

## Задача А. Постфиксная запись

Имя входного файла: `postfix.in`  
Имя выходного файла: `postfix.out`  
Ограничение по времени: 1 second  
Ограничение по памяти: 64 megabytes

В постфиксной записи (или обратной польской записи) операция записывается после двух операндов. Например, сумма двух чисел  $A$  и  $B$  записывается как  $AB+$ . Запись  $BC+D*$  обозначает привычное нам  $(B+C)*D$ , а запись  $ABC+D*+$  означает  $A+(B+C)*D$ . Достоинство постфиксной записи в том, что она не требует скобок и дополнительных соглашений о приоритете операторов для своего чтения.

Дано выражение в обратной польской записи. Определите его значение.

### Формат входных данных

В единственной строке записано выражение в постфиксной записи, содержащее однозначные числа и операции  $+$ ,  $-$ ,  $*$ . Строка содержит не более 100 чисел и операций.

### Формат выходных данных

Необходимо вывести значение записанного выражения. Гарантируется, что результат выражения, а также результаты всех промежуточных вычислений по модулю меньше  $2^{31}$ .

### Примеры

<code>postfix.in</code>	<code>postfix.out</code>
8 9 + 1 7 - *	-102

## Задача В. Скобки

Имя входного файла: `brackets.in`  
Имя выходного файла: `brackets.out`  
Ограничение по времени: 0.5 second  
Ограничение по памяти: 64 megabytes

Требуется определить, является ли правильной данная последовательность круглых, квадратных и фигурных скобок.

### Формат входных данных

В единственной строке входного файла записано подряд  $N$  скобок ( $1 \leq N \leq 10^5$ ).

### Формат выходных данных

В выходной файл вывести «YES», если данная последовательность является правильной, и «NO» в противном случае.

### Примеры

<code>brackets.in</code>	<code>brackets.out</code>
()	YES
([])	YES

## Задача С. Игра в пьяницу

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

В игре в пьяницу карточная колода раздаётся поровну двум игрокам. Далее они вскрывают по одной верхней карте, и тот, чья карта старше, забирает себе обе вскрытые карты, которые кладутся под низ его колоды. Тот, кто остаётся без карт — проигрывает.

Для простоты будем считать, что все карты различны по номиналу, а также, что самая младшая карта побеждает самую старшую карту («шестерка берет туза»).

Игрок, который забирает себе карты, сначала кладёт под низ своей колоды карту первого игрока, затем карту второго игрока (то есть карта второго игрока оказывается внизу колоды).

Напишите программу, которая моделирует игру в пьяницу и определяет, кто выигрывает. В игре участвует  $n$  карт, имеющих значения от 0 до  $n-1$ , большая карта побеждает меньшую, карта со значением 0 побеждает карту  $n-1$ .

### Формат входных данных

Программа получает на вход три строки. В первой строке содержится целое чётное число  $n$  ( $2 \leq n \leq 100\,000$ ). Вторая строка содержит  $\frac{n}{2}$  чисел — карты первого игрока, а третья —  $\frac{n}{2}$  карт второго игрока. Карты перечислены сверху вниз, то есть каждая строка начинается с той карты, которая будет открыта первой. Гарантируется, что каждая из карт встречается в колодах игроков ровно один раз.

### Формат выходных данных

Программа должна определить, кто выигрывает при данной раздаче, и вывести слово «first» или «second», после чего вывести количество ходов, сделанных до выигрыша. Если на протяжении  $2 \cdot 10^5$  ходов игра не заканчивается, программа должна вывести слово «draw».

### Примеры

<code>card-game.in</code>	<code>card-game.out</code>
10	second 5
1 3 5 7 9	
2 4 6 8 0	

## Задача D. Очередь

Имя входного файла: `queue.in`  
Имя выходного файла: `queue.out`  
Ограничение по времени: 3 секунды  
Ограничение по памяти: 256 мегабайт

Гоблины Мглистых гор очень любят ходить к своим шаманам. Так как гоблинов много, к шаманам часто образуются очень длинные очереди. А поскольку много гоблинов в одном

месте быстро образуют шумную толпу, которая мешает шаманам производить сложные медицинские манипуляции, последние решили установить некоторые правила касательно порядка в очереди.

Обычные гоблины при посещении шаманов должны вставать в конец очереди. Привилегированные же гоблины, знающие особый пароль, встают ровно в ее середину, причем при нечетной длине очереди они встают сразу за центром.

Так как гоблины также широко известны своим непочтительным отношением ко всяческим правилам и законам, шаманы попросили вас написать программу, которая бы отслеживала порядок гоблинов в очереди.

### Формат входных данных

В первой строке записано одно целое число  $N$  ( $1 \leq N \leq 10^5$ ) — число запросов к вашей программе. В следующих  $N$  строках заданы описания запросов в следующем формате:

- «+  $i$ » — к очереди присоединяется гоблин  $i$  ( $1 \leq i \leq N$ ) и встает в ее конец;
- «\*  $i$ » — привилегированный гоблин  $i$  встает в середину очереди ( $1 \leq i \leq N$ );
- «-» — гоблин выходит из очереди и заходит к шаманам. Гарантируется, что на момент каждого такого запроса очередь будет не пуста.

### Формат выходных данных

Для каждого запроса третьего типа в отдельной строке выведите номер гоблина, который должен зайти к шаманам.

### Примеры

queue.in	queue.out
7	1
+ 1	2
+ 2	3
-	
+ 3	
+ 4	
-	
-	
10	1
+ 1	3
+ 2	2
* 3	5
-	4
+ 4	
* 5	
-	
-	
-	
-	

### Задача Е. Рыцарский турнир

Имя входного файла: `knight-in-the-row.in`  
Имя выходного файла: `knight-in-the-row.out`  
Ограничение по времени: 3 секунды  
Ограничение по памяти: 256 мегабайт

Ура! Король Берлядии Берл II устраивает рыцарский турнир. Король уже разослал послание всем рыцарям королевства, а они, в свою очередь, дали согласие на участие в этом грандиозном событии.

Что же касается вас, то вы — простой крестьянин. Не удивительно, что рыцарский турнир вы проспали (ведь он проводился в выходной), и теперь вам очень хочется узнать его результаты. В этот раз рыцарский турнир в Берляндии проходил следующим образом:

- В турнире принимали участие  $n$  рыцарей. Каждому рыцарю был присвоен уникальный номер — целое число от 1 до  $n$ .
- Турнир проводился в  $m$  сражений, в  $i$ -ом сражении все еще не выбывшие рыцари с номерами не меньше  $l_i$  и не больше  $r_i$  боролись за право продолжить участие в турнире.
- После  $i$ -го сражения среди всех рыцарей, которые боролись, победил только один — рыцарь с номером  $x_i$ , он продолжил участие в турнире. Остальные рыцари выбыли из турнира.
- Победитель самого последнего ( $m$ -го) сражения (рыцарь с номером  $x_m$ ) был объявлен победителем всего турнира.

Вы узнали у своих друзей информацию про все сражения, и теперь для каждого рыцаря вам интересно знать, каким рыцарем он был побежден. Считается, что рыцарь с номером  $a$  победил рыцаря с номером  $b$ , если было такое сражение, в котором боролись оба этих рыцаря, а победителем из этого сражения вышел рыцарь с номером  $a$ .

Напишите программу, вычисляющую для каждого рыцаря, каким рыцарем он был побежден.

### Формат входных данных

В первой строке записано два целых числа  $n, m$  ( $2 \leq n \leq 3 \cdot 10^5$ ;  $1 \leq m \leq 3 \cdot 10^5$ ) — количество рыцарей и количество сражений. В каждой из следующих  $m$  строк записано три целых числа  $l_i, r_i, x_i$  ( $1 \leq l_i < r_i \leq n$ ;  $l_i \leq x_i \leq r_i$ ) — описание  $i$ -го сражения.

Гарантируется, что входные данные корректны и соответствуют условию задачи. Гарантируется, что в каждом сражении участвовали как минимум два рыцаря.

### Формат выходных данных

Выведите  $n$  целых чисел. Если  $i$ -ый рыцарь был побежден, то  $i$ -ое число должно быть равно номеру рыцаря, который победил рыцаря с номером  $i$ . Если  $i$ -ый рыцарь является победителем турнира, то  $i$ -ое число должно быть равно 0.

### Примеры

<code>knight-in-the-row.in</code>	<code>knight-in-the-row.out</code>
-----------------------------------	------------------------------------

**Замечание**

Рассмотрим первый тестовый пример. В первом сражении бились рыцари 1 и 2, победил рыцарь 1. Во втором сражении бились рыцари 1 и 3, победил рыцарь 3. В последнем сражении бились рыцари 3 и 4, победил рыцарь 4.

Задача взята с сайта [codeforces.ru](http://codeforces.ru)