**You shall not pass!**

Молодой, но очень перспективный разработчик Олег к концу испытательного срока настолько преисполнился идеей аккаунтинга, лимитирования и квотирования запросов, что решил, что каждый его сервис будет поддерживать эти технологии. Проведя ночь с ноутбуком в обнимку, он написал целую библиотеку, которая позволяла бы ограничивать рейт запросов в свой сервис, основываясь на аутентификационной информации (проще говоря, токену).

Первый MVP библиотеки позволял отдельно взятому пользователю ходить в его сервис не более *N* раз в секунду. Таким образом, если в момент времени *t* пользователь *u* отправил запрос, то он будет пропущен (принят), если пользователь *u* сделал с момента времени *t*−1 сек. меньше *N* запросов.

Смекнув, что в случае нерабочего кода тут пахнет серьезными денежными потерями, Олег покрыл практически всю свою библиотеку unit-тестами.

Тесты подавали на вход записанный упрощенный лог запросов, вида:

1 10000

1676217286000 AYjcyMzY3ZDhiNmJkNTY

1676217286009 RjY2NjM5NzA2OWJjuE7c

1676217286010 AYjcyMzY3ZDhiNmJkNTY

1676217286010 RjY2NjM5NzA2OWJjuE7c

1676217286100 AYjcyMzY3ZDhiNmJkNTY

1676217286199 RjY2NjM5NzA2OWJjuE7c

1676217287012 RjY2NjM5NzA2OWJjuE7c

1676217287612 RjY2NjM5NzA2OWJjuE7c

...

1676217892348 Th7UHJdLswIYQxwSg29D

Где первая строка указывала рейт запросов в секунду и количество записей лога запроса, а последующие - лог запросов, в которых первое поле - расширенный unix timestamp с миллисекундной точностью, а второе - OAuth токен пользователя.

На выходе тесты проверяли статус запроса: будет ли он пропущен или нет. При этом выводились **только пропущенные** запросы:

1676217286000 AYjcyMzY3ZDhiNmJkNTY

1676217286009 RjY2NjM5NzA2OWJjuE7c

...

1676217892348 Th7UHJdLswIYQxwSg29D

К сожалению, утром за день до конца испытательного срока Олег узнал, что его библиотека умерла вместе с жестким диском, но зато остались живы тесты, которые он успел закоммитить. Олег просит Вас помочь с его проблемой, восстановив утерянную функциональнось.

**Формат ввода**

На вход даются два числа *N* и *K*, разделенные пробельными символами, где 1≤*N*≤106, 1≤*K*≤106.

Далее следуют *K* строк, каждая из которых содержит 2 поля: расширенный Unix-timestamp с **миллисекундной точностью** (unsigned 64-bit integer), и строку - токен, идентифицирующий пользователя. Токен имеет размер в точности равный 20 символов из [a-zA-Z0-9] алфавита.

Гарантируется, что ввод отсортирован по первому полю (timestamp) и что каждая строка всегда оканчивается символом \n. Суммарное количество пользователей не превышает 1000.

**Формат вывода**

В ответе нужно вывести тот же самый лог запросов, оставив из него только те запросы, которые будут приняты (не будут отфильтрованы).

Обратите внимание, что каждую строку следует выводить без изменений, включая символ \n в конце строки.

**Пример 1**

Ввод

1 1

1679981968781 0fdc847ecc495d6dd584

Вывод

1679981968781 0fdc847ecc495d6dd584

**Пример 2**

Ввод

2 4

1679982033000 441079aa62dc3cd57df3

1679982033998 441079aa62dc3cd57df3

1679982033999 441079aa62dc3cd57df3

1679982034000 441079aa62dc3cd57df3

Вывод

1679982033000 441079aa62dc3cd57df3

1679982033998 441079aa62dc3cd57df3

1679982034000 441079aa62dc3cd57df3

**Примечание**

В этой задаче большой объем входных и выходных данных. Рекомендуется использовать построчный ввод/вывод, иначе возможен выход за пределы допустимой памяти.

Ограничение памяти

64.0 Мб

Ограничение времени

4 с

Ввод

стандартный ввод или input.txt

Вывод

стандартный вывод или output.txt