

## Задача А. Книжная коллекция Губки Боба

Имя входного файла: `swap.in`  
Имя выходного файла: `swap.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Недавно Губка Боб купил себе коллекцию из  $2n$  книг по истории дна. Коллекция устроена следующим образом: каждая книга имеет свой уникальный порядковый номер, первые  $n$  номеров коллекции содержат в себе раннюю историю дна, а следующие  $n$  книг с номерами от  $n + 1$  до  $2n$  повествуют новую историю. Для коллекции Боб выделил целую полку и поставил книги в порядке возрастания их порядковых номеров.

Губка Боб ушел на работу, а дома остался его любимый питомец улитка Гэрри. Гэрри очень шаловливый для своего вида, поэтому начал сразу же переставлять книги. Переставляя книги, он вынимает ровно две из них, затем ставит первую на старую позицию второй, а вторую на старую позицию первой. Таким образом, он просто меняет книги местами.

Вернувшись домой, Губка Боб обнаружил шалость Гэрри и был сильно недоволен. К счастью, в комнате стояла камера, которая зафиксировала каждое действие Гэрри. Губка Боб очень любит коллекцию книг по ранней истории дна, и теперь по данным камеры он хочет узнать, сколько книг с номерами от 1 до  $n$  было среди первых  $n$  книг на полке после каждого переставления Гэрри. Более формально, после каждой смены книг местами Губка Боб хочет знать количество чисел  $x$ , таких, что  $1 \leq x \leq n$  и книга с номером  $x$  находится среди первых  $n$  на полке.

Губка Боб готов предоставить вам записи перестановок книг, а также просит вас написать программу, которая после каждой перестановки будет вычислять количество книг по ранней истории дна, которые находятся на полке в позициях от 1 до  $n$ .

### Формат входного файла

В первой строке входного файла содержится одно целое число  $n$  ( $1 \leq n \leq 100\,000$ ) — количество книг по старой истории дна, равное количеству книг по новой истории дна.

Во второй строке входного файла дано число  $m$  ( $1 \leq m \leq 100\,000$ ) — количество перестановок, которые совершил Гэрри.

В следующих  $m$  строках записаны пара чисел  $i_k, j_k$  ( $1 \leq i_k, j_k \leq 2n$ ), которые означают, что  $k$ -ой перестановкой Гэрри поменял местами книги, которые стояли в позициях  $i_k, j_k$ .

### Формат выходного файла

В единственной строке выходного файла выведите  $m$  чисел — количество книг по старой истории дна, которые находятся на полке в позициях от 1 до  $n$ , после каждой перестановки Гэрри.

### Пример

<code>swap.in</code>	<code>swap.out</code>
3	2
3	1
1 4	0
2 5	
3 6	
4	4
3	3
8 7	3
1 5	
1 2	

## Комментарий

Изначально книги стояли на полке в порядке  $\{1, 2, 3, 4, 5, 6\}$ . После перестановки 1-й и 4-й книг полка стала выглядеть так:  $\{4, 2, 3, 1, 5, 6\}$ . После следующей перестановки книги на полке стояли в порядке  $\{4, 5, 3, 1, 2, 6\}$ , а после следующей перестановки в порядке  $\{4, 5, 6, 1, 2, 3\}$ . Количество чисел от 1 до  $n$  в первой половине массива после каждой из перестановок равны 2, 1 и 0 соответственно.

## Система оценивания

Первая группа тестов состоит из тестов, для которых выполняется ограничение  $1 \leq n, m \leq 1000$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 50 баллов.

Вторая группа тестов состоит из тестов, для которых выполняются полные ограничения. Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 50 баллов.

Обратите внимание на возможность узнать результат проверки вашего решения на всех тестах, нажав на ссылку «Request feedback» на вкладке «Runs».

## Задача В. Ловушки

Имя входного файла:	trap.in
Имя выходного файла:	trap.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Мало кому известно, что под самым посещаемым рестораном Бикини Боттом — Красти Крабс — находится бункер. В нем хранится самое дорогое сокровище Мистера Крабса — секретная формула Крабсбургера. Каким-то образом Планктон узнал о местонахождении вожделенной формулы и решил ее похитить! Но провести Юджина Крабса не так-то просто. Практически на каждом шагу в этом бункере расположена сигнализация. И конечно же неуклюжий Планктон задел один из механизмов, который поднял шумную тревогу. Планктон бросился бежать, но за ним уже мчался Губка Боб.

Планктон и Губка Боб бегут по координатной прямой. В начальный момент времени Губка Боб находится в точке 0, а Планктон в точке  $s$ . Скорость Планктона —  $v_1$  метров в секунду, скорость Боба —  $v_2$  метров в секунду. У Планктона также имеется  $k$  одинаковых ловушек. Он знает, что если оставить ловушку в точке  $x_i$ , Губка Боб потратит дополнительно  $a_i$  секунд на то, чтобы ее преодолеть. Ваша задача состоит в том, чтобы узнать, как долго может продолжаться погоня.

Считается, что Губка Боб догнал планктона, если их координаты совпадают. Если Губка Боб догонит Планктона в тот момент, когда тот устанавливает ловушку, Планктон считается пойманным.

### Формат входного файла

В первой строке входного файла даны два числа  $n, k$  ( $1 \leq n, k \leq 100\,000$ ) — количество точек, в которых можно установить ловушки и количество ловушек у Планктона соответственно.

В следующей строке входного файла даны два числа  $v_1, v_2$  ( $1 \leq v_1, v_2 \leq 1000$ ) — скорость Планктона и Губки Боба соответственно.

В следующей строке дано число  $s$  ( $0 \leq s \leq 100\,000\,000$ ) — координата, с которой начал свое движение Планктон.

В следующих  $n$  строках дано по два числа  $x_i, a_i$  ( $0 \leq x_i \leq 100\,000\,000, 0 \leq a_i \leq 1000$ ) — описание  $i$ -й точки, в которую можно поставить ловушку.

Гарантируется, что для всех  $1 \leq i \leq n - 1$  верно  $x_{i+1} > x_i$ .

### Формат выходного файла

В единственной строке выходного файла выведите единственное число — ответ на задачу.

Ответ будет считаться верным, если он имеет относительную или абсолютную погрешность не более  $10^{-6}$ .

Если погоня будет продолжаться бесконечно, в единственной строке выходного файла выведите «inf» (без кавычек).

## Пример

trap.in	trap.out
6 2 1 2 3 0 1 5 2 7 3 10 4 11 5 12 6	13.000000
1 1 2 1 1 1 1	inf

## Система оценивания

Первая группа тестов состоит из тестов, для которых выполняются ограничения  $n, k \leq 1000$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 40 баллов.

Вторая группа тестов состоит из тестов, для которых выполняются ограничения  $n, k \leq 100\,000$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 60 баллов.

Обратите внимание на возможность узнать результат проверки вашего решения на всех группах тестов, нажав на ссылку «Request feedback» на вкладке «Runs».

## Задача С. Занимательная игра

Имя входного файла: `game.in`  
Имя выходного файла: `game.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Губка Боб и Патрик уже битый час не могут придумать, чем им заняться. Сквидвард решил их выручить и предложил сыграть в следующую игру: есть доска, на которой изначально написано число 1. На своем ходе каждый из игроков может стереть число, написанное на доске, и написать вместо него либо в два раза большее число, либо число, большее на единицу. Например, из числа 3 можно получить число 4 и число 6, а из числа 5 можно получить 6 или 10. Также в начале игры задается число  $n$  — верхняя граница. Она означает максимальное число, которое может быть записано на доску. Например, если  $n = 9$ , а на доске написано число 5, то следующим может быть выписано только число 6, потому что  $10 > 9$ , то есть число 10 недопустимо. Проигрывает в этой игре тот, кто не может сделать ход, то есть тот, перед чьим ходом на доске написано число  $n$ . Первым решил ходить Патрик.

Сыграв множество партий, Патрик задался следующим вопросом: как понять, сколько чисел являются выигрышными среди чисел от  $l$  до  $r$  включительно? Число  $n$  называется выигрышным, если при верхней границей равной  $n$  при оптимальной игре выигрывает Патрик.

Помогите ему справиться с этой нелегкой задачей, посчитайте по данным числам  $l, r$  количество выигрышных чисел на этом отрезке.

### Формат входного файла

В первой и единственной строке входного файла даны два числа  $l, r$  ( $1 \leq l \leq r \leq 10^{18}$ ) — границы отрезка.

### Формат выходного файла

В единственной строке выходного файла выведите ответа на задачу — количество выигрышных чисел на отрезке  $[l, r]$ .

### Пример

<code>game.in</code>	<code>game.out</code>
2 3	1
8 10	2

### Комментарий

В первом тестовом примере выигрышным является число 2: начав с числа 1, Патрик может сразу получить 2 и выиграть. Число 3 не является выигрышным, потому что из числа 1 Патрик может получить только 2, а тогда Губка Боб следующим ходом получит число 3 и выиграет.

### Система оценивания

Первая группа тестов состоит из тестов, для которых выполняется ограничение  $l \leq r \leq 10$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 10 баллов.

Вторая группа тестов состоит из тестов, для которых выполняется ограничение  $l \leq r \leq 1000$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 30 баллов.

Третья группа тестов состоит из тестов, для которых выполняются ограничения  $l \leq r \leq 10^9$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 30 баллов.

Четвертая группа тестов состоит из тестов, для которых выполняются ограничения  $l \leq r \leq 10^{18}$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 30 баллов.

Обратите внимание на возможность узнать результат проверки вашего решения на всех группах тестов, нажав на ссылку **«Request feedback»** на вкладке **«Runs»**.

## Задача D. Крабсбургеры

Имя входного файла:	xor.in
Имя выходного файла:	xor.out
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Как известно, главная особенность рецепта Крабсбургера — это отсутствие какого-либо рецепта. Для того, чтобы иметь хоть какую-то информацию о бургерах, каждый ингредиент имеет некоторую характеристику — КрабсИндекс. Известно лишь одно: если Крабсбургер был приготовлен с использованием  $m$  ингредиентов с КрабсИндексами  $a_1, a_2, \dots, a_m$ , то КрабсИндекс получившегося бургера будет равен  $a_1 \oplus a_2 \oplus \dots \oplus a_m$ .

Одним солнечным утром в КрастиКраб зашли  $k$  посетителей. Каждый заказал себе фирменный Крабсбургер, причем  $i$ -й посетитель пожелал, чтобы КрабсИндекс его бургера был не меньше, чем  $l_i$ , но и не больше, чем  $r_i$ .

Перед поваром стоит непростая задача — он должен приготовить все  $k$  бургеров, удовлетворив пожелания клиентов, и обязательно использовав все имеющиеся ингредиенты. Перед ним лежит в ряд  $n$  ингредиентов. Он  $k$  раз должен взять несколько (не меньше одного) ингредиентов из левого конца ряда, и приготовить из них новый бургер, проследив, чтобы он удовлетворял заказу. Повар готовит бургеры в том же порядке, в котором поступили заказы от клиентов.

Помогите ему узнать, сколько существует способов справиться с задачей. Два способа считаются различными, если есть хотя бы один ингредиент, попавший в бургеры к разным клиентам в этих разбиениях.

### Формат входного файла

В первой строке входного файла даны два целых числа  $n, k$  ( $1 \leq n \cdot k \leq 100\,000, k \leq n$ ) — количество ингредиентов и число клиентов.

В следующей строке дано  $n$  целых чисел  $a_i$  ( $0 \leq a_i \leq 1\,000\,000\,000$ ) — КрабсИндексы ингредиентов.

В следующих  $k$  строках дано по 2 целых числа  $l_i, r_i$  ( $0 \leq l_i \leq r_i \leq 1\,000\,000\,000$ ) — пожелания о КрабсИндексе от  $i$ -го клиента.

### Формат выходного файла

Выведите одно целое число — количество искомых разбиений по модулю  $1\,000\,000\,007$ .

### Комментарий

Исключающее или ( $\oplus$ ) — логическая операция, которая имеет следующую таблицу истинности:

- $0 \oplus 0 = 0$
- $0 \oplus 1 = 1$
- $1 \oplus 0 = 1$
- $1 \oplus 1 = 0$

«Исключающее или» чисел, состоящих из нескольких бит, считается побитово. Например,  $2 \oplus 3 = 1$ ,  $2 \oplus 5 = 7$ ,  $5 \oplus 5 = 0$ .

Более подробно про «исключающее или» можно почитать тут:  
[https://ru.wikipedia.org/wiki/Сложение\\_по\\_модулю\\_2](https://ru.wikipedia.org/wiki/Сложение_по_модулю_2)

## Пример

xor.in	xor.out
7 3	6
1 0 1 0 1 0 1	
1 1	
0 0	
1 1	

## Система оценивания

Первая группа тестов состоит из тестов, для которых выполняется ограничение  $1 \leq n \leq 100, 1 \leq k \leq 4$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 20 баллов.

Вторая группа тестов состоит из тестов, для которых выполняется ограничение  $1 \leq n \leq 500, 1 \leq k \leq 100$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 40 баллов.

Третья группа тестов состоит из тестов, для которых выполняется ограничение  $1 \leq n \cdot k \leq 100\,000, k \leq n$ . Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 40 баллов.

Обратите внимание на возможность узнать результат проверки вашего решения на всех группах тестов, нажав на ссылку «Request feedback» на вкладке «Runs».



## Задача Е. Ковер

Имя входного файла: `carpet.in`  
Имя выходного файла: `carpet.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Сегодня Сквидвард купил длинный рулон ткани. Он состоит из  $n$  сшитых подряд кусочков ткани, каждый из которых покрашен в какой-то из 26 цветов. Будем считать этот рулон строкой  $s$  длины  $n$ , причем, каждому цвету соответствует строчная буква латинского алфавита.

Сквидвард хочет вырезать из этого рулона ткани коврик для своей мамы. Для этого он выберет какой-то непрерывный непустой подотрезок этого рулона, вырежет его — это и будет коврик для его мамы. Но это еще не все. Из оставшихся частей он хочет сшить точно такой же коврик. Для этого он хочет вырезать из левой и правой оставшихся частей по кусочку ткани и сшить их вместе в таком же порядке — это будет коврик для него. При этом Сквидвард хочет, чтобы эти коврики были абсолютно одинаковыми — тем самым он хочет показать маме свою любовь к ней. Ему стало интересно, сколькими способами можно вырезать коврик.

Формально говоря, он хочет выяснить, сколько существует таких пар индексов  $i, j$ , ( $1 \leq i \leq j \leq n$ ), для которых найдутся такие  $i_1, j_1, i_2, j_2$  ( $1 \leq i_1 \leq j_1 < i, j < i_2 \leq j_2 \leq n$ ), что  $s[i..j] = s[i_1..j_1] + s[i_2..j_2]$ . ( $s[l..r]$  — это подстрока строки  $s$ , с  $l$ -го символа по  $r$ -й символ включительно).

Сам Сквидвард не в состоянии посчитать это число, поэтому он попросил вас помочь ему.

### Формат входного файла

В первой строке входного файла дана строка  $s$  ( $1 \leq |s| \leq 100\,000$ ), состоящая из строчных букв латинского алфавита.

### Формат выходного файла

В единственной строке выходного файла выведите одно число — количество способов вырезать коврик.

### Пример

<code>carpet.in</code>	<code>carpet.out</code>
aaaa	1
abababb	3

### Комментарий

В первом примере существует единственная подходящая пара индексов (2, 3). Во втором примере существуют три подходящие пары индексов (3, 4), (5, 6) и (4, 6).

### Система оценивания

Первая группа тестов состоит из тестов, для которых длина строки не превосходит 100. Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 20 баллов.

Вторая группа тестов состоит из тестов, для которых длина строки не превосходит 2 000. Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 30 баллов.

Третья группа тестов состоит из тестов, для которых длина строки не превосходит 100 000. Баллы за эту группу начисляются только при прохождении всех тестов группы. Стоимость группы составляет 50 баллов.

Обратите внимание на возможность узнать результат проверки вашего решения на всех тестах, нажав на ссылку «Request feedback» на вкладке «Runs».