

## у2018-1-2. Стек, очередь, СММ

### А. Минимум на стеке

ограничение по времени на тест: 2 секунды  
ограничение по памяти на тест: 256 мегабайт  
ввод: стандартный ввод  
вывод: стандартный вывод

Вам требуется реализовать структуру данных, выполняющую следующие операции:

1. Добавить элемент  $x$  в конец структуры.
2. Удалить последний элемент из структуры.
3. Выдать минимальный элемент в структуре.

#### Входные данные

В первой строке входного файла задано одно целое число  $n$  — количество операций ( $1 \leq n \leq 10^6$ ). В следующих  $n$  строках заданы сами операции. В  $i$ -ой строке число  $t_i$  — тип операции (1, если операция добавления. 2, если операция удаления. 3, если операция минимума). Если задана операция добавления, то через пробел записано целое число  $x$  — элемент, который следует добавить в структуру ( $-10^9 \leq x \leq 10^9$ ). Гарантируется, что перед каждой операцией удаления или нахождения минимума структура не пуста.

#### Выходные данные

Для каждой операции нахождения минимума выведите одно число — минимальный элемент в структуре. Ответы разделяйте переводом строки.

#### Пример

**входные данные**

Скопировать

```
8
1 2
```

```
1 3
1 -3
3
2
3
2
3
```

выходные данные

Скопировать

```
-3
2
2
```

## В. Шарики

ограничение по времени на тест: 2 секунды  
ограничение по памяти на тест: 256 мегабайт  
ввод: стандартный ввод  
вывод: стандартный вывод

В одной компьютерной игре игрок выставляет в линию шарики разных цветов. Когда образуется непрерывная цепочка из трех и более шариков одного цвета, она удаляется из линии. Все шарики при этом сдвигаются друг к другу, и ситуация может повториться.

Напишите программу, которая по данной ситуации определяет, сколько шариков будет сейчас уничтожено. Естественно, непрерывных цепочек из трех и более одноцветных шаров в начальный может быть не более одной.

### Входные данные

Даны количество шариков в цепочке (не более  $10^5$ ) и цвета шариков (от 0 до 9, каждому цвету соответствует свое целое число).

### Выходные данные

Требуется вывести количество шариков, которое будет уничтожено.

### Примеры

входные данные

Скопировать

```
5 1 3 3 3 2
```

выходные данные

Скопировать

3

#### ВХОДНЫЕ ДАННЫЕ

Скопировать

10 3 3 2 1 1 1 2 2 3 3

#### ВЫХОДНЫЕ ДАННЫЕ

Скопировать

10

## С. Астроград

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

ввод: стандартный ввод

вывод: стандартный вывод

В Астрополисе прошел концерт популярной группы Астроград. За пару дней до концерта перед кассой выстроилась огромная очередь из людей, желающих туда попасть. Изначально очередь была пуста. В каждый из  $n$  моментов времени происходило следующее:

1. В очередь пришел новый человек с уникальным номером  $id$ , он встает в очередь последним.
2. Человеку, стоящему спереди очереди, удалось купить билет. Он уходит.
3. Человеку, стоящему последнему в очереди, надоело ждать. Он уходит.
4. Человек с уникальным номером  $q$  хочет знать, сколько людей стоит в очереди спереди него.
5. Очередь хочет знать, человек с каким уникальным номером стоит сейчас первым и задерживает всех.

Вам необходимо написать программу, которая умеет обрабатывать описанные события.

### Входные данные

В первой строке дано целое число  $n$  ( $1 \leq n \leq 10^5$ ) — количество событий. В каждой из следующих  $n$  строк дано описание событий: номер события, а также число  $id$  ( $1 \leq id \leq 10^5$ ) для событий типа 1 и число  $q$  для событий типа 4. События происходили в том порядке, в каком они описаны во входном файле. Гарантируется корректность всех событий.

### Выходные данные

Выведите ответы для событий типа 4 и 5 в том порядке, в каком они описаны во входном файле.

### Пример

входные данные	Скопировать
<pre> 7 1 1 5 1 3 3 2 1 2 4 2 </pre>	
выходные данные	Скопировать
<pre> 1 0 </pre>	

### Примечание

В примере из условия происходили следующие события:

1. В очередь пришел человек с  $id = 1$ . Очередь: [ 1 ]
2. Первым в очереди стоит человек с  $id = 1$ . Очередь: [ 1 ]
3. В очередь пришел человек с  $id = 3$ . Очередь: [ 1, 3 ]
4. Последнему в очереди надоело стоять и он уходит. Очередь: [ 1 ]
5. Первому в очереди удалось купить билет и он уходит. Очередь: [ ]
6. В очередь пришел человек с  $id = 2$ . Очередь: [ 2 ]
7.  $q = 2$  хочет знать, сколько человек стоит перед ним. Очередь: [ 2 ]

## D. Гоблины и шаманы

ограничение по времени на тест: 2 секунды  
ограничение по памяти на тест: 256 мегабайт  
ввод: стандартный ввод  
вывод: стандартный вывод

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 толку, которая мешает шаманам проводить 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.

### Входные данные

В первой строке входных данных записано число  $N$  ( $1 \leq N \leq 5 \cdot 10^5$ ) - количество запросов к программе. Следующие  $N$  строк содержат описание запросов в формате:

- „+ i” - гоблин с номером  $i$  ( $1 \leq i \leq N$ ) встает в конец очереди.
- „\* i” - привилегированный гоблин с номером  $i$  встает в середину очереди.
- „-” - первый гоблин из очереди уходит к шаманам. Гарантируется, что на момент такого запроса очередь не пуста.

### Выходные данные

Для каждого запроса типа „-” программа должна вывести номер гоблина, который должен уйти к шаманам.

### Пример

входные данные	Скопировать
7 + 1 + 2 - + 3 + 4 - -	
выходные данные	Скопировать
1 2 3	

## Е. Постфиксная запись

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

ввод: стандартный ввод  
вывод: стандартный вывод

В постфиксной записи (или обратной польской записи) операция записывается после двух операндов. Например, сумма двух чисел  $A$  и  $B$  записывается как  $A\ B\ +$ . Запись  $B\ C\ +\ D\ *$  обозначает привычное нам  $(B + C) * D$ , а запись  $A\ B\ C\ +\ D\ * +$  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.

### Входные данные

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

### Выходные данные

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

### Пример

<b>входные данные</b>	Скопировать
8 9 + 1 7 - *	
<b>выходные данные</b>	Скопировать
-102	

## F. Сортировка стеком

ограничение по времени на тест: 1 секунда  
ограничение по памяти на тест: 256 мегабайт  
ввод: стандартный ввод  
вывод: стандартный вывод

### Примеры

<b>входные данные</b>	Скопировать

5 5 3 1 2 4	
<b>выходные данные</b>	Скопировать
push push push pop push pop pop pop push pop pop	

<b>входные данные</b>	Скопировать
3 2 3 1	
<b>выходные данные</b>	Скопировать
impossible	

## G. Система непересекающихся множеств

ограничение по времени на тест: 2 секунды  
ограничение по памяти на тест: 256 мегабайт  
ввод: стандартный ввод  
вывод: стандартный вывод

Реализуйте систему непересекающихся множеств. Вместе с каждым множеством храните минимальный, максимальный элемент в этом множестве и их количество.

### Входные данные

Первая строка входного файла содержит  $n$  — количество элементов в носителе ( $1 \leq n \leq 300\,000$ ). Далее операций с множеством. Операция `get` должна возвращать минимальный, максимальный элемент в соответствующем множестве, а также их количество.

### Выходные данные

Выведите последовательно результат выполнения всех операций `get`.

### Пример

входные данные	Скопировать
5 union 1 2 get 3 get 2 union 2 3 get 2 union 1 3 get 5 union 4 5 get 5 union 4 1 get 5	
выходные данные	Скопировать
3 3 1 1 2 2 1 3 3 5 5 1 4 5 2 1 5 5	

## Н. Подсчет опыта

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 64 мегабайта

ввод: стандартный ввод

вывод: стандартный вывод

В очередной онлайн игре игроки, как обычно, сражаются с монстрами и набирают опыт. Для того, чтобы сражаться с монстрами, они объединяются в кланы. После победы над монстром, всем участникам клана, победившего его, добавляется одинаковое число единиц опыта. Особенностью этой игры является то, что кланы никогда не распадаются и из клана нельзя выйти. Единственная доступная операция — объединение двух кланов в один.

Поскольку игроков стало уже много, вам поручили написать систему учета текущего опыта игроков.



## Входные данные

В первой строке входного файла содержатся числа  $n$  ( $1 \leq n \leq 200000$ ) и  $m$  ( $1 \leq m \leq 200000$ ) — число зарегистрированных игроков и число запросов.

В следующих  $m$  строках содержатся описания запросов. Запросы бывают трех типов:

- `join X Y` — объединить кланы, в которые входят игроки  $X$  и  $Y$  (если они уже в одном клане, то ничего не меняется).
- `add X V` — добавить  $V$  единиц опыта всем участникам клана, в который входит игрок  $X$  ( $1 \leq V \leq 100$ ).
- `get X` — вывести текущий опыт игрока  $X$ .

Изначально у всех игроков 0 опыта и каждый из них состоит в клане, состоящим из него одного.

## Выходные данные

Для каждого запроса `get X` выведите текущий опыт игрока  $X$ .

### Пример

входные данные	Скопировать
<pre>3 6 add 1 100 join 1 3 add 1 50 get 1 get 2 get 3</pre>	
выходные данные	Скопировать
<pre>150 0 50</pre>	