имя выходного файла:	`condense2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Требуется найти количество ребер в конденсации ориентированного графа. Примечание: конденсация графа не содержит кратных ребер.

Формат входного файла

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($n \le 10\,000$, $m \le 100\,000$). Следующие m строк содержат описание ребер, по одному на строке. Ребро номер i описывается двумя натуральными числами b_i, e_i — началом и концом ребра соответственно ($1 \le b_i, e_i \le n$). В графе могут присутствовать кратные ребра и петли.

Формат выходного файла

Первая строка выходного файла должна содержать одно число — количество ребер в конденсации графа.

Пример

condense2.in	condense2.out
4 4 2 1 3 2 2 3 4 3	2

Задача C. Выпуклая оболочка

Имя входного файла:	`convex.in`
Имя выходного файла:	`convex.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

На плоскости даны N точек. Вам требуется построить выпуклую оболочку данного множества точек и вывести длину её периметра.

Формат входного файла

Первая строка содержит количество точек N, ($1 \le N \le 20\,000$). Каждая из последующих N строк содержит два целых числа — координаты x_i и y_i.

Формат выходного файла

Выведите в выходной файл длину периметра выпуклой оболочки.

Пример

convex.in	convex.out
5 0 0 1 0 0 1 -1 0 0 -1	5.65685

Задача D. Циклические суффиксы

Имя входного файла:	`cyclic.in`
Имя выходного файла:	`cyclic.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Рассмотрим строку $S = s_1 s_2 s_3 \dots s_{n-1} s_n$ над алфавитом Σ. *Циклическим расширением* порядка m строки S назовем строку $s_1 s_2 s_3 \dots s_{n-1} s_n s_1 s_2 \dots$ из m символов; это значит, что мы приписываем строку S саму к себе, пока не получим требуемую длину, и берем префикс длины m.

Циклической строкой S' назовем бесконечное циклическое расширение строки S.

Рассмотрим суффиксы циклической строки S'. Очевидно, существует не более $|S|$ различных суффиксов: $(n+1)$-ый суффикс совпадает с первым, $(n+2)$-ой совпадает со вторым, и так далее. Более того, различных суффиксов может быть даже меньше. Например, если $S =$ `abab`, первые четыре суффикса циклической строки S' — это:

$$\begin{aligned} S'_1 &= \texttt{abababab}\dots \\ S'_2 &= \texttt{bababababa}\dots \\ S'_3 &= \texttt{abababab}\dots \\ S'_4 &= \texttt{bababababa}\dots \end{aligned}$$

Здесь существует всего два различных суффикса, в то время как $|S| = 4$.

Отсортируем первые $|S|$ суффиксов S' лексикографически. Если два суффикса совпадают, первым поставим суффикс с меньшим индексом. Теперь нас интересует следующий вопрос: на каком месте в этом списке стоит сама строка S'?

Например, рассмотрим строку $S =$ `cabcab`:

(1) S'_2 = `abcabcabca...`
(2) S'_5 = `abcabcabca...`
(3) S'_3 = `bcabcabcab...`
(4) S'_6 = `bcabcabcab...`
(5) S'_1 = `cabcabcabc...`
(6) S'_4 = `cabcabcabc...`

Здесь циклическая строка $S' = S'_1$ находится на пятом месте.

Вам дана строка S. Ваша задача — найти позицию циклической строки S' в описанном порядке.

Формат входного файла

Во входном файле записана единственная строка S ($1 \le |S| \le 1\,000\,000$), состоящая из прописных латинских букв.

Формат выходного файла

В выходной файл выведите единственное число — номер строки S' в описанном порядке среди первых $|S|$ суффиксов.

Примеры

cyclic.in	cyclic.out
abracadabra	3
cabcab	5

Задача E. План эвакуации

Имя входного файла:	`evacuate.in`
Имя выходного файла:	`evacuate.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В городе есть муниципальные здания и бомбоубежища, которые были специально построены для эвакуации служащих в случае ядерной войны. Каждое бомбоубежище имеет ограниченную вместительность по количеству людей, которые могут в нем находиться. В идеале все работники из одного муниципального здания должны были бы бежать к ближайшему бомбоубежищу. Однако, в таком случае, некоторые бомбоубежища могли бы переполниться, в то время как остальные остались бы наполовину пустыми.

Чтобы резрешить эту проблему Городской Совет разработал специальный план эвакуации. Вместо того того, чтобы каждому служащему индивидуально приписать, в какое бомбоубежище он должен бежать, для каждого муниципального здания определили, сколько служащих из него в какое бомобоубежище должны бежать. Задача индивидального распределения была переложена на внутреннее управление муниципальных зданий.

План эвакуации учитывает количество служащих в каждом здании — каждый служащий должен быть учтен в плане и в каждое бомбоубежище может быть направлено количество служащих, не превосходящее вместимости бомбоубежища.

Городской Совет заявляет, что их план эвакуации оптиматен в том смысле, что суммарное время эвакуации всех служащих города минимально.

Мэр города, находящийся в постоянной конфронтации с Городским Советом, не слишком то верит этому заявлению. Поэтому он нанял Вас в качестве независимого эксперта для проверки плана эвакуации. Ваша задача состоит в том, чтобы либо убедиться в оптимальности плана Городского Совета, либо доказать обратное, представив в качестве доказательства другой план эвакуации с меньшим суммарным временем для эвакуации всех служащих.

Карта города может быть представлена в виде квадратной сетки. Расположение муниципальных зданий и бомбоубежищ задается парой целых чисел, а время эвакуации из муниципального здания с координатами (X_i, Y_i) в бомбоубежище с координатами (P_j, Q_j) составляет $D_{ij} = |X_i - P_j| + |Y_i - Q_j| + 1$ минут.

Формат входного файла

Входной файл содержит описание карты города и плана эвакуации, предложенного Городским Советом. Первая строка входного файла содержит два целых числа N ($1 \le N \le 100$) и M ($1 \le M \le 100$), разделенных пробелом. N — число муниципальных зданий в городе (все они занумерованы числами от 1 до N), M — число бомбоубежищ (все они занумерованы числами от 1 до M).

Последующие N строк содержат описания муниципальных зданий. Каждая строка содержит целые числа X_i, Y_i и B_i, разделенные пробелами, где X_i, Y_i ($-1\,000 \le X_i, Y_i \le 1\,000$) — координаты здания, а B_i ($1 \le B_i \le 1000$) — число служащих в здании.

Описание бомбоубежищ содержится в последующих M строках. Каждая строка содержит целые числа P_j, Q_j и C_j, разделенные пробелами, где P_j, Q_j ($-1\,000 \le P_j, Q_j \le 1\,000$) — координаты бомбоубежища, а C_j ($1 \le C_j \le 1000$) — вместимость бомбоубежища.

В последующихся N строках содержится описание плана эвакуации. Каждая строка представляет собой описание плана эвакуации для отдельного здания. План эвакуации из i-го здания состоит из M целых чисел E_{ij}, разделенных пробелами. E_{ij} ($0 \le E_{ij} \le 10\,000$) — количество служащих, которые должны эвакуироваться из i-го здания в j-е бомбоубежище.

Гарантируется, что план, заданный во входном файле, корректен.

Формат выходного файла

Если план эвакуации Городского Совета оптимален, то выведите одно слово `OPTIMAL`. В противном случае выведите на первой строке слово `SUBOPTIMAL`, а в последующих N строках выведите Ваш план эвакуации (более оптимальный) в том же формате, что и во входном файле. Ваш план не обязан быть оптимальным, но должен быть лучше плана Городского Совета.

Пример

evacuate.in	evacuate.out
3 4 -3 3 5 -2 2 6 2 2 5 -1 1 3 1 1 4 -2 -2 7 0 -1 3 3 1 1 0 0 0 6 0 0 3 0 2	SUBOPTIMAL 3 0 1 1 0 0 6 0 0 4 0 1
3 4 -3 3 5 -2 2 6 2 2 5 -1 1 3 1 1 4 -2 -2 7 0 -1 3 3 0 1 1 0 0 6 0 0 4 0 1	OPTIMAL

Задача F. Пересечение двух отрезков

Имя входного файла:	intersection.in
Имя выходного файла:	intersection.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Формат входного файла

Восемь чисел — координаты концов двух отрезков.

Формат выходного файла

Одна строка "YES", если отрезки имеют общие точки, и "NO" в противном случае.

Пример

intersection.in	intersection.out
5 1 2 6 1 1 7 8	YES

Задача G. Малыш и Карлсон

Имя входного файла:	karlsson.in
Имя выходного файла:	karlsson.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

На свой День рождения Малыш позвал своего лучшего друга Карлсона. Мама испекла его любимый пирог прямоугольной формы $a \times b \times c$ сантиметров. Карлсон знает, что у Малыша еще есть килограмм колбасы. Чтобы заполучить ее, он предложил поиграть следующим образом: они по очереди разрезают пирог на две ненулевые по объему прямоугольные части с целыми измерениями и съедают меньшую часть (в случае, когда части равные, можно съесть любую). Проигрывает тот, кто не может сделать хода (то есть когда размеры будут $1 \times 1 \times 1$). Естественно, победителю достается колбаса.

Малыш настаивает на том, чтобы он ходил вторым.

Помогите Карлсону выяснить, сможет ли он выиграть, и если сможет — какой должен быть его первый ход для этого.

Считается, что Малыш всегда ходит оптимально.

Формат входного файла

Во входном файле содержится 3 целых числа a, b, c ($1 \le a, b, c \le 5\,000$) — размеры пирога.

Формат выходного файла

В случае, если Карлсон не сможет выиграть в Малыша, выведите NO. В противном случае в первой строке выведите YES, во второй — размеры пирога после первого хода Карлсона в том же порядке, что и во входном файле.

Примеры

karlsson.in	karlsson.out
1 1 1	NO
2 1 1	YES 1 1 1

Задача H. Вставка ключевых значений

Имя входного файла:	key.in
Имя выходного файла:	key.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Вас наняла на работу компания MacroHard, чтобы вы разработали новую структуру данных для хранения целых ключевых значений.

Эта структура выглядит как массив A бесконечной длины, ячейки которого нумеруются с единицы. Изначально все ячейки пусты. Единственная операция, которую необходимо поддерживать — это операция $Insert(L, K)$, где L — положение в массиве, а K — некоторое положительное целое ключевое значение.

Операция выполняется следующим образом:

- Если ячейка $A[L]$ пуста, то присвоить $A[L] := K$.
- Если ячейка $A[L]$ непуста, выполнить $Insert(L + 1, A[L])$, а затем присвоить $A[L] := K$.

По заданной последовательности из N целых чисел $L_1, L_2, \ldots, L_N$ вам необходимо вывести содержимое этого массива после выполнения следующей последовательности операций:

$Insert(L_1, 1)$
$Insert(L_2, 2)$
...
$Insert(L_N, N)$

Формат входного файла

В первой строке входного файла содержится N — число операций $Insert$ и M — максимальный номер позиции, которую можно использовать в операции $Insert$. ($1 \le N \le 131\,072$, $1 \le M \le 131\,072$).

В следующей строке даны N целых чисел L_i, которые описывают операции $Insert$ ($1 \le L_i \le M$).

Формат выходного файла

Выведите содержимое массива после выполнения данной последовательности операций $Insert$. На первой строке выведите W — номер последней несвободной позиции в массиве. Далее выведите W целых чисел — $A[1], A[2], \ldots, A[W]$. Для пустых ячеек выводите нули.

Пример

key.in	key.out
5 4 3 3 4 1 3	6 4 0 5 2 3 1

Задача I. Матрица

Имя входного файла:	`matrix.in`
Имя выходного файла:	`matrix.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Вам дана матрица целых чисел размера $n \times n$. Ваша задача — найти такой набор координат (k_i, l_i), в котором каждая координата k_i и каждая координата l_i встречается ровно один раз, такой, чтобы минимизировать сумму выбранных элементов.

Формат входного файла

Первая строка входного файла содержит одно целое число n ($1 \le n \le 239$). Следующие n строк содержат по n целых чисел в каждой. Все эти числа не превосходят по абсолютной величине 10^6.

Формат выходного файла

Первая строка должна содержать значение оптимизируемой функции. В следующие n строк необходимо записать пары чисел, описывающих выбранные ячейки. Первой координатой выводится номер строки.

Пример

matrix.in	matrix.out
2 1 1 1 1	2 1 1 2 2

Задача J. Лабиринт знаний

Имя входного файла:	`maze.in`
Имя выходного файла:	`maze.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Участникам сборов подарили билеты на аттракцион "Лабиринт знаний". Лабиринт представляет собой N комнат, занумерованных от 1 до N, между некоторыми из которых есть двери. Когда человек проходит через дверь, показатель его знаний изменяется на определенную величину, фиксированную для данной двери. Вход в лабиринт находится в комнате 1, выход — в комнате N. Каждый участник сборов проходит лабиринт ровно один раз и набирает некоторое количество знаний (при входе в лабиринт этот показатель равен нулю). Ваша задача — показать наилучший результат.

Формат входного файла

Первая строка входного файла содержит целые числа N ($1 \le N \le 2\,000$) — количество комнат и M ($1 \le M \le 10\,000$) — количество дверей. В каждой из следующих M строк содержится описание двери — номера комнат, из которой она ведет и в которую она ведет (через дверь в лабиринте можно ходить только в одну сторону), а также целое число, которое прибавляется к количеству знаний при прохождении через дверь (это число по модулю не превышает 10 000). Двери могут вести из комнаты в нее саму, между двумя комнатами может быть более одной двери.

Формат выходного файла

В выходной файл выведите ":)" — если можно получить неограниченно большой запас знаний, ":(" — если лабиринт пройти нельзя, и максимальное количество набранных знаний в противном случае.

Пример

maze.in	maze.out
2 2 1 2 5 1 2 -5	5
6 6 1 2 1 2 6 1 2 3 2 3 4 1 4 5 1 5 3 1	2

Задача K. Мега-инверсии

Имя входного файла:	`mega.in`
Имя выходного файла:	`mega.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Инверсией в перестановке $p_1, p_2, ..., p_N$ называется пара (i, j) такая, что $i < j$ и $p_i > p_j$. Назовем мега-инверсией в перестановке $p_1, p_2, ..., p_N$ тройку (i, j, k) такую, что $i < j < k$ и $p_i > p_j > p_k$. Придумайте алгоритм для быстрого подсчета количества мега-инверсий в перестановке.

Формат входного файла

Первая строка входного файла содержит целое число N ($1 \le N \le 100\,000$). Следующие N чисел описывают перестановку: $p_1, p_2, ..., p_N$ ($1 \le p_i \le N$), все p_i попарно различны. Числа разделяются пробелами и/или переводами строк.

Формат выходного файла

Единственная строка выходного файла должна содержать одно число, равное количеству мега-инверсий в перестановке $p_1, p_2, \cdots, p_N$.

Примеры

mega.in	mega.out
4 4 3 2 1	4

Задача L. Наименьшее кратное

Имя входного файла:	`multiple.in`
Имя выходного файла:	`multiple.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дано число X и множество цифр D. Требуется дописать к X минимальное количество цифр из D, чтобы получившееся число делилось на k. При этом получившееся число должно быть минимально возможным.

Формат входного файла

Первая строка входного файла содержит два натуральных числа X и k ($1 \le X \le 10^{1000}\, 2 \le k \le 100\,000$). Во второй строке записано количество цифр во множестве D. В третьей строке через пробел записаны эти цифры.

Формат выходного файла

Единственная строка должна содержать минимальное число, полученное из X дописыванием цифр из D и кратное k. Если такого числа не существует, выведите `-1`.

Пример

multiple.in	multiple.out
102 101 3 1 0 3	10201
202 101 3 1 0 3	202

Задача M. Период строки

Имя входного файла:	period.in
Имя выходного файла:	period.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Строка S имеет период T, если

$$\exists n : S = T^n = \underbrace{T, \ldots, T}_{n}$$

Вам дана строка S. Ваша задача — найти минимальную по длине T такую, что $S = T^n$ для некоторого $n \in \mathbb{N}$

Формат входного файла

Строка S длиной от 1 до 10^6 символов.

Формат выходного файла

Единственное чило — длина T.

Примеры

period.in	period.out
abaabaabaabaaba	3

Задача N. Точки сочленения

Имя входного файла:	points.in
Имя выходного файла:	points.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан неориентированный граф. Требуется найти все точки сочленения в нем.

Формат входного файла

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($n \le 20\,000$, $m \le 200\,000$).

Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается двумя натуральными числами b_i, e_i — номерами концов ребра ($1 \le b_i, e_i \le n$).

Формат выходного файла

Первая строка выходного файла должна содержать одно натуральное число b — количество точек сочленения в заданном графе. На следующей строке выведите b целых чисел — номера вершин, которые являются точками сочленения, в возрастающем порядке.

Пример

points.in	points.out
9 12 1 2 1 3 2 3 1 4 4 5 1 5 2 6 6 7 2 7 3 8 8 9 3 9	3 1 2 3

Задача O. Простая задача

Имя входного файла:	prosto.in
Имя выходного файла:	prosto.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Найдите количество натуральных чисел на данном отрезке от a до b включительно, не делящихся нацело ни на одно из заданных различных простых чисел p_i.

Формат входного файла

В первой строке входного файла заданы два числа a и b — границы отрезка ($1 \le a \le b \le 10^{18}$). Во второй строке задано количество простых чисел n ($1 \le n \le 9$). В третьей строке перечислены сами простые числа p_i. Все числа p_i различны и не превосходят 100.

Формат выходного файла

Необходимо вывести ответ на задачу.

Пример

prosto.in	prosto.out
5 10 2 2 3	2
20 40 2 3 7	12
50 100 1 17	48
100 200 3 2 3 5	28

Задача P. Корень из перестановки

Имя входного файла:	root.in
Имя выходного файла:	root.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Девочка Маша написала на доске последовательность из N различных натуральных чисел из диапазона от 1 до N. Затем к доске подошел хулиган Вася и написал свою последовательность: если на i-м месте в Машиной последовательности стоит число j, а на j — число k, то на i-ое место в своей последовательности Вася пишет число k. Затем он безжалостно стер Машину последовательность. Помогите бедной девушке восстановить утерянную последовательность!

Формат входного файла

В первой строке входного файла записано число N ($1 \le N \le 1\,000$). Во второй строке идут N чисел — последовательность, написанная Васей.

Формат выходного файла

В выходной файл выведите последовательность, написанную Машей. Если существует несколько вариантов — выведите любой.

Пример

root.in	root.out
3 2 3 1	3 1 2

Задача Q. Остовное дерево 2

Имя входного файла:	`spantree2.in`
Имя выходного файла:	`spantree2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Требуется найти в связном графе остовное дерево минимального веса.

Формат входного файла

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно. Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается тремя натуральными числами b_i, e_i и w_i — номера концов ребра и его вес соответственно ($1 \le b_i, e_i \le n$, $0 \le w_i \le 100\,000$). $n \le 20\,000$, $m \le 100\,000$.

Граф является связным.

Формат выходного файла

Первая строка выходного файла должна содержать одно натуральное число — вес минимального остовного дерева.

Примеры

spantree2.in	spantree2.out
4 4 1 2 1 2 3 2 3 4 5 4 1 4	7

Задача R. Звезды

Имя входного файла:	`stars.in`
Имя выходного файла:	`stars.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Вася любит наблюдать за звездами. Но следить за всем небом сразу ему тяжело. Поэтому он наблюдает только за частью пространства, ограниченной кубом размером $n \times n \times n$. Этот куб поделен на маленькие кубики размером $1 \times 1 \times 1$. Во время его наблюдений могут происходить следующие события:

1. В каком-то кубике появляются или исчезают несколько звезд.
2. К нему может заглянуть его друг Петя и поинтересоваться, сколько видно звезд в части пространства, состоящей из нескольких кубиков.

Формат входного файла

Первая строка входного файла содержит натуральное число $1 \le n \le 128$. Координаты кубиков — целые числа от 0 до $n-1$. Далее следуют записи о происходивших событиях по одной в строке. В начале строки записано число m. Если m равно:

1, то за ним следуют 4 числа — x, y, z ($0 \le x, y, z < N$) и k ($-20000 \le k \le 20000$) — координаты кубика и величина, на которую в нем изменилось количество видимых звезд;

2, то за ним следуют 6 чисел — x_1, y_1, z_1, x_2, y_2, z_2 ($0 \le x_1 \le x_2 < N$, $0 \le y_1 \le y_2 < N$, $0 \le z_1 \le z_2 < N$), которые означают, что Петя попросил подсчитать количество звезд в кубиках (x, y, z) из области: $x_1 \le x \le x_2$, $y_1 \le y \le y_2$, $z_1 \le z \le z_2$;

3, то это означает, что Васе надоело наблюдать за звездами и отвечать на вопросы Пети. Эта запись встречается во входном файле только один раз и будет последней записью.

Количество записей во входном файле не больше 100 002.

Формат выходного файла

Для каждого Петиного вопроса выведите на отдельной строке одно число — искомое количество звезд.

Пример

stars.in	stars.out
2 2 1 1 1 1 1 1 1 0 0 0 1 1 0 1 0 3 2 0 0 0 0 0 0 2 0 0 0 0 1 0 1 0 1 0 -2 2 0 0 0 1 1 1 3	0 1 4 2

Задача S. И снова сумма...

Имя входного файла:	`sum.in`
Имя выходного файла:	`sum.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Реализуйте структуру данных, которая поддерживает множество S целых чисел, с котором разрешается производить следующие операции:

- $add(i)$ — добавить в множество S число i (если он там уже есть, то множество не меняется);
- $sum(l, r)$ — вывести сумму всех элементов x из S, которые удовлетворяют неравенству $l \le x \le r$.

Формат входного файла

Исходно множество S пусто. Первая строка входного файла содержит n — количество операций ($1 \le n \le 300\,000$). Следующие n строк содержат операции. Каждая операция имеет вид либо «`+` i», либо «`?` l r». Операция «`?` l r» задает запрос $sum(l, r)$.

Если операция «`+` i» идет во входном файле в начале или после другой операции «`+`», то она задает операцию $add(i)$. Если же она идет после запроса «`?`», и результат этого запроса был y, то выполняется операция $add((i + y) \bmod 10^9)$.

Во всех запросах и операциях добавления параметры лежат в интервале от 0 до 10^9.

Формат выходного файла

Для каждого запроса выведите одно число — ответ на запрос.

Пример

sum.in	sum.out
6 + 1 + 3 + 3 ? 2 4 + 1 ? 2 4	3 7

Задача T. Обмен

Имя входного файла:	swap.in
Имя выходного файла:	swap.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Пусть все натуральные числа исходно организованы в список в естественном порядке. Разрешается выполнить следующую операцию: $swap(a, b)$. Эта операция возвращает в качестве результата расстояние в текущем списке между числами a и b и меняет их местами.

Задана последовательность операций *swap*. Требуется вывести в выходной файл результат всех этих операций.

Формат входного файла

Первая строка входного файла содержит число n ($1 \le n \le 200\,000$) — количество операций. Каждая из следующих n строк содержит по два числа в диапазоне от 1 до 10^9 — аргументы операций *swap*.

Формат выходного файла

Для каждой операции во входном файле выведите ее результат.

Пример

swap.in	swap.out
4 1 4 1 3 4 5 1 4	3 1 4 2