


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## A. Легионы Цезаря

ограничение по времени на тест: 2 seconds

ограничение по памяти на тест: 256 megabytes

Знаменитый полководец Гай Юлий Цезарь любил выстраивать воинов своей армии в шеренгу. Всего в армии было  $n_1$  пехотинцев и  $n_2$  всадников. Цезарь считал, что расстановка **не** красивая, если где-то в строю стоит подряд строго больше  $k_1$  пехотинцев или строго больше  $k_2$  всадников. Найдите количество **красивых** расстановок воинов.

Учтите, что в каждой расстановке должны присутствовать все  $n_1 + n_2$  воинов. Все пехотинцы считаются неразличимыми между собой. Аналогично, все всадники считаются неразличимыми между собой.

### Входные данные

В единственной строке через пробел записаны четыре целых числа  $n_1, n_2, k_1, k_2$  ( $1 \leq n_1, n_2 \leq 100, 1 \leq k_1, k_2 \leq 10$ ) — количество пехотинцев и всадников в армии, а также наибольшее допустимое количество стоящих подряд пехотинцев и всадников, соответственно.

### Выходные данные

Выведите количество красивых расстановок войск по модулю  $100000000$  ( $10^8$ ), то есть количество таких расстановок, где подряд стоит не более  $k_1$  пехотинцев и не более  $k_2$  всадников.

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
2 1 1 10	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
1	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
2 3 1 2	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
5	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
2 4 1 1	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
0	

### Примечание

Обозначим пехотинца как 1, а всадника как 2.

В первом примере единственное красивое построение: 121

Во втором примере существует 5 красивых построений: 12122, 12212, 21212, 21221, 22121




[ЗАДАЧИ](#)   [ОТОСЛАТЬ](#)   [МОИ ПОСЫЛКИ](#)   [СТАТУС](#)   [ПОЛОЖЕНИЕ](#)   [ЗАПУСК](#)

## B. Ехаб снова делит массив

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Ехаб снова решил развлечься. У него есть массив  $a$  длины  $n$ . Он называет массив хорошим, если его нельзя разбить на 2 непересекающихся подпоследовательности такие, что сумма элементов первой равна сумме элементов второй. Сейчас он хочет удалить минимальное количество элементов из  $a$  так, что оставшийся массив будет хорошим. Вы можете ему в этом помочь?

Последовательность  $a$  является подпоследовательностью  $b$ , если  $a$  может быть получена из  $b$  удалением нескольких (возможно, ни одного или всех) элементов. Разбить массив это разделить его на 2 подпоследовательности так, чтобы каждый элемент входил ровно в одну из них. Вы должны использовать все элементы по одному разу.

### Входные данные

В первой строке записано одно целое число  $n$  ( $2 \leq n \leq 100$ ) — размер массива  $a$ .

Во второй строке записаны  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 2000$ ) — элементы массива  $a$ .

### Выходные данные

Первая строка должна содержать минимальное количество элементов, которое нужно удалить.

Вторая строка должна содержать индексы этих элементов, записанные через пробел.

Если существует несколько решений, выведите любое из них. Можно показать, что решение всегда существует.

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
4 6 3 9 12	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
1 2	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
2 1 2	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
0	

### Примечание

В первом примере вы можете разбить массив на  $[6, 9]$  и  $[3, 12]$ , так что необходимо удалить хотя бы 1 элемент. Удаление 3 подходит.

Во втором примере массив уже хороший.




[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## C. Нулевой путь

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам дана таблица с  $n$  строк и  $m$  столбцов. Обозначим ячейку в  $i$ -й ( $1 \leq i \leq n$ ) строке и  $j$ -м ( $1 \leq j \leq m$ ) столбце через  $(i, j)$ , а число в ней через  $a_{ij}$ . Все числа равны 1 или  $-1$ .

Вы начинаете с ячейки  $(1, 1)$  и можете перемещаться на одну ячейку вниз или вправо за один раз. В конце концов, вы хотите оказаться в ячейке  $(n, m)$ .

Можно ли двигаться таким образом, чтобы сумма значений, записанных во всех посещенных ячейках (включая  $a_{11}$  и  $a_{nm}$ ), была равна 0?

1	-1	-1	-1
-1	1	1	-1
1	1	1	-1

### Входные данные

Каждый тест содержит несколько наборов входных данных. Первая строка содержит количество наборов входных данных  $t$  ( $1 \leq t \leq 10^4$ ). Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа  $n$  и  $m$  ( $1 \leq n, m \leq 1000$ ) — размер таблицы.

Каждая из следующих  $n$  строк содержит  $m$  целых чисел.  $j$ -е целое число в  $i$ -й строке это  $a_{ij}$  ( $a_{ij} = 1$  или  $-1$ ) — элемент в ячейке  $(i, j)$ .

Гарантируется, что сумма  $n \cdot m$  по всем наборам входных данных не превышает  $10^6$ .

### Выходные данные

Для каждого набора входных данных выведите «YES», если существует путь из левого верхнего угла в правый нижний, который дает в сумме 0, и «NO» в противном случае. Вы можете выводить каждую букву в любом регистре.

### Пример

входные данные	Скопировать
<pre>5 1 1 1 1 2 1 -1 1 4 1 -1 1 -1 3 4 1 -1 -1 -1 -1 1 1 -1 1 1 1 -1 3 4 1 -1 1 1 -1 1 -1 1 1 -1 1 1</pre>	
выходные данные	Скопировать
<pre>NO YES YES YES NO</pre>	

### Примечание

Один из возможных путей для четвертого набора входных данных приведен на рисунке в утверждении.


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## D. Running Miles

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Есть улица с  $n$  достопримечательностями, достопримечательность с номером  $i$  находится в  $i$  милях от начала улицы. Достопримечательность номер  $i$  обладает красотой  $b_i$ . Вы хотите стартовать утром пробежку в  $l$  милях и закончить в  $r$  милях от начала улицы. Пока вы бежите, вы будете пробегать мимо каких-то достопримечательностей (включая достопримечательности в  $l$  и  $r$  милях от начала). Вам интересны 3 наиболее красивых достопримечательности с вашей пробежки, но вы устаете с каждой милей, которую пробегаете.

Выберите  $l$  и  $r$  такие, что вы пробежите мимо хотя бы 3 достопримечательностей, и сумма красот 3 самых красивых достопримечательностей минус дистанция в милях, которую вы пробегаете, максимальна. Более формально, выберите  $l$  и  $r$  такие, что  $b_{i_1} + b_{i_2} + b_{i_3} - (r - l)$  максимально, где  $i_1, i_2, i_3$  — индексы трех самых красивых достопримечательностей в диапазоне  $[l, r]$ .

### Входные данные

В первой строке находится единственное целое число  $t$  ( $1 \leq t \leq 10^5$ ) — число наборов входных данных.

Первая строка каждого набора входных данных содержит единственное целое число  $n$  ( $3 \leq n \leq 10^5$ ).

Вторая строка каждого набора входных данных содержит  $n$  целых чисел  $b_i$  ( $1 \leq b_i \leq 10^8$ ) — красоты достопримечательностей в  $i$  милях от начала улицы.

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит  $10^5$ .

### Выходные данные

Для каждого набора входных данных выведите одно целое число — максимальное значение  $b_{i_1} + b_{i_2} + b_{i_3} - (r - l)$  для некоторого отрезка  $[l, r]$ .

### Пример

входные данные	Скопировать
4 5 5 1 4 2 3 4 1 1 1 1 6 9 8 7 6 5 4 7 100000000 1 100000000 1 100000000 1 100000000	
выходные данные	Скопировать
8 1 22 299999996	

### Примечание

В первом примере мы можем выбрать  $l$  и  $r$  равными 1 и 5. Так мы посетим все достопримечательности, и три достопримечательности с максимальной красотой имеют индексы 1, 3 и 5 с красотами 5, 4 и 3 соответственно. Откуда суммарное значение равно  $5 + 4 + 3 - (5 - 1) = 8$ .

Во втором примере отрезок  $[l, r]$  может быть равен  $[1, 3]$  или  $[2, 4]$ , с суммарным значением, равным  $1 + 1 + 1 - (3 - 1) = 1$ .

## E. Оптимизация дорог

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 128 мегабайт

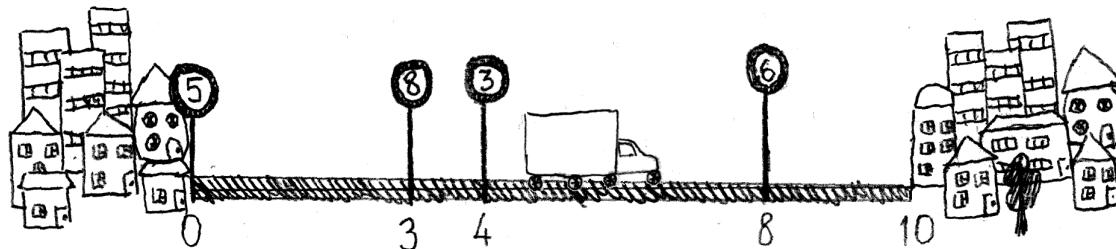
Правительство Марса заинтересовано не только в оптимизации космических перелетов, но и в улучшении дорожной системы планеты.

Одна из самых важных магистралей Марса соединяет Олимп-Сити и Кцолоп, столицу Кидонии. В рамках этой задачи будем рассматривать только путь от Кцолопа до Олимп-Сити, но не обратный путь (от Олимп-Сити до Кцолопа).

Дорога от Кцолопа до Олимп-Сити имеет длину  $\ell$  километров. Каждая точка дороги имеет координату  $x$  ( $0 \leq x \leq \ell$ ) — расстояние до Кцолопа в километрах. Таким образом, Кцолоп находится в точке с координатой 0, а Олимп-Сити — в точке с координатой  $\ell$ .

На дороге стоит  $n$  дорожных знаков, на  $i$ -м из которых написано ограничение скорости  $a_i$ . Это ограничение означает, что следующий километр следует проехать за  $a_i$  минут и действует до тех пор, пока по пути не встретится следующий дорожный знак. Один из дорожных знаков стоит в самом начале дороги (т. е. в точке с координатой 0) и задает начальную скорость.

Зная расположение всех дорожных знаков, нетрудно вычислить время поездки от Кцолопа до Олимп-Сити. Рассмотрим пример:



В данном случае необходимо проехать первые три километра за пять минут каждый, затем — один километр за восемь минут, затем — еще четыре километра за три минуты каждый и, наконец, последние два километра за шесть минут каждый. Итого время в пути составит  $3 \cdot 5 + 1 \cdot 8 + 4 \cdot 3 + 2 \cdot 6 = 47$  минут.

С целью оптимизации движения правительство Марса решило убрать не более  $k$  дорожных знаков. При этом знак в начале дороги убирать нельзя, иначе на начальном отрезке пути не будет никаких ограничений. Убирать знаки необходимо таким образом, чтобы минимизировать время в пути от Кцолопа до Олимп-Сити.

В Кидонии сосредоточены крупные промышленные предприятия Марса, поэтому оптимизация дороги до Олимп-Сити является приоритетной задачей. По этой причине правительство Марса поручило Вам решить эту задачу и выяснить, какие знаки необходимо убрать.

### Входные данные

В первой строке входных данных находится три целых числа  $n$ ,  $\ell$  и  $k$  ( $1 \leq n \leq 500$ ,  $1 \leq \ell \leq 10^5$ ,  $0 \leq k \leq n - 1$ ) — количество знаков на дороге от Кцолопа до Олимп-Сити, расстояние между этими городами и максимальное количество знаков, которые разрешается убрать.

Во второй строке входных данных находится  $n$  целых чисел  $d_i$  ( $d_1 = 0$ ,  $d_i < d_{i+1}$ ,  $0 \leq d_i \leq \ell - 1$ ) — координаты знаков.

В третьей строке входных данных находится  $n$  целых чисел  $a_i$  ( $1 \leq a_i \leq 10^4$ ) — ограничения скорости.

### Выходные данные

Выведите одно целое число — минимально возможное время в пути от Кцолопа до Олимп-Сити (в минутах), если убрать не более  $k$  дорожных знаков.

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
4 10 0 0 3 4 8 5 8 3 6	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
47	

**входные данные**

```
4 10 2
0 3 4 8
5 8 3 6
```

**выходные данные**

```
38
```

**Примечание**

В первом примере знаки удалять нельзя, и ответ равен 47, как сказано в условии задачи выше.

Во втором примере следует удалить второй и четвертый знаки. Тогда придется проехать первые четыре километра за  $4 \cdot 5 = 20$  минут, а последние шесть километров – за  $6 \cdot 3 = 18$  минут. Итого получаем  $4 \cdot 5 + 6 \cdot 3 = 38$  минут.

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов

Соревнования по программированию 2.0

Время на сервере: 05.05.2025 15:30:58<sup>UTC+5</sup> (n2).

Мобильная версия, переключиться на [десктопную](#).

[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке




[ЗАДАЧИ](#)   [ОТОСЛАТЬ](#)   [МОИ ПОСЫЛКИ](#)   [СТАТУС](#)   [ПОЛОЖЕНИЕ](#)   [ЗАПУСК](#)

## F. Монстры и заклинания

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Монокарп снова играет в компьютерную игру. Он ученик мага, поэтому знает только одно заклинание. К счастью, это заклинание может наносить урон монстрам.

Уровень, на котором он сейчас, содержит  $n$  монстров.  $i$ -й из них появляется через  $k_i$  секунд с начала уровня и имеет  $h_i$  очков здоровья. Дополнительно есть ограничение  $h_i \leq k_i$  для всех  $1 \leq i \leq n$ . Все  $k_i$  различны.

Монокарп может использовать заклинание в моменты времени, которые являются положительным целым количеством секунд с начала уровня:  $1, 2, 3, \dots$ . Урон от заклинания считается следующим образом. Если он не использовал заклинание в предыдущую секунду, то урон равен 1. Иначе, пусть урон в прошлую секунду был равен  $x$ . Тогда он может выбрать, чтобы урон был либо  $x + 1$ , либо 1. Заклинание использует ману: использование заклинания с уроном  $x$  использует  $x$  маны. Мана не восстанавливается.

Чтобы убить  $i$ -го монстра, Монокарп должен использовать заклинание с уроном хотя бы  $h_i$  ровно в тот момент, когда появляется монстр, что равно  $k_i$ .

Обратите внимание, что Монокарп может использовать заклинание и тогда, когда в текущую секунду монстра нет.

Количество маны, необходимое для использования всех заклинаний, — это сумма маны по всем использованным заклинаниям. Посчитайте наименьшее количество маны, которое потребуется Монокарпу, чтобы убить всех монстров.

Можно показать, что всегда можно убить всех монстров в рамках ограничений задачи.

### Входные данные

В первой строке записано одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных.

В первой строке каждого набора входных данных записано одно целое число  $n$  ( $1 \leq n \leq 100$ ) — количество монстров на уровне.

Во второй строке каждого набора входных данных записаны  $n$  целых чисел  $k_1 < k_2 < \dots < k_n$  ( $1 \leq k_i \leq 10^9$ ) — количество секунд с начала уровня до появления  $i$ -го монстра. Все  $k_i$  различны,  $k_i$  заданы в порядке возрастания.

В третьей строке записаны  $n$  целых чисел  $h_1, h_2, \dots, h_n$  ( $1 \leq h_i \leq k_i \leq 10^9$ ) — здоровье  $i$ -го монстра.

Сумма  $n$  по всем наборам входных данных не превосходит  $10^4$ .

### Выходные данные

На каждый набор входных данных выведите одно целое число — наименьшее количество маны, которое потребуется Монокарпу, чтобы убить всех монстров.

### Пример

входные данные	Скопировать
3 1 6 4 2 4 5 2 2 3 5 7 9 2 1 2	
выходные данные	Скопировать
10 6 7	

### Примечание

В первом наборе входных данных Монокарп может использовать заклинание через 3, 4, 5 и 6 секунд с начала уровня с уроном 1, 2, 3 и 4, соответственно. Урон в 6 секунду равен 4, что действительно больше или равно здоровью монстра, который появляется.


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## G. Палиндром из трех блоков (простая версия)

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

**Единственное отличие между простой и сложной версиями — ограничения.**

Вам дана последовательность  $a$ , состоящая из  $n$  положительных чисел.

Определим **палиндром из трех блоков** как последовательность, которая состоит из не более, чем двух различных элементов (назовем эти элементы  $a$  и  $b$ ,  $a$  может быть равно  $b$ ) и выглядит следующим образом  $\underbrace{[a, a, \dots, a]}_x, \underbrace{[b, b, \dots, b]}_y, \underbrace{[a, a, \dots, a]}_x$ .

Здесь  $x, y$  — числа, большие или равные 0. Например, последовательности  $[]$ ,  $[2]$ ,  $[1, 1]$ ,  $[1, 2, 1]$ ,  $[1, 2, 2, 1]$  и  $[1, 1, 2, 1, 1]$  — **палиндромы из трех блоков**, но  $[1, 2, 3, 2, 1]$ ,  $[1, 2, 1, 2, 1]$  и  $[1, 2]$  — нет.

Ваша задача — выбрать максимальную по длине **подпоследовательность**  $a$ , которая является **палиндромом из трех блоков**.

Вам нужно ответить на  $t$  независимых наборов тестовых данных.

Напомним, что последовательность  $t$  является подпоследовательностью последовательности  $s$ , если  $t$  может быть получена из  $s$  после удаления нуля или более элементов без изменения порядка оставшихся элементов. Например, если  $s = [1, 2, 1, 3, 1, 2, 1]$ , то возможные подпоследовательности:  $[1, 1, 1, 1]$ ,  $[3]$  и  $[1, 2, 1, 3, 1, 2, 1]$ , но не  $[3, 2, 3]$  and  $[1, 1, 1, 1, 2]$ .

### Входные данные

Первая строка теста содержит одно целое число  $t$  ( $1 \leq t \leq 2000$ ) — количество наборов тестовых данных. Затем следуют  $t$  наборов тестовых данных.

Первая строка каждого набора содержит одно целое число  $n$  ( $1 \leq n \leq 2000$ ) — длину  $a$ . Вторая строка набора содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 26$ ), где  $a_i$  — это  $i$ -й элемент в  $a$ . **Заметьте, что максимальное значение  $a_i$  может быть до 26.**

Гарантируется, что сумма чисел  $n$  по всем наборам тестовых данных не превосходит 2000 ( $\sum n \leq 2000$ ).

### Выходные данные

Для каждого набора тестовых данных выведите ответ на него — максимально возможную длину некоторой подпоследовательности  $a$ , которая является **палиндромом из трех блоков**.

### Пример

входные данные	Скопировать
<pre>6 8 1 1 2 2 3 2 1 1 3 1 3 3 4 1 10 10 1 1 26 2 2 1 3 1 1 1</pre>	
выходные данные	Скопировать
<pre>7 2 4 1 1 3</pre>	


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## H. Подстрока

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

Дан граф из  $n$  вершин и  $m$  **ориентированных** ребер. В каждой вершине записана некоторая строчная латинская буква. Определим величину пути как наибольшее количество раз, которое какая-то буква встречалась на этом пути. Например, если буквы на пути образуют строку «abaca», то величина этого пути равна 3. Ваша задача — найти путь с наибольшей величиной.

### Входные данные

Первая строка содержит два целых числа  $n, m$  ( $1 \leq n, m \leq 300\,000$ ), означающих, что в графе  $n$  вершин и  $m$  ориентированных ребер.

Вторая строка содержит строку  $s$ , состоящую только из строчных латинских букв. Символ номер  $i$  — это буква, записанная в вершине номер  $i$ .

Далее следуют  $m$  строк. Каждая из этих строк содержит два целых числа  $x, y$  ( $1 \leq x, y \leq n$ ), описывающих ориентированное ребро из  $x$  в  $y$ . Обратите внимание,  $x$  может быть равно  $y$ , и могут быть несколько ребер между  $x$  и  $y$ . Кроме того, граф может быть несвязным.

### Выходные данные

Выведите одно число — максимальную величину пути. Если существуют пути со сколь угодно большой величиной, выведите -1.

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
5 4 abaca 1 2 1 3 3 4 4 5	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
3	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
6 6 xzyabc 1 2 3 1 2 3 5 4 4 3 6 4	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
-1	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
10 14 xzyzyzqzx 1 2 2 4 3 5 4 5 2 6 6 8 6 5 2 10 3 9 10 9 4 6 1 10 2 8 3 7	

[Скопировать](#)**выходные данные**

4

**Примечание**

В первом примере путь с максимальной величиной — это  $1 \rightarrow 3 \rightarrow 4 \rightarrow 5$ . Величина этих путей равна 3, т. к. буква «a» встречается 3 раза.

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов

Соревнования по программированию 2.0

Время на сервере: 05.05.2025 15:31:04<sup>UTC+5</sup> (n2).

Мобильная версия, переключиться на [десктопную](#).

[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ITMO


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## I. Выбор столицы Древляндии

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

Страна Древляндия состоит из  $n$  городов, некоторые пары из которых соединены односторонними дорогами. Всего в этой стране  $n - 1$  дорога. Известно, что если не обращать внимание на направление дорог, то из любого города можно добраться до любого другого.

Недавно советом старейшин было решено выбрать столицу Древляндии. Конечно, это должен быть один из городов страны. Предполагается, что совет будет заседать в столице и регулярно перемещаться из столицы в другие города (об обратном перемещении пока никто не думает). По этой причине, если город  $a$  будет выбран столицей, то все дороги должны быть ориентированы так, что, двигаясь по ним, можно из города  $a$  добраться до любого другого города. Для этого, возможно, некоторые дороги придется переориентировать.

Помогите старейшинам выбрать столицу так, что необходимо переориентировать минимальное количество дорог в стране.

### Входные данные

В первой строке входных данных записано целое число  $n$  ( $2 \leq n \leq 2 \cdot 10^5$ ) — количество городов в Древляндии. Следующие  $n - 1$  строк содержат описания дорог, по одной дороге в строке. Дорога описывается парой целых чисел  $s_i, t_i$  ( $1 \leq s_i, t_i \leq n; s_i \neq t_i$ ) — номерами соединяемых дорогой городов,  $i$ -ая дорога ориентирована из города  $s_i$  в город  $t_i$ . Считайте, что города в Древляндии пронумерованы от 1 до  $n$ .

### Выходные данные

В первую строку выведите минимальное количество переориентируемых дорог при оптимальном выборе столицы. Во вторую строку выведите все возможные способы выбрать столицу — последовательность номеров городов в порядке возрастания.

### Примеры

входные данные	<a href="#">Скопировать</a>
3	
2 1	
2 3	
выходные данные	<a href="#">Скопировать</a>
0	
2	
входные данные	<a href="#">Скопировать</a>
4	
1 4	
2 4	
3 4	
выходные данные	<a href="#">Скопировать</a>
2	
1 2 3	

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов

Соревнования по программированию 2.0

Время на сервере: 05.05.2025 15:31:08<sup>UTC+5</sup> (n2).

Мобильная версия, переключиться на [десктопную](#).

[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## J. На лифте или по лестнице?

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вы планируете купить квартиру в  $n$ -этажном здании. Этажи пронумерованы от 1 до  $n$  в порядке снизу вверх. Сначала вы хотите узнать для каждого этажа минимально возможное время, необходимое для того, чтобы добраться до него с первого (нижнего) этажа.

Пусть:

- $a_i$  для всех  $i$  от 1 до  $n - 1$  равно времени, необходимому для того, чтобы добраться с  $i$ -го этажа на  $(i + 1)$ -й (а также с  $(i + 1)$ -го на  $i$ -й) по лестнице;
- $b_i$  для всех  $i$  от 1 до  $n - 1$  равно времени, необходимому для того, чтобы добраться с  $i$ -го этажа на  $(i + 1)$ -й (а также с  $(i + 1)$ -го на  $i$ -й) при помощи лифта, но здесь также есть дополнительное время  $c$  — дополнительное время за использование лифта (вам необходимо ждать его, также двери лифта очень медленные!).

За один ход вы можете добраться с этажа, на котором вы сейчас стоите,  $x$  до любого этажа  $y$  ( $x \neq y$ ) двумя различными способами:

- Если вы идете по лестнице, то это просто сумма соответствующих значений  $a_i$ . Формально, это займет  $\sum_{i=\min(x,y)}^{\max(x,y)-1} a_i$  единиц времени.
- Если вы используете лифт, то это просто сумма  $c$  и соответствующих значений  $b_i$ . Формально, это займет  $c + \sum_{i=\min(x,y)}^{\max(x,y)-1} b_i$  единиц времени.

Вы можете выполнять любое количество ходов (возможно, нулевое).

Таким образом, ваша задача — определить для каждого  $i$  минимально возможное суммарное время, чтобы достичь  $i$ -й этаж с 1-го (нижнего) этажа.

### Входные данные

Первая строка входных данных содержит два целых числа  $n$  и  $c$  ( $2 \leq n \leq 2 \cdot 10^5$ ,  $1 \leq c \leq 1000$ ) — количество этажей и дополнительное время, которое тратится при использовании лифта.

Вторая строка входных данных содержит  $n - 1$  целых чисел  $a_1, a_2, \dots, a_{n-1}$  ( $1 \leq a_i \leq 1000$ ), где  $a_i$  равно времени, необходимому для того, чтобы добраться с  $i$ -го этажа на  $(i + 1)$ -й (а также с  $(i + 1)$ -го на  $i$ -й) по лестнице.

Третья строка входных данных содержит  $n - 1$  целое число  $b_1, b_2, \dots, b_{n-1}$  ( $1 \leq b_i \leq 1000$ ), где  $b_i$  равно времени, необходимому для того, чтобы добраться с  $i$ -го этажа на  $(i + 1)$ -й (а также с  $(i + 1)$ -го на  $i$ -й) при помощи лифта.

### Выходные данные

Выведите  $n$  целых чисел  $t_1, t_2, \dots, t_n$ , где  $t_i$  равно минимально возможному суммарному времени, чтобы достичь  $i$ -й этаж с первого этажа, если вы можете выполнять любое количество ходов.

### Примеры

входные данные	Скопировать
10 2 7 6 18 6 16 18 1 17 17 6 9 3 10 9 1 10 1 5	
выходные данные	Скопировать
0 7 13 18 24 35 36 37 40 45	

входные данные	Скопировать
10 1 3 2 3 1 3 3 1 4 1 1 2 3 4 4 1 2 1 3	
выходные данные	Скопировать
0 2 4 7 8 11 13 14 16 17	


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## K. Программа

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Задана программа, состоящая из  $n$  инструкций. Изначально единственная переменная  $x$  присваивается 0. После этого следуют инструкции двух типов:

- увеличить  $x$  на 1;
- уменьшить  $x$  на 1.

Даны  $m$  запросов следующего типа:

- запрос  $l \ r$  — сколько различных значений принимает переменная  $x$ , если все инструкции между  $l$ -й и  $r$ -й включительно пропускаются, а остальные исполняются без изменения порядка?

### Входные данные

В первой строке записано одно целое число  $t$  ( $1 \leq t \leq 1000$ ) — количество наборов входных данных.

Затем следует описание  $t$  наборов входных данных.

В первой строке каждого набора входных данных записаны два целых числа  $n$  и  $m$  ( $1 \leq n, m \leq 2 \cdot 10^5$ ) — количество инструкций в программе и количество запросов.

Во второй строке каждого набора входных данных записана программа — последовательность из  $n$  символов: каждый символ равен либо '+', либо '-' — инструкция увеличения и уменьшения, соответственно.

В каждой из следующих  $m$  строк записаны по два целых числа  $l$  и  $r$  ( $1 \leq l \leq r \leq n$ ) — описание запроса.

Сумма  $n$  по всем наборам входных данных не превосходит  $2 \cdot 10^5$ . Сумма  $m$  по всем наборам входных данных не превосходит  $2 \cdot 10^5$ .

### Выходные данные

На каждый набор входных данных выведите  $m$  целых чисел — для каждого запроса  $l, r$  выведите количество различных значений, которые принимает переменная  $x$ , если все инструкции между  $l$ -й и  $r$ -й включительно пропускаются, а остальные исполняются без изменения порядка.

### Пример

входные данные	Скопировать
<pre>2 8 4 -+---+- 1 8 2 8 2 5 1 1 4 10 +++ 1 1 1 2 2 2 1 3 2 3 3 3 1 4 2 4 3 4 4 4</pre>	
выходные данные	Скопировать
<pre>1 2 4 4 3 3 4 2 3</pre>	


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## L. Требуемая длина

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

Даны два целых числа,  $n$  и  $x$ . Вы можете провести несколько операций с числом  $x$ .

Каждая операция заключается в следующем: выбрать любую цифру  $y$ , которая встречается в десятичной записи  $x$  хотя бы один раз, и заменить значение  $x$  на  $x \cdot y$ .

Вы должны сделать так, чтобы длина десятичной записи  $x$  (без ведущих нулей) стала равна  $n$ . Какое минимальное количество операций вам потребуется для этого?

### Входные данные

В единственной строке заданы два целых числа  $n$  и  $x$  ( $2 \leq n \leq 19$ ;  $1 \leq x < 10^{n-1}$ ).

### Выходные данные

Выведите одно целое число — минимальное количество операций, необходимое, чтобы сделать длину десятичной записи  $x$  (без ведущих нулей) равной  $n$ , или  $-1$ , если это невозможно.

### Примеры

<b>входные данные</b>	<b>Скопировать</b>
2 1	
<b>выходные данные</b>	<b>Скопировать</b>
-1	
<b>входные данные</b>	<b>Скопировать</b>
3 2	
<b>выходные данные</b>	<b>Скопировать</b>
4	
<b>входные данные</b>	<b>Скопировать</b>
13 42	
<b>выходные данные</b>	<b>Скопировать</b>
12	

### Примечание

Во втором примере следующая последовательность операций позволяет достичь цели:

1. умножить  $x$  на 2, тогда  $x = 2 \cdot 2 = 4$ ;
2. умножить  $x$  на 4, тогда  $x = 4 \cdot 4 = 16$ ;
3. умножить  $x$  на 6, тогда  $x = 16 \cdot 6 = 96$ ;
4. умножить  $x$  на 9, тогда  $x = 96 \cdot 9 = 864$ .




[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## M. Медведь и простые числа

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

Недавно медведь начал изучать структуры данных и столкнулся со следующей задачей.

Задана последовательность целых чисел  $x_1, x_2, \dots, x_n$  длины  $n$  и  $m$  запросов, каждый из которых характеризуется двумя целыми числами  $l_i, r_i$ . Обозначим за  $f(p)$  — количество таких индексов  $k$ , что  $x_k$  делится на  $p$ . Ответом на запрос  $l_i, r_i$  является сумма:  $\sum_{p \in S(l_i, r_i)} f(p)$ , где  $S(l_i, r_i)$  — множество простых чисел из отрезка  $[l_i, r_i]$  (обе границы включаются в отрезок).

Помогите медведю справиться с этой задачей.

### Входные данные

В первой строке записано целое число  $n$  ( $1 \leq n \leq 10^6$ ). Во второй строке записаны  $n$  целых чисел  $x_1, x_2, \dots, x_n$  ( $2 \leq x_i \leq 10^7$ ). Числа не обязательно различные.

В третьей строке записано целое число  $m$  ( $1 \leq m \leq 50000$ ). В каждой из  $m$  следующих строк записана пара целых чисел через пробел  $l_i$  и  $r_i$  ( $2 \leq l_i \leq r_i \leq 2 \cdot 10^9$ ) — числа, характеризующие текущий запрос.

### Выходные данные

Выведите  $m$  целых чисел — ответы на запросы в порядке появления запросов во входных данных.

### Примеры

входные данные	<a href="#">Скопировать</a>
<pre>6 5 5 7 10 14 15 3 2 11 3 12 4 4</pre>	

выходные данные	<a href="#">Скопировать</a>
<pre>9 7 0</pre>	

входные данные	<a href="#">Скопировать</a>
<pre>7 2 3 5 7 11 4 8 2 8 10 2 123</pre>	

выходные данные	<a href="#">Скопировать</a>
<pre>0 7</pre>	

### Примечание

Рассмотрим первый тестовый пример. Всего в первом примере 3 запроса.

- Поступает первый запрос  $l = 2, r = 11$ . Нужно посчитать  $f(2) + f(3) + f(5) + f(7) + f(11) = 2 + 1 + 4 + 2 + 0 = 9$ .
- Поступает второй запрос  $l = 3, r = 12$ . Нужно посчитать  $f(3) + f(5) + f(7) + f(11) = 1 + 4 + 2 + 0 = 7$ .
- Поступает третий запрос  $l = 4, r = 4$ . Так как на этом промежутке нет простых чисел, то сумма равна 0.


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## N. Минимизация суммы

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Задан целочисленный массив  $a$  размера  $n$ .

Вы можете выполнить следующую операцию: выбрать элемент массива и заменить его значением любого из его соседей.

Например, если  $a = [3, 1, 2]$ , вы можете получить один из массивов  $[3, 3, 2]$ ,  $[3, 2, 2]$  и  $[1, 1, 2]$  за одну операцию, но не  $[2, 1, 2]$  и  $[3, 4, 2]$ .

Ваша задача — посчитать минимально возможную сумму массива, если вы можете выполнить вышеописанную операцию не более  $k$  раз.

### Входные данные

Первая строка содержит одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных.

Первая строка каждого набора содержит два целых числа  $n$  и  $k$  ( $1 \leq n \leq 3 \cdot 10^5$ ;  $0 \leq k \leq 10$ ).

Вторая строка содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ).

Дополнительное ограничение на входные данные: сумма  $n$  по всем наборам входных данных не превышает  $3 \cdot 10^5$ .

### Выходные данные

Для каждого набора входных данных выведите одно целое число — минимально возможную сумму массива, если вы можете выполнить вышеописанную операцию не более  $k$  раз.

### Пример

входные данные	<a href="#">Скопировать</a>
<pre>4 3 1 3 1 2 1 3 5 4 2 2 2 1 3 6 3 4 1 2 2 4 3</pre>	
выходные данные	<a href="#">Скопировать</a>
<pre>4 5 5 10</pre>	

### Примечание

В первом примере одна из возможных последовательностей операций:  $[3, 1, 2] \rightarrow [1, 1, 2]$ .

Во втором примере не нужно применять операцию.

В третьем примере одна из возможных последовательностей операций:  $[2, 2, 1, 3] \rightarrow [2, 1, 1, 3] \rightarrow [2, 1, 1, 1]$ .

В четвертом примере одна из возможных последовательностей операций:

$[4, 1, 2, 2, 4, 3] \rightarrow [1, 1, 2, 2, 4, 3] \rightarrow [1, 1, 1, 2, 4, 3] \rightarrow [1, 1, 1, 2, 2, 3]$ .


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## О. Геометрическая прогрессия

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Поликарп очень любит геометрические прогрессии. Так как ему только три года, он любит прогрессии только длины три. Также у него есть любимое целое число  $k$  и последовательность  $a$ , состоящая из  $n$  целых чисел.

Он хочет узнать: сколько подпоследовательностей длины три можно выбрать из  $a$  так, чтобы они образовывали геометрическую прогрессию со знаменателем  $k$ .

Подпоследовательностью длины три называются три таких индекса  $i_1, i_2, i_3$ , что  $1 \leq i_1 < i_2 < i_3 \leq n$ . То есть подпоследовательности длины три — это такие тройки элементов, которые необязательно идут подряд в последовательности, но их индексы строго возрастают.

Геометрическая прогрессия со знаменателем  $k$  — это последовательность чисел вида  $b \cdot k^0, b \cdot k^1, \dots, b \cdot k^{r-1}$ .

Поликарпу всё ещё три года, поэтому он не может посчитать это количество самостоятельно. Помогите ему сделать это.

### Входные данные

В первой строке входных данных содержится два целых числа  $n$  и  $k$  ( $1 \leq n, k \leq 2 \cdot 10^5$ ) — количество чисел в последовательности Поликарпа и его любимое число.

Вторая строка содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $-10^9 \leq a_i \leq 10^9$ ) — элементы последовательности.

### Выходные данные

Выведите единственное число — количество способов выбрать такую подпоследовательность длины три, что она образует геометрическую прогрессию со знаменателем  $k$ .

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
5 2 1 1 2 2 4	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
4	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
3 1 1 1 1	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
1	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
10 3 1 2 6 2 3 6 9 18 3 9	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
6	

### Примечание

В первом примере ответ четыре, так как в качестве первого элемента можно выбрать любую из двух единиц, в качестве второго — любую из двух двоек, а третий элемент подпоследовательности обязательно должен равняться четырём.


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## P. MEX и инкременты

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

У Дмитрия есть массив из  $n$  неотрицательных целых чисел  $a_1, a_2, \dots, a_n$ .

За одну операцию он может выбрать произвольный индекс  $j$  ( $1 \leq j \leq n$ ) и увеличить значение элемента  $a_j$  на 1. Он может выбирать один и тот же индекс  $j$  многократно.

Для каждого  $i$  от 0 до  $n$  определите, сможет ли Дмитрий сделать **MEX** массива равным ровно  $i$ . Если сможет, то определите, за какое минимальное количество операций.

**MEX** массива равен минимальному целому неотрицательному числу, которого нет в массиве. Например, **MEX** массива  $[3, 1, 0]$  равен 2, а массива  $[3, 3, 1, 4]$  — 0.

### Входные данные

Первая строка входных данных содержит единственное целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных в teste.

Далее следуют описания наборов входных данных.

Первая строка описания каждого набора входных данных содержит одно целое число  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ) — длину массива  $a$ .

Вторая строка описания каждого набора входных данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq n$ ) — элементы массива  $a$ .

Гарантируется, что сумма значений  $n$  по всем наборам входных данных в teste не превосходит  $2 \cdot 10^5$ .

### Выходные данные

Для каждого набора входных данных выведите  $n + 1$  целое число —  $i$ -е число равно минимальному числу операций, за которое можно сделать **MEX** массива равным  $i$  ( $0 \leq i \leq n$ ), или -1, если этого сделать нельзя.

### Пример

входные данные	Скопировать
<pre>5 3 0 1 3 7 0 1 2 3 4 3 2 4 3 0 0 0 7 4 6 2 3 5 0 5 5 4 0 1 0 4</pre>	
выходные данные	Скопировать
<pre>1 1 0 -1 1 1 2 2 1 0 2 6 3 0 1 4 3 1 0 -1 -1 -1 -1 -1 -1 2 1 0 2 -1 -1</pre>	

### Примечание

В первом наборе входных данных примера  $n = 3$ :

- чтобы получить **MEX** = 0, достаточно выполнить один инкремент:  $a_1++$ ;
- чтобы получить **MEX** = 1, достаточно выполнить один инкремент:  $a_2++$ ;
- MEX** = 2 для заданного массива, поэтому выполнять инкременты не надо;
- получить **MEX** = 3, выполняя инкременты, невозможно.


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## Q. Штрихкод

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Имеется картинка размера  $n \times m$  пикселов. Каждый пиксель может быть белым или черным. Требуется изменить цвета как можно меньшего количества пикселов так, чтобы получилась картинка-штрихкод.

Картинка является штрихкодом если выполняются следующие условия:

- В каждом столбце все пиксели одного цвета.
- Ширина каждой одноцветной вертикальной полосы не менее  $x$  и не более  $y$  пикселов. Другими словами, если сгруппировать все соседние столбцы пикселов одного цвета, то не должно получиться группы размера менее  $x$  или более  $y$ .

### Входные данные

В первой строке записаны четыре целых числа через пробел  $n, m, x$  и  $y$  ( $1 \leq n, m, x, y \leq 1000; x \leq y$ ).

Далее идет  $n$  строк, описывающих исходную картинку. В каждой из этих строк содержится ровно  $m$  символов. Символ «`.`» обозначает белый пиксель, а «`#`» — черный. Никаких других символов кроме «`.`» и «`#`» в описании картинки не содержится.

### Выходные данные

В первой строке выведите наименьшее количество пикселов, которое нужно перекрасить. Гарантируется, что ответ существует.

### Примеры

входные данные	Скопировать
<pre>6 5 1 2 ##.# .###. ###.. #...# .#.#. ##..</pre>	
выходные данные	Скопировать
<pre>11</pre>	
входные данные	Скопировать
<pre>2 5 1 1 ##### ....</pre>	
выходные данные	Скопировать
<pre>5</pre>	

### Примечание

В первом тестовом примере картинка после перекрашивания может выглядеть следующим образом:

```
.##..
.##..
.##..
.##..
.##..
.##..
```

Во втором тестовом примере картинка после перекрашивания может выглядеть следующим образом:

```
.#.#
.#.#.
```


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## R. Делимость

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

Вам дан массив целых чисел  $a_1, a_2, \dots, a_n$ .

Массив  $b$  называется подпоследовательностью  $a$  если можно удалить некоторые элементы  $a$ , чтобы остался массив  $b$ .

Массив  $b_1, b_2, \dots, b_k$  называется хорошим если он не пуст и для каждого  $i$  ( $1 \leq i \leq k$ )  $b_i$  делится на  $i$ .

Посчитайте количество хороших подпоследовательностей  $a$  по модулю  $10^9 + 7$ . Две подпоследовательности считаются различными, если множества индексов входящих в них чисел различны, то есть значения элементов не учитываются при сравнении подпоследовательностей. В частности, у массива  $a$  есть ровно  $2^n - 1$  различных подпоследовательностей (не считая пустую).

### Входные данные

Первая строка содержит одно целое число  $n$  ( $1 \leq n \leq 100\,000$ ) — длину массива  $a$ .

Вторая строка содержит целые числа  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^6$ ).

### Выходные данные

Выведите одно целое число — количество хороших подпоследовательностей по модулю  $10^9 + 7$ .

### Примеры

<b>входные данные</b>	<b>Скопировать</b>
2 1 2	
<b>выходные данные</b>	<b>Скопировать</b>
3	
<b>входные данные</b>	<b>Скопировать</b>
5 2 2 1 22 14	
<b>выходные данные</b>	<b>Скопировать</b>
13	

### Примечание

В первом примере все три непустые подпоследовательности являются хорошими:  $\{1\}$ ,  $\{1, 2\}$ ,  $\{2\}$

Во втором примере хорошими являются следующие подпоследовательности:  $\{2\}$ ,  $\{2, 2\}$ ,  $\{2, 22\}$ ,  $\{2, 14\}$ ,  $\{2\}$ ,  $\{2, 22\}$ ,  $\{2, 14\}$ ,  $\{1\}$ ,  $\{1, 22\}$ ,  $\{1, 14\}$ ,  $\{22\}$ ,  $\{22, 14\}$ ,  $\{14\}$ .

Обратите внимание, что некоторые подпоследовательности упомянуты несколько раз, так как они входят в исходный массив несколько раз.




[ЗАДАЧИ](#)   [ОТОСЛАТЬ](#)   [МОИ ПОСЫЛКИ](#)   [СТАТУС](#)   [ПОЛОЖЕНИЕ](#)   [ЗАПУСК](#)

## S. Омкар и Война кроватей

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Омкар играет в свою любимую видеоигру Война кроватей! В Войне кроватей  $n$  игроков располагаются по кругу, так что для всех  $j$  таких, что  $2 \leq j \leq n$ , игрок  $j - 1$  находится слева от игрока  $j$ , а игрок  $j$  — справа от игрока  $j - 1$ . Кроме того, слева от игрока 1 находится игрок  $n$ , а справа от игрока  $n$  — игрок 1.

В настоящее время каждый игрок атакует либо левого, либо правого игрока. Это означает, что каждого игрока в настоящее время атакует либо 0, 1 либо 2 других игроков. Ключевым элементом стратегии Войны кроватей является то, что если на игрока нападает ровно 1 другой игрок, то в ответ он, логически, должен атаковать этого игрока. Если же на игрока нападают либо 0 либо 2 других игроков, то стратегия Войны кроватей гласит, что игрок может атаковать любого из соседних игроков.

К сожалению, может случиться так, что некоторые игроки в этой игре не следуют правильной стратегии Войны кроватей нужным образом. Омкар знает, на кого в настоящее время нападает каждый игрок, и он может поговорить с любым количеством из  $n$  игроков в игре, чтобы заставить их вместо этого атаковать другого игрока — то есть, если игрок в настоящее время атаковал игрока слева от него, Омкар может убедить его вместо этого атаковать игрока справа от него; если он в настоящее время атаковал игрока справа от него, Омкар может убедить его вместо этого атаковать игрока слева от него.

Омкар хотел бы, чтобы все игроки действовали логично. Найдите минимальное количество игроков, с которыми Омкар должен поговорить, чтобы после того, как все игроки, с которыми он поговорил (если таковые имеются), изменили игрока, которого они атакуют, все игроки действовали логически в соответствии со стратегией Войны кроватей.

### Входные данные

Каждый тест содержит несколько наборов входных данных. В первой строке указано количество наборов входных данных  $t$  ( $1 \leq t \leq 10^4$ ). Ниже приведены описания наборов входных данных.

Первая строка каждого набора входных данных содержит одно целое число  $n$  ( $3 \leq n \leq 2 \cdot 10^5$ ) — количество игроков (а значит и кроватей) в этой игре Bed Wars.

Вторая строка каждого набора входных данных содержит строку  $s$  длиной  $n$ .  $j$ -й символ  $s$  равен L, если  $j$ -й игрок атакует игрока слева от него, и R, если  $j$ -й игрок атакует игрока справа от него.

Гарантируется, что сумма  $n$  по всем наборам входных данных не превышает  $2 \cdot 10^5$ .

### Выходные данные

Для каждого набора входных данных выведите одно целое число: минимальное количество игроков, с которыми Омкар должен поговорить, чтобы сделать так, чтобы все игроки действовали логично в соответствии со стратегией Войны кроватей.

Можно доказать, что Омкар всегда может достичь этого при заданных ограничениях.

### Пример

входные данные	<a href="#">Скопировать</a>
5 4 RLRL 6 LRRRRL 8 RLLRRRLL 12 LLLLRRLRRRL 5 RRRRR	
выходные данные	<a href="#">Скопировать</a>
0 1 1 3 2	

### Примечание

В первом наборе входных данных игроки 1 и 2 атакуют друг друга, а игроки 3 и 4 — друг друга. Каждый игрок атакует ровно 1 другого игрока, и каждый игрок атакует игрока, который атакует их, так что все игроки уже действуют логично в соответствии со

[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## T. Kavi разбивает на пары

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

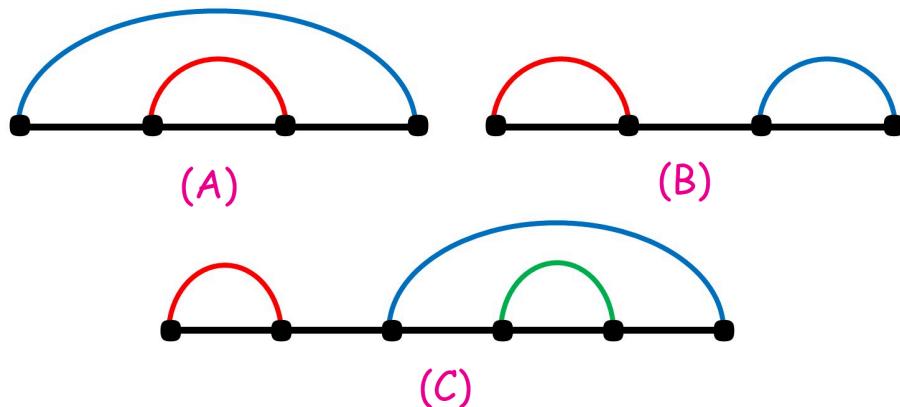
У Kavi есть  $2n$  точек, лежащих на оси  $OX$ ,  $i$ -я из которых расположена в точке  $x = i$ .

Kavi рассматривает все способы разбить эти  $2n$  точек на  $n$  пар. Среди этих способов его интересуют **хорошие** способы, которые определяются следующим образом:

Рассмотрим  $n$  отрезков с концами в точках в соответствующих парах. Пара называется хорошей, если для каждого из  $2$  различных отрезков  $A$  и  $B$  из этих, для них выполняется хотя бы одно из следующих условий:

- Один из отрезков  $A$  и  $B$  полностью содержится внутри другого.
- $A$  и  $B$  имеют одинаковую длину.

Рассмотрим следующий пример:



$A$  — хорошее разбиение на пары, так как красный отрезок полностью лежит внутри синего отрезка.

$B$  — хорошее разбиение, так как красный и синий отрезки имеют одинаковую длину.

$C$  не является хорошим разбиением, так как ни один из красного и синего отрезков не лежит внутри другого, и они имеют разную длину.

Kavi интересует количество хороших разбиений на пары, поэтому он хочет, чтобы вы нашли его для него. Поскольку результат может быть большим, найдите это число по модулю 998244353.

Два разбиения на пары называются различными, если в одном из них какие-то две точки находятся в одной паре, а в другом — в разных парах.

### Входные данные

Единственная строка ввода содержит единственное целое число  $n$  ( $1 \leq n \leq 10^6$ ).

### Выходные данные

Выведите количество хороших пар по модулю 998244353.

### Примеры

<b>входные данные</b>	<b>Скопировать</b>
1	
<b>выходные данные</b>	<b>Скопировать</b>
1	
<b>входные данные</b>	<b>Скопировать</b>
2	
<b>выходные данные</b>	<b>Скопировать</b>
3	

**входные данные****Скопировать**

3

**выходные данные****Скопировать**

6

**входные данные****Скопировать**

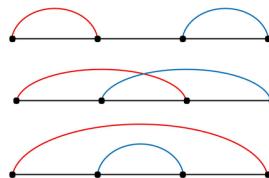
100

**выходные данные****Скопировать**

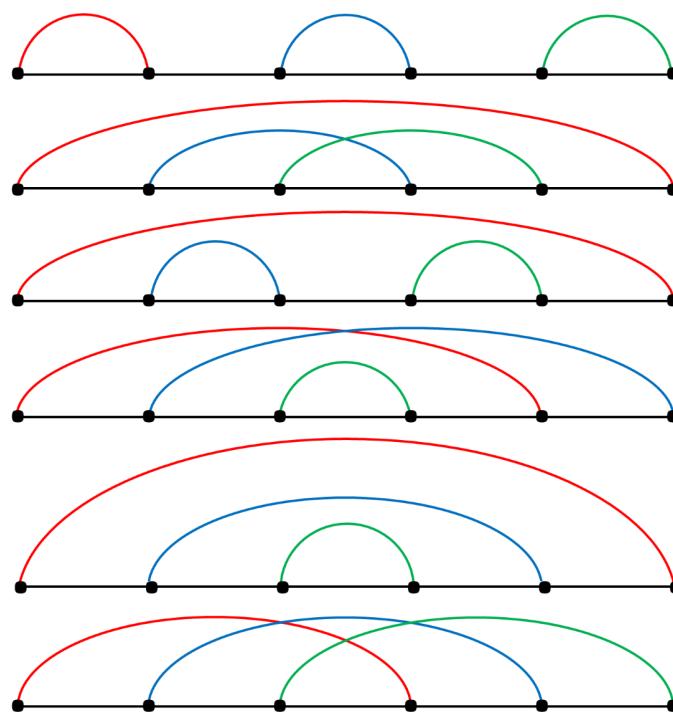
688750769

**Примечание**

Хорошими разбиениями на пары для второго примера являются:



Хорошими разбиениями на пары для третьего примера являются:

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов

Соревнования по программированию 2.0

Время на сервере: 05.05.2025 15:31:30<sup>UTC+5</sup> (n2).Мобильная версия, переключиться на [десктопную](#).[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке

**ИТМО**


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## U. Moamen и XOR

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Moamen и Ezzat играют в игру. Они создают массив  $a$  длины  $n$ , состоящий из неотрицательных целых чисел, где каждый элемент меньше  $2^k$ .

Moamen победит, если  $a_1 \& a_2 \& a_3 \& \dots \& a_n \geq a_1 \oplus a_2 \oplus a_3 \oplus \dots \oplus a_n$ .

Здесь  $\&$  обозначает операцию битового И, а  $\oplus$  обозначает операцию битового исключающего ИЛИ.

Теперь Moamen хочет узнать, сколько существует таких массивов  $a$ , где он побеждает?

Так как ответ может быть очень большим, выведите его по модулю  $1\,000\,000\,007$  ( $10^9 + 7$ ).

### Входные данные

Первая строка ввода содержит одно целое число  $t$  ( $1 \leq t \leq 5$ ) — количество наборов входных данных.

Каждый набор входных данных состоит из одной строки, содержащей два целых числа  $n$  и  $k$  ( $1 \leq n \leq 2 \cdot 10^5$ ,  $0 \leq k \leq 2 \cdot 10^5$ ).

### Выходные данные

Для каждого набора входных данных выведите одно целое число — количество различных массивов, в которых Moamen победит.

Выведите результат по модулю  $1\,000\,000\,007$  ( $10^9 + 7$ ).

### Пример

входные данные	<span style="border: 1px solid black; padding: 2px;">Скопировать</span>
3	
3 1	
2 1	
4 0	
выходные данные	<span style="border: 1px solid black; padding: 2px;">Скопировать</span>
5	
2	
1	

### Примечание

В первом примере  $n = 3$ ,  $k = 1$ . В результате все возможные массивы — это  $[0, 0, 0]$ ,  $[0, 0, 1]$ ,  $[0, 1, 0]$ ,  $[1, 0, 0]$ ,  $[1, 1, 0]$ ,  $[0, 1, 1]$ ,  $[1, 0, 1]$  и  $[1, 1, 1]$ .

Moamen победит только в 5 из них:  $[0, 0, 0]$ ,  $[1, 1, 0]$ ,  $[0, 1, 1]$ ,  $[1, 0, 1]$ ,  $[1, 1, 1]$ .

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов

Соревнования по программированию 2.0

Время на сервере: 05.05.2025 15:31:31 UTC+5 (n2).

Мобильная версия, переключиться на [десктопную](#).

[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке




[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## V. Фейковые пластиковые деревья

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Нам дано корневое дерево, состоящее из  $n$  вершин, пронумерованных от 1 до  $n$ . Корнем дерева является вершина 1, а родителем вершины  $v$  является  $p_v$ .

В каждой вершине записано число, изначально все числа равны 0. Обозначим число, записанное в вершине  $v$ , как  $a_v$ .

Для каждой  $v$  мы хотим, чтобы  $a_v$  находилось между  $l_v$  и  $r_v$  ( $l_v \leq a_v \leq r_v$ ).

За одну операцию мы делаем следующее:

- Выбираем некоторую вершину  $v$ . Пусть  $b_1, b_2, \dots, b_k$  — вершины на пути от вершины 1 до вершины  $v$  (то есть  $b_1 = 1$ ,  $b_k = v$  и  $b_i = p_{b_{i+1}}$ ).
- Выберем неубывающий массив  $c$  длины  $k$  из неотрицательных целых чисел:  $0 \leq c_1 \leq c_2 \leq \dots \leq c_k$ .
- Для каждого  $i$  ( $1 \leq i \leq k$ ), увеличим  $a_{b_i}$  на  $c_i$ .

Какое минимальное количество операций необходимо для достижения нашей цели?

### Входные данные

Первая строка содержит целое число  $t$  ( $1 \leq t \leq 1000$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит одно целое число  $n$  ( $2 \leq n \leq 2 \cdot 10^5$ ) — количество вершин в дереве.

Вторая строка каждого набора входных данных содержит  $n - 1$  целых чисел,  $p_2, p_3, \dots, p_n$  ( $1 \leq p_i < i$ ), где  $p_i$  обозначает родителя вершины  $i$ .

В  $i$ -й из следующих  $n$  строк содержится два целых числа  $l_i$  и  $r_i$  ( $1 \leq l_i \leq r_i \leq 10^9$ ).

Гарантируется, что сумма  $n$  по всем наборам входных данных не превышает  $2 \cdot 10^5$ .

### Выходные данные

Для каждого набора входных данных выведите минимально необходимое количество операций.

### Пример

входные данные	<span style="border: 1px solid black; padding: 2px;">Скопировать</span>
<pre>4 2 1 1 5 2 9 3 1 1 4 5 2 4 6 10 4 1 2 1 6 9 5 6 4 5 2 4 5 1 2 3 4 5 5 4 4 3 3 2 2 1 1</pre>	
выходные данные	<span style="border: 1px solid black; padding: 2px;">Скопировать</span>
<pre>1 2 2 5</pre>	

**Примечание**

В первом наборе входных данных мы можем достичь цели с помощью одной операции: выбрать  $v = 2$  и  $c = [1, 2]$ , в результате чего  $a_1 = 1, a_2 = 2$ .

Во втором наборе входных данных мы можем достичь цели с помощью двух операций: сначала выбираем  $v = 2$  и  $c = [3, 3]$ , в результате чего  $a_1 = 3, a_2 = 3, a_3 = 0$ . Затем выбираем  $v = 3, c = [2, 7]$ , в результате  $a_1 = 5, a_2 = 3, a_3 = 7$ .

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов

Соревнования по программированию 2.0

Время на сервере: 05.05.2025 15:31:33<sup>UTC+5</sup> (n2).

Мобильная версия, переключиться на [десктопную](#).

[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## W. Путешествие

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

В некоторой стране  $n + 1$  городов, пронумерованных от 0 до  $n$ .  $n$  дорог соединяют эти города,  $i$ -я дорога соединяет города  $i - 1$  и  $i$  ( $i \in [1, n]$ ).

У каждой дороги есть направление. Направления задаются строкой из  $n$  символов, каждый символ — либо L, либо R. Если  $i$ -й символ — L, то  $i$ -я дорога сначала направлена из города  $i$  в город  $i - 1$ ; иначе она направлена из города  $i - 1$  в город  $i$ .

Путешественник хочет посетить как можно больше городов этой страны. Сначала он выбирает город, из которого начнет свое путешествие. Каждый день путешественник **должен** перейти из текущего города в соседний город по дороге, и он может перейти по дороге только в соответствии с ее направлением; то есть, если дорога направлена из города  $i$  в город  $i + 1$ , он может перейти из города  $i$  в город  $i + 1$ , но не из  $i + 1$  в  $i$ . После того как путешественник перемещается в соседний город, **все** дороги меняют свое направление **на противоположное**. Если путешественник не может перейти в соседний город, он обязан закончить свое путешествие; также он может закончить путешествие в любой момент, в который захочет.

Цель путешественника — посетить как можно больше городов за одно путешествие (каждый город можно посетить любое количество раз, но только первое посещение считается). Для каждого города  $i$  посчитайте максимальное количество городов, которое путешественник может посетить **за одно путешествие**, если оно начинается в городе  $i$ .

### Входные данные

В первой строке задано одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных.

Каждый набор входных данных состоит из двух строк. В первой строке задано одно целое число  $n$  ( $1 \leq n \leq 3 \cdot 10^5$ ). Во второй строке — последовательность  $s$ , состоящая из ровно  $n$  символов, каждый из которых — L или R.

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит  $3 \cdot 10^5$ .

### Выходные данные

Для каждого набора входных данных выведите  $n + 1$  целое число.  $i$ -е число должно быть равно максимальному количеству городов, которое можно посетить за одно путешествие, если это путешествие начинается в  $i$ -м городе.

### Пример

входные данные	<a href="#">Скопировать</a>
2	
6	
LRRRLL	
3	
LRL	
выходные данные	<a href="#">Скопировать</a>
1 3 2 3 1 3 2	
1 4 1 4	

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов

Соревнования по программированию 2.0

Время на сервере: 05.05.2025 15:31:35 UTC+5 (n2).

Мобильная версия, переключиться на [десктопную](#).

[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## X. Не вините меня

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

К сожалению, автор задачи не смог придумать интересную историю, поэтому он просто просит вас решить следующую задачу.

Дан массив  $a$ , состоящий из  $n$  положительных целых чисел. Посчитайте количество подпоследовательностей, для которых побитовое AND элементов в подпоследовательности имеет ровно  $k$  единичных битов в двоичном представлении. Ответ может быть большим, поэтому выведите его по модулю  $10^9 + 7$ .

Напомним, что подпоследовательность массива  $a$  - это последовательность, которую можно получить из  $a$ , удалив некоторые (возможно, ни одного) элементы. Например,  $[1, 2, 3]$ ,  $[3]$ ,  $[1, 3]$  являются подпоследовательностями  $[1, 2, 3]$ , но  $[3, 2]$  и  $[4, 5, 6]$  - нет.

Обратите внимание, что **AND** обозначает [логическую операцию AND](#).

### Входные данные

Каждый тест содержит несколько наборов входных данных. Первая строка содержит количество наборов  $t$  ( $1 \leq t \leq 10^4$ ).

Затем следует описание наборов входных данных.

Первая строка каждого тестового случая состоит из двух целых чисел  $n$  и  $k$  ( $1 \leq n \leq 2 \cdot 10^5$ ,  $0 \leq k \leq 6$ ) — длина массива и количество единичных битов, которое должно быть у побитового AND подсчитанных подпоследовательностей в двоичном представлении.

Вторая строка каждого тестового случая состоит из  $n$  целых чисел  $a_i$  ( $0 \leq a_i \leq 63$ ) — массив  $a$ .

Гарантируется, что сумма  $n$  по всем тестовым случаям не превышает  $2 \cdot 10^5$ .

### Выходные данные

Для каждого тестового случая выведите одно целое число - количество подпоследовательностей, у которых в двоичном представлении значение побитового AND имеет ровно  $k$  установленных битов. Ответ может быть большим, поэтому выведите его по модулю  $10^9 + 7$ .

### Пример

входные данные	<a href="#">Скопировать</a>
<pre>6 5 1 1 1 1 1 4 0 0 1 2 3 5 1 5 5 7 4 2 1 2 3 12 0 0 2 0 2 0 2 0 2 0 2 10 6 63 0 63 5 5 63 63 4 12 13</pre>	
выходные данные	<a href="#">Скопировать</a>
<pre>31 10 10 1 4032 15</pre>	


[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## Y. Бесквадратное разбиение (простая версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Это простая версия задачи. Единственное отличие — в этой версии задачи  $k = 0$ .**

Дан массив  $a_1, a_2, \dots, a_n$  из  $n$  положительных целых чисел. Необходимо разбить его на наименьшее количество непрерывных отрезков так, чтобы ни на каком отрезке не было двух чисел (на разных позициях), произведение которых является полным квадратом.

При этом до разбиения разрешается сделать не более  $k$  раз следующую операцию: выбрать любое число в массиве и заменить его значение на произвольное целое положительное число. Но в этой версии задачи  $k = 0$ , поэтому это не важно.

Какое наименьшее количество непрерывных отрезков нужно использовать, если сделать изменения оптимально?

### Входные данные

В первой строке находится единственное целое число  $t$  ( $1 \leq t \leq 1000$ ) — количество наборов входных данных.

Первая строка описания каждого набора входных данных содержит два целых числа  $n, k$  ( $1 \leq n \leq 2 \cdot 10^5, k = 0$ ).

Вторая строка описания каждого набора входных данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^7$ ).

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит  $2 \cdot 10^5$ .

### Выходные данные

Для каждого набора входных данных выведите одно число — ответ на задачу.

### Пример

входные данные	<a href="#">Скопировать</a>
3 5 0 18 6 2 4 1 5 0 6 8 1 24 8 1 0 1	
выходные данные	<a href="#">Скопировать</a>
3 2 1	

### Примечание

В первом наборе входных можно сделать такое разбиение:

- [18, 6]
- [2, 4]
- [1]




[ЗАДАЧИ](#) [ОТОСЛАТЬ](#) [МОИ ПОСЫЛКИ](#) [СТАТУС](#) [ПОЛОЖЕНИЕ](#) [ЗАПУСК](#)

## Z. Короткая задача

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

Обозначим за  $d(n)$  сумму всех делителей числа  $n$ , т.е.  $d(n) = \sum_{k|n} k$ .

Например,  $d(1) = 1$ ,  $d(4) = 1 + 2 + 4 = 7$ ,  $d(6) = 1 + 2 + 3 + 6 = 12$ .

Для заданного числа  $c$ , найдите минимальное  $n$  такое, что  $d(n) = c$ .

### Входные данные

В первой строке содержится одно целое число  $t$  ( $1 \leq t \leq 10^4$ ). Далее следуют  $t$  наборов входных данных.

Каждый набор входных данных характеризуется одним целым числом  $c$  ( $1 \leq c \leq 10^7$ ).

### Выходные данные

Для каждого набора входных данных выведите:

- «-1», если не существует  $n$ , такого что  $d(n) = c$ ;
- $n$ , иначе.

### Пример

входные данные	<a href="#">Скопировать</a>
<pre>12 1 2 3 4 5 6 7 8 9 10 39 691</pre>	
выходные данные	<a href="#">Скопировать</a>
<pre>1 -1 2 3 -1 5 4 7 -1 -1 18 -1</pre>	

