

Задача А. Стек (!) (1 балл)

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

Реализуйте работу стека. Для каждой операции изъятия элемента выведите ее результат.

На вход программе подаются строки, содержащие команды. Каждая строка содержит одну команду. Команда — это либо “+ N”, либо “-”. Команда “+ N” означает добавление в стек числа N , по модулю не превышающего 10^9 . Команда “-” означает изъятие элемента из стека. Гарантируется, что не происходит извлечения из пустого стека. Гарантируется, что размер стека в процессе выполнения команд не превысит 10^6 элементов.

Формат входного файла

В первой строке входного файла содержится количество команд — M ($1 \leq M \leq 10^6$). Каждая последующая строка исходного файла содержит ровно одну команду.

Формат выходного файла

Выведите числа, которые удаляются из стека, по одному в каждой строке. Гарантируется, что изъятий из пустого стека не производится.

Пример

stack.in	stack.out
6	10
+ 1	1234
+ 10	
-	
+ 2	
+ 1234	
-	

Задача В. Очередь (1 балл)

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

Реализуйте работу очереди. Для каждой операции изъятия элемента выведите ее результат.

На вход программе подаются строки, содержащие команды. Каждая строка содержит одну команду. Команда — это либо “+ N”, либо “-”. Команда “+ N” означает добавление в очередь числа N , по модулю не превышающего 10^9 . Команда “-” означает изъятие элемента из очереди. Гарантируется, что размер очереди в процессе выполнения команд не превысит 10^6 элементов.

Формат входного файла

В первой строке содержится количество команд — M ($1 \leq M \leq 10^6$). В последующих строках содержатся команды, по одной в каждой строке.

Формат выходного файла

Выведите числа, которые удаляются из очереди, по одному в каждой строке. Гарантируется, что извлечения из пустой очереди не производится.

Пример

<code>queue.in</code>	<code>queue.out</code>
4	1
+ 1	10
+ 10	
-	
-	

Задача С. Правильная скобочная последовательность (1 балл)

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

Входной файл содержит несколько строк, каждая из которых содержит последовательность символов '(', ')', '[' и ']'. Выясните, является ли она правильной скобочной последовательностью с двумя типами скобок.

Подсказка: используйте стек.

Формат входного файла

Входной файл содержит $1 \leq n \leq 500$ строк, каждая из которых содержит скобочную последовательность длиной $1 \leq l \leq 10^4$.

Формат выходного файла

Для каждой строки входного файла выведите в выходной файл «YES», если соответствующая последовательность является правильной скобочной последовательностью, или «NO», если не является.

Пример

<code>brackets.in</code>	<code>brackets.out</code>
<code>()()</code>	YES
<code>([])</code>	YES
<code>([]]</code>	NO
<code>(([]</code>	NO
<code>)()</code>	NO

Задача D. Постфиксная запись (2 балла)

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

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.

Подсказка: используйте стек.

Формат входного файла

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

Формат выходного файла

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

Пример

postfix.in	postfix.out
8 9 + 1 7 - *	-102

Задача Е. Приоритетная очередь (3 балла)

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

Реализуйте приоритетную очередь. Ваша очередь должна поддерживать следующие операции: добавить элемент, извлечь минимальный элемент, уменьшить элемент, добавленный во время одной из операций.

Все операции нумеруются по порядку, начиная с единицы. Гарантируется, что размер очереди в процессе выполнения команд не превысит 10^6 элементов.

Формат входного файла

Входной файл содержит описание операций с очередью. Операции могут быть следующими:

- `push x` — требуется добавить элемент x в очередь.
- `extract-min` — требуется удалить из очереди минимальный элемент и вывести его в выходной файл. Если очередь пуста, в выходной файл требуется вывести звездочку `*`.
- `decrease-key x y` — требуется заменить значение элемента, добавленного в очередь операцией `push` в строке входного файла номер x , на y . Гарантируется, что на строке x действительно находится операция `push`, что этот элемент не был ранее удален операцией `extract-min`, и что y меньше, чем предыдущее значение этого элемента.

В очередь помещаются и извлекаются только целые числа, не превышающие по модулю 10^9 .

Формат выходного файла

Выведите последовательно результат выполнения всех операций `extract-min`, по одному в каждой строке выходного файла. Если перед очередной операцией `extract-min` очередь пуста, выведите вместо числа звездочку `*`.

Пример

<code>priorityqueue.in</code>	<code>priorityqueue.out</code>
<code>push 3</code>	<code>2</code>
<code>push 4</code>	<code>1</code>
<code>push 2</code>	<code>3</code>
<code>extract-min</code>	<code>*</code>
<code>decrease-key 2 1</code>	
<code>extract-min</code>	
<code>extract-min</code>	
<code>extract-min</code>	