# Задание по программированию: Электронная книга

## Условие

Разработайте систему стимулирования чтения электронных книг. Для простоты будем считать, что книга всего одна, но её одновременно читают много людей. Необходимо следить за прогрессом чтения у всех пользователей и выводить мотивирующие уведомления. А именно, ваша программа должна обрабатывать следующие события:

* **READ user page** — сохранить факт того, что пользователь под номером ***user*** дочитал книгу до страницы ***page***. Если ранее такой пользователь не встречался, необходимо его добавить. Гарантируется, что в рамках одного пользователя номера страниц в соответствующих ему событиях возрастают.
* **CHEER user** — сообщить пользователю ***user***, какая доля существующих пользователей (не считая его самого) прочитала меньшую часть книги, чем он. Если этот пользователь на данный момент единственный, доля считается равной 1. Если для данного пользователя пока не было ни одного события **READ**, доля считается равной 0, а сам пользователь не учитывается при вычислении долей для других пользователей до тех пор, пока для него не случится событие **READ**.

## Формат входных данных

В первой строке вводится количество запросов Q — натуральное число, не превосходящее 10^6. В следующих Q строках в соответствии с описанным выше форматом вводятся запросы. Гарантируется, что все вводимые числа целые и положительные, при этом номера пользователей не превосходят 10^5, а номера страниц не превосходят 1000.

## Формат выходных данных

Для каждого запроса **CHEER user** выведите единственное вещественное число от 0 до 1 — долю пользователей, прочитавших меньше страниц, чем ***user***. Формат вывода этого числа должен быть в точности таким же, как в опубликованном ниже медленном решении.

## Ограничения

4 секунды на выполнение всех запросов. Все описанные в условии гарантии действительно справедливы для всех тестов, на которых будет запускаться ваша программа. Проверять корректность тестов не нужно.

## Пример

### Ввод

12

CHEER 5

READ 1 10

CHEER 1

READ 2 5

READ 3 7

CHEER 2

CHEER 3

READ 3 10

CHEER 3

READ 3 11

CHEER 3

CHEER 1

### Вывод

0

1

0

0.5

0.5

1

0.5

### Комментарии к примеру

Пользователь 5 не учитывается при вычислении долей, потому что для него не произошло ни одного события **READ**.

Пользователь 1 изначально был единственным, но в конце его обогнал 3-й, но не обогнал 2-й, поэтому он оказался продуктивнее 50 % пользователей.

Пользователь 3 изначально обгонял только 2-го и потому получал долю 50 %, но в конце обогнал 1-го и получил долю 100 %.

## Правильное, но медленное решение

[slow.cpp](https://d3c33hcgiwev3.cloudfront.net/7YdiJmS5EeiQ1RJfHSpJKg_ee2eec2064b911e8a488d1e3a26d9d2c_slow.cpp?Expires=1617062400&Signature=M9UlBUc6YKrKGW4ZQdLXbhnne9Dq0dH3eTicQ2LnZw1kntDiqXa-s0OWGJQKgGwwJollPl9ARRxDchaQOZvo3nxDN6pZYcaFinPFcpQ6GNanIf~kzBiJH44Pl5f3guBJT-P80KE-~f45JQLLOMc23lNVZmzUgsBzG3LZ0HyzB9k_&Key-Pair-Id=APKAJLTNE6QMUY6HBC5A)

## Подсказки

Если задача вызывает непреодолимые трудности, вы можете воспользоваться подсказками. Они помогут быстрее сдать задачу, но в итоге вы получите меньше пользы от процесса поиска решения.

## Подсказка 1

Обратите внимание: номера страниц не превосходят 1000. Это довольно маленькие числа, поэтому решение не обязано быть линейным.

## Подсказка 2

Так или иначе, в Read и/или Cheer будут вложенные циклы. Подумайте, как организовать их так, чтобы амортизированная сложность обработки этих запросов была приемлемой. (Именно этим страдает медленное решение.)

## Подсказка 3

Часто в таких задачах помогает придумать решение со следующей точки зрения: можно ли использовать такие структуры данных, чтобы запросы на чтение (имеются в виду запросы на чтение данных из класса, в данном случае это Cheer) работали за O(1)?

## Подсказка 4

Порассуждаем над предыдущей подсказкой. Чтобы Cheer работал за O(1), нужно достать откуда-то чуть ли не готовый ответ. Параметр этого метода — user\_id. Можно ли хранить где-то готовый рейтинг каждого пользователя?

## Подсказка 5

Чтобы поддерживать текущий рейтинг для каждого пользователя, пришлось обновлять бы его в Read для всех пользователей с одинаковым рейтингом, а это именно та проблема, которой страдает медленное решение. Поэтому стоит подумать на шаг вперёд: раз пользователей много, страниц мало, и вдобавок у всех пользователей с одинаковым количеством прочитанных страниц одинаковый рейтинг, можно хранить некий «рейтинг» каждой страницы; обновлять его в Read и обращаться к рейтингу нужной страницы в Cheer.