

# N. Радио

Ограничение времени	1 секунда
Ограничение памяти	32.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Компания «Аудио Квадрат» захватила значительную долю рынка музыкальных товаров и услуг. В частности она делает персональное радио для каждого человека, которое учитывает его музыкальные предпочтения и текущее настроение.

Но параллельно, just for fun, компания запускает проект общественного радио. Это радио, в котором песня выбирается в соответствии с пожеланиями множества проголосовавших пользователей.

Алгоритм общественного радио безумно прост. Посетители интернет сайта «Аудио Квадрата» постоянно голосуют за отдельные композиции (с одного IP можно проголосовать в течении 10 минут только за одну композицию). Голос может иметь вес *score*. Для каждой композиции считаются очки — *track\_score*. Следующей композицией, которая будет играть на радио, станет та, которая имеет максимальное количество очков. Если таких композиций несколько, то должна играть та, которая имеет минимальный идентификатор. В момент, когда композиция запускается на радио, её *track\_score* становится равным -1.

Чтобы исключить попытки накручивания счётчика определёнными группами лиц, компания решила принимать голоса с одного IP не чаще раза в 10 минут (два голоса могут быть приняты, если между ними не менее шестисот секунд).

Вы являетесь главным разработчиком компании «Аудио Квадрат». И хотя у вас в подчинении множество высококлассных программистов, вы решили вспомнить старые добрые времена и запрограммировать этот алгоритм сами.

Общественных радио будет много, поэтому важно, чтобы каждое из них работало достаточно эффективно, то есть не отнимало много процессорных ресурсов и оперативной памяти.

Необходимо написать программу, реализующую алгоритм работы одного общественного радио. Взаимодействие с программой будет осуществляться через стандартный поток ввода/вывода согласно заданному ниже протоколу.

Для решения этой задачи используйте ассоциативные контейнеры и очереди с приоритетами.

## Формат ввода

Каждая строка входа — это определенная команда. Всего 3 типа команд:

- *VOTE* *ip* *track\_id* *score* *time*
- *GET*
- *EXIT*

Команда *VOTE* меняет число очков *track\_score* определённой композиции. Она получает четыре аргумента:

- *ip* — IP адрес компьютера, с которого пришел голос; четыре целых числа из промежутка  $[0, 255]$ , разделённых точкой
- *track\_id* — численный идентификатор композиции; натуральное число из отрезка  $[1, 2 \cdot 10^7]$ ;
- *score* — количество очков, которое нужно добавить к текущим очкам композиции *track\_score* — целое число из отрезка  $[-100, 100]$ ;
- *time* — момент времени в секундах от некоторого фиксированного момента времени — целое число из промежутка  $[0, 2 \cdot 10^9]$ .

На команду *VOTE* нужно отвечать новым значением очков музыкальной композиции *track\_id* (даже если эти очки не изменились).

Команда *GET* используется для получения следующей композиции. Ваша программа должна отвечать на неё парой *track\_id* и *track\_score* (вывести эти два числа в одной строке, разделив их пробелом). Сразу после выполнения этой команды новое значение *track\_score* для этой композиции должно стать равным *-1*.

Команда *EXIT* находится в последней строке входа. При её получении необходимо вывести строку «ОК». Число команд на входе не более 100001. Изначально значение *track\_score* для всех композиций равно 0.


Команды во входных данных упорядочены по параметру *time* (для двух из трёх команд этот параметр просто не передаётся, так как не нужен).

## Формат вывода

Выход должен содержать ровно столько же строчек, что и вход. Каждая строка выхода — это ответ на соответствующую команду из входа.

## Пример 1


Ввод 

Вывод 

GET	1 0
VOTE 1.1.1.1 1 -1 1	-2
VOTE 1.1.1.1 2 1 2	0
VOTE 1.1.1.1 1 2 4	-2
VOTE 1.1.1.1 1 4 20045	2
GET	1 2
GET	2 0
GET	3 0
VOTE 194.85.81.128 3 -1 20049	-2
EXIT	OK

## Пример 2

Ввод 

Вывод 

GET	1 0
GET	2 0
GET	3 0
VOTE 192.168.0.1 2 2 11111111	1
EXIT	OK