



## С. Командир Ciel

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Лиса Ciel становится командиром Древоземелья. В Древоземелье, как это следует из названия, есть  $n$  городов, соединенных  $n - 1$  ненаправленными дорогами, а между любыми двумя городами существует путь по дорогам Древоземелья.

Лиса Ciel должна назначить каждому городу офицера. У каждого офицера есть ранг — буква от 'A' до 'Z'. Таким образом, имеется 26 различных рангов, самый высокий — 'A', самый низкий — 'Z'.

У Ciel имеется достаточно офицеров каждого ранга. Но не все так просто, должно быть выполнено особое условие: если  $x, y$  — два различных города и у их офицеров одинаковые ранги, то на простом пути между  $x$  и  $y$  должен быть город  $z$ , имеющий офицера более высокого ранга. Таким образом, общение между офицерами одного ранга будет гарантированно проходить под присмотром офицера более высокого ранга.

Помогите Ciel составить подходящий план назначения офицеров городам. Если это невозможно, выведите «Impossible!».

### Входные данные

В первой строке записано целое число  $n$  ( $2 \leq n \leq 10^5$ ) — количество городов в Древоземелье.

В каждой из следующих  $n - 1$  строк записано два целых числа  $a$  и  $b$  ( $1 \leq a, b \leq n, a \neq b$ ) — это значит, что существует дорога между городами  $a$  и  $b$ . Считайте, что города пронумерованы от 1 до  $n$  некоторым образом.

Гарантируется, что заданный граф будет деревом.

### Выходные данные

Если подходящий план существует, выведите  $n$  символов, разделенных пробелами —  $i$ -ый символ обозначает ранг офицера в городе  $i$ .

В противном случае, выведите «Impossible!».

### Примеры

входные данные	<a href="#">Скопировать</a>
4	
1 2	
1 3	
1 4	
выходные данные	<a href="#">Скопировать</a>
A B B B	
входные данные	<a href="#">Скопировать</a>
10	
1 2	
2 3	
3 4	
4 5	
5 6	
6 7	
7 8	
8 9	
9 10	
выходные данные	<a href="#">Скопировать</a>
D C B A D C B D C D	

### Примечание

В первом примере для любых двух офицеров ранга 'B', офицер с рангом 'A' будет на пути между ними. То есть, такое решение подходит.

## E. Сжатие массива

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам задан массив  $a_1, a_2, \dots, a_n$ . Вы можете выполнять на нем следующую операцию произвольное количество раз:

- Выбираем два равных соседних элемента  $a_i = a_{i+1}$  (если такие существуют).
- Заменяем их на один элемент со значением  $a_i + 1$ .

После каждой такой операции длина массива уменьшается на один (и элементы перенумеровываются соответствующим образом). Массив  $a$  какой минимальной длины вы можете получить?

### Входные данные

В первой строке задано одно целое число  $n$  ( $1 \leq n \leq 500$ ) — первоначальная длина массива  $a$ .

Во второй строке заданы  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 1000$ ) — первоначальный массив  $a$ .

### Выходные данные

Выведите единственное число — минимально возможную длину массива, которую можно получить после применения операции, описанной выше, произвольное количество раз.

### Примеры

<b>входные данные</b>	Скопировать
5 4 3 2 2 3	
<b>выходные данные</b>	Скопировать
2	
<b>входные данные</b>	Скопировать
7 3 3 4 4 4 3 3	
<b>выходные данные</b>	Скопировать
2	
<b>входные данные</b>	Скопировать
3 1 3 5	
<b>выходные данные</b>	Скопировать
3	
<b>входные данные</b>	Скопировать
1 1000	
<b>выходные данные</b>	Скопировать
1	

### Примечание

В первом примере, одна из оптимальных последовательностей операций такая:  $4 3 2 2 3 \rightarrow 4 3 3 3 \rightarrow 4 4 3 \rightarrow 5 3$ .

Во втором примере, одна из оптимальных последовательностей операций такая:  $3 3 4 4 4 3 3 \rightarrow 4 4 4 4 3 3 \rightarrow 4 4 4 4 4 \rightarrow 5 4 4 4 \rightarrow 5 5 4 \rightarrow 6 4$ .

В третьем и четвертом примере, не возможно применить операцию ни разу.



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Омкар и круг

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Дэнни, местный математик-маньяк, очарован кругами, самым последним творением Омкара. Помогите ему решить эту задачу о круге!

Вам даны  $n$  неотрицательных целых чисел  $a_1, a_2, \dots, a_n$ , расположенных по кругу, где  $n$  гарантированно будет нечетным (т.е.  $n - 1$  делится на 2). Формально для всех  $i$  таких, что  $2 \leq i \leq n$ , элементы  $a_{i-1}$  и  $a_i$  считаются соседними, а  $a_n$  и  $a_1$  также рассматриваются как соседние. За одну операцию вы выбираете число в круге, заменяете его суммой двух соседних элементов, а затем удаляете два соседних элемента из круга. Это повторяется до тех пор, пока в круге не останется только одно число, которое мы называем круговым значением.

Помогите Дэнни найти максимально возможное круговое значение после некоторой последовательности операций.

### Входные данные

Первая строка содержит одно нечетное целое число  $n$  ( $1 \leq n < 2 \cdot 10^5$ ,  $n$  нечетно) — начальный размер круга.

Вторая строка содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^9$ ) — начальные числа в круге.

### Выходные данные

Выведите максимально возможное круговое значение после применения некоторой последовательности операций к данному кругу.

### Примеры

входные данные	Скопировать
3 7 10 2	
выходные данные	Скопировать
17	
входные данные	Скопировать
1 4	
выходные данные	Скопировать
4	

### Примечание

Для первого примера круговое значение 17 получается так:

Выберите число с индексом 3. Сумма соседних элементов равна 17. Удалите 7 и 10 из круга и замените 2 на 17.

Обратите внимание, что ответ может выйти за пределы 32-разрядного целого числа.

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:21 UTC+5 (h1).

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

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

При поддержке



ITMO



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## E. Реставрационное расстояние

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Вам нужно отреставрировать стену. Стена состоит из  $N$  столбиков кирпичей, высота  $i$ -го столбика изначально равна  $h_i$ , высота измеряется количеством кирпичей. После реставрации все  $N$  столбиков должны иметь одинаковую высоту.

Вам доступны следующие операции:

- положить кирпич на верх одного из столбиков, стоимость этой операции равна  $A$ ;
- убрать кирпич с верха одного из непустых столбиков, стоимость этой операции равна  $R$ ;
- переложить один кирпич с верха одного из непустых столбиков на верх другого столбика, стоимость этой операции равна  $M$ .

Вы не можете создавать дополнительные столбики или игнорировать какие-то из существующих столбиков, даже если их высота стала равна 0.

За какую минимальную суммарную стоимость возможно провести реставрацию, то есть сделать высоты всех столбиков равными?

### Входные данные

В первой строке записаны четыре целых числа  $N, A, R, M$  ( $1 \leq N \leq 10^5, 0 \leq A, R, M \leq 10^4$ ) — количество столбиков кирпичей в стене, а также стоимости доступных операций.

Во второй строке записаны  $N$  целых чисел  $h_i$  ( $0 \leq h_i \leq 10^9$ ) — изначальные высоты столбиков.

### Выходные данные

Выведите одно целое число — минимальную суммарную стоимость реставрации стены.

### Примеры

входные данные	Скопировать
3 1 100 100 1 3 8	
выходные данные	Скопировать
12	

входные данные	Скопировать
3 100 1 100 1 3 8	
выходные данные	Скопировать
9	

входные данные	Скопировать
3 100 100 1 1 3 8	
выходные данные	Скопировать
4	

входные данные	Скопировать
5 1 2 4 5 5 3 6 5	
выходные данные	Скопировать
4	

входные данные	Скопировать
5 1 2 2 5 5 3 6 5	
выходные данные	Скопировать



|    
[Kap6502](#) | [Выходи](#)

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

## E. Распределение весов

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам задан неориентированный невзвешенный граф, состоящий из  $n$  вершин и  $m$  ребер (который представляет карту Бертауна), а также массив цен  $p$  длины  $m$ . Гарантируется, что между каждой парой вершин (районов) существует путь.

Майк планирует совершить путешествие из вершины (района)  $a$  в вершину (район)  $b$ , а затем из вершины (района)  $b$  в вершину (район)  $c$ . Он может посещать один и тот же район дважды или даже большее число раз. Но есть небольшая проблема: власти города хотят ввести цену за использование дорог, таким образом, что если кто-то проходит по ней, то он должен заплатить цену, соответствующую этой дороге (**он платит каждый раз, когда проходит по дороге**). Список используемых цен  $p$  готов и они просто хотят распределить его между всеми дорогами в городе таким образом, что каждая цена из массива соответствует ровно одной дороге.

Вы являетесь хорошим другом Майка (и, внезапно, мэром Бертауна) и хотите помочь сделать его путешествие настолько дешевым, насколько это возможно. Таким образом, ваша задача заключается в том, чтобы распределить цены между дорогами таким образом, что если Майк выберет оптимальный путь, то цена его путешествия будет **минимально возможной**. **Заметьте, что вы не можете перераспределять цены после начала путешествия.**

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

### Входные данные

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

Первая строка набора тестовых данных содержит пять целых чисел  $n, m, a, b$  и  $c$  ( $2 \leq n \leq 2 \cdot 10^5$ ,  $n - 1 \leq m \leq \min(\frac{n(n-1)}{2}, 2 \cdot 10^5)$ ,  $1 \leq a, b, c \leq n$ ) — количество вершин, количество ребер и районы в путешествии Майка.

Вторая строка набора тестовых данных содержит  $m$  целых чисел  $p_1, p_2, \dots, p_m$  ( $1 \leq p_i \leq 10^9$ ), где  $p_i$  равно  $i$ -й цене из массива.

Следующие  $m$  строк набора тестовых данных описывают ребра: ребро  $i$  предствалено парой целых чисел  $v_i, u_i$  ( $1 \leq v_i, u_i \leq n$ ,  $u_i \neq v_i$ ), которые означают индексы вершин, соединенных ребром. Гарантируется, что в графе не существует петель и кратных ребер, то есть для каждой пары  $(v_i, u_i)$  в списке ребер не существует других пар  $(v_i, u_i)$  или  $(u_i, v_i)$ , а также для каждой пары  $(v_i, u_i)$  выполняется условие  $v_i \neq u_i$ . Гарантируется, что заданный граф является связным.

Гарантируется, что сумма  $n$  (также как и сумма  $m$ ) не превосходит  $2 \cdot 10^5$  ( $\sum n \leq 2 \cdot 10^5$ ,  $\sum m \leq 2 \cdot 10^5$ ).

### Выходные данные

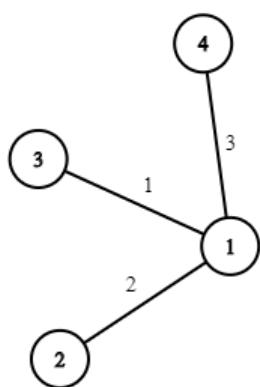
Выведите ответ на каждый набор тестовых данных — **минимально возможную** цену путешествия Майка, если вы распределите цены между ребрами оптимально.

### Пример

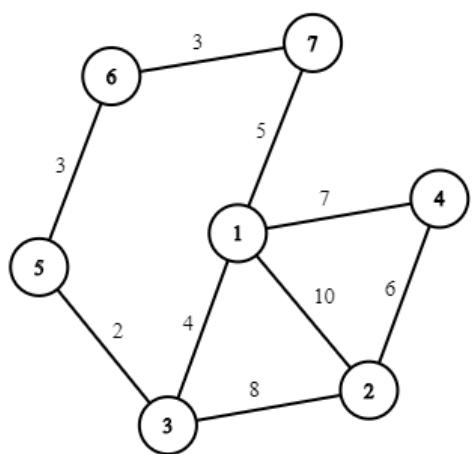
входные данные	Скопировать
<pre>2 4 3 2 3 4 1 2 3 1 2 1 3 1 4 7 9 1 5 7 2 10 4 8 5 6 7 3 3 1 2 1 3 1 4 3 2 3 5 4 2 5 6 1 7 6 7</pre>	
выходные данные	Скопировать
<pre>7 12</pre>	

### Примечание

Одно из возможных решение первого набора тестовых данных из примера:



Одно из возможных решение второго набора тестовых данных из примера:



---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 07.08.2025 13:01:25 UTC+5 (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке





|   
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## E. Сдвиги подпоследовательностей

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

У Naman есть две бинарные строки  $s$  и  $t$  длины  $n$  (бинарной строкой называется строка, состоящая из символов «0» и «1»). Он хочет сделать из строки  $s$  строку  $t$  используя следующую операцию как можно меньшее количество раз.

За одну операцию, он может выбрать некоторую подпоследовательность  $s$  и сдвинуть символы на позициях этой подпоследовательности по часовой стрелке один раз.

Например, если  $s = 1110100$ , он может выбрать подпоследовательность, соответствующую позициям (нумерация с 1)  $\{2, 6, 7\}$  и сдвинуть символы на этих позициях один раз по часовой стрелке. В результате строка, которая получится, будет  $s = 1010110$ .

Строка  $a$  называется подпоследовательностью строки  $b$ , если  $a$  может быть получена из  $b$  с помощью удаления некоторых символов без изменения порядка оставшихся символов.

Для того, чтобы сделать сдвиг последовательности  $s$  размера  $k$  по часовой стрелке один раз, нужно сделать следующие замены символов  $c_1 := c_k, c_2 := c_1, c_3 := c_2, \dots, c_k := c_{k-1}$  одновременно.

Определите минимальное число операций, которое необходимо сделать Naman, чтобы получить из строки  $s$  строку  $t$  или определите, что это сделать невозможно.

### Входные данные

В первой строке находится единственное целое число  $n$  ( $1 \leq n \leq 10^6$ ) — длина строк.

Во второй строке находится бинарная строка  $s$  длины  $n$ .

В третьей строке находится бинарная строка  $t$  длины  $n$ .

### Выходные данные

Если невозможно получить из строки  $s$  строку  $t$  после применения любого числа операций, выведите  $-1$ .

Иначе выведите минимальное количество операций, которое для этого требуется.

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
6 010000 000001	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
1	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
10 1111100000 0000011111	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
5	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
8 10101010 01010101	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
1	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
10 1111100000 1111100001	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>

-1

**Примечание**

В первом teste, Naman может выбрать подпоследовательность, соответствующую позициям  $\{2, 6\}$  и сдвинуть символы строки, на этих позициях один раз по часовой стрелке. В результате, он сможет получить из строки  $s$  строку  $t$  за одну операцию.

Во втором teste, он может сделать операцию на подпоследовательности из всех индексов 5 раз подряд. Можно доказать, что это минимальное необходимое количество операций.

В последнем teste, невозможно получить из строки  $s$  строку  $t$ .

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:26<sub>UTC+5</sub> (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ITMO



|   
[Kap6502](#) | [Выходи](#)

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

## C2. Армия покемонов (сложная версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Это сложная версия задачи. Различия между версиями заключаются в том, что в простой версии нет запросов обмена. Вы можете делать взломы только если обе версии задачи сданы.**

Пикачу — милый и дружелюбный покемон, живущий в стае диких пикачу.

Однако недавно стало известно, что команда R хочет украдь всех этих покемонов! Тренер покемонов Андрей решил помочь Пикачу собрать армию для борьбы с командой R.

В первую очередь Андрей посчитал всех покемонов: их оказалось ровно  $n$  штук. Затем он установил силу каждого покемона и так получилось, что  $i$ -й покемон имеет силу, равную  $a_i$ , и силы всех покемонов различны.

В качестве армии Андрей может выбрать любую непустую подпоследовательность покемонов. Иными словами, Андрей выбирает какой-то массив  $b$  из  $k$  индексов таких, что  $1 \leq b_1 < b_2 < \dots < b_k \leq n$ , и его армия будет состоять из покемонов с силами  $a_{b_1}, a_{b_2}, \dots, a_{b_k}$ .

Сила армии вычисляется как знакопеременная сумма элементов подпоследовательности, то есть  $a_{b_1} - a_{b_2} + a_{b_3} - a_{b_4} + \dots$

Андрей экспериментирует с построением покемонов. Он  $q$  раз меняет двух покемонов местами, а именно, в  $i$ -й раз он менял местами покемонов с номерами  $l_i$  и  $r_i$ .

Андрею надо знать: какую максимальную силу армии он мог получить при начальной расстановке покемонов, а также после каждого изменения строя?

Помогите Андрею и покемонам, иначе команда R удастся воплотить в жизнь свой коварный план!

### Входные данные

Каждый тест содержит несколько наборов входных данных.

В первой строке находится одно целое положительное число  $t$  ( $1 \leq t \leq 10^3$ ) — количество наборов входных данных.

Описание наборов входных данных приведено ниже.

В первой строке каждого набора входных данных находятся два целых числа  $n$  и  $q$  ( $1 \leq n \leq 3 \cdot 10^5$ ,  $0 \leq q \leq 3 \cdot 10^5$ ) — количество покемонов и количество обменов соответственно.

Во второй строке находятся  $n$  различных целых положительных чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq n$ ) — силы покемонов.

$i$ -я из следующих  $q$  строк содержит два целых положительных числа  $l_i$  и  $r_i$  ( $1 \leq l_i \leq r_i \leq n$ ) — номера обмениваемых покемонов в  $i$ -й операции.

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

### Выходные данные

Для каждого набора входных данных выведите  $q + 1$  число — максимально возможную силу армии до изменений и после каждого изменения.

### Пример

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

**выходные данные****Скопировать**

```
3  
4  
2  
2  
2  
9  
10  
10  
10  
9  
11
```

**Примечание**

Рассмотрим третий набор входных данных:

Изначально мы можем выбрать армию следующим образом: [1 2 5 4 3 6 7], при этом ее итоговая сила будет  $5 - 3 + 7 = 9$ .

После первого изменения выбор армии будет таким: [2 1 5 4 3 6 7], при этом ее итоговая сила будет  $2 - 1 + 5 - 3 + 7 = 10$ .

Далее выбор армии будет таким: [2 1 5 4 3 7 6], при этом ее итоговая сила будет  $2 - 1 + 5 - 3 + 7 = 10$ .

После третьего изменения армию можно выбрать так: [2 1 4 5 3 7 6], при этом ее итоговая сила будет  $2 - 1 + 5 - 3 + 7 = 10$ .

Далее выбор армии будет таким: [1 2 4 5 3 7 6], при этом ее итоговая сила будет  $5 - 3 + 7 = 9$ .

После всех запросов выбор армии будет таким: [1 4 2 5 3 7 6], при этом ее итоговая сила будет  $4 - 2 + 5 - 3 + 7 = 11$ .

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 07.08.2025 13:01:27<sub>UTC+5</sub> (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## G. Подарочный набор

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

У Поликарпа есть  $x$  красных и  $y$  синих конфет. Из них он хочет составить подарочные наборы. Каждый подарочный набор будет содержать либо  $a$  красных конфет и  $b$  синих конфет, либо наоборот —  $a$  синих конфет и  $b$  красных конфет.

Помогите Поликарпу понять, какое наибольшее число подарочных наборов он может составить.

Например, если  $x = 10$ ,  $y = 12$ ,  $a = 5$  и  $b = 2$ , то он может составить три подарочных набора:

- В первом наборе будет 5 красных конфет и 2 синие;
- Во втором наборе будет 5 синих конфет и 2 красные;
- В третьем наборе будет 5 синих конфет и 2 красные.

У Поликарпа останется одна красная конфета, используя которую, он не сможет собрать еще один подарочный набор.

### Входные данные

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

Каждый набор входных данных состоит из одной строки в которой находятся четыре целых числа  $x$ ,  $y$ ,  $a$  и  $b$  ( $1 \leq x, y, a, b \leq 10^9$ ).

### Выходные данные

Для каждого набора входных данных выведите одно число — максимальное количество подарочных наборов, которое может собрать Поликарп.

### Пример

Входные данные	Скопировать
9 10 12 2 5 1 1 2 2 52 311 13 27 10000000000 10000000000 1 1 1000000000 1 1 1000000000 1 1000000000 1000000000 1 1 2 1 1 7 8 1 2 4 1 2 3	
Выходные данные	Скопировать
3 0 4 1000000000 1 1 1 5 0	





| [English](#) | [Russian](#)  
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## F2. Ближайшее красивое число (усложнённая версия)

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Это усложнённая версия задачи F1. Они отличаются только ограничениями (F1:  $k \leq 2$ , F2:  $k \leq 10$ ).

Дано число  $n$ . Найдите **минимальное** целое число  $x$  такое, что  $x \geq n$  и  $x$  является  $k$ -красивым числом.

Число называется  $k$ -красивым, если в его десятичной записи, не содержащей лидирующих нулей, встречается **не более  $k$**  различных цифр. Например, если  $k = 2$ , числа 3434443, 55550, 777 и 21 являются  $k$ -красивыми, а числа 120, 445435 и 998244353 — нет.

### Входные данные

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

Каждый набор входных данных состоит из одной строки, содержащей два целых числа  $n$  и  $k$  ( $1 \leq n \leq 10^9$ ,  $1 \leq k \leq 10$ ).

### Выходные данные

Для каждого набора входных данных в отдельной строке выведите  $x$  — минимальное целое  $k$ -красивое число, так чтобы  $x \geq n$ .

### Пример

входные данные	Скопировать
6 2021 3 177890 2 34512 3 724533 4 998244353 1 12345678 10	
выходные данные	Скопировать
2021 181111 34533 724542 999999999 12345678	

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:34 UTC+5 (h1).

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

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

При поддержке



ITMO



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Последнее следствие Ехаба

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Для связного неориентированного графа с  $n$  вершинами и целого числа  $k$ , вы должны, на ваш выбор:

- или найти независимое множество с **ровно**  $\lceil \frac{k}{2} \rceil$  вершинами.
- или найти **простой** цикл длины **не более**  $k$ .

Независимое множество — это набор вершин такой, что никакие две из них не связаны ребром. Простой цикл — это цикл, который не содержит ни одной вершины дважды.

У меня есть доказательство, что для любых входных данных вы всегда можете решить по крайней мере одну из этих задач, но оно слишком тривиально, чтобы поместиться здесь.

### Входные данные

Первая строка содержит три целых числа  $n$ ,  $m$ , and  $k$  ( $3 \leq k \leq n \leq 10^5$ ,  $n - 1 \leq m \leq 2 \cdot 10^5$ ) — количество вершин и ребер в графе и параметр  $k$  из условия.

Каждая из следующих  $m$  строк содержит два целых числа  $u$  и  $v$  ( $1 \leq u, v \leq n$ ), которые обозначают, что между вершинами  $u$  и  $v$  есть ребро. Гарантируется, что граф связен и не содержит петель или кратных ребер.

### Выходные данные

Если вы решили решить первую задачу, то в первой строке выведите 1, а затем строку, содержащую  $\lceil \frac{k}{2} \rceil$  различных целых чисел, не превышающих  $n$  — вершины в желаемом независимом наборе.

Если же вы решили решить вторую задачу, то в первой строке выведите 2, затем строку, содержащую одно целое число,  $c$ , представляющее длину найденного цикла, а затем строку, содержащую  $c$  различных целых чисел, не превышающих  $n$  — вершины в нужном цикле, **в порядке их появления в цикле**.

### Примеры

входные данные	Скопировать
4 4 3 1 2 2 3 3 4 4 1	
выходные данные	Скопировать
1 1 3	
входные данные	Скопировать
4 5 3 1 2 2 3 3 4 4 1 2 4	
выходные данные	Скопировать
2 3 2 3 4	
входные данные	Скопировать
4 6 3 1 2 2 3 3 4 4 1 1 3 2 4	
выходные данные	Скопировать
2 3	

1 2 3

**Входные данные**

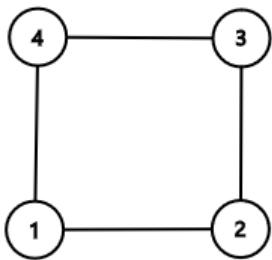
5 4 5  
1 2  
1 3  
2 4  
2 5

**Выходные данные**

1  
1 4 5

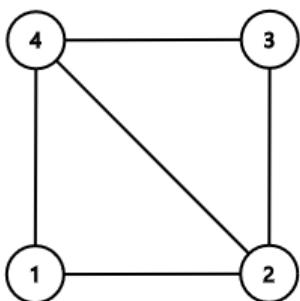
**Примечание**

В первом примере:



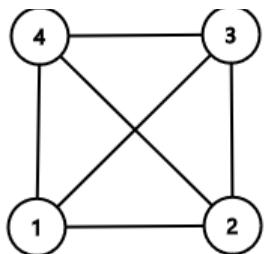
Обратите внимание, что вывод независимого множества  $\{2, 4\}$  тоже зачтется, но вывод цикла  $1 - 2 - 3 - 4$  — нет, потому что его длина должна быть не более 3.

Во втором примере:

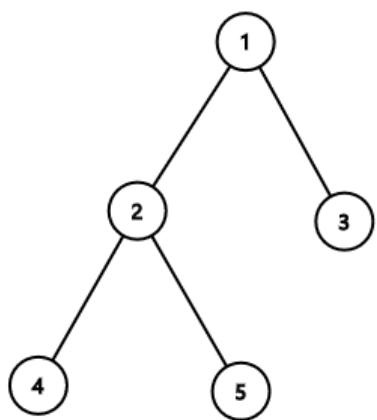


Обратите внимание, что вывод независимого множества  $\{1, 3\}$  или цикла  $2 - 1 - 4$  также зачтутся.

В третьем примере:



В четвертом примере:



---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:37 UTC+5 (h1).

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

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

При поддержке



ITMO

## D. Пересечения отрезков

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам заданы два списка отрезков  $[al_1, ar_1], [al_2, ar_2], \dots, [al_n, ar_n]$  и  $[bl_1, br_1], [bl_2, br_2], \dots, [bl_n, br_n]$ .

Первоначально, все отрезки  $[al_i, ar_i]$  равны  $[l_1, r_1]$  и все отрезки  $[bl_i, br_i]$  равны  $[l_2, r_2]$ .

За один шаг вы можете выбрать один отрезок (либо из первого списка, либо из второго) и удлинить его на 1. Другими словами, предположим, вы выбрали отрезок  $[x, y]$ , тогда вы можете его превратить либо в  $[x - 1, y]$ , либо в  $[x, y + 1]$ .

Объявим суммарное пересечение  $I$  как сумму длин пересечений соответствующих пар отрезков, то есть

$\sum_{i=1}^n \text{intersection\_length}([al_i, ar_i], [bl_i, br_i])$ . Пустое пересечение имеет длину 0, а длина отрезка  $[x, y]$  равна  $y - x$ .

Какое минимальное количество шагов необходимо выполнить, чтобы сделать  $I$  большим или равным  $k$ ?

### Входные данные

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

В первой строке каждого набора заданы два целых числа  $n$  и  $k$  ( $1 \leq n \leq 2 \cdot 10^5$ ;  $1 \leq k \leq 10^9$ ) — длина каждого из списков и минимально необходимое суммарное пересечение.

Во второй строке каждого набора заданы два целых числа  $l_1$  и  $r_1$  ( $1 \leq l_1 \leq r_1 \leq 10^9$ ) — отрезок, которому первоначально равны все  $[al_i, ar_i]$ .

В третьей строке каждого набора заданы два целых числа  $l_2$  и  $r_2$  ( $1 \leq l_2 \leq r_2 \leq 10^9$ ) — отрезок, которому первоначально равны все  $[bl_i, br_i]$ .

Гарантируется, что сумма  $n$  не превосходит  $2 \cdot 10^5$ .

### Выходные данные

Выведите  $t$  чисел — по одному на набор входных данных. Для каждого набора, выведите минимальное количество необходимых шагов, чтобы сделать  $I$  больше или равным  $k$ .

### Пример

входные данные	Скопировать
3 3 5 1 2 3 4 2 1000000000 1 1 999999999 999999999 10 3 5 10 7 8	
выходные данные	Скопировать
7 2000000000 0	

### Примечание

В первом наборе входных данных, мы можем достигнуть суммарного пересечения 5, используя, например, следующую стратегию:

- превратим  $[al_1, ar_1]$  из  $[1, 2]$  в  $[1, 4]$  за 2 шага;
- превратим  $[al_2, ar_2]$  из  $[1, 2]$  в  $[1, 3]$  за 1 шаг;
- превратим  $[bl_1, br_1]$  из  $[3, 4]$  в  $[1, 4]$  за 2 шага;
- превратим  $[bl_2, br_2]$  из  $[3, 4]$  в  $[1, 4]$  за 2 шага.

В результате,

$$I = \text{intersection\_length}([al_1, ar_1], [bl_1, br_1]) + \text{intersection\_length}([al_2, ar_2], [bl_2, br_2]) + \\ + \text{intersection\_length}([al_3, ar_3], [bl_3, br_3]) = 3 + 2 + 0 = 5$$

Во втором наборе, мы можем сделать  $[al_1, ar_1] = [0, 1000000000]$  за 1000000000 шагов и  $[bl_1, br_1] = [0, 1000000000]$  за 1000000000 шагов.

В третьем наборе, суммарное пересечение  $I$  уже равно  $10 > 3$ , а потому, не нужно делать ни одного шага.

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 07.08.2025 13:01:40<sup>UTC+5</sup> (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке





|   
[Kap6502](#) | [Выйти](#)

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

## F. Разделение массива

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам задан массив  $a$ , состоящий из  $n$  целых чисел.

Пусть  $\min(l, r)$  равно минимальному значению среди  $a_l, a_{l+1}, \dots, a_r$ , а  $\max(l, r)$  равно максимальному значению среди  $a_l, a_{l+1}, \dots, a_r$ .

Ваша задача — выбрать три таких **положительных** (больших 0) целых числа  $x$ ,  $y$  и  $z$ , что:

- $x + y + z = n$ ;
- $\max(1, x) = \min(x + 1, x + y) = \max(x + y + 1, n)$ .

Другими словами, вам необходимо разделить массив  $a$  на три последовательные непустые части, которые покрывают весь массив, и максимум в первой части равен минимуму во второй части, а также равен максимуму в третьей части (или же определить, что такое разбиение невозможно найти).

Среди всех возможных троек (разделений) можно выбрать любое.

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

### Входные данные

Первая строка входных данных содержит одно целое число  $t$  ( $1 \leq t \leq 2 \cdot 10^4$ ) — количество наборов тестовых данных. Затем следуют  $t$  наборов тестовых данных.

Первая строка набора тестовых данных содержит одно целое число  $n$  ( $3 \leq n \leq 2 \cdot 10^5$ ) — длину  $a$ .

Вторая строка набора тестовых данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ), где  $a_i$  равно  $i$ -му элементу  $a$ .

Гарантируется, что сумма  $n$  не превосходит  $2 \cdot 10^5$  ( $\sum n \leq 2 \cdot 10^5$ ).

### Выходные данные

Выведите ответ на каждый набор тестовых данных: NO в единственной строке, если не существует такого разбиения  $a$ , которое удовлетворяет ограничениям из условия задачи. Иначе выведите YES в первой строке и три целых числа  $x$ ,  $y$  и  $z$  ( $x + y + z = n$ ) во второй строке.

Если существует несколько возможных ответов, вы можете вывести любой.

### Пример

входные данные	Скопировать
<pre>6 11 1 2 3 3 3 4 4 3 4 2 1 8 2 9 1 7 3 9 4 1 9 2 1 4 2 4 3 3 1 2 7 4 2 1 1 4 1 4 5 1 1 1 1 1 7 4 3 4 3 3 3 4</pre>	
выходные данные	Скопировать
<pre>YES 6 1 4 NO YES 2 5 2 YES 4 1 2 YES 1 1 3 YES 2 1 4</pre>	



|   
[Kap6502](#) | [Выход](#)

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

## D2. Черепаха и задача MEX (сложная версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Две версии — это разные задачи. В этой версии задачи нельзя выбирать одно и то же целое число дважды или более. Можно делать взломы только в том случае, если обе версии решены.**

Однажды Черепаха играла с  $n$  последовательностями. Пусть длина  $i$ -й последовательности равна  $l_i$ . Тогда  $i$ -я последовательность была  $a_{i,1}, a_{i,2}, \dots, a_{i,l_i}$ .

Свинка дала Черепахе задачу, когда Черепаха играла. Условие задачи было следующим:

- Сначала было неотрицательное целое число  $x$ . Черепаха могла выполнять произвольное количество (возможно, ноль) операций с этим числом.
- В каждой операции Черепаха могла выбрать целое число  $i$ , такое что  $1 \leq i \leq n$  и  $i$  не было выбрано ранее, и сделать  $x$  равным  $\text{mex}^{\dagger}(x, a_{i,1}, a_{i,2}, \dots, a_{i,l_i})$ .
- Черепаха должна была найти ответ, который был максимальным значением  $x$  после выполнения произвольного количества операций.

Черепаха без труда решила вышеуказанную задачу. Она определила  $f(k)$  как ответ на вышеуказанную задачу, когда начальное значение  $x$  было  $k$ .

Затем Свинка дала Черепахе неотрицательное целое число  $m$  и попросил Черепаху найти значение  $\sum_{i=0}^m f(i)$  (т.е. значение  $f(0) + f(1) + \dots + f(m)$ ). К сожалению, она не смогла решить эту задачу. Пожалуйста, помогите ей!

$\dagger \text{mex}(c_1, c_2, \dots, c_k)$  определяется как наименьшее **неотрицательное** целое число  $x$ , которое не встречается в последовательности  $c$ . Например,  $\text{mex}(2, 2, 0, 3)$  равно 1,  $\text{mex}(1, 2)$  равно 0.

### 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  $n, m$  ( $1 \leq n \leq 2 \cdot 10^5, 0 \leq m \leq 10^9$ ).

Каждая из следующих  $n$  строк содержит несколько целых чисел. Первое целое число  $l_i$  ( $1 \leq l_i \leq 2 \cdot 10^5$ ) представляет длину  $i$ -й последовательности, а следующие  $l_i$  целых числа  $a_{i,1}, a_{i,2}, \dots, a_{i,l_i}$  ( $0 \leq a_{i,j} \leq 10^9$ ) представляют элементы  $i$ -й последовательности.

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

### Выходные данные

Для каждого набора входных данных выведите одно целое число — значение  $\sum_{i=0}^m f(i)$ .

### Пример

входные данные	Скопировать
<pre>6 3 4 2 0 2 3 2 3 3 4 7 0 1 5 3 4 5 0 2 0 4 11 1 1 5 1 3 0 3 3 2 50 2 1 2 2 1 2 1 1 7 1 2 4 1 4 9 5 4 114514 2 2 2 5 7 3 6 0 3 3 0 1 1</pre>	

```
5 0 9 2 1 5
5 1919810
1 2
2 324003 0
3 1416324 2 1460728
4 1312631 2 0 1415195
5 1223554 192248 2 1492515 725556
```

**Входные данные**[Скопировать](#)

```
16
18
1281
4
6556785365
1842836177961
```

**Примечание**

В первом наборе входных данных, когда  $x$  изначально равен 2, Черепаха может выбрать  $i = 3$  и сделать  $x$  равным  $\text{tex}(x, a_{3,1}, a_{3,2}, a_{3,3}, a_{3,4}) = \text{tex}(2, 7, 0, 1, 5) = 3$ . Можно доказать, что Черепаха не может сделать значение  $x$  больше 3, поэтому  $f(2) = 3$ .

Можно увидеть, что  $f(0) = 3$ ,  $f(1) = 3$ ,  $f(2) = 3$ ,  $f(3) = 3$ , и  $f(4) = 4$ . Таким образом,  $f(0) + f(1) + f(2) + f(3) + f(4) = 3 + 3 + 3 + 3 + 4 = 16$ .

Во втором наборе входных данных, когда  $x$  изначально равен 1, Черепаха может выбрать  $i = 1$  и сделать  $x$  равным  $\text{tex}(x, a_{1,1}, a_{1,2}, a_{1,3}, a_{1,4}, a_{1,5}) = \text{tex}(1, 0, 2, 0, 4, 11) = 3$ . Можно доказать, что Черепаха не может сделать значение  $x$  больше 3, поэтому  $f(1) = 3$ .

Можно увидеть, что  $f(0) = 4$ ,  $f(1) = 3$ ,  $f(2) = 4$ ,  $f(3) = 3$ , и  $f(4) = 4$ . Таким образом,  $f(0) + f(1) + f(2) + f(3) + f(4) = 4 + 3 + 4 + 3 + 4 = 18$ .

В четвертом наборе входных данных можно увидеть, что  $f(0) = 3$  и  $f(1) = 1$ . Таким образом,  $f(0) + f(1) = 3 + 1 = 4$ .

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:43<sub>UTC+5</sub> (h1).

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

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

При поддержке



ИТМО

## D. Зимний поход

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Циклическая страна представляет клетчатое поле  $2n \times 2n$ . Строки этого поля пронумерованы числами от 1 до  $2n$  сверху вниз, а столбцы пронумерованы числами от 1 до  $2n$  слева направо. Клетка  $(x, y)$  — это клетка на пересечении строки  $x$  и столбца  $y$  для  $1 \leq x \leq 2n$  и  $1 \leq y \leq 2n$ .

Сейчас Ваши  $n^2$  друзей находятся в левом верхнем углу этого поля. Иными словами, в каждой клетке  $(x, y)$ , где  $1 \leq x, y \leq n$  находится ровно один Ваш друг. В некоторых из остальных клеток поля находятся сугробы.

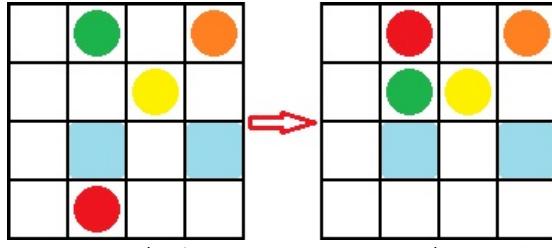
Ваши друзья хотят переместиться в правый нижний угол этого поля. Для этого в каждой клетке  $(x, y)$ , для которой  $n + 1 \leq x, y \leq 2n$  должен оказаться ровно один друг. При этом неважно, в какой клетке окажется каждый друг.

Вы решили помочь своим друзьям совершить поход.

Вы будете давать друзьям инструкции по перемещению одного из следующих видов:

- Выбрать некоторую строку  $x$ . Все друзья, находящиеся в этой строке перемещаются в следующую клетку этой строки. После этой операции друг из клетки  $(x, y)$  окажется в клетке  $(x, y + 1)$  для  $1 \leq y < 2n$ , а друг из клетки  $(x, 2n)$  окажется в клетке  $(x, 1)$ .
- Выбрать некоторую строку  $x$ . Все друзья, находящиеся в этой строке перемещаются в предыдущую клетку этой строки. После этой операции друг из клетки  $(x, y)$  окажется в клетке  $(x, y - 1)$  для  $1 < y \leq 2n$ , а друг из клетки  $(x, 1)$  окажется в клетке  $(x, 2n)$ .
- Выбрать некоторый столбец  $y$ . Все друзья, находящиеся в этом столбце перемещаются в следующую клетку этого столбца. После этой операции друг из клетки  $(x, y)$  окажется в клетке  $(x + 1, y)$  для  $1 \leq x < 2n$ , а друг из клетки  $(2n, y)$  окажется в клетке  $(1, y)$ .
- Выбрать некоторый столбец  $y$ . Все друзья, находящиеся в этом столбце перемещаются в предыдущую клетку этого столбца. После этой операции друг из клетки  $(x, y)$  окажется в клетке  $(x - 1, y)$  для  $1 < x \leq 2n$ , а друг из клетки  $(1, y)$  окажется в клетке  $(2n, y)$ .

Обратите внимание, как ведут себя друзья, находящиеся на границах клетчатого поля в этих инструкциях.



Пример применения третьей операции ко второму столбцу. Здесь разноцветные кружки обозначают Ваших друзей, а синим покрашены клетки с сугробами.

Вы можете давать инструкции любое количество раз. Вы можете давать инструкции разных видов. Если после одной из этих инструкций один из друзей окажется в сугробе, то он замёрзнет.

Чтобы друзья не замёрзли, Вы можете до объявления первой инструкции убрать некоторые сугробы с помощью следующей операции:

- Выбрать клетку  $(x, y)$ , в которой сейчас находится сугроб и убрать его за  $c_{x,y}$  монет.

Вы можете делать эту операцию любое количество раз.

Вы хотите потратить как можно меньше монет на уборку сугробов, а затем дать друзьям инструкции таким образом, чтобы они переместились в правый нижний угол поля и никто из них не замёрз.

Определите, сколько монет Вы потратите.

### Входные данные

В первой строке задано одно целое число  $t$  ( $1 \leq t \leq 100$ ) — количество наборов входных данных. Далее следуют описания этих наборов.

В первой строке дано одно число  $n$  ( $1 \leq n \leq 250$ ) — половина длины стороны поля.

В следующих  $2n$  строках дано по  $2n$  чисел  $c_{i,1}, c_{i,2}, \dots, c_{i,2n}$  ( $0 \leq c_{i,j} \leq 10^9$ ) — стоимости уборки сугробов в каждой клетке. Если  $c_{i,j} = 0$  для некоторых  $i, j$ , то в клетке  $(i, j)$  нет сугроба. Иначе в клетке  $(i, j)$  есть сугроб.

Гарантируется, что  $c_{i,j} = 0$  для  $1 \leq i, j \leq n$ .

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит 250.

## Выходные данные

Для каждого набора входных данных выведите одно число — минимальное количество монет, которое Вам нужно потратить.

### Пример

#### входные данные

[Скопировать](#)

```
4
1
0 8
1 99
2
0 0 0 0
0 0 0 0
9 9 2 2
9 9 9 9
2
0 0 4 2
0 0 2 4
4 2 4 2
2 4 2 4
4
0 0 0 0 0 0 0 2
0 0 0 0 0 0 2 0
0 0 0 0 0 2 0 0
0 0 0 0 2 0 0 0
0 0 0 2 2 0 2 2
0 0 2 0 1 6 2 1
0 2 0 0 2 4 7 4
2 0 0 0 2 0 1 6
```

#### выходные данные

[Скопировать](#)

```
100
22
14
42
```

### Примечание

В первом наборе входных данных можно убрать сугробы в клетках  $(2, 1)$  и  $(2, 2)$  за 100 монет. После этого Вы можете дать инструкции

- Перемещение в предыдущую клетку в первом столбце. После этого друг окажется в клетке  $(2, 1)$ .
- Перемещение в следующую клетку во второй строке. После этого друг окажется в клетке  $(2, 2)$  и закончит поход.

Во втором наборе входных данных можно убрать все сугробы на клетках столбцов 3 и 4 за 22 монеты. После этого Вы можете дать инструкции

- Перемещение в следующую клетку в первой строке.
- Перемещение в следующую клетку в первой строке.
- Перемещение в следующую клетку во второй строке.
- Перемещение в следующую клетку во второй строке.
- Перемещение в следующую клетку в третьем столбце.
- Перемещение в следующую клетку в третьем столбце.
- Перемещение в следующую клетку в четвёртом столбце.
- Перемещение в следующую клетку в четвёртом столбце.

Можно показать, что в результате этих инструкций ни один из друзей не замёрзнет, и что нельзя убрать сугробы меньшей суммарной стоимости.





| [Kap6502](#) | [Выйти](#)

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

## B2. Покраска массива II

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

**Единственное различие между двумя версиями задачи заключается в том, что в этой версии вам нужно найти минимальный возможный ответ.**

Гомеру очень нравятся массивы. Сегодня он красит массив  $a_1, a_2, \dots, a_n$  двумя видами цветов, **белым** и **черным**. Покраска массива  $a_1, a_2, \dots, a_n$  описывается массивом  $b_1, b_2, \dots, b_n$ , где  $b_i$  обозначает цвет  $a_i$  (0 для белого и 1 для черного).

Согласно покраске  $b_1, b_2, \dots, b_n$  массив  $a$  разделяется на два новых массива  $a^{(0)}$  и  $a^{(1)}$ , где  $a^{(0)}$  — это подпоследовательность всех белых элементов в  $a$ , а  $a^{(1)}$  — это подпоследовательность всех черных элементов в  $a$ . Например, если  $a = [1, 2, 3, 4, 5, 6]$  и  $b = [0, 1, 0, 1, 0, 0]$ , то  $a^{(0)} = [1, 3, 5, 6]$  и  $a^{(1)} = [2, 4]$ .

Количество отрезков в массиве  $c_1, c_2, \dots, c_k$ , обозначаемое  $\text{seg}(c)$ , — это количество элементов, которое останется, если объединить все соседние элементы с одинаковым значением в  $c$ . Например, количество отрезков в  $[1, 1, 2, 2, 3, 3, 3, 2]$  равно 4, так как массив станет равным  $[1, 2, 3, 2]$  после объединения соседних элементов с одинаковым значением. В частности, количество отрезков в пустом массиве равно 0.

Гомер хочет найти покраску  $b$ , согласно которой суммарное количество отрезков в  $a^{(0)}$  и  $a^{(1)}$ , т. е.  $\text{seg}(a^{(0)}) + \text{seg}(a^{(1)})$ , **минимально**. Найдите, чему равно это значение.

### Входные данные

Первая строка содержит целое число  $n$  ( $1 \leq n \leq 10^5$ ).

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

### Выходные данные

Выведите единственное число — **минимально** возможное суммарное количество отрезков.

### Примеры

входные данные	<a href="#">Скопировать</a>
6 1 2 3 1 2 2	
выходные данные	<a href="#">Скопировать</a>
4	
входные данные	<a href="#">Скопировать</a>
7 1 2 1 2 1 2 1	
выходные данные	<a href="#">Скопировать</a>
2	

### Примечание

В первом примере можно выбрать  $a^{(0)} = [1, 1, 2, 2]$ ,  $a^{(1)} = [2, 3]$  и  $\text{seg}(a^{(0)}) = \text{seg}(a^{(1)}) = 2$ . Таким образом, ответ  $2 + 2 = 4$ .

Во втором примере можно выбрать  $a^{(0)} = [1, 1, 1, 1]$ ,  $a^{(1)} = [2, 2, 2]$  и  $\text{seg}(a^{(0)}) = \text{seg}(a^{(1)}) = 1$ . Таким образом, ответ  $1 + 1 = 2$ .





|   
[Kap6502](#) | [Выйти](#)

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

## D. 01-дерево

ограничение по времени на тест: 1 секунда  
 ограничение по памяти на тест: 256 мегабайт

Есть ребро-взвешенное полное двоичное дерево с  $n$  листьями. Полное двоичное дерево определяется как дерево, в котором каждая нелистовая вершина имеет ровно 2 дочерних вершины. Для каждой нелистовой вершины обозначим одного из ее детей как левого ребенка, а другого — как правого.

Это двоичное дерево обладает очень странным свойством. Для каждой нелистовой вершины одно из ребер к ее детям имеет вес 0, а другое ребро — вес 1. Обратите внимание, что ребро с весом 0 может идти как к левому ребенку, так и к правому.

Вы забыли, как выглядит дерево, но, к счастью, вы помните некоторую информацию о листьях в виде массива  $a$  длины  $n$ . Для каждого  $i$  от 1 до  $n$ ,  $a_i$  представляет собой расстояние<sup>†</sup> от корня до  $i$ -го листа (в порядке обхода dfs<sup>‡</sup>). Определите, существует ли полное двоичное дерево, удовлетворяющее массиву  $a$ . Обратите внимание, что восстанавливать дерево **не нужно**.

<sup>†</sup> Расстояние от вершины  $u$  до вершины  $v$  определяется как сумма весов ребер на пути от вершины  $u$  до вершины  $v$ .

<sup>‡</sup> Порядок листьев в dfs определяется вызовом следующей функции dfs на корне бинарного дерева.

`dfs_order = [] — порядок обхода dfs`

`функция dfs(v):`

```

    если v — лист:
        добавить v в конец dfs_order
    иначе:
        dfs(левый ребенок v)
        dfs(правый ребенок v)
```

`dfs(корень)`

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

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

Вторая строка каждого набора входных данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq n - 1$ ) — расстояние от корня до  $i$ -го листа.

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

### Выходные данные

Для каждого набора входных данных выведите «YES», если существует полное двоичное дерево, удовлетворяющее массиву  $a$ , и «NO» в противном случае.

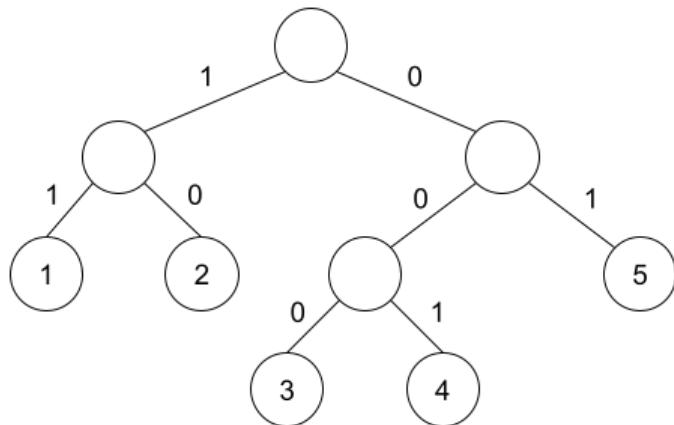
Вы можете выводить каждую букву в любом регистре (строчную или заглавную). Например, строки «yEs», «yes», «Yes» и «YES» будут приняты как положительный ответ.

### Пример

<b>входные данные</b>	<button>Скопировать</button>
2 5 2 1 0 1 1 5 1 0 2 1 3	
<b>выходные данные</b>	<button>Скопировать</button>
YES NO	

### Примечание

В первом наборе входных данных массиву удовлетворяет следующее дерево.



Для второго набора входных данных можно доказать, что не существует полного двоичного дерева, удовлетворяющего массиву.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:50<sub>UTC+5</sub> (h1).

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

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

При поддержке



ИТМО



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## E. Делимость на 25

ограничение по времени на тест: 1 секунда  
ограничение по памяти на тест: 256 мегабайт

Задано целое число  $n$  от 1 до  $10^{18}$ , записанное без лидирующих нулей.

За один ход можно поменять в нём любую такую пару соседних цифр местами, что в результате получается число без лидирующих нулей. Иными словами, **после каждого хода** должно получаться число без лидирующих нулей.

Какое наименьшее количество ходов надо последовательно совершить, чтобы получилось число, которое делится на 25?  
Выведите -1, если не существует такой последовательности ходов, которая приводит к числу, кратному 25.

### Входные данные

В первой строке входных данных задано целое число  $n$  ( $1 \leq n \leq 10^{18}$ ). Гарантируется, что первая (левая) цифра числа  $n$  отлична от нуля.

### Выходные данные

Если не существует такой последовательности ходов, которая приводит к числу, кратному 25, выведите -1. В противном случае выведите минимальное количество ходов в такой последовательности.

Обратите внимание, что вы можете менять местами только соседние цифры.

### Примеры

входные данные	выходные данные	Скопировать
5071	4	
входные данные	выходные данные	Скопировать
705	1	
входные данные	выходные данные	Скопировать
1241367	-1	

### Примечание

В первом тестовом примере одна из возможных последовательностей ходов следующая: 5071 → 5701 → 7501 → 7510 → 7150.

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 07.08.2025 13:01:52 UTC+5 (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО

## D. Турнир боевого искусства

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Монокарп планирует провести соревнование по смешанным единоборствам. В нем будет три весовых категории: легкий вес, средний вес и тяжелый вес. Победитель в каждой категории определяется по системе плей-офф.

В частности, это подразумевает, что количество участников в каждой категории должно быть степенью двойки. Дополнительно в каждой категории должно быть ненулевое количество участников.

Всего участников уже зарегистрировались на соревнование,  $i$ -й из них весит  $a_i$ . Чтобы разделить участников на категории, Монокарп планирует установить две целых границы весов  $x$  и  $y$  ( $x < y$ ).

Все участники с весом строго меньше  $x$  будут считаться легким весом. Все участники с весом больше или равно  $y$ , будут считаться тяжелым весом. Все остальные участники будут считаться средним весом.

Возможно, что распределение не сделает количество участников в каждой категории степенью двойки. Оно так же может сделать и пустые категории. Чтобы решить эти проблемы, Монокарп может пригласить любое количество участников в каждую категорию.

Обратите внимание, что Монокарп не может выгнать никого из  $n$  участников, которые уже зарегистрировались на соревнование.

Однако, он хочет пригласить как можно меньше дополнительных участников. Помогите Монокарпу выбрать  $x$  и  $y$  таким образом, чтобы суммарное количество необходимых дополнительных участников было как можно меньше. Выведите это количество.

### 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  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ) — количество зарегистрированных участников.

Во второй строке каждого набора входных данных записаны  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq n$ ) — веса зарегистрированных участников.

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

### Выходные данные

На каждый набор входных данных выведите одно целое число — наименьшее количество дополнительных участников, которых Монокарпу необходимо пригласить после того, как он выберет границы весов  $x$  и  $y$ .

### Пример

входные данные	Скопировать
4 4 3 1 2 1 1 1 6 2 2 2 1 1 1 8 6 3 6 3 6 3 6 6	
выходные данные	Скопировать
0 2 3 2	

### Примечание

В первом наборе входных данных Монокарп может выбрать  $x = 2$  и  $y = 3$ . Легкая, средняя и тяжелая весовые категории будут содержать 2, 1 и 1 участников, соответственно. Они все являются степенями двойки, поэтому дополнительных участников не нужно.

Во втором наборе входных данных вне зависимости от выбора  $x$  и  $y$  в одной категории будет 1 участник, в остальных — 0. Монокарпу придется пригласить 1 участника в обе из оставшихся категорий.

В третьем наборе входных данных Монокарп может выбрать  $x = 1$  и  $y = 2$ . Легкая, средняя и тяжелая весовые категории будут содержать 0, 3 и 3 участников, соответственно. В каждую категорию придется пригласить по одному участнику.

В четвертом наборе входных данных Монокарп может выбрать  $x = 8$  и  $y = 9$ . Легкая, средняя и тяжелая весовые категории будут содержать 8, 0 и 0 участников, соответственно. В среднюю и тяжелую категории придется пригласить по одному участнику.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:54<sup>UTC+5</sup> (h1).

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

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

При поддержке





|   
[Kap6502](#) | [Выйти](#)

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

## C. Водный баланс

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

$n$  цистерн с водой стоят в ряд,  $i$ -я из них содержит  $a_i$  литров воды. Цистерны пронумерованы от 1 до  $n$  слева направо.

Вы можете выполнить следующую операцию: выбрать некоторый подотрезок  $[l, r]$  ( $1 \leq l \leq r \leq n$ ), и перераспределить воду с цистерн  $l, l+1, \dots, r$  между ними равномерно. Другими словами, заменить каждое из  $a_l, a_{l+1}, \dots, a_r$  на  $\frac{a_l + a_{l+1} + \dots + a_r}{r-l+1}$ . К примеру, если для объемов  $[1, 3, 6, 7]$  вы выберете  $l = 2, r = 3$ , вы получите объемы  $[1, 4.5, 4.5, 7]$ . **Вы можете выполнять данную операцию любое количество раз.**

Какую лексикографически минимальную последовательность объемов воды вы можете получить?

Напомним:

Последовательность  $a$  лексикографически меньше последовательности  $b$  равной длины, если и только если выполняется следующее: в первой (слева) позиции, где  $a$  и  $b$  отличаются, в последовательности  $a$  стоит меньший элемент, чем в  $b$ .

### Входные данные

Первая строка содержит целое число  $n$  ( $1 \leq n \leq 10^6$ ) — количество цистерн с водой.

Вторая строка содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^6$ ) — объемы воды в цистернах изначально, в литрах.

Из-за большого размера входных данных, **не рекомендуется** считывать их как doubles.

### Выходные данные

Выведите лексикографически минимальную последовательность, которую вы можете получить. В  $i$ -й строке выведите итоговый объем воды в  $i$ -й цистерне.

Ваш ответ будет считаться правильным, если его абсолютная или относительная ошибка каждого  $a_i$  не превосходит  $10^{-9}$ .

Формально, пусть ваш ответ равен  $a_1, a_2, \dots, a_n$ , а ответ жюри равен  $b_1, b_2, \dots, b_n$ . Ваш ответ будет засчитан, если и только если  $\frac{|a_i - b_i|}{\max(1, |b_i|)} \leq 10^{-9}$  для каждого  $i$ .

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
4 7 5 5 7	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
5.6666666667 5.666666667 5.666666667 7.000000000	

<b>входные данные</b>	<input type="button" value="Скопировать"/>
5 7 8 8 10 12	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
7.000000000 8.000000000 8.000000000 10.000000000 12.000000000	

<b>входные данные</b>	<input type="button" value="Скопировать"/>
10 3 9 5 5 1 7 5 3 8 7	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
3.000000000 5.000000000 5.000000000 5.000000000 5.000000000	

```
| 5.000000000
| 5.000000000
| 5.000000000
| 7.500000000
| 7.500000000
```

**Примечание**

В первом примере, вы можете получить лексикографически минимальную последовательность, применив операцию к подотрезку [1, 3].

Во втором примере, вы не можете получить меньшую лексикографически последовательность.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:01:58<sup>UTC+5</sup> (h1).

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

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

При поддержке



ИТМО



|   
[Kap6502](#) | [Выйти](#)

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

## D. Досуг в школе №41

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

В средней школе №41 учатся  $n$  детей. Всем известно, что они хорошие математики. Однажды на перемене ребята решили провести исследование. Они выстроились в один ряд и повернули головы налево или направо.

Дети проделывали следующую операцию: каждую секунду несколько пар соседних в ряду детей, **смотрящих друг на друга**, могли **одновременно** развернуться в противоположную сторону. Таким образом, ребенок, смотрящий на правого соседа, повернется налево, и наоборот для второго ребенка. Более того, каждую секунду **хотя бы одна** пара соседних детей проделывала подобную операцию. Процесс заканчивается, когда нет пары соседних детей смотрящих друг на друга.

Дано число детей  $n$ , изначальная расстановка детей в ряду и целое положительное число  $k$ . Необходимо найти последовательность действий детей, завершающий процесс ровно за  $k$  секунд. Более формально, на каждый из  $k$  ходов нужно вывести номера детей, которые повернутся налево во время хода.

Для примера, дети могут действовать с конфигурацией приведенной ниже и  $k = 2$  следующим образом:



На первом ходу развернутся две пары: (1, 2) и (3, 4). После этого получается такая конфигурация:



На втором ходу развернется только пара (2, 3). В итоговой конфигурации никакая пара детей не смотрит друг на друга.  
Хорошая работа.



Если решение существует, то гарантируется что дети совершают не более чем  $n^2$  разворотов.

### Входные данные

Первая строка входных данных содержит два целых числа  $n, k$  ( $2 \leq n \leq 3000, 1 \leq k \leq 3000000$ ) — количество детей и количество ходов через которые нужно закончить.

Вторая строка входных данных содержит строку длины  $n$ , которая состоит только из символов L и R. L значит, что ребенок смотрит налево, а R — направо.

### Выходные данные

Если решения не существует, выведите  $-1$ .

Иначе, вывод должен состоять из  $k$  строк. Каждая строка должна начинаться с положительного целого числа  $n_i$  ( $1 \leq n_i \leq \frac{n}{2}$ ) — количества пар детей, которые развернутся на этом ходу. Далее в этой же строке должно следовать  $n_i$  **парно различных** чисел — номера детей, которые повернут голову налево во время этого хода.

После проведения всех разворотов, не должно быть пары соседних детей, смотрящих друг на друга.

Если существует более одного решения, выведите любое из них.

### Примеры

входные данные	Скопировать
2 1 RL	
выходные данные	Скопировать
1 1	

<b>входные данные</b>	<input type="button" value="Скопировать"/>
2 1 LR	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
-1	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
4 2 RLRL	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
2 1 3 1 2	

**Примечание**

Первый тест описывает пару детей смотрящих друг на друга. За один ход они развернутся.

Во втором teste дети не могут сделать ни одного хода. В итоге они не смогут закончить за  $k > 0$  ходов.

Третий тест описан в условии.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:02:00<sup>UTC+5</sup> (h1).

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

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

При поддержке



ИТМО

## E. Коробки с карандашами

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

На день рождения Мишка получил в подарок разноцветные карандаши! К сожалению, он живет в монохромном мире, где все одного и того же цвета, и имеет смысл исключительно насыщенность цвета. Эту упаковку можно представить в виде последовательности  $a_1, a_2, \dots, a_n$  из  $n$  целых чисел — насыщенность цвета каждого карандаша. Теперь Мишка хочет навести порядок в этой упаковке. Для этого он подготовил бесконечное количество пустых коробок. Он хочет их заполнить таким образом, чтобы:

- Каждый карандаш лежал **ровно** в одной коробке;
- В каждой непустой коробке лежало не меньше  $k$  карандашей;
- Если карандаши  $i$  и  $j$  лежат в одной и той же коробке, то  $|a_i - a_j| \leq d$ , где  $|x|$  означает модуль числа  $x$ . Обратите внимание, что обратное условие может не выполняться, могут быть такие карандаши  $i$  и  $j$ , что  $|a_i - a_j| \leq d$  и они лежат в различных коробках.

Помогите Мишке определить, можно ли распределить все карандаши по коробкам. Выведите "YES", если существует такое распределение. Иначе выведите "NO".

### Входные данные

В первой строке записаны три целых числа  $n, k$  и  $d$  ( $1 \leq k \leq n \leq 5 \cdot 10^5$ ,  $0 \leq d \leq 10^9$ ) — количество карандашей, минимальный размер любой непустой коробки и максимальная разница в насыщенности какой-либо пары карандашей, лежащих в одной коробке, соответственно.

Во второй строке записаны  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ) — насыщенности цветов каждого карандаша.

### Выходные данные

Выведите "YES", если можно распределить все карандаши по коробкам так, чтобы все условия выполнялись. Иначе выведите "NO".

### Примеры

входные данные	Скопировать
6 3 10 7 2 7 7 4 2	
выходные данные	Скопировать
YES	
входные данные	Скопировать
6 2 3 4 5 3 13 4 10	
выходные данные	Скопировать
YES	
входные данные	Скопировать
3 2 5 10 16 22	
выходные данные	Скопировать
NO	

### Примечание

В первом примере можно разделить все карандаши на 2 коробки по 3 карандаша любым образом. Также можно положить все карандаши в одну коробку, разница в насыщенности между любой парой карандашей в ней не будет превосходить 10.

Во втором примере можно разделить карандаши насыщенностей [4, 5, 3, 4] на 2 коробки размера 2 и положить оставшиеся в еще одну коробку.



|   
[Kap6502](#) | [Выйти](#)

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

## D1. Разбиение XOR — сольная версия

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Это сольная версия задачи. Обратите внимание, что решение этой задачи может иметь или не иметь общих идей с решением игровой версии. Вы можете сдавать и получать баллы за каждую из версий независимо.

Вы можете делать взломы только тогда, когда обе версии задачи решены.

Дана целочисленная переменная  $x$ , которая изначально имеет значение  $n$ . Операция разбиения определена как:

- Выберите значение  $y$  такое, что  $0 < y < x$  и  $0 < (x \oplus y) < x$ .
- Обновите  $x$ , приравняв его  $x = y$  либо  $x = x \oplus y$ .

Определите, возможно ли преобразовать  $x$  в  $m$  за не более чем 63 операции, используя такую операцию разбиения. Если это возможно, предъявите последовательность операций для достижения  $x = m$ .

Здесь  $\oplus$  обозначает операцию [побитового исключающего ИЛИ](#).

### Входные данные

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

Первая строка каждого набора входных данных содержит два целых числа  $n$  и  $m$  ( $1 \leq m < n \leq 10^{18}$ ) — изначальное значение  $x$  и требуемое значение  $x$ .

### Выходные данные

Для каждого набора тестовых данных выведите ваш ответ в следующем формате.

Если невозможно достичь требуемого значения  $m$  за 63 операции, то выведите  $-1$  в единственной строке.

Иначе,

Первая строка должна содержать целое число  $k$  ( $1 \leq k \leq 63$ ) — где  $k$  является количеством произведенных операций.

Следующая строка должна содержать  $k + 1$  целое число — последовательность, в виде которой переменная  $x$  изменяется с ходом операций разбиения. 1-е и  $(k + 1)$ -ое числа должны равняться  $n$  и  $m$  соответственно.

### Пример

входные данные	Скопировать
3 7 3 4 2 481885160128643072 45035996273704960	
выходные данные	Скопировать
1 7 3 -1 3 481885160128643072 337769972052787200 49539595901075456 45035996273704960	

### Примечание

В первом тестовом случае  $n = 7$ , для первой операции  $x = 7$ , если мы выберем  $y = 3$ , тогда  $(7 \oplus 3) < 7$ , следовательно мы можем обновить  $x$  сделав его равным 3 уже на первой операции, чему и равно  $m$ .

Во втором тестовом случае  $n = 4$ , для первой операции  $x = 4$ .

Если мы выберем:

- $y = 1$  тогда  $(4 \oplus 1) > 4$
- $y = 2$  тогда  $(4 \oplus 2) > 4$
- $y = 3$  тогда  $(4 \oplus 3) > 4$

Следовательно мы не можем выполнить первую операцию и невозможно сделать  $x = 2$ .



|    
[Kap6502](#) | [Выход](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Определите победные острова в гонке

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Муууууууууууууууууууууу

— Корова Бесси, Искусство гонок по островам

Две коровы фермера Джона, Бесси и Элси, планируют участвовать в гонках по  $n$  островам. Существует  $n - 1$  основных мостов, соединяющих остров  $i$  с островом  $i + 1$  для всех  $1 \leq i \leq n - 1$ . Кроме того, существует  $m$  альтернативных мостов. Элси может использовать **основные, и альтернативные** мосты, а Бесси — только основные. Все мосты являются **односторонними** и могут использоваться только для перемещения с острова с меньшим индексом на остров с большим индексом.

Изначально Элси находится на острове 1, а Бесси — на острове  $s$ . Коровы ходят по очереди, причем Бесси ходит первой. Предположим, что корова находится на острове  $i$ . Во время хода коровы, если есть мост, соединяющий остров  $i$  с островом  $j$ , то корова может переместиться на остров  $j$ . Затем остров  $i$  разрушается, и все мосты, соединяющие остров  $i$ , также разрушаются. Обратите внимание на следующее:

- Если нет мостов, соединяющих остров  $i$  с другим островом, то остров  $i$  разрушается, и эта корова выбывает из гонки.
- Если другая корова также находится на острове  $i$ , то после того, как эта корова переместится на другой остров, остров разрушится, и другая корова выбывает из гонки.
- После того, как остров или мост разрушились, ни одна корова не может ими воспользоваться.
- Если корова выбывает из гонки, ее ход пропускается до конца гонки.

Гонка заканчивается, как только одна из коров достигнет острова  $n$ . Можно показать, что независимо от стратегий коров, хотя бы одна из них достигнет острова  $n$ . Бесси побеждает тогда и только тогда, когда она первой достигает острова  $n$ .

Для каждого  $1 \leq s \leq n - 1$  определите, выиграет ли Бесси, если она начнет гонку на острове  $s$ . Предположим, что обе коровы следуют оптимальным стратегиям, чтобы обеспечить себе победу.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. Первая строка содержит одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит два числа  $n$  и  $m$  ( $2 \leq n \leq 2 \cdot 10^5$ ,  $0 \leq m \leq 2 \cdot 10^5$ ) — количество островов и количество альтернативных мостов.

Следующие  $m$  строк каждого набора входных данных содержат по два числа  $u$  и  $v$  ( $1 \leq u < v \leq n$ ) — острова, которые соединяет альтернативный мост. Гарантируется, что все альтернативные мосты различны и не совпадают с основными мостами.

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

### Выходные данные

Для каждого набора входных данных выведите в отдельной строке бинарную строку длины  $n - 1$ .  $i$ -й символ равен 1, если Бесси может выиграть, если она начнет с острова  $i$ . В противном случае он равен 0.

### Пример

входные данные	Скопировать
5 6 0 6 1 2 6 6 1 1 5 10 4 1 3 1 6 2 7 3 8 15 3 2 8 4 9 8 15	
выходные данные	Скопировать

```
11111  
11011  
10011  
100001111  
11000111000111
```

### Примечание

В первом наборе входных данных нет альтернативных мостов, по которым Элси могла бы обогнать Бесси и первой достичь острова  $n$ , поэтому Бесси выиграет на всех островах, потому что она всегда ходит первой.

Во втором случае Бесси проиграет, если начнет с острова 3, потому что:

- Ход Бесси: Перейти по основному мосту с острова 3 на остров 4.
- Ход Элси: Перейти по основному мосту с острова 1 на остров 2.
- Ход Бесси: Перейти по основному мосту с острова 4 на остров 5.
- Ход Элси: Перейти по альтернативному мосту с острова 2 на остров 6. Элси первой достигает острова  $n$ .

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 07.08.2025 13:02:07<sup>UTC+5</sup> (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке





|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## F1. Завершение проектов (простая версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Единственное отличие между простой и сложной версиями — то, что вам необходимо завершить все проекты в легкой версии, но это не обязательно в сложной версии.**

Поликарп — очень известный фрилансер. Его текущий рейтинг равен  $r$ .

Некоторые очень богатые заказчики попросили его завершить некоторые проекты для их компаний. Чтобы завершить  $i$ -й проект, Поликарпу необходимо иметь хотя бы  $a_i$  единиц рейтинга; после того, как он завершит этот проект, его рейтинг изменится на  $b_i$  (его рейтинг увеличится или уменьшится на  $b_i$ ) ( $b_i$  может быть положительным и отрицательным). Рейтинг Поликарпа не может падать ниже нуля, потому что люди перестанут доверять фрилансеру с таким низким рейтингом.

Поликарп может выбрать порядок, в котором он выполняет проекты. Сможет ли он выполнить все проекты, если выберет подходящий порядок?

Иными словами, проверьте, что существует такой порядок исполнения проектов, что у Поликарпа достаточно рейтинга перед каждым из них, а после каждого из проектов его рейтинг неотрицателен.

### Входные данные

Первая строка входных данных содержит два целых числа  $n$  и  $r$  ( $1 \leq n \leq 100, 1 \leq r \leq 30000$ ) — количество проектов и изначальный рейтинг Поликарпа, соответственно.

Следующие  $n$  строк содержат проекты, один на строку.  $i$ -й проект представлен парой целых чисел  $a_i$  и  $b_i$  ( $1 \leq a_i \leq 30000, -300 \leq b_i \leq 300$ ) — рейтинг, необходимый для того, чтобы завершить  $i$ -й проект и изменение рейтинга после завершения проекта.

### Выходные данные

Выведите «YES» или «NO».

### Примеры

<b>входные данные</b>	<b>Скопировать</b>
3 4 4 6 10 -2 8 -1	
<b>выходные данные</b>	<b>Скопировать</b>
YES	
<b>входные данные</b>	<b>Скопировать</b>
3 5 4 -5 4 -2 1 3	
<b>выходные данные</b>	<b>Скопировать</b>
YES	
<b>входные данные</b>	<b>Скопировать</b>
4 4 5 2 5 -3 2 1 4 -2	
<b>выходные данные</b>	<b>Скопировать</b>
YES	
<b>входные данные</b>	<b>Скопировать</b>
3 10 10 0 10 -10 30 0	

**выходные данные****Скопировать**

NO

**Примечание**

В первом примере возможная последовательность имеет вид: 1, 2, 3.

Во втором примере возможная последовательность имеет вид: 2, 3, 1.

В третьем примере возможная последовательность имеет вид: 3, 1, 4, 2.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:02:09<sub>UTC+5</sub> (h1).

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

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

При поддержке





|   
[Kap6502](#) | [Выйти](#)

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

### C. Точки на плоскости

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

На плоскости расположены  $n$  точек  $(x_i, y_i)$  с целочисленными координатами от 0 до  $10^6$ . Расстоянием между двумя точками с номерами  $a$  и  $b$  назовем следующую величину:  $\text{dist}(a, b) = |x_a - x_b| + |y_a - y_b|$  (расстояние, вычисляемое по такой формуле, называется *манхэттенским расстоянием*).

Назовем гамильтоновым путем некоторую перестановку  $p_i$  от 1 до  $n$ . Назовем длиной этого пути величину  $\sum_{i=1}^{n-1} \text{dist}(p_i, p_{i+1})$ .

Найдите какой-нибудь гамильтонов путь с длиной не более, чем  $25 \times 10^8$ . Обратите внимание, минимизировать длину гамильтонова пути не требуется.

#### Входные данные

В первой строке дано число  $n$  ( $1 \leq n \leq 10^6$ ).

В  $i + 1$ -й строке даны координаты точки с номером  $i$ :  $x_i$  и  $y_i$  ( $0 \leq x_i, y_i \leq 10^6$ ).

Гарантируется, что никакие две точки не совпадают.

#### Выходные данные

Выведите перестановку чисел  $p_i$  от 1 до  $n$  — искомый гамильтонов путь. Перестановка должна удовлетворять неравенству

$$\sum_{i=1}^{n-1} \text{dist}(p_i, p_{i+1}) \leq 25 \times 10^8.$$

Если возможных ответов несколько, разрешается вывести любой.

Гарантируется, что существует подходящая перестановка.

#### Примеры

входные данные	<a href="#">Скопировать</a>
5 0 7 8 10 3 4 5 0 9 12	
выходные данные	<a href="#">Скопировать</a>
4 3 1 2 5	

#### Примечание

В teste из условия, суммарное расстояние равно:

$$\text{dist}(4, 3) + \text{dist}(3, 1) + \text{dist}(1, 2) + \text{dist}(2, 5) =$$

$$(|5 - 3| + |0 - 4|) + (|3 - 0| + |4 - 7|) + (|0 - 8| + |7 - 10|) + (|8 - 9| + |10 - 12|) = 2 + 4 + 3 + 3 + 8 + 3 + 1 + 2 = 26$$





|   
[Kap6502](#) | [Выйти](#)

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

## C. Денежные операции

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

В Васином городе есть  $n$  банков, расположенных по кругу так, что соседними являются банки, номера которых различаются на 1, а также банк номер 1 и банк номер  $n$  (если  $n > 1$ ). При этом, сам себе банк соседом не является.

В каждом банке у Васи есть счёт с каким-нибудь балансом. Заметим, что он может быть и отрицательным, если Вася должен банку.

Вася доступна следующая операция: взять два **соседних** банка и перечислить со счёта в одном банке на счёт в другом любое количество денег. При этом нет никаких ограничений на размер переводимой суммы или текущее состояние счёта в любом из банков.

Вася путается в больших числах, поэтому он просит вас определить, за какое минимальное количество операций он может обнулить сумму денег, лежащую на счёте в каждом из банков. Гарантируется, что Вася может это сделать, то есть суммарный баланс Васи по всем банкам равен нулю.

### Входные данные

В первой строке входных данных записано единственное целое число  $n$  ( $1 \leq n \leq 100\,000$ ) — количество банков.

Во второй строке записаны  $n$  целых чисел  $a_i$  ( $-10^9 \leq a_i \leq 10^9$ ),  $i$ -е из которых определяет баланс Васи в  $i$ -м банке.

Гарантируется, что сумма всех  $a_i$  равна 0.

### Выходные данные

Выведите минимальное количество операций, за которое Вася может обнулить баланс в каждом банке.

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
3 5 0 -5	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
1	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
4 -1 0 1 0	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
2	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
4 1 2 3 -6	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
3	

### Примечание

В первом примере можно перечислить из первого банка в третий сумму 5.

Во втором примере можно перечислить сначала из третьего банка во второй, а потом из второго в первый сумму 1.

В третьем примере одним из оптимальных ответов является следующая последовательность действий:

1. перечислить из первого банка во второй сумму 1;
2. перечислить из второго банка в третий сумму 3;
3. перечислить из третьего банка в четвертый сумму 6.



|   
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## F. Тимофей и черно-белое дерево

ограничение по времени на тест: 4 секунды

ограничение по памяти на тест: 256 мегабайт

Тимофей приехал в известную летнюю школу и нашел там дерево на  $n$  вершинах. Дерево — это связный неориентированный граф без циклов.

Каждая вершина этого дерева, кроме  $c_0$ , покрашена в **белый** цвет. Вершина  $c_0$  покрашена в **черный** цвет.

Тимофей хочет покрасить все вершины этого дерева в **черный** цвет. Для этого он выполняет  $n - 1$  операцию. Во время  $i$ -й операции он выбирает **белую** вершину  $c_i$  и красит ее в **черный** цвет.

Назовем **позитивностью** дерева минимальное расстояние между всеми парами различных **черных** вершин в нем. Расстоянием между вершинами  $v$  и  $u$  называется количество ребер на пути от  $v$  до  $u$ .

После каждой очередной покраски Тимофей хочет знать **позитивность** текущего дерева.

### Входные данные

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

В первой строке каждого набора входных данных записаны числа  $n, c_0$  ( $2 \leq n \leq 2 \cdot 10^5, 1 \leq c_0 \leq n$ ) — количество вершин в дереве и номер начальной **черной** вершины.

Во второй строке каждого набора входных данных записано  $n - 1$  различных чисел  $c_1, c_2, \dots, c_{n-1}$  ( $1 \leq c_i \leq n, c_i \neq c_0$ ), где  $c_i$  это вершина, покрашенная в **черный** цвет во время  $i$ -й операции.

В следующей  $n - 1$  строке каждого набора входных данных записаны числа  $v_i, u_i$  ( $1 \leq v_i, u_i \leq n$ ) — ребра в дереве.

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

### Выходные данные

Для каждого набора входных данных выведите в отдельной строке  $n - 1$  число.

Число с номером  $i$  должно быть равно **позитивности** дерева, полученного первыми  $i$  покрасками.

### Пример

входные данные	Скопировать
6 6 6 4 1 3 5 2 2 4 6 5 5 3 3 4 1 3 4 2 4 1 3 3 1 2 3 1 4 10 3 10 7 6 5 2 9 8 1 4 1 2 1 3 4 5 4 3 6 4 8 7 9 8 10 8 1 8 7 3 7 5 1 2 4 6 1 2 3 2 4 5 3 4 6 5 7 6 9 7	

```

9 3 1 4 2 6 8 5
4 1
8 9
4 8
2 6
7 3
2 4
3 5
5 4
10 2
1 8 5 10 6 9 4 3 7
10 7
7 8
3 6
9 7
7 6
4 2
1 6
7 5
9 2

```

**входные данные****Скопировать**

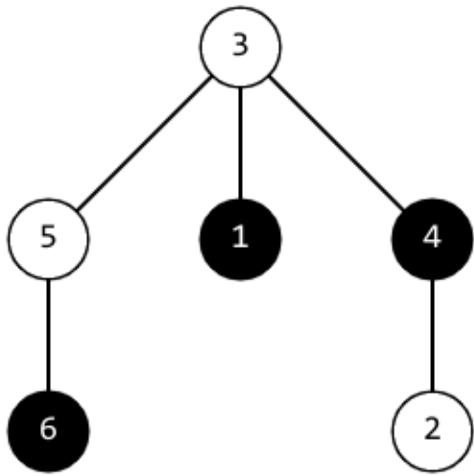
```

3 2 1 1 1
3 1 1
3 2 2 2 2 2 1 1 1
4 2 2 1 1 1
5 1 1 1 1 1 1 1
4 3 2 2 1 1 1 1 1 1

```

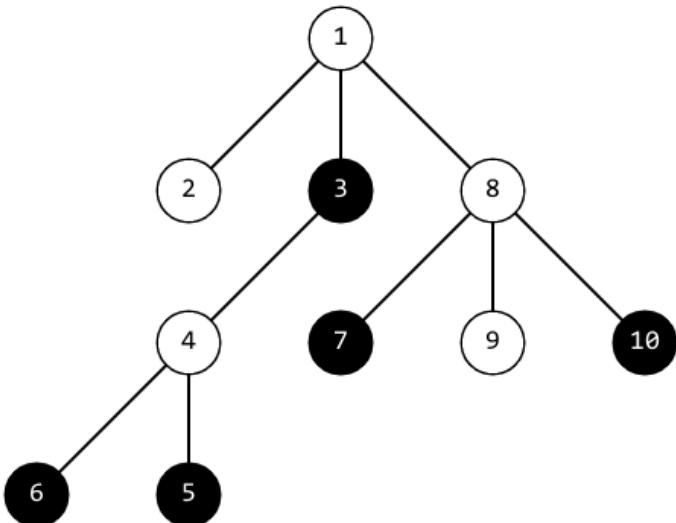
**Примечание**

В первом наборе входных данных, после второй покраски, дерево выглядит так:



Расстояние между вершинами 1 и 6 равно 3, расстояние между вершинами 4 и 6 равно 3, расстояние между вершинами 1 и 4 равно 2. Позитивность такого дерева равна минимуму из этих расстояний, то есть 2.

В третьем наборе входных данных, после четвертой покраски, дерево выглядит так:



Позитивность такого дерева равна 2.

## E. Уравнивание людей

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Мальчик Петя и его друг, Petya++, отправились на BFDMONCON, где сегодня проходит конкурс костюмов.



Проходя по фестивалю, они наткнулись на научный стенд имени профессора Оука и Гольфболла, где им предложили решить интересную задачу.

Дана последовательность из чисел  $a_1, a_2, \dots, a_n$  и вы можете производить несколько операций над этой последовательностью.

Каждая операция должна выглядеть следующим образом. Вы выбираете некоторую подпоследовательность<sup>†</sup>. Далее вы называете все числа, находящиеся на нечетных позициях в этой подпоследовательности — *северными*, а все числа, находящиеся на четных позициях в этой подпоследовательности — *южными*. При этом учитывается лишь позиция числа в подпоследовательности, а не в исходной последовательности.

Например, рассмотрим последовательность **1, 4, 2, 8, 5, 7, 3, 6, 9** и ее подпоследовательность (выделена жирным) **1, 4, 2, 8, 5, 7, 3, 6, 9**. Тогда числа 4 и 5 будут *северными*, а числа 2 и 6 — *южными*.

После этого можно выполнить одно из следующих действий:

- прибавить 1 ко всем северным числам и отнять 1 от всех южных; или
- прибавить 1 ко всем южным числам и отнять 1 от всех северных.

Таким образом, из последовательности **1, 4, 2, 8, 5, 7, 3, 6, 9** при выборе описанной выше подпоследовательности можно получить либо **1, 5, 1, 8, 6, 7, 3, 5, 9**, либо **1, 3, 3, 8, 4, 7, 3, 7, 9**.

Затем операция заканчивается. Обратите также внимание, что все операции являются независимыми, т. е. по окончании одной операции числа больше не называются *северными* или *южными*.

Необходимо с помощью описанных выше операций превратить все числа последовательности в нули. Поскольку до конкурса костюмов осталось совсем немного времени, ребята хотят узнать, какое минимальное количество операций для этого придется совершить.

Увы, ребятам такая задача оказалась не по силам, поэтому они позвали Вас на помощь.

<sup>†</sup> Последовательность  $c$  является подпоследовательностью  $d$ , если  $c$  может быть получена из  $d$  удалением нескольких (возможно, ни одного или всех) элементов.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит целое число  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ) — количество чисел в последовательности.

Во второй строке находится  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $-10^9 \leq a_i \leq 10^9$ ) — описание самой последовательности.

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

### Выходные данные

Для каждого набора входных данных выведите по одному целому числу в отдельной строке — минимальное количество операций, необходимое, чтобы последовательность состояла только из нулей.

### Пример

#### входные данные

[Скопировать](#)

```
5
3
1 2 -3
5
1 0 0 -1 -1
6
2 -4 3 -5 4 1
5
1 -1 1 -1 1
7
0 0 0 0 0 0 0
```

#### выходные данные

[Скопировать](#)

```
3
2
6
1
0
```

### Примечание

В первом тестовом примере последовательность операций выглядит так: **1, 2, -3**  $\rightarrow$  **0, 2, -2**  $\rightarrow$  **0, 1, -1**  $\rightarrow$  **0, 0, 0**.

Во втором тестовом примере последовательность выглядит так: **1, 0, 0, -1, -1**  $\rightarrow$  **0, 0, 0, 0, -1**  $\rightarrow$  **0, 0, 0, 0, 0**.

В четвертом тестовом примере достаточно выбрать всю последовательность в качестве подпоследовательности, а затем отнять единицу от северных чисел и прибавить единицу к южным. Таким образом, последовательность занулится за одну операцию.

В пятом тестовом примере не нужно делать никаких операций, поскольку последовательность уже и так состоит из нулей.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:02:18<sup>UTC+5</sup> (h1).

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

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

При поддержке



ITMO



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## E. Монстры

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам дан неориентированный граф с  $n$  вершинами и  $m$  ребрами. Изначально для каждой вершины  $i$  существует монстр с опасностью  $a_i$  в этой вершине. Вы можете победить монстра с опасностью  $a_i$  тогда и только тогда, когда вы уже победили как минимум  $a_i$  других монстров.

Вы хотите победить всех монстров. Сначала вы выбираете некоторую вершину  $s$  и побеждаете монстра в этой вершине (поскольку вы еще не побеждали монстров,  $a_s$  должно быть равно 0). Затем вы можете перемещаться в соседние вершины. Если вы хотите переместиться из вершины  $u$  в вершину  $v$ , то должно выполняться следующее: либо монстр в вершине  $v$  уже побежден ранее, либо вы можете победить его сейчас. Во втором случае вы побеждаете монстра в вершине  $v$  и достигаете вершины  $v$ . Вы можете проходить вершины и ребра любое количество раз.

Определите, сможете ли вы победить всех монстров или нет.

### 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 их описание.

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

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

Следующие  $m$  строк содержат по два целых числа  $u, v$  ( $1 \leq u, v \leq n$ ), описывающих ребро между вершинами  $u$  и  $v$ . Гарантируется, что в графе нет кратных ребер и петель.

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

### Выходные данные

Для каждого набора входных данных выведите «YES», если вы можете победить всех монстров, и «NO» иначе.

Вы можете выводить каждую букву в любом регистре (строчную или заглавную). Например, строки «yEs», «yes», «Yes» и «YES» будут приняты как положительный ответ.

### Пример

входные данные	Скопировать
5 4 3 2 1 0 3 1 2 2 3 3 4 6 6 0 1 2 3 0 1 1 2 2 3 3 4 4 5 4 6 5 6 4 3 0 1 2 0 1 2 2 3 1 3 4 6 1 1 1 0 1 2 3 2 4 3 2 4 4 1 1 3 5 5 0 1 3 2 0 1 2 2 3 3 4 4 5	

3 5	
<b>входные данные</b>	<a href="#">Скопировать</a>
YES	
YES	
NO	
YES	
NO	

**Примечание**

В первом наборе входных данных мы можем начать с вершины 3, победить монстра в ней, затем перейти в вершины 2, 1 в этом порядке, победив монстров в них. Затем вернуться к вершине 3 и пойти к вершине 4, победив в ней монстра.

В третьем наборе входных данных нет пути к вершине 4, если мы начинаем с вершины 1. Также нет пути к вершинам 1, 2, 3, если мы начнем с вершины 4.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:02:19<sub>UTC+5</sub> (h1).

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

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

При поддержке



ITMO



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Прогулка по скобкам

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

Дана строка  $s$  длины  $n$ , состоящая из символов '(' и ')'. Вы идете по этой строке. Вы начинаете с первого символа  $s$  и хотите сделать последовательность шагов так, чтобы закончить на  $n$ -м символе. За один шаг вы можете переместиться на один символ влево (если вы стоите не на первом символе) или на один символ вправо (если вы стоите не на последнем символе). Вы не можете оставаться на одном и том же месте, однако вы можете посетить любой символ, включая первый и последний, любое количество раз.

В каждый момент времени вы записываете символ, на котором вы сейчас стоите. Мы говорим, что строка *проходима*, если существует некоторая последовательность ходов, начинающаяся в первом символе и заканчивающаяся в последнем, такая, что записанная вами строка является правильной скобочной последовательностью.

Правильная скобочная последовательность — это последовательность скобок, которая может быть преобразована в правильное арифметическое выражение путем добавления символов '+' и '\*' между исходными символами последовательности. Например, последовательности скобок «()()», «(( ))» являются правильными (итоговыми выражениями являются: «(1)+(1)», «((1+1)+1)»), а последовательности «)» и «(» не являются правильными.

$t = 0:$	$((())()))$	(	$t = 5:$	$((())())()$	$((((($
$t = 1:$	$((\cdot))())$	$(($	$t = 6:$	$((())(\cdot))$	$((((($
$t = 2:$	$((\cdot))())$	$((\cdot$	$t = 7:$	$((())(\cdot))$	$((((($
$t = 3:$	$((\cdot))())$	$((\cdot\cdot$	$t = 8:$	$((\cdot\cdot))(\cdot)$	$((((($
$t = 4:$	$((\cdot\cdot))()$	$((\cdot\cdot\cdot$	$t = 9:$	$((\cdot\cdot\cdot))(\cdot)$	$((((($

Один возможный корректный путь по  $s = ((())())$ . Красная точка указывает на ваше текущее положение, а красная строка — на записанную вами строку. Обратите внимание, что итоговая красная строка — это правильная скобочная последовательность.

Вам даны  $q$  запросов. Каждый запрос меняет значение символа с '(' на ')' или наоборот. После каждого изменения определите, является ли данная строка проходимой.

Запросы **сохраняются**, то есть эффект от каждого запроса распространяется на последующие запросы.

### Входные данные

Первая строка входных данных содержит два целых числа  $n$  и  $q$  ( $1 \leq n, q \leq 2 \cdot 10^5$ ) — размер строки и количество запросов, соответственно.

Вторая строка ввода содержит строку  $s$  длины  $n$ , состоящую из символов '(' и ')' — начальная строка.

Каждая из следующих  $q$  строк содержит одно целое число  $i$  ( $1 \leq i \leq n$ ) — индекс символа, который нужно изменить в данном запросе.

### Выходные данные

Для каждого запроса выведите «YES», если строка проходима после этого запроса, и «NO» в противном случае.

Вы можете выводить ответ в любом регистре (верхнем или нижнем). Например, строки «YES», «Yes», «Yes» и «YES» будут распознаны как положительные ответы.

### Примеры

#### Входные данные

```
10 9
((())()))
9
7
```

```
2
6
3
6
7
4
8
```

**входные данные****Скопировать**

```
YES
YES
NO
NO
YES
NO
YES
NO
NO
```

**входные данные****Скопировать**

```
3 2
(())
2
3
```

**входные данные****Скопировать**

```
NO
NO
```

**Примечание**

В первом примере:

- После первого запроса строка имеет вид ( () ) ( ) ( ). Эта строка является правильной скобочной последовательностью, поэтому ее можно пройти, просто перемещаясь вправо.
- После второго запроса строка имеет вид ( () ) ( ) ( ). Если вы сдвинетесь один раз вправо, затем влево, затем пройдете вправо до конца строки, вы получите строку ( ( ( ( ) ( ) ( ), которая является правильной скобочной последовательностью.
- После третьего запроса строка имеет вид ( ) ) ( ) ( ) ( ). Мы можем показать, что эта строка не является проходимой.

Во втором примере, строки после запросов равны ( ) ) и ( ) (, ни одна из которых не является проходимой.

---

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

Соревнования по программированию 2.0

Время на сервере: 07.08.2025 13:02:21<sub>UTC+5</sub> (h1).

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

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

При поддержке



ИТМО



| [English](#) | [Russian](#)  
[Кар6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## F1. Летающая сортировка (простая версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Это простая версия задачи. В этой версии все числа в заданном массиве различны и ограничения на  $n$  меньше, чем в сложной версии задачи.**

Вам дан массив  $a$  из  $n$  целых чисел (**в массиве нет одинаковых элементов**). Вы можете производить над элементами массива следующие операции:

1. выбрать любой индекс  $i$  ( $1 \leq i \leq n$ ) и переместить элемент  $a[i]$  в **начало** массива;
2. выбрать любой индекс  $i$  ( $1 \leq i \leq n$ ) и переместить элемент  $a[i]$  в **конец** массива.

Например, если  $n = 5$ ,  $a = [4, 7, 2, 3, 9]$ , то можно применить следующую последовательность операций:

- после применения операции первого типа ко второму элементу массив  $a$  станет равным  $[7, 4, 2, 3, 9]$ ;
- после применения операции второго типа ко второму элементу массив  $a$  станет равным  $[7, 2, 3, 9, 4]$ .

Вы можете проводить операции любого типа произвольное количество раз в любом порядке.

Найдите минимальное суммарное количество операций первого и второго типа, которые сделают массив  $a$  отсортированным по неубыванию. Иными словами, сколько минимум операций надо применить, чтобы массив удовлетворял неравенствам  $a[1] \leq a[2] \leq \dots \leq a[n]$ ?

### Входные данные

В первой строке записано одно целое число  $t$  ( $1 \leq t \leq 100$ ) — количество наборов тестовых данных в тесте. Далее следуют  $t$  наборов тестовых данных.

Каждый набор начинается со строки, в которой записано целое число  $n$  ( $1 \leq n \leq 3000$ ) — размер массива  $a$ .

Далее следуют  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^9$ ) — массив, который требуется отсортировать заданными операциями. **Все числа в заданном массиве различны**.

Сумма  $n$  по всем наборам тестовых данных в одном teste не превосходит 3000.

### Выходные данные

Для каждого набора тестовых данных выведите одно целое число — минимальное суммарное количество операций первого и второго типа, которые сделают массив отсортированным по неубыванию.

### Пример

входные данные	Скопировать
4 5 4 7 2 3 9 5 3 5 8 1 7 5 1 4 5 7 12 4 0 2 1 3	
выходные данные	Скопировать
2 2 0 2	

### Примечание

В первом тестовом наборе нужно переместить сначала тройку, а потом двойку в начало массива. Следовательно, искомая последовательность операций может иметь вид:  $[4, 7, 2, 3, 9] \rightarrow [3, 4, 7, 2, 9] \rightarrow [2, 3, 4, 7, 9]$ .

Во втором тестовом наборе нужно переместить единицу в начало массива, а восьмерку — в конец. Искомая последовательность операций имеет вид:  $[3, 5, 8, 1, 7] \rightarrow [1, 3, 5, 8, 7] \rightarrow [1, 3, 5, 7, 8]$ .

В третьем тестовом наборе массив уже отсортирован.



|   
[Kap6502](#) | [Выйти](#)

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

## E. MEX против DIFF

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

У Тёти Люсине, как и у всех гениальных людей, имеются свои странности. Сегодня в качестве развлечения она решила посмотреть на противостояние двух могущественных функций. В зависимости от результата она решит, добавлять ли эти функции в CrowdScript.

Дан массив  $a$  из  $n$  неотрицательных целых чисел. За одну операцию вы можете заменить любое число в массиве на любое неотрицательное целое число.

Назовём стоимостью массива  $\text{DIFF}(a) - \text{MEX}(a)$ , где  $\text{MEX}$  множества чисел — это минимальное целое неотрицательное число, которого нет в множестве, а  $\text{DIFF}$  — количество различных чисел в массиве.

Например,  $\text{MEX}(\{1, 2, 3\}) = 0$ ,  $\text{MEX}(\{0, 1, 2, 4, 5\}) = 3$ .

Вам необходимо определить, какую минимальную стоимость массива  $a$  можно получить, если разрешено сделать не больше  $k$  операций.

### Входные данные

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

В первой строке описания каждого набора входных данных находится два целых числа  $n$  и  $k$  ( $1 \leq n \leq 10^5$ ,  $0 \leq k \leq 10^5$ ) — размер массива  $a$  и максимальное количество операций, которое можно сделать.

Во второй строке описания каждого набора входных данных находятся  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^9$ ) — массив  $a$ .

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

### Выходные данные

Для каждого набора входных данных выведите единственное число — минимальная стоимость, которую можно получить, сделав не больше  $k$  операций.

### Пример

входные данные	<a href="#">Скопировать</a>
<pre>4 4 1 3 0 1 2 4 1 0 2 4 5 7 2 4 13 0 0 13 1337 1000000000 6 2 1 2 8 0 0 0</pre>	
выходные данные	<a href="#">Скопировать</a>
<pre>0 1 2 0</pre>	

### Примечание

В первом наборе входных данных не требуется делать какие-либо операции, чтобы минимизировать  $\text{DIFF} - \text{MEX}$ .

Во втором наборе входных данных можно заменить 5 на 1. После этого массив  $a$  будет  $[0, 2, 4, 1]$ ,  $\text{DIFF} = 4$ ,  $\text{MEX} = \text{MEX}(\{0, 1, 2, 4\}) = 3$ , поэтому ответ равен 1.

В третьем наборе входных данных один из возможных массивов  $a$  будет  $[4, 13, 0, 0, 13, 1, 2]$ ,  $\text{DIFF} = 5$ ,  $\text{MEX} = 3$ .

В четвертом наборе входных данных один из возможных массивов  $a$  будет  $[1, 2, 3, 0, 0, 0]$ .



|   
[Kap6502](#) | [Выйти](#)

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

## H. Тест Сакурако

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Сакурако вскоре будет сдавать тест. Тест можно описать как массив целых чисел  $n$  и задание на нем:

Задав целое число  $x$ , Сакурако может выполнить следующую операцию любое количество раз:

- Выбрать целое число  $i$  ( $1 \leq i \leq n$ ) такое, что  $a_i \geq x$ ;
- Изменить значение  $a_i$  на  $a_i - x$ .

Применяя эту операцию произвольное количество раз, она хочет найти минимальную возможную медиану\* массива  $a$ .

Сакурако знает массив, но не знает целое число  $x$ . Кто-то проболтался, что одно из  $q$  значений  $x$  будет в следующем тесте, поэтому Сакурако спрашивает вас, каков ответ для каждого такого  $x$ .

\*Медиана массива длины  $n$  — это элемент, который стоит в середине отсортированного массива (в  $\frac{n+2}{2}$ -й позиции для чётного  $n$ , и в  $\frac{n+1}{2}$ -й для нечётного)

### 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  $q$  ( $1 \leq n, q \leq 10^5$ ) — количество элементов массива и количество запросов.

Вторая строка каждого набора содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq n$ ) — элементы массива.

Следующие  $q$  строк содержат по одному целому числу  $x$  ( $1 \leq x \leq n$ ).

Гарантируется, что сумма  $n$  по всем наборам входных данных не превысит  $10^5$ . То же самое гарантируется и для суммы  $q$  по всем наборам входных данных.

### Выходные данные

Для каждого набора входных данных выведите  $q$  целых чисел — ответ для каждого запроса.

### Пример

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

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:19:20<sup>UTC+5</sup> (h1).

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

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

При поддержке



ITMO



| [English](#) | [Russian](#)  
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D2. Кёрк и бинарная строка (сложная версия)

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

**Единственное отличие между легкой и сложной версиями — длины строк. Вы можете взламывать эту задачу тогда, когда решите ее. Но предыдущую только тогда, когда решите обе задачи.**

У Кёрка есть бинарная строка  $s$  (строка, содержащая только нули и единицы) длины  $n$ , и он просит вас найти бинарную строку  $t$  такой же длины, для которой выполняются следующие условия:

- Для любых  $l$  и  $r$  ( $1 \leq l \leq r \leq n$ ) длина наибольшей неубывающей подпоследовательности в подстроке  $s_l s_{l+1} \dots s_r$  равна длине наибольшей неубывающей подпоследовательности в подстроке  $t_l t_{l+1} \dots t_r$ ;
- Количество нулей в строке  $t$  — максимально возможное.

Неубывающая подпоследовательность в строке  $p$  — это такая последовательность индексов  $i_1, i_2, \dots, i_k$ , что  $i_1 < i_2 < \dots < i_k$  и  $p_{i_1} \leq p_{i_2} \leq \dots \leq p_{i_k}$ . Число  $k$  называется длиной подпоследовательности.

Если существует несколько строк, удовлетворяющих условиям, то выведите любую.

### Входные данные

Единственная строка содержит бинарную строку длины не больше  $10^5$ .

### Выходные данные

Выполните бинарную строку, для которой выполняются указанные условия. Если существует несколько таких строк, выведите любую.

### Примеры

входные данные	<a href="#">Скопировать</a>
110	
выходные данные	<a href="#">Скопировать</a>
010	

входные данные	<a href="#">Скопировать</a>
010	
выходные данные	<a href="#">Скопировать</a>
010	

входные данные	<a href="#">Скопировать</a>
0001111	
выходные данные	<a href="#">Скопировать</a>
0000000	

входные данные	<a href="#">Скопировать</a>
0111001100111011101000	
выходные данные	<a href="#">Скопировать</a>
0011001100001011101000	

### Примечание

В первом примере:

- Для подстрок длины 1 длина наибольшей неубывающей подпоследовательности равна 1;
- Для  $l = 1, r = 2$  наибольшая неубывающая подпоследовательность для подстроки  $s_1 s_2$  — 11, для подстроки  $t_1 t_2$  — 01;
- Для  $l = 2, r = 3$  наибольшая неубывающая подпоследовательность для подстроки  $s_2 s_3$  — 1, для подстроки  $t_2 t_3$  — 1;
- Для  $l = 1, r = 3$  наибольшая неубывающая подпоследовательность для подстроки  $s_1 s_3$  — 11, для подстроки  $t_1 t_3$  — 00;

Второй пример аналогичен первому.



|   
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## B2. Отправьте коробки Алисе (усложнённая версия)

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Это — сложная версия задачи. В ней  $1 \leq n \leq 10^6$  и  $0 \leq a_i \leq 10^6$ . Вы можете взламывать эту версию тогда, когда решите ее (но предыдущую только тогда, когда заблокировали обе версии).

Приближается Рождество, и наш главный герой, Боб, готовит эффектный подарок для своей давней лучшей подруги Алисы. В этом году он решил приготовить  $n$  коробок шоколада, пронумерованных от 1 до  $n$ . Изначально  $i$ -я коробка содержит  $a_i$  кусочков шоколада.

Поскольку Боб — типичный приятный парень, он не отправит Алисе  $n$  пустых ящиков. Другими словами, **минимум одно число из  $a_1, a_2, \dots, a_n$  является положительным**. Поскольку Алиса не любит взаимно простые множества, она будет счастлива, только если существует какое-то целое число  $k > 1$ , такое, что количество кусочков в каждой коробке делится на  $k$ . Обратите внимание, что Алиса не будет возражать, если будут какие-то пустые коробки.

Чарли, парень Алисы, также является вторым лучшим другом Боба, поэтому он решает помочь Бобу, переставляя кусочки шоколада. За одну секунду Чарли может взять кусок в  $i$ -й коробке, и поместить его либо в  $i - 1$ -ю коробку или  $i + 1$ -ю коробку (если такие коробки существуют). Конечно, он хочет помочь своему другу как можно быстрее. Поэтому он просит вас подсчитать минимальное количество секунд, которое понадобится ему, чтобы сделать Алису счастливой.

### Входные данные

Первая строка содержит одно целое число  $n$  ( $1 \leq n \leq 10^6$ ) — количество коробок шоколада.

Вторая строка содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^6$ ) — количество кусков шоколада в  $i$ -й коробке.

Гарантируется, что как минимум одно число среди  $a_1, a_2, \dots, a_n$  положительное.

### Выходные данные

Если Чарли не может сделать Алису счастливой, выведите  $-1$ .

Иначе выведите  $x$  — минимальное количество секунд, которые нужны Чарли, чтобы помочь Бобу сделать Алису счастливой.

### Примеры

входные данные	<a href="#">Скопировать</a>
3 4 8 5	
выходные данные	<a href="#">Скопировать</a>
9	

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

входные данные	<a href="#">Скопировать</a>
4 0 5 15 10	
выходные данные	<a href="#">Скопировать</a>
0	

входные данные	<a href="#">Скопировать</a>
1 1	
выходные данные	<a href="#">Скопировать</a>
-1	

### Примечание

В первом примере Чарли может передвинуть все шоколадные кусочки во вторую коробку. Каждая коробка будет делиться на



|   
[Kap6502](#) | [Выйти](#)

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

## E. Разделение на клики

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 512 мегабайт

Даны два целых числа,  $n$  и  $k$ . Рассмотрим граф на  $n$  вершинах, пронумерованных от 1 до  $n$ , в котором изначально нет ребер.

Вам нужно назначить каждой вершине число; пусть  $a_i$  — число, назначенное вершине  $i$ . Все  $a_i$  должны быть различными целыми числами от 1 до  $n$ .

После того как вы назначили числа для вершин, вы добавляете в граф ребра. Для пары вершин  $(i, j)$  добавляется ребро между ними, если  $|i - j| + |a_i - a_j| \leq k$ .

Ваша цель — создать граф, который можно разбить на минимально возможное (для заданных значений  $n$  и  $k$ ) количество клик. Каждая вершина графа должна принадлежать ровно одной клике. Напомним, что клика — это такое множество вершин, что каждая пара вершин в этом множестве соединена ребром.

Поскольку *BledDest* не особо подтянул свои навыки программирования, он не может решить задачу «дан граф, разбейте его на минимальное количество клик». Поэтому мы также просим вас вывести само разбиение.

### Входные данные

Первая строка содержит одно целое число  $t$  ( $1 \leq t \leq 1600$ ) — количество наборов входных данных.

Каждый набор входных данных состоит из одной строки, содержащей два целых числа  $n$  и  $k$  ( $2 \leq n \leq 40$ ;  $1 \leq k \leq 2n$ ).

### Выходные данные

Для каждого набора входных данных выведите три строки:

- первая строка должна содержать  $n$  **различных** целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq n$ ) — значения, которые вы назначаете вершинам;
- вторая строка должна содержать одно целое число  $q$  ( $1 \leq q \leq n$ ) — количество клик, на которые вы разбиваете граф;
- третья строка должна содержать  $n$  целых чисел  $c_1, c_2, \dots, c_n$  ( $1 \leq c_i \leq q$ ) — разбиение графа на клики. Где две вершины  $u$  и  $v$  в одной клике, если  $c_u = c_v$ .

Если существует несколько ответов, выведите любой из них.

### Пример

входные данные	<a href="#">Скопировать</a>
3 2 3 5 4 8 16	
выходные данные	<a href="#">Скопировать</a>
2 1 1 1 1 3 1 5 2 4 2 1 1 2 1 2 1 2 3 4 5 6 7 8 1 1 1 1 1 1 1 1	



ITMO



|   
[Kap6502](#) | [Выйти](#)

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

## D. Порталы

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вы играете в стратегическую компьютерную игру (да, у нас закончились идеи для легенд задач). В этой игре вы управляете огромной армией, и ваша задача — захватить  $n$  замков противника.

Рассмотрим игровой процесс более подробно. Изначально у вас в управлении армия из  $k$  воинов. Противник контролирует  $n$  замков; для захвата  $i$ -го замка вам потребуется  $a_i$  воинов (считается, что каждый захват замка проводится без потерь, поэтому размер армии после захвата не меняется). После захвата каждого замка вы увеличиваете свою армию, нанимая воинов в новом замке — в  $i$ -м замке вы сможете нанять  $b_i$  воинов. Помимо захвата замков, в каждом из них можно оставить охрану (только после захвата): если вы оставляете хотя бы одного воина в замке, то этот замок будет считаться *охраняемым*. У каждого замка есть параметр важности  $c_i$ , и успешность вашей стратегии определяется суммарной важностью всех охраняемых замков в конце игры. Оставлять охрану в замке можно двумя способами:

- если вы находитесь в замке  $i$ , вы можете оставить одного воина для охраны замка  $i$ ;
- замки связаны  $m$  односторонними порталами. Каждый портал характеризуется двумя номерами соединяемых замков  $u$  и  $v$  (для каждого портала  $u > v$ ). Порталами можно пользоваться следующим способом: если вы находитесь в замке  $u$ , вы можете отправить одного воина охранять замок  $v$  через портал.

Вы захватываете замки по очереди: сначала первый, потом второй, и так далее. После захвата замка  $i$ , пока вы не захватили замок  $i + 1$ , вы можете нанять воинов в замке  $i$ , оставить охрану в замке  $i$ , а также воспользоваться любым количеством порталов, ведущих из замка  $i$  в замки с меньшими номерами. После того, как вы захватите следующий замок, эти действия для замка  $i$  будут уже недоступны.

Если в какой-то момент вам не хватает воинов для захвата следующего замка, вы проигрываете. Ваша цель — максимизировать суммарную важность всех охраняемых замков после захвата последнего замка ( обратите внимание, что вы можете нанимать воинов в последнем замке после его захвата, оставлять в нем охрану и пользоваться порталами — суммарная важность всех охраняемых замков будет подсчитана после всех ваших действий).

Можете ли вы разработать оптимальную стратегию захвата замков и оставления охраны?

### Входные данные

В первой строке входных данных заданы три целых числа  $n$ ,  $m$  и  $k$  ( $1 \leq n \leq 5000$ ,  $0 \leq m \leq \min(\frac{n(n-1)}{2}, 3 \cdot 10^5)$ ,  $0 \leq k \leq 5000$ ) — количество замков, количество порталов и изначальное количество воинов в вашей армии, соответственно.

Далее следуют  $n$  строк, в  $i$ -й из которых содержится описание  $i$ -го замка — три целых числа  $a_i$ ,  $b_i$  и  $c_i$  ( $0 \leq a_i, b_i, c_i \leq 5000$ ) — количество воинов, требуемых для захвата замка  $i$ , количество воинов, которых можно нанять после захвата, и важность замка, соответственно.

Далее следуют  $m$  строк, в  $i$ -й из которых содержится описание  $i$ -го портала — два целых числа  $u_i$  и  $v_i$  ( $1 \leq v_i < u_i \leq n$ ), соответствующих порталу, который ведет из замка  $u_i$  в замок  $v_i$ . Не существует двух одинаковых порталов.

Гарантируется, что размер вашей армии ни при каких условиях не может превысить 5000 (то есть  $k + \sum_{i=1}^n b_i \leq 5000$ ).

### Выходные данные

Если невозможно захватить все замки ни при каких обстоятельствах, выведите одно целое число  $-1$ .

Иначе выведите одно целое число — максимальную суммарную важность всех охраняемых замков в конце игры.

### Примеры

входные данные	Скопировать
4 3 7 7 4 17 3 0 8 11 2 0 13 3 5 3 1 2 1 4 3	
выходные данные	Скопировать
5	

Входные данные	Скопировать
4 3 7 7 4 17 3 0 8 11 2 0 13 3 5 3 1 2 1 4 1	
Выходные данные	Скопировать
22	
Входные данные	Скопировать
4 3 7 7 4 17 3 0 8 11 2 0 14 3 5 3 1 2 1 4 3	
Выходные данные	Скопировать
-1	

### Примечание

Лучшая стратегия в первом примере — следующая:

1. захватить первый замок;
2. нанять солдат в первом замке, теперь у вас 11 солдат;
3. захватить второй замок;
4. захватить третий замок;
5. нанять солдат в третьем замке, теперь у вас 13 солдат;
6. захватить четвертый замок;
7. оставить одного воина в четвертом замке, теперь у вас 12 солдат.

Эта стратегия (и некоторые другие) дает в результате суммарную важность охраняемых замков, равную 5.

Лучшая стратегия во втором примере — следующая:

1. захватить первый замок;
2. нанять солдат в первом замке, теперь у вас 11 солдат;
3. захватить второй замок;
4. захватить третий замок;
5. нанять солдат в третьем замке, теперь у вас 13 солдат;
6. захватить четвертый замок;
7. оставить одного воина в четвертом замке, теперь у вас 12 солдат;
8. отправить одного воина в первый замок через третий портал, теперь у вас 11 солдат.

Эта стратегия (и некоторые другие) дает в результате суммарную важность охраняемых замков, равную 22.

В третьем примере нельзя захватить последний замок: для этого нужны 14 воинов, но вы можете собрать не более 13 до захвата последнего замка.





|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## F. Унификация MST

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

Задан неориентированный взвешенный **связный** граф, состоящий из  $n$  вершин и  $m$  ребер **без петель и кратных ребер**.

$i$ -е ребро —  $e_i = (u_i, v_i, w_i)$ ; расстояние между вершинами  $u_i$  и  $v_i$  по ребру  $e_i$  равно  $w_i$  ( $1 \leq w_i$ ). Граф является **связным**, то есть для каждой пары вершин существует хотя бы один путь между ними, состоящий только из ребер заданного графа.

Минимальное оствое дерево (MST) в случае **положительных** весов — это подмножество ребер связного взвешенного неориентированного графа, соединяющее все его вершины и имеющее минимальный суммарный вес среди всех таких подмножеств (суммарный вес — это сумма весов выбранных ребер).

Вы можете модифицировать заданный граф. Единственная операция, которую вы можете проводить, заключается в следующем: увеличить вес какого-либо ребра на 1. Вы **можете** увеличивать вес каждого ребра любое (возможно, нулевое) количество раз.

Предположим, что изначальный вес MST был равен  $k$ . Ваша задача — увеличить веса некоторых ребер **за минимально возможное количество операций** таким образом, чтобы вес MST в получившемся графе остался равен  $k$ , но MST стало **独一无二ным** (это означает, что есть только один способ выбрать MST в получившемся графе).

Ваша задача — посчитать **минимальное** количество операций, необходимое для того, чтобы это сделать.

### Входные данные

Первая строка входных данных содержит два целых числа  $n$  и  $m$  ( $1 \leq n \leq 2 \cdot 10^5$ ,  $n - 1 \leq m \leq 2 \cdot 10^5$ ) — количество вершин и количество ребер в изначальном графе.

The next  $m$  строк содержат по три целых числа.  $i$ -я строка содержит описание  $i$ -го ребра  $e_i$ . Оно определяется тремя целыми числами  $u_i$ ,  $v_i$  и  $w_i$  ( $1 \leq u_i, v_i \leq n$ ,  $u_i \neq v_i$ ,  $1 \leq w_i \leq 10^9$ ), где  $u_i$  и  $v_i$  — вершины, соединяемые  $i$ -м ребром, а  $w_i$  — вес этого ребра.

Гарантируется, что заданный граф **не содержит петель и кратных ребер** (то есть для каждого  $i$  от 1 до  $m$   $u_i \neq v_i$  и для каждой неориентированной пары  $(u, v)$  существует не более одного ребра, соединяющего эту пару вершин). Также гарантируется, что заданный граф является **связным**.

### Выходные данные

Выведите одно целое число — **минимальное** количество операций, чтобы унифицировать MST заданного графа без изменения веса MST.

### Примеры

входные данные	выходные данные
8 10 1 2 1 2 3 2 2 4 5 1 4 2 6 3 3 6 1 3 3 5 2 3 7 1 4 8 1 6 2 4	1
входные данные	выходные данные
4 3 2 1 3 4 3 4 2 4 1	0
входные данные	выходные данные

```
3 3
1 2 1
2 3 2
1 3 3
```

**входные данные****Скопировать**

0

**входные данные****Скопировать**

```
3 3
1 2 1
2 3 3
1 3 3
```

**входные данные****Скопировать**

1

**входные данные****Скопировать**

1 0

**входные данные****Скопировать**

0

**входные данные****Скопировать**

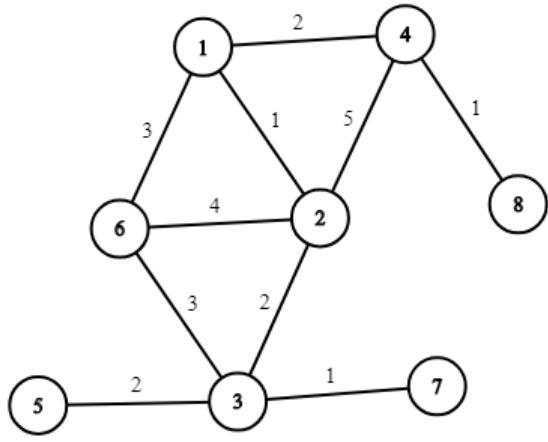
```
5 6
1 2 2
2 3 1
4 5 3
2 4 2
1 4 2
1 5 3
```

**входные данные****Скопировать**

2

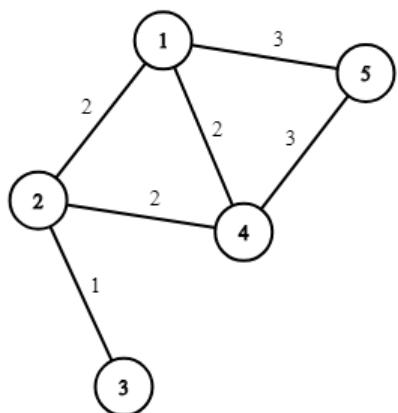
**Примечание**

Картинка, соответствующая первому тестовому примеру:



Вы можете, например, увеличить вес ребра (1, 6) или (6, 3) на 1 для унификации MST.

Картинка, соответствующая последнему тестовому примеру:



Вы можете, например, увеличить веса ребер (1, 5) и (2, 4) на 1 для унификации MST.

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 08.08.2025 11:19:31 UTC+5 (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке





|   
[Kap6502](#) | [Выходи](#)

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

## F. Новогодняя головоломка

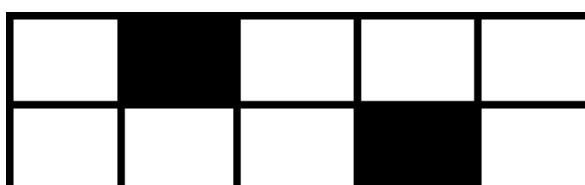
ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

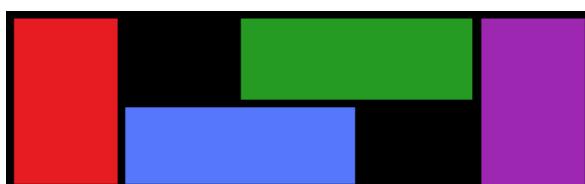
*Каждый год Дед Мороз дарит подарки всем детям. Однако в каждой стране есть свои традиции, и этот процесс происходит по разному. Например в Берлгандии нужно решить новогоднюю головоломку.*

Поликарпу досталась следующая задача: дана клетчатая полоска размером  $2 \times n$ , некоторые клетки на ней заблокированы. Нужно проверить, можно ли замостить все незаблокированные клетки с помощью дощечек  $2 \times 1$  и  $1 \times 2$ .

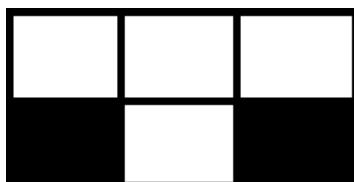
Например, если  $n = 5$  и полоска имеет следующий вид (черные клетки заблокированы):



То ее можно замостить, например, используя две вертикальных и две горизонтальных, как на картинке ниже (разные дощечки обозначаются разным цветом).



А если  $n = 3$  и полоска имеет следующий вид:



То замостить свободные клетки невозможно.

Поликарп легко справился с этой задачей и получил свой новогодний подарок. А сможете ли вы решить ее?

### Входные данные

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

Перед каждым набором входных данных расположена пустая строка.

В первой строке каждого набора содержится два целых числа  $n$  и  $m$  ( $1 \leq n \leq 10^9$ ,  $1 \leq m \leq 2 \cdot 10^5$ ) — длина полоски и количество заблокированных клеток на ней.

В следующих  $m$  строках находится по два целых числа  $r_i, c_i$  ( $1 \leq r_i \leq 2$ ,  $1 \leq c_i \leq n$ ) — номера строк и столбцов заблокированных клеток. Гарантируется, что все заблокированные клетки различные, т.е.  $(r_i, c_i) \neq (r_j, c_j)$ ,  $i \neq j$ .

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

### Выходные данные

Для каждого набора входных данных в отдельной строке выведите:

- «YES», если можно ли замостить все незаблокированные клетки с помощью дощечек  $2 \times 1$  и  $1 \times 2$ ;
- «NO» в противном случае.

Вы можете выводить «YES» и «NO» в любом регистре (например, строки yEs, yes, Yes и YES будут распознаны как положительный ответ).

### Пример

**входные данные**

**Скопировать**

```
3  
5 2  
2 2  
1 4  
  
3 2  
2 1  
2 3  
  
6 4  
2 1  
2 3  
2 4  
2 6
```

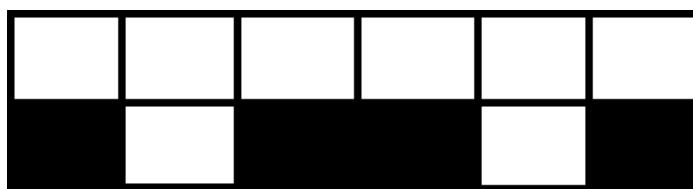
**входные данные****Скопировать**

```
YES  
NO  
NO
```

**Примечание**

Первые два набора входных данных разобраны в условии.

В третьем наборе входных данных полоска выглядит следующим образом



Несложно убедиться, что свободные клетки на ней нельзя замостить

---

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:19:38<sup>UTC+5</sup> (h1).

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

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

При поддержке



ИТМО

## D1. Красно-синие операции (простая версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Единственное отличие между простой и сложной версиями — максимальные значения  $n$  и  $q$ .**

Задан массив, состоящий из  $n$  целых чисел. Изначально все элементы красные.

К массиву можно несколько раз применить следующую операцию. На  $i$ -й операции вы выбираете элемент массива; затем:

- если элемент красный, то он увеличивается на  $i$  и становится синим;
- если элемент синий, то он уменьшается на  $i$  и становится красным.

Операции нумеруются с 1, т. е. на первой операции некоторый элемент изменяется на 1 и так далее.

К массиву задаются  $q$  запросов следующего вида:

- дано целое число  $k$ , какой может быть наибольший минимум в массиве после того как вы примените **ровно  $k$**  операций к нему?

Обратите внимание, что операции не изменяют массив между запросами, все запросы задаются к начальному массиву  $a$ .

### Входные данные

В первой строке записано два целых числа  $n$  и  $q$  ( $1 \leq n, q \leq 1000$ ) — количество элементов массива и количество запросов.

Во второй строке записаны  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ).

В третьей строке записаны  $q$  целых чисел  $k_1, k_2, \dots, k_q$  ( $1 \leq k_j \leq 10^9$ ).

### Выходные данные

На каждый запрос выведите одно целое число — наибольший минимум, который может быть в массиве после того как вы примените **ровно  $k$**  операций к нему.

### Примеры

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

входные данные	<a href="#">Скопировать</a>
5 10 5 2 8 4 4 1 2 3 4 5 6 7 8 9 10	
выходные данные	<a href="#">Скопировать</a>
3 4 5 6 7 8 9 8 11 8	

входные данные	<a href="#">Скопировать</a>
2 5 2 3 10 6 8 1 3	
выходные данные	<a href="#">Скопировать</a>
10 7 8 3 3	



| [English](#) | [Russian](#)  
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## E. Дерево коротких расстояний

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Задано неориентированное дерево, состоящее из  $n$  вершин. Неориентированное дерево — это связный граф с  $n - 1$  ребром.

Ваша задача — добавить минимальное количество ребер таким образом, что длина кратчайшего пути из вершины 1 до любой другой вершины не превышает 2. Заметьте, что вы не можете добавлять петли и кратные ребра.

### Входные данные

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

Следующие  $n - 1$  строк описывают ребра: ребро  $i$  задано как пара вершин  $u_i, v_i$  ( $1 \leq u_i, v_i \leq n$ ). Гарантируется, что заданный граф является деревом. Гарантируется, что среди заданных ребер петли и кратные ребра отсутствуют.

### Выходные данные

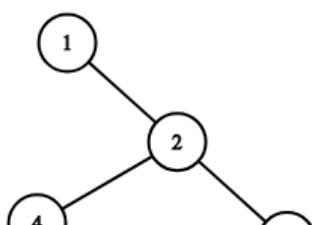
Выведите одно целое число — минимальное количество ребер, которое необходимо добавить, чтобы длина кратчайшего пути из вершины 1 до любой другой вершины не превышает 2. Заметьте, что вы не можете добавлять петли и кратные ребра.

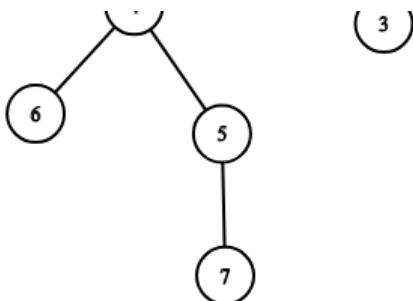
### Примеры

входные данные	скопировать
7 1 2 2 3 2 4 4 5 4 6 5 7	
выходные данные	скопировать
2	
входные данные	скопировать
7 1 2 1 3 2 4 2 5 3 6 1 7	
выходные данные	скопировать
0	
входные данные	скопировать
7 1 2 2 3 3 4 3 5 3 6 3 7	
выходные данные	скопировать
1	

### Примечание

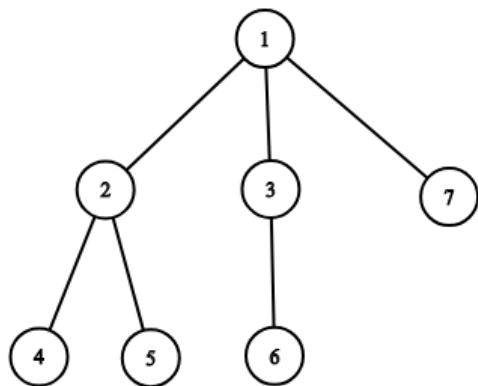
Дерево, соответствующее первому тестовому примеру:





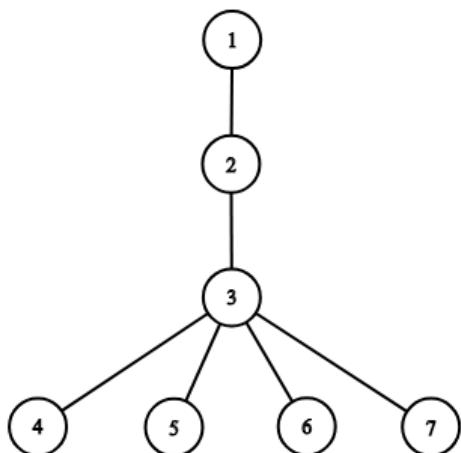
Ответ равен 2, вот несколько возможных ответов:  $[(1, 5), (1, 6)]$ ,  $[(1, 4), (1, 7)]$ ,  $[(1, 6), (1, 7)]$ .

Дерево, соответствующее второму тестовому примеру:



Ответ равен 0.

Дерево, соответствующее третьему тестовому примеру:



Ответ равен 1, единственный способ достичь такого ответа — добавить ребро  $(1, 3)$ .



| [English](#) | [Russian](#)  
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## E1. Игра Подугольник (простая версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Это простая версия задачи. Разница между двумя версиями заключается в ограничениях на переменные. Вы можете делать взломы, только если решены обе версии задачи.**

Цовак и Нарек играют в игру. У них есть массив  $a$  и матрица целых чисел  $b$  с  $n$  строками и  $m$  столбцами, пронумерованных с 1. Ячейка в  $i$ -й строке и  $j$ -м столбце обозначается  $(i, j)$ .

Они ищут элементы  $a$  по очереди; первым начинает Цовак. Каждый раз игрок ищет в матрице клетку, содержащую текущий элемент  $a$  (Цовак ищет первый, Нарек — второй и т.д.). Допустим, игрок выбрал клетку  $(r, c)$ . Следующий игрок должен выбрать свою клетку в подматрице, начинающейся в  $(r + 1, c + 1)$  и заканчивающейся в  $(n, m)$  (подматрица может быть пустой, если  $r = n$  или  $c = m$ ). Если игрок не может найти такую клетку (или оставшаяся подматрица пуста) или массив заканчивается (предыдущий игрок выбрал последний элемент), то он проигрывает.

Ваша задача — определить победителя, если игроки играют оптимально.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. Первая строка содержит одно целое число  $t$  ( $1 \leq t \leq 300$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит три целых числа  $l, n$  и  $m$  ( $1 \leq l, n, m \leq 300$ ) — длина массива и размер матрицы.

Вторая строка содержит  $l$  целых чисел  $a_1, a_2, a_3, \dots, a_l$  ( $1 \leq a_i \leq \min(7, n \cdot m)$ ) — элементы массива  $a$ .

В  $i$ -й из последующих  $n$  строк содержится  $m$  целых чисел  $b_{i,1}, b_{i,2}, b_{i,3}, \dots, b_{i,m}$  ( $1 \leq b_{i,j} \leq \min(7, n \cdot m)$ ), представляющих  $i$ -ю строку матрицы.

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

Гарантируется, что сумма  $l$  по всем наборам входных данных не превосходит 300.

### Выходные данные

Вы должны вывести  $t$  строк,  $i$ -я из которых содержит символ, представляющий ответ на  $i$ -й набор входных данных: «T», если победил Цовак, или «N», в противном случае (без кавычек).

### Пример

входные данные	<a href="#">Скопировать</a>
3 2 2 3 1 2 1 3 5 4 5 2 2 2 4 1 2 1 1 3 2 4 2 5 1 2 4 2 1 2 3 4 5 5 5 5 5 5	
выходные данные	<a href="#">Скопировать</a>
N T N	

### Примечание

В первом наборе входных данных Цовак начинает с поиска 1. Существует только одно вхождение 1 в  $(1, 1)$ , поэтому он выбирает его. Затем Нареку нужно найти 2 в подматрице  $(2, 2)$ , которая состоит только из двух последних элементов: 5 и 2. Он выбирает 2, и Цовак проигрывает, так как массив закончился.

Во втором наборе входных данных Цовак должен выбрать 1. В клетке  $(n, m)$  есть 1, поэтому он выбирает эту клетку. Затем, поскольку подматрица  $(n + 1, m + 1)$  пуста, Нарек не может найти 2, поэтому он проигрывает.



|    
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## E. Минимакс

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

Префикс-функция от строки  $t = t_1 t_2 \dots t_n$  и позиции  $i$  в ней — это длина  $k$  наибольшего собственного (не равного всей подстроке) префикса подстроки  $t_1 t_2 \dots t_i$ , который одновременно является суффиксом этой подстроки.

Например, для строки  $t = abacaba$  значения префикс-функции от позиций  $1, 2, \dots, 7$  равны  $[0, 0, 1, 0, 1, 2, 3]$ .

Введём функцию  $f(t)$ , равную **максимальному** значению префикс-функции строки  $t$  по всем её позициям. Например,  $f(abacaba) = 3$ .

Вам дана строка  $s$ . Переставьте её символы произвольным образом, чтобы получить строку  $t$  (количество вхождений любого символа в строки  $s$  и  $t$  должно совпадать). Значение  $f(t)$  должно быть **минимальным** возможным. Среди всех вариантов минимизировать  $f(t)$  выберите тот, где строка  $t$  лексикографически **минимальна**.

### Входные данные

Во входных данных находятся несколько наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^5$ ) — количество наборов входных данных. Далее следуют наборы входных данных.

Каждый набор входных данных состоит из одной строки  $s$  ( $1 \leq |s| \leq 10^5$ ), состоящей из строчных латинских букв.

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

### Выходные данные

Для каждого набора входных данных выведите одну строку  $t$ .

Мульти множество букв в строках  $s$  и  $t$  должно совпадать. Значение  $f(t)$ , максимума префикс-функции в строке  $t$ , должно быть минимальным возможным. Страна  $t$  должна быть лексикографически минимальной среди всех строк, удовлетворяющих предыдущим условиям.

### Пример

входные данные	Скопировать
3 vkcup abababa zzzzzz	
выходные данные	Скопировать
ckriuv aababab zzzzzz	

### Примечание

Строка  $a$  лексикографически меньше строки  $b$ , если и только если выполняется один из следующих пунктов:

- $a$  — префикс  $b$ , но  $a \neq b$ ;
- в первой позиции, где  $a$  и  $b$  различны, в строке  $a$  находится буква, которая встречается в алфавите раньше, чем соответствующая буква в  $b$ .

В первом наборе входных данных  $f(t) = 0$  и значения префикс-функции равны  $[0, 0, 0, 0, 0]$  для любой перестановки букв.

Строка `ckriuv` является лексикографически минимальной перестановкой букв строки `vkcup`.

Во втором наборе входных данных  $f(t) = 1$ , значения префикс-функции равны  $[0, 1, 0, 1, 0, 1, 0]$ .

В третьем наборе входных данных  $f(t) = 5$ , значения префикс-функции равны  $[0, 1, 2, 3, 4, 5]$ .



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Учимся рисовать

ограничение по времени на тест: 4.5 секунд

ограничение по памяти на тест: 512 мегабайт

*Pristine Beat - Touhou*

Элси учится рисовать. У нее есть холст из  $n$  клеток, пронумерованных от 1 до  $n$ , и она может раскрасить любое (возможно, пустое) подмножество клеток.

1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---

У Элси есть двумерный массив  $a$ , который она будет использовать для оценки картин. Пусть максимальные непрерывные отрезки закрашенных клеток в картине равны  $[l_1, r_1], [l_2, r_2], \dots, [l_x, r_x]$ . Красота картины — это сумма  $a_{l_i, r_i}$  по всем  $1 \leq i \leq x$ . На изображении выше максимальные непрерывные отрезки закрашенных клеток равны  $[2, 4], [6, 6], [8, 9]$ , а красота картины равна  $a_{2,4} + a_{6,6} + a_{8,9}$ .

Всего есть  $2^n$  различных способов раскрасить холст. Помогите Элси найти  $k$  наибольших значений красоты картины, которые она может получить, среди всех способов. Обратите внимание, что эти  $k$  значений не обязательно должны быть различными. Гарантируется, что существует не менее  $k$  различных способов раскрасить холст.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^3$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит 2 целых числа  $n$  и  $k$  ( $1 \leq n \leq 10^3$ ,  $1 \leq k \leq \min(2^n, 5 \cdot 10^3)$ ) — количество клеток и количество наибольших значений красоты картины, которые вы должны найти.

Следующие  $n$  строк каждого набора входных данных описывают двумерный массив  $a$ ,  $i$ -я строка содержит  $n - i + 1$  целых чисел  $a_{i,i}, a_{i,i+1}, \dots, a_{i,n}$  ( $-10^6 \leq a_{i,j} \leq 10^6$ ).

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

### Выходные данные

Для каждого набора входных данных выведите в отдельной строке  $k$  целых чисел:  $i$ -е из них должно равняться  $i$ -му наибольшему значению красоты картины, которое может получить Элси.

### Пример

входные данные	Скопировать
<pre>4 1 2 -5 2 4 2 -3 -1 3 8 2 4 3 1 3 5 6 20 0 -6 -3 0 -6 -2 -7 -5 -2 -3 -4 7 0 -9 -4 2 -1 1 1 -2 -6</pre>	
выходные данные	Скопировать
<pre>0 -5 2 0 -1 -3 7 5 4 3 3 2 1 0 8 8 7 7 5 5 2 2 1 1 1 1 1 0 0 0 0 0 -1</pre>	

### Примечание

В первом наборе входных данных Элси может либо закрасить единственную клетку, либо не закрашивать ее. Если она раскрасит единственную клетку, красота картины составит  $-5$ . Если она решит не раскрашивать ее, красота картины составит

0. Таким образом, наибольшая красота, которую она может получить, равна 0, а вторая по величине красота, которую она может получить, равна  $-5$ .

Ниже приведена иллюстрация третьего набора входных данных.

1	2	3	7
1	2	3	5
1	2	3	4
1	2	3	3
1	2	3	3
1	2	3	2
1	2	3	1
1	2	3	0

---

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:19:54 UTC+5 (h1).

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

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

При поддержке



ИТМО


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

## E. Сделайте это нулем

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам дан массив  $a$ , состоящий из  $n$  целых положительных чисел. Вам разрешено выполнять следующую операцию:

- Выберите массив  $b$  размером  $n$ , такой что выполняются следующие условия:
  - $0 \leq b_i \leq a_i$  для каждого  $1 \leq i \leq n$
  - Существует индекс  $1 \leq i < n$ , такой что  $b_1 + b_2 + \dots + b_i = b_{i+1} + b_{i+2} + \dots + b_n$ . То есть сумма префикса длины  $i$  равна сумме суффикса длины  $n - i$ .
- Затем вычтите  $b_i$  из  $a_i$  для каждого  $1 \leq i \leq n$ .

Ваша задача состоит в том, чтобы сделать все элементы равными 0. Найдите **минимальное** количество операций, необходимых для этого.

Также вам необходимо вывести сами операции. Если невозможно сделать все элементы массива  $a$  равными 0, независимо от количества использованных операций, выведите одно число  $-1$ . Можно показать, что при ограничениях данной задачи минимальное количество необходимых операций не превосходит 17.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит целое число  $n$  ( $2 \leq n \leq 5 \cdot 10^4$ ) — длина массива  $a$ .

Вторая строка каждого набора входных данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^{12}$ ) — массив  $a$ .

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит  $5 \cdot 10^4$ .

### Выходные данные

Для каждого набора входных данных выведите  $-1$ , если решения нет.

В противном случае, сначала выведите целое число  $s$  ( $1 \leq s \leq 17$ ) — минимальное количество операций для изменения всех элементов массива  $a$  на 0.

Затем в следующих  $s$  строках выведите по  $n$  целых чисел  $b_1, b_2, \dots, b_n$  ( $0 \leq b_i \leq a_i$ ), обозначающих массив  $b$  для каждой операции.

После выполнения операций все элементы массива  $a$  должны стать 0.

### Пример

входные данные	<a href="#">Скопировать</a>
<pre>3 3 3 1 2 3 2 2 5 4 5 3 1 5</pre>	
выходные данные	<a href="#">Скопировать</a>
<pre>1 1 2 3 -1 2 3 1 1 1 2 2 0 4</pre>	

### Примечание

В первом наборе входных данных мы можем просто выбрать  $b = a$  в нашей операции. Это допустимо, потому что  $b_1 + b_2 = b_3$ .

Во втором наборе входных данных можно доказать, что невозможно сделать все элементы массива  $a$  равными 0.



|   
[Kap6502](#) | [Выйти](#)

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

## B. Гелифиш и камелия японская

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

У Гелифиша есть массив из  $n$  целых чисел  $c_1, c_2, \dots, c_n$ . Изначально  $c = [a_1, a_2, \dots, a_n]$ .

Гелифиш сделает  $q$  модификаций массива  $c$ .

Для  $i = 1, 2, \dots, q$  Гелифишу даются три целых числа  $x_i, y_i$  и  $z_i$  в диапазоне от 1 до  $n$ . Затем Гелифиш присваивает  $c_{z_i} := \min(c_{x_i}, c_{y_i})$ .

После  $q$  модификаций  $c = [b_1, b_2, \dots, b_n]$ .

Теперь Флауэр знает значение  $b$  и значения целых чисел  $x_i, y_i$  и  $z_i$  для всех  $1 \leq i \leq q$ , но она не знает значение  $a$ .

Флауэр хочет найти любое возможное значение массива  $a$  или сообщить, что такого  $a$  не существует.

Если существует несколько возможных значений массива  $a$ , вы можете вывести любое из них.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа  $n$  и  $q$  ( $1 \leq n, q \leq 3 \cdot 10^5$ ) — размер массива и количество модификаций.

Вторая строка каждого набора входных данных содержит  $n$  целых чисел  $b_1, b_2, \dots, b_n$  ( $1 \leq b_i \leq 10^9$ ) — значение массива  $c$  после  $q$  модификаций.

Следующие  $q$  строк каждая содержат три целых числа  $x_i, y_i$  и  $z_i$  ( $1 \leq x_i, y_i, z_i \leq n$ ) — описание  $i$ -й модификации.

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

### Выходные данные

Для каждого набора входных данных, если искомое  $a$  существует, выведите  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^9$ ) на отдельной строке. В противном случае выведите «-1» на отдельной строке.

Если существует несколько решений, выведите любое из них.

### Пример

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

### Примечание

В первом наборе входных данных, основываясь на данной последовательности модификаций, мы знаем, что  $b_1 = a_1$  и  $b_2 = \min(a_1, a_2)$ . Поэтому необходимо, чтобы  $b_2 \leq b_1$ . Однако для данного  $b$  у нас  $b_1 < b_2$ . Следовательно, решения не существует.

Во втором наборе входных данных мы можем видеть, что данный  $c$  становится  $b$  из  $a$  после заданных модификаций, и  $c$  не изменяется при каждой модификации.

## D. Деревонумерация

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Eikoos и Sushi играют в игру.

Игра проводится на дереве из  $n$  вершин, пронумерованных от 1 до  $n$ . Напомним, что дерево с  $n$  вершинами это неориентированный связный граф с  $n - 1$  ребрами.

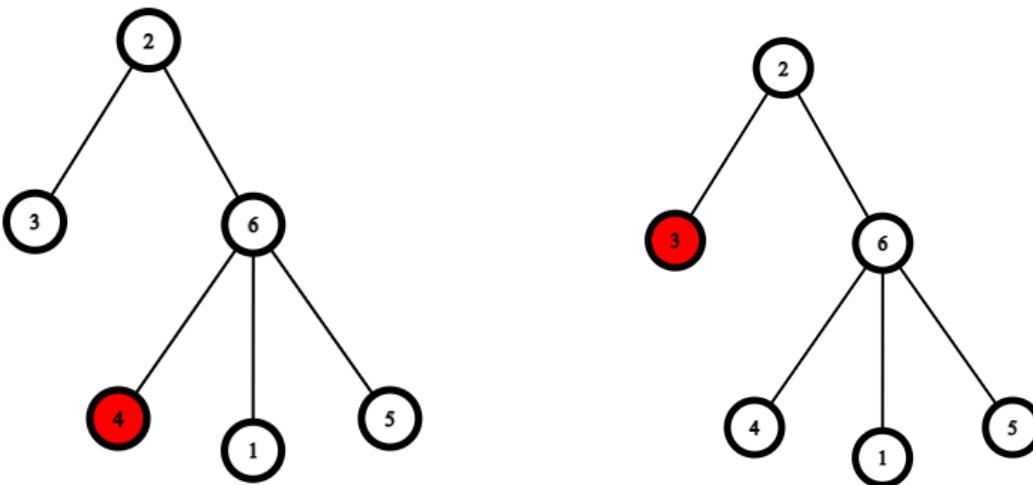
Игроки поочередно перемещают фишку по дереву. **Eikoos делает первый ход, помещая фишку** на любую вершину по своему выбору. Sushi делает следующий ход, затем Eikoos, затем Sushi, и так далее. В каждый ход после первого, игрок должен переместить фишку в какую-то вершину  $u$  такую, что:

- Между вершинами  $u$  и  $v$  (на которой фишка находится данный момент), есть ребро
- $u$  не была посещена ранее
- $u \oplus v \leq \min(u, v)$

Здесь  $x \oplus y$  обозначает операцию [побитового исключающего ИЛИ](#) чисел  $x$  и  $y$ .

Оба игрока играют оптимально. Игрок, который не может сделать ход, проигрывает.

Ниже приведены примеры, демонстрирующие правила игры.



Предположим, Eikoos начинает игру, помещая фишку в вершину 4. Затем Sushi перемещает фишку в вершину 6, которая не была посещена и является соседней к 4. Также  $6 \oplus 4 = 2 \leq \min(6, 4)$ . На следующем ходу Eikoos перемещает фишку в вершину 5, которая не была посещена и является соседней к 6. Также,  $5 \oplus 6 = 3 \leq \min(5, 6)$ . У Sushi больше нет ходов для игры, поэтому она проигрывает.

Перед началом игры Eikoos решает тайком перенумеровать вершины дерева в свою пользу. Формально, перенумерация — это перестановка  $p$  длины  $n$  (последовательность из  $n$  целых чисел, где каждое целое число от 1 до  $n$  встречается ровно один раз), где  $p_i$  обозначает новый номер вершины  $i$ .

Она хочет максимизировать количество вершин, которые она может выбрать в первый ход, для которых она сможет гарантировать себе победу. Помогите Eikoos найти любую перенумерацию, которая поможет ей в этом.

### Входные данные

Первая строка содержит одно целое число  $t$  ( $1 \leq t \leq 10^5$ ) — количество наборов входных данных. Описание каждого набора входных данных выглядит следующим образом.

Предположим, Eikoos начинает игру, помещая фишку в вершину 3. Sushi перемещает фишку в вершину 2, которая не была посещена и является соседней к 3. Также  $3 \oplus 2 = 1 \leq \min(3, 2)$ . Eikoos не может переместить фишку в вершину 6, так как  $6 \oplus 2 = 4 \not\leq \min(6, 2)$ . Поскольку у Eikoos нет ходов для игры, она проигрывает.

Первая строка каждого набора входных данных содержит целое число  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ) — количество вершин в дереве.

Следующие  $n - 1$  строки содержат по два целых числа  $u$  и  $v$  ( $1 \leq u, v \leq n; u \neq v$ ) — обозначающие ребро между вершинами  $u$  и  $v$ .

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

### Выходные данные

Для каждого набора входных данных выведите любую подходящую перестановку — перестановку длины  $n$ , которая максимизирует количество вершин, которые Eikoos может выбрать в первый ход и иметь выигрышную стратегию. Если таких перестановок несколько, вы можете вывести любую из них.

### Пример

Входные данные	Скопировать
3 1 2 1 2 3 1 2 1 3	
Выходные данные	Скопировать
1 2 1 1 2 3	

### Примечание

В первом наборе входных данных у Eikoos есть только одна вершина. У Sushi не будет ходов после того, как Eikoos выберет эту вершину, и Eikoos выиграет.

Во втором наборе входных данных  $1 \oplus 2 = 3 \not\in \min(1, 2)$ . Следовательно, после того, как Eikoos выберет любую из вершин, у Sushi не останется ходов, и Eikoos выиграет. И  $\{1, 2\}$ , и  $\{2, 1\}$  являются допустимыми перенумерациями.

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 08.08.2025 11:20:00<sup>UTC+5</sup> (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО



|   
[Kap6502](#) | [Выйти](#)

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

## F. Магазин лопат

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

В магазине поблизости продаются  $n$  лопат.  $i$ -я лопата стоит  $a_i$  бурлей.

Миша хочет купить **ровно**  $k$  лопат. Ежедневно лопата может быть куплена **не более одного раза**.

Миша может покупать лопаты в несколько покупок. В течение одной покупки он выбирает любое подмножество оставшихся (еще не купленных) лопат и покупает это подмножество.

Также в магазине есть  $m$  специальных предложений.  $j$ -е из них задано парой  $(x_j, y_j)$  и означает, что если Миша купит **ровно**  $x_j$  лопат в **течение одной покупки**, то  $y_j$  **самых дешевых** из них он получит бесплатно (то есть он не будет платить за  $y_j$  самых дешевых лопат в **течение текущей покупки**).

Миша может использовать любое предложение любое (возможно, нулевое) количество раз, но он не может использовать **более одного** предложения в **течение одной покупки** (но он может покупать лопаты без использования каких-либо предложений).

Ваша задача — посчитать минимальную стоимость покупки  $k$  лопат, если Миша будет покупать их оптимальным образом.

### Входные данные

Первая строка входных данных содержит три целых числа  $n, m$  и  $k$  ( $1 \leq n, m \leq 2 \cdot 10^5, 1 \leq k \leq \min(n, 2000)$ ) — количество лопат в магазине, количество специальных предложений и количество лопат, которое Миша хочет купить соответственно.

Вторая строка входных данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 2 \cdot 10^5$ ), где  $a_i$  равно стоимости  $i$ -й лопаты.

Следующие  $m$  строк содержат специальные предложения.  $j$ -е из них задано парой целых чисел  $(x_j, y_j)$  ( $1 \leq y_j \leq x_j \leq n$ ) и означает, что если Миша купит ровно  $x_j$  лопат в **течение одной покупки**, то он сможет взять  $y_j$  **самых дешевых** из них бесплатно.

### Выходные данные

Выведите одно целое число — минимальную стоимость покупки  $k$  лопат, если Миша будет покупать их оптимальным образом.

### Примеры

входные данные	Скопировать
7 4 5 2 5 4 2 6 3 1 2 1 6 5 2 1 3 1	
выходные данные	Скопировать
7	
входные данные	Скопировать
9 4 8 6 8 5 1 8 1 1 2 1 9 2 8 4 5 3 9 7	
выходные данные	Скопировать
17	
входные данные	Скопировать
5 1 4 2 5 7 4 6 5 4	
выходные данные	Скопировать
17	

### Примечание

В первом тестовом примере Миша может купить лопаты на позициях 1 и 4 (обе со стоимостями 2) в течение первой покупки и получить одну из них бесплатно при помощи первого или третьего специального предложения. И затем он может купить лопаты на позициях 3 и 6 (со стоимостями 4 и 3) в течение второй покупки и получить вторую бесплатно при помощи первого или третьего специального предложения. Затем он может купить лопату на позиции 7 со стоимостью 1. Таким образом, общая стоимость равна  $4 + 2 + 1 = 7$ .

Во втором тестовом примере Миша может купить лопаты на позициях 1, 2, 3, 4 и 8 (цены равны 6, 8, 5, 1 и 2) и получить три самые дешевые (со стоимостями 5, 1 и 2) бесплатно. И затем он может купить лопаты на позициях 6, 7 и 9 (все со стоимостями 1) без использования любых специальных предложений. Таким образом, общая стоимость равна  $6 + 8 + 1 + 1 + 1 = 17$ .

В третьем тестовом примере Миша может купить четыре самые дешевые лопаты без использования любых специальных предложений и получить общую стоимость 17.

---

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:20:02 UTC+5 (h1).

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

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

При поддержке



ИТМО



## E. Спидран

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вы играете в компьютерную игру, в которой есть  $n$  заданий, которые вам нужно выполнить. Однако  $j$ -е задание можно выполнить только в начале  $h_j$ -го часа игрового дня. Каждый игровой день длится ровно  $k$  часов. Часы каждого игрового дня пронумерованы числами  $0, 1, \dots, k - 1$ . После конца первого дня начинается следующий, и так далее.

Между заданиями есть зависимости: для некоторых пар  $(a_i, b_i)$  задание с номером  $b_i$  может быть выполнено только после выполнения задания с номером  $a_i$ . Гарантируется, что **циклических зависимостей нет**, поскольку иначе игру пройти было бы невозможно и никто не стал бы в неё играть.

Вы очень хорошо умеете играть в эту игру, поэтому вы можете выполнить любое число заданий за пренебрежимо малое время (т. е. вы можете выполнить любое число заданий в начале одного и того же часа, даже если между ними есть зависимости).

Вам нужно выполнить все задания как можно быстрее. Вы можете выполнять их в любом доступном порядке. Время прохождения игры равно разнице между временем завершения последнего задания и временем завершения первого задания в этом порядке.

Найдите наименьшее время, за которое можно пройти игру.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 100\,000$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит три целых числа  $n, m$  и  $k$  ( $1 \leq n \leq 200\,000, 0 \leq m \leq 200\,000, 1 \leq k \leq 10^9$ ) — количество заданий, количество зависимостей между ними, а также длительность игрового дня в часах.

Следующая строка содержит  $n$  целых чисел  $h_1, h_2, \dots, h_n$  ( $0 \leq h_i < k$ ).

Следующие  $m$  строк описывают зависимости. В  $i$ -й из них содержатся два целых числа  $a_i$  и  $b_i$  ( $1 \leq a_i < b_i \leq n$ ): это означает, что задание с номером  $b_i$  может быть выполнено только после выполнения задания  $a_i$ . Гарантируется, что все зависимости попарно различны.

Гарантируется, что сумма значений  $n$  по всем наборам входных данных не превосходит 200 000.

Гарантируется, что сумма значений  $m$  по всем наборам входных данных не превосходит 200 000.

### Выходные данные

Для каждого набора входных данных выведите одно число — наименьшее время, необходимое для прохождения игры.

### Пример

#### Входные данные

[Скопировать](#)

```
6
4 4 24
12 16 18 12
1 2
1 3
2 4
3 4
4 3 10
2 6 5 9
1 4
2 4
3 4
2 1 10
5 5
1 2
5 0 1000
8 800 555 35 35
5 0 10
3 2 5 4 7
3 2 5
4 3 2
1 2
2 3
```

#### Выходные данные

[Скопировать](#)

```
24
7
```

0
480
5
8

### Примечание

В первом наборе входных данных задания 1 и 4 нужно выполнить в начале 12-го часа дня, но они не могут быть выполнены в течение одного часа, поскольку между ними нужно выполнить задания 2 и 3. Однако всё это можно сделать за 24 часа. Для этого можно начать в 12 часов первого игрового дня, выполнив первое задание. В 16 часов можно выполнить задание 2. В 18 часов можно выполнить задание 3. Наконец, в 12 часов следующего дня можно выполнить задание 4. С момента выполнения первого задания до момента выполнения последнего прошло 24 часа.

В третьем наборе входных данных можно выполнить первое задание, а затем сразу же выполнить второе. Можно начать в 5 часов первого дня, выполнив первое задание. Сразу после этого становится доступным второе задание, его можно выполнять немедленно. Суммарное прошедшее время равно 0.

В четвёртом наборе входных данных можно начать с третьего задания. Можно начать в 555 часов первого дня и закончить в 35 часов второго дня. Суммарное прошедшее время равно  $1035 - 555 = 480$ .

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 08.08.2025 11:20:03<sub>UTC+5</sub> (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО



|    
[Kap6502](#) | [Выходи](#)

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

## F. Красоты Берляндии

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 256 мегабайт

В Берляндии есть  $n$  железнодорожных станций, которые соединены  $n - 1$  участками железной дороги, по которым может ходить поезд в любом из двух направлений. Железнодорожная сеть связана, то есть представляет собой неориентированное дерево.

У вас есть карта этой сети, таким образом для каждого участка железной дороги известно, какие станции он соединяет.

У каждого из  $n - 1$  участков есть целочисленный параметр *красота пейзажа за окном*, однако эти значения на карте не указаны и вам неизвестны. Все эти значения лежат в границах от 1 до  $10^6$ , включительно.

Вы произвели опрос  $m$  пассажиров:  $j$ -й пассажир сообщил три значения:

- станция отправления  $a_j$ ;
- станция прибытия  $b_j$ ;
- минимальную из красот пейзажа за окном на пути следования из  $a_j$  в  $b_j$  (поезд ехал кратчайшим путём из  $a_j$  в  $b_j$ ).

Вы хотите улучшить карту и на каждом участке железной дороги указать величину  $f_i$  — красота пейзажа за окном. Эти значения должны быть согласованы с опросом пассажиров.

Выведите любой возможный набор значений  $f_1, f_2, \dots, f_{n-1}$ , которые согласуются с опросами или укажите, что такой не существует.

### Входные данные

В первой строке входных данных записано целое число  $n$  ( $2 \leq n \leq 5000$ ) — количество железнодорожных станций в Берляндии.

В следующих  $n - 1$  строках записаны описания участков железных дорог:  $i$ -й участок задаётся двумя целыми числами  $x_i$  и  $y_i$  ( $1 \leq x_i, y_i \leq n, x_i \neq y_i$ ), где  $x_i$  и  $y_i$  — номера станций, которые соединены  $i$ -м участком железной дороги. Все участки железной дороги — двунаправлены. Из любой станции можно проехать в любую другую по железной дороге.

Следующая строка содержит число  $m$  ( $1 \leq m \leq 5000$ ) — количество опрошенных пассажиров. Далее содержится  $m$  строк,  $j$ -я строка содержит три целых числа  $a_j, b_j$  и  $g_j$  ( $1 \leq a_j, b_j \leq n; a_j \neq b_j; 1 \leq g_j \leq 10^6$ ) — станция отправления, прибытия и минимальная красота пейзажа за окном во время пути.

### Выходные данные

Если ответа не существует, то выведите единственное число **-1**.

В противном случае выведите  $n - 1$  целое число  $f_1, f_2, \dots, f_{n-1}$  ( $1 \leq f_i \leq 10^6$ ), где  $f_i$  — возможная красота пейзажа за окном для  $i$ -й дороги.

Если существует несколько возможных ответов, вы можете вывести любой из них.

### Примеры

входные данные	Скопировать
4 1 2 3 2 3 4 2 1 2 5 1 3 3	
выходные данные	Скопировать
5 3 5	
входные данные	Скопировать
6 1 2 1 6 3 1 1 5 4 1 4	

```
6 1 3  
3 4 1  
6 5 2  
1 2 5
```

**входные данные****Скопировать**

```
5 3 1 2 1
```

**входные данные****Скопировать**

```
6  
1 2  
1 6  
3 1  
1 5  
4 1  
4  
6 1 1  
3 4 3  
6 5 3  
1 2 4
```

**входные данные****Скопировать**

```
-1
```

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:20:05 UTC+5 (h1).

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

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

При поддержке

**ИТМО**



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Обход яблони

ограничение по времени на тест: 5 секунд

ограничение по памяти на тест: 512 мегабайт

Существует яблоневое дерево с  $n$  вершинами, изначально на каждой вершине находится одно яблоко. У вас есть бумага, на которой изначально ничего не написано.

Вы обходите яблоневое дерево, выполняя следующие действия, пока на ней остаётся хотя бы одно яблоко:

- Выберите **яблочный путь**  $(u, v)$ . Путь  $(u, v)$  называется **яблочным путём**, если и только если на каждой вершине на пути  $(u, v)$  есть яблоко.
- Пусть  $d$  — это количество яблок на пути. Запишите три числа  $(d, u, v)$  в этом порядке на бумаге.
- Затем удалите все яблоки на пути  $(u, v)$ .

Здесь путь  $(u, v)$  обозначает последовательность вершин на единственном кратчайшем пути от  $u$  до  $v$ .

Пусть в результате на бумаге была записана последовательность чисел  $a$ . Ваша задача — найти лексикографически наибольшую возможную последовательность  $a$ .

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит число  $n$  ( $1 \leq n \leq 1.5 \cdot 10^5$ ).

Следующие  $n - 1$  строк каждого набора входных данных содержат два числа  $u, v$  ( $1 \leq u, v \leq n$ ). Гарантируется, что входные рёбра образуют дерево.

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит  $1.5 \cdot 10^5$ .

### Выходные данные

Для каждого набора входных данных выведите лексикографически наибольшую возможную последовательность  $a_1, a_2, \dots, a_{|a|}$ . Можно показать, что  $|a| \leq 3 \cdot n$ .

### Пример

входные данные	Скопировать
6	
4	
1 2	
1 3	
1 4	
4	
2 1	
2 4	
2 3	
5	
1 2	
2 3	
3 4	
4 5	
1	
8	
6 3	
3 5	
5 4	
4 2	
5 1	
1 8	
3 7	
6	
3 2	
2 6	
2 5	
5 4	
4 1	
выходные данные	Скопировать
3 4 3 1 2 2	
3 4 3 1 1 1	

5	5	1						
1	1	1						
5	8	7	2	4	2	1	6	6
5	6	1	1	3	3			

### Примечание

В первом наборе входных данных можно выполнить следующие шаги:

- Выбрать яблочный путь  $(4, 3)$ . Этот путь состоит из вершин 4, 1, 3, и на каждой из них есть яблоко (так что это действительно яблочный путь).  $d = 3$ , так как на этом пути 3 яблока. Добавить 3, 4, 3 в таком порядке в нашу последовательность  $a$ . Далее удалить яблоки с этих 3 вершин.
- Осталось только яблоко на вершине 2. Выбрать яблочный путь  $(2, 2)$ . Этот путь состоит только из одной вершины 2.  $d = 1$ , так как на этом пути 1 яблоко. Добавить 1, 2, 2 в таком порядке в нашу последовательность  $a$  и удалить яблоко с вершины 2.

Таким образом, конечная последовательность будет  $[3, 4, 3, 1, 2, 2]$ . Можно показать, что это лексикографически наибольшая возможная последовательность.

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 08.08.2025 11:20:11 UTC+5 (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке





|   
[Kap6502](#) | [Выйти](#)

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

## F. Кошмар Евклида

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вы должно быть знаете, что Евклид был математиком. Ну, как оказалось, Морфей об этом тоже знал. Поэтому, когда он захотел разыграть Евклида, он послал ему соответствующий кошмар.

В своем плохом сне, у Евклида был набор  $S$  из  $n$   $m$ -мерных векторов над полем  $\mathbb{Z}_2$ , и он мог их складывать. Другими словами, у него были вектора с  $m$  координатами, каждая координата равнялась либо 0, либо 1. Сложение векторов определяется следующим образом: пусть  $u + v = w$ , тогда  $w_i = (u_i + v_i) \bmod 2$ .

Евклид может сложить любое подмножество векторов из  $S$  и сохранить получившийся  $m$ -мерный вектор над  $\mathbb{Z}_2$ . В частности, он может сложить пустое подмножество векторов. В таком случае, все координаты получившегося вектора будут равны 0.

Пусть  $T$  будет множеством всех векторов, которые могут быть получены как сумма некоторого подмножества векторов из  $S$ . Теперь Евклид интересуется, каков размер  $T$ , и может ли он использовать только какое-то подмножество  $S'$  множества  $S$ , чтобы получить все вектора из  $T$ . Как всегда и бывает в таких ситуациях, он не проснется до тех пор, пока не выяснит это. Пока что дела для философа обстоят довольно мрачно. Однако, появилась надежда, когда он заметил, что все вектора из  $S$  содержат **максимум 2** координаты, равные 1.

Помогите Евклиду вычислить  $|T|$ , количество  $m$ -мерных векторов над  $\mathbb{Z}_2$ , которые могут быть получены как сумма какого-то подмножества векторов из  $S$ . Так как оно может быть довольно велико, выведите его по модулю  $10^9 + 7$ . Также, вы должны найти  $S'$ , **минимальное** такое подмножество  $S$ , что все вектора из  $T$  могут быть получены как как сумма какого-то подмножества векторов из  $S'$ . В случае, если существует несколько таких подмножеств с минимальным количеством элементов, выведите лексикографически минимальный по индексам взятых элементов.

Рассмотрим два множества  $A$  и  $B$ , такие что  $|A| = |B|$ . Пусть  $a_1, a_2, \dots, a_{|A|}$  и  $b_1, b_2, \dots, b_{|B|}$  будут возрастающие массивы индексов элементов, взятых в  $A$  и  $B$ , соответственно.  $A$  лексикографически меньше, чем  $B$ , если существует такое  $i$ , что  $a_j = b_j$  для всех  $j < i$  и  $a_i < b_i$ .

### Входные данные

В первой строке даны два целых числа  $n, m$  ( $1 \leq n, m \leq 5 \cdot 10^5$ ), обозначающие количество векторов в  $S$  и количество измерений.

В следующих  $n$  строках дано описание векторов из  $S$ . В каждой из них дано целое число  $k$  ( $1 \leq k \leq 2$ ), а затем  $k$  различных целых чисел  $x_1, \dots, x_k$  ( $1 \leq x_i \leq m$ ). Это обозначает  $m$ -мерный вектор, у которого координаты номер  $x_1, \dots, x_k$  равны 1, а все остальные координаты равны 0.

Среди этих  $n$  векторов нет одинаковых.

### Выходные данные

В первой строке выведите два целых числа: остаток от деления  $|T|$  на  $10^9 + 7$ , и  $|S'|$ . Во второй строке выведите  $|S'|$  целых чисел — индексы элементов, входящих в  $S'$ , в возрастающем порядке. Элементы  $S$  нумеруются с 1 в порядке, в котором они даны во входных данных.

### Примеры

<b>входные данные</b>	<b>Скопировать</b>
3 2 1 1 1 2 2 2 1	
<b>выходные данные</b>	<b>Скопировать</b>
4 2 1 2	
<b>входные данные</b>	<b>Скопировать</b>
2 3 2 1 3 2 1 2	
<b>выходные данные</b>	<b>Скопировать</b>
4 2 1 2	

Входные данные	Скопировать
3 5 2 1 2 1 3 1 4	
Выходные данные	Скопировать
8 3 1 2 3	

### Примечание

В первом примере нам даны три вектора:

- 10
- 01
- 11

Оказывается, что мы можем представить все вектора из 2-мерного пространства используя эти вектора:

- 00, сумма пустого подмножества векторов;
- 01 = 11 + 10, сумма первого и третьего векторов;
- 10 = 10, просто первый вектор;
- 11 = 10 + 01, сумма первого и второго векторов.

Поэтому,  $T = \{00, 01, 10, 11\}$ . Мы можем выбрать любые два из трех векторов  $S$ , и все еще будем способны получить все вектора из  $T$ . В таком случае, мы выберем два первых вектора. Так как мы не можем получить все вектора из  $T$ , используя только один вектор из  $S$ ,  $|S'| = 2$  и  $S' = \{10, 01\}$  (индексы 1 и 2), как множество  $\{1, 2\}$  — лексикографически минимально. Мы можем получить все вектора из  $T$ , используя только вектора из  $S'$ , как показано ниже:

- 00, сумма пустого подмножества;
- 01 = 01, просто второй вектор;
- 10 = 10, просто первый вектор;
- 11 = 10 + 01, сумма первого и второго вектора.

---

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:20:16<sup>UTC+5</sup> (h1).

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

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

При поддержке



ITMO



|   
[Kap6502](#) | [Выйти](#)

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

## F. Wi-Fi

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вы работаете администратором в общежитии, состоящем из  $n$  комнат, расположенных в ряд вдоль коридора. Комнаты пронумерованы слева направо от 1 до  $n$ .

Вы хотите подключить все  $n$  комнат к интернету.

Каждая из комнат может быть подключена к интернету кабелем напрямую, стоимость подключения  $i$ -й комнаты равна  $i$  монет.

В некоторых комнатах также есть место для установки роутера. Стоимость установки роутера в  $i$ -й комнате также равна  $i$  монет. Вы не можете поставить роутер в комнате, в которой нет места для его установки. Когда вы ставите роутер в комнате  $i$ , вы подключаете все комнаты с номерами от  $\max(1, i - k)$  до  $\min(n, i + k)$  включительно к интернету, где  $k$  — это радиус действия роутера. Значение  $k$  одинаково для всех роутеров.

Определите минимальную стоимость подключения всех  $n$  комнат к интернету. Считайте, что количество комнат, в которые можно установить роутеры, не превосходит количества роутеров, которые у вас есть.

### Входные данные

В первой строке следуют два целых числа  $n$  и  $k$  ( $1 \leq n, k \leq 2 \cdot 10^5$ ) — количество комнат и радиус действия каждого роутера.

Во второй строке следует строка  $s$  длины  $n$ , состоящая только из нулей и единиц. Если  $i$ -й символ строки равен единице, то в  $i$ -й комнате есть место для установки роутера. Если  $i$ -й символ строки равен нулю, то в  $i$ -й комнате нет места для установки роутера.

### Выходные данные

Выведите одно целое число — минимальную стоимость подключения всех  $n$  комнат к интернету.

### Примеры

входные данные	выходные данные
5 2 00100	3
входные данные	выходные данные
6 1 000000	21
входные данные	выходные данные
4 1 0011	4
входные данные	выходные данные
12 6 000010000100	15

### Примечание

В первом примере достаточно установить роутер в комнату 3, тогда все комнаты будут подключены к интернету. Стоимость подключения равна 3.

Во втором примере нельзя устанавливать роутеры ни в одну из комнат, поэтому все комнаты нужно подключать напрямую кабелем. Таким образом, стоимость подключения всех комнат равна  $1 + 2 + 3 + 4 + 5 + 6 = 21$ .


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

## D2. Сумма по всем подстрокам (сложная версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

**Это простая версия задачи. Единственное отличие между версиями заключается в ограничениях на  $t$  и  $n$ . Вы можете делать взломы только тогда, когда обе версии задачи решены.**

Для бинарного<sup>†</sup> шаблона  $p$  и бинарной строки  $q$  одинаковой длины  $m$ , скажем, что строка  $q$  называется  $p$ -хорошой, если для каждого  $i$  ( $1 \leq i \leq m$ ) существуют индексы  $l$  и  $r$  такие, что:

- $1 \leq l \leq i \leq r \leq m$ , и
- $p_i$  является модой<sup>‡</sup> строки  $q_l q_{l+1} \dots q_r$ .

Для шаблона  $p$  определим  $f(p)$  как минимально возможное количество 1 в  $p$ -хорошой бинарной строке (такой же длины, как и шаблон).

Вам дана бинарная строка  $s$  длины  $n$ . Найдите

$$\sum_{i=1}^n \sum_{j=i}^n f(s_i s_{i+1} \dots s_j).$$

Другими словами, вам нужно просуммировать значения  $f$  по всем  $\frac{n(n+1)}{2}$  подстрокам  $s$ .

<sup>†</sup> Бинарный шаблон — это строка, состоящая только из символов 0 и 1.

<sup>‡</sup> Символ  $c$  является модой строки  $t$  длины  $m$ , если число вхождений  $c$  в  $t$  не меньше  $\lceil \frac{m}{2} \rceil$ . Например, 0 является модой 010, 1 не является модой 010, и оба символа 0 и 1 являются модами 011010.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^5$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит одно целое число  $n$  ( $1 \leq n \leq 10^6$ ) — длина двоичной строки  $s$ .

Вторая строка каждого набора входных данных содержит бинарную строку  $s$  длины  $n$ , состоящую только из символов 0 и 1.

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит  $10^6$ .

### Выходные данные

Для каждого набора входных данных выведите сумму значений  $f$  по всем подстрокам  $s$ .

### Пример

входные данные	<a href="#">Скопировать</a>
<pre>4 1 1 2 10 5 00000 20 11110110000000111111</pre>	
выходные данные	<a href="#">Скопировать</a>
<pre>1 2 0 346</pre>	

### Примечание

В первом наборе входных данных единственной 1-хорошой строкой является 1. Таким образом,  $f(1) = 1$ .

Во втором наборе входных данных  $f(10) = 1$ , так как 01 является 10-хорошой, а 00 не является 10-хорошой. Таким образом, ответ равен  $f(1) + f(10) + f(0) = 1 + 1 + 0 = 2$ .

В третьем наборе входных данных  $f$  равна 0 для всех  $1 \leq i \leq j \leq 5$ . Таким образом, ответ равен 0.



|   
[Kap6502](#) | [Выйти](#)

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

## G. Звонок во время пути

ограничение по времени на тест: 4 секунды

ограничение по памяти на тест: 256 мегабайт

Вы живёте в городе, состоящем из  $n$  перекрёстков и  $m$  улиц, соединяющих некоторые пары перекрёстков. По каждой улице можно перемещаться в любую сторону. Никакие две улицы не соединяют одну и ту же пару перекрёстков, и никакая улица не соединяет перекрёсток с самим собой. Из любого перекрёстка можно добраться до любого другого, возможно, проходя через какие-то другие перекрёстки.

Каждую минуту вы можете сесть на автобус на перекрёстке  $u_i$  и доехать за  $l_{i1}$  минут до перекрёстка  $v_i$ . И наоборот доехать от перекрёстка  $v_i$  до перекрёстка  $u_i$  за  $l_{i2}$  минут. Садиться в автобус и выходить из него можно только на перекрёстках. Сесть в автобус на перекрёстке можно только находясь в нём.

Также по каждой улице можно пройти пешком за  $l_{i2} > l_{i1}$  минут.

Вы можете делать остановки на перекрёстках.

Вы живёте на перекрёстке с номером 1. Сегодня вы проснулись в момент времени 0, и у вас запланировано важное мероприятие на перекрёстке с номером  $n$ , на которое вы должны добраться не позднее момента времени  $t_0$ . Также у вас запланирован телефонный звонок, который продлится с  $t_1$  по  $t_2$  минуту ( $t_1 < t_2 < t_0$ ).

Во время телефонного разговора нельзя ехать на автобусе, но можно идти по любым улицам пешком, сделать остановку или находиться у себя дома. Вы можете выходить из автобуса в минуту  $t_1$  и заходить в автобус в минуту  $t_2$ .

Так как вы хотите выспаться, вам стало интересно — как поздно вы можете выйти из дома, чтобы успеть поговорить по телефону и при этом не опоздать на мероприятие?

### Входные данные

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

Первая строка описания каждого набора входных данных содержит два целых числа  $n, m$  ( $2 \leq n \leq 10^5, 1 \leq m \leq 10^5$ ) — количество перекрёстков и улиц в городе.

Вторая строка описания каждого набора входных данных содержит три целых числа  $t_0, t_1, t_2$  ( $1 \leq t_1 < t_2 < t_0 \leq 10^9$ ) — время начала мероприятия, время начала телефонного звонка и время его конца, соответственно.

Следующие  $m$  строк каждого набора данных содержат описания улиц.

$i$ -я строка содержит четыре целых числа  $u_i, v_i, l_{i1}, l_{i2}$  ( $1 \leq u_i, v_i \leq n, u_i \neq v_i, 1 \leq l_{i1} < l_{i2} \leq 10^9$ ) — номера перекрёстков, соединяемых  $i$ -й улицей, а также время пути по улице на автобусе и пешком. Гарантируется, что никакие две улицы не соединяют одну и ту же пару перекрёстков, а также что из любого перекрёстка можно добраться до любого другого.

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

### Выходные данные

Для каждого набора входных данных выведите одно целое число — максимальный момент времени, когда вы можете выйти из дома, чтобы успеть поговорить по телефону и не опоздать на мероприятие. Если вы не можете попасть на мероприятие вовремя, выведите -1.

### Пример

входные данные	Скопировать
<pre>7 5 5 100 20 80 1 5 30 100 1 2 20 50 2 3 20 50 3 4 20 50 4 5 20 50 2 1 100 50 60 1 2 55 110 4 4 100 40 60 1 2 30 100</pre>	

```
2 4 30 100
1 3 20 50
3 4 20 50
3 3
100 80 90
1 2 1 10
2 3 10 50
1 3 20 21
3 2
58 55 57
2 1 1 3
2 3 3 4
2 1
12 9 10
2 1 6 10
5 5
8 5 6
2 1 1 8
2 3 4 8
4 2 2 4
5 3 3 4
4 5 2 6
```

**Выходные данные****Скопировать**

```
0
-1
60
80
53
3
2
```

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:20:22<sub>UTC+5</sub> (h1).

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

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

При поддержке



ITMO



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## F. Нестабильная сортировка строки

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Авторы загадали строку  $s$ , состоящую из  $n$  строчных букв латинского алфавита.

Вам задано две перестановки ее индексов (не обязательно одинаковых)  $p$  и  $q$  (обе длины  $n$ ). Напомним, что перестановкой называется массив длины  $n$ , содержащий каждое целое число от 1 до  $n$  ровно один раз.

Для всех  $i$  от 1 до  $n - 1$  выполняются следующие свойства:  $s[p_i] \leq s[p_{i+1}]$  и  $s[q_i] \leq s[q_{i+1}]$ . Это значит, что если вы выпишете все символы  $s$  в порядке индексов в перестановке, результирующая строка получится отсортированной в неубывающем порядке.

Ваша задача — восстановить **любую** строку  $s$  длины  $n$ , состоящую **хотя бы из  $k$  различных строчных букв латинского алфавита**, которая подходит к заданным перестановкам.

Если существует несколько возможных ответов, вы можете вывести любой.

### Входные данные

Первая строка входных данных содержит два целых числа  $n$  и  $k$  ( $1 \leq n \leq 2 \cdot 10^5$ ,  $1 \leq k \leq 26$ ) — длину строки и необходимое количество различных символов.

Вторая строка входных данных содержит  $n$  целых чисел  $p_1, p_2, \dots, p_n$  ( $1 \leq p_i \leq n$ , все  $p_i$  являются различными целыми числами от 1 до  $n$ ) — перестановку  $p$ .

Третья строка входных данных содержит  $n$  целых чисел  $q_1, q_2, \dots, q_n$  ( $1 \leq q_i \leq n$ , все  $q_i$  являются различными целыми числами от 1 до  $n$ ) — перестановку  $q$ .

### Выходные данные

Если невозможно найти подходящую строку, выведите «NO» в первой строке.

Иначе выведите «YES» в первой строке и строку  $s$  во второй строке. Она должна состоять ровно из  $n$  строчных букв латинского алфавита, содержать хотя бы  $k$  различных символов и подходить заданным перестановкам.

Если существует несколько возможных ответов, вы можете вывести любой.

### Пример

Входные данные	<button>Скопировать</button>
3 2 1 2 3 1 3 2	
Выходные данные	<button>Скопировать</button>
YES abb	

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:20:24 UTC+5 (h1).

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

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

При поддержке



ITMO



| [English](#) [Russian](#)  
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## C2. Шейх (сложная версия)

ограничение по времени на тест: 4 секунды

ограничение по памяти на тест: 256 мегабайт

Это сложная версия задачи. Единственное различие состоит в том, что в этой версии  $q = n$ .

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

Стоимостью подотрезка массива  $[l, r]$ ,  $1 \leq l \leq r \leq n$ , назовем величину  $f(l, r) = \text{sum}(l, r) - \text{xor}(l, r)$ , где  $\text{sum}(l, r) = a_l + a_{l+1} + \dots + a_r$ , а  $\text{xor}(l, r) = a_l \oplus a_{l+1} \oplus \dots \oplus a_r$  ( $\oplus$  обозначает побитовое исключающее ИЛИ).

У вас будет  $q$  запросов. Каждый запрос задается парой чисел  $L_i, R_i$ , где  $1 \leq L_i \leq R_i \leq n$ . Вам нужно найти подотрезок  $[l, r]$ ,  $L_i \leq l \leq r \leq R_i$  с максимальным значением  $f(l, r)$ . Если ответов несколько, то среди них нужно найти подотрезок с минимальной длиной, то есть минимальным значением  $r - l + 1$ .

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. Первая строка содержит целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа  $n$  и  $q$  ( $1 \leq n \leq 10^5$ ,  $q = n$ ) — длину массива и количество запросов.

Вторая строка каждого набора входных данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^9$ ) — элементы массива.

$i$ -я из следующих  $q$  строк каждого набора входных данных содержит два целых числа  $L_i$  и  $R_i$  ( $1 \leq L_i \leq R_i \leq n$ ) — границы, в которых нужно найти отрезок.

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

Гарантируется, что  $L_1 = 1$  и  $R_1 = n$ .

### Выходные данные

Для каждого набора входных данных выведите  $q$  пар чисел  $L_i \leq l \leq r \leq R_i$ , таких что значение  $f(l, r)$  максимально, а среди таких длина  $r - l + 1$  минимальна. Если существует несколько правильных ответов, выведите любой из них.

### Пример

входные данные	Скопировать
6 1 1 0 1 1 2 2 5 10 1 2 2 2 3 3 0 2 4 1 3 1 2 2 3 4 4 0 12 8 3 1 4 1 3 2 4 2 3 5 5 21 32 32 32 10 1 5 1 4 1 3 2 5 3 5 7 7 0 1 0 1 0 1 0 1 7 3 6 2 5 1 4 4 7	

```
2 6  
2 7
```

**Входные данные****Скопировать**

```
1 1  
1 1  
2 2  
1 1  
1 1  
2 2  
2 3  
2 3  
2 3  
2 3  
2 3  
2 3  
2 3  
2 3  
2 3  
3 4  
2 4  
4 6  
2 4  
2 4  
4 6  
2 4  
2 4
```

**Примечание**

Во всех наборах входных данных рассматривается первый запрос.

В первом наборе входных данных  $f(1, 1) = 0 - 0 = 0$ .

Во втором наборе входных данных  $f(1, 1) = 5 - 5 = 0$ ,  $f(2, 2) = 10 - 10 = 0$ . Заметим, что  $f(1, 2) = (10 + 5) - (10 \oplus 5) = 0$ , но нам среди максимальных значений  $f(l, r)$  нужно найти подотрезок с минимальной длиной. Поэтому правильными ответами будут только отрезки  $[1, 1]$  и  $[2, 2]$ .

В четвертом наборе входных данных  $f(2, 3) = (12 + 8) - (12 \oplus 8) = 16$ .

В пятом наборе входных данных есть два правильных ответа, так как  $f(2, 3) = f(3, 4)$  и их длины равны.

---

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:20:26<sup>UTC+5</sup> (h1).

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

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

При поддержке



ITMO

## E. Раскраски и домино

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 512 мегабайт

У вас есть большая прямоугольная доска, разделенная на  $n \times m$  клетка (у доски  $n$  строк и  $m$  столбцов). Каждая клетка — либо белая, либо черная.

Вы красите каждую белую клетку либо в красный цвет, либо в синий. Очевидно, количество различных вариантов раскраски доски равно  $2^w$ , где  $w$  — количество белых клеток.

После перекрашивания всех белых клеток вы начинаете кладь на доску доминошки по следующим правилам:

- каждая доминошка накрывает две соседние клетки;
- каждая клетка накрыта не более чем одной доминошкой;
- если доминошка расположена горизонтально (она накрывает две соседние клетки в строке), обе накрытые ею клетки должны быть красными;
- если доминошка расположена вертикально (она накрывает две соседние клетки в столбце), обе накрытые ею клетки должны быть синими.

Пусть **ценность** раскраски доски — максимальное количество доминошек, которое можно расположить. Посчитайте сумму ценностей по всем  $2^w$  способам раскрасить доску. Так как сумма может быть очень большой, выведите ее по модулю 998 244 353.

### Входные данные

В первой строке заданы два целых числа  $n$  и  $m$  ( $1 \leq n, m \leq 3 \cdot 10^5$ ;  $nm \leq 3 \cdot 10^5$ ) — количество строк и столбцов, соответственно.

Затем следуют  $n$  строк, в каждой из которых задана последовательность из  $m$  символов. Символ под номером  $j$  в  $i$ -й последовательности — \*, если  $j$ -я клетка  $i$ -й строки черная; иначе этот символ — 0.

### Выходные данные

Выведите одно целое число — сумму ценностей по всем  $2^w$  способам раскрасить доску, взятую по модулю 998 244 353.

### Примеры

<b>входные данные</b>	<input type="button" value="Скопировать"/>
3 4 **00 00*0 **00	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
144	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
3 4 **00 00** **00	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
48	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
2 2 00 0*	
<b>выходные данные</b>	<input type="button" value="Скопировать"/>
4	
<b>входные данные</b>	<input type="button" value="Скопировать"/>
1 4 0000	



| [English](#) | [Russian](#)  
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## B. Дождь

ограничение по времени на тест: 4 секунды

ограничение по памяти на тест: 256 мегабайт

Вы владеете полем, который можно представить как бесконечную числовую прямую, все позиции на которой отмечены целыми числами.

В следующие  $n$  дней будет идти дождь. В  $i$ -й день центр дождя будет располагаться в точке  $x_i$ , а интенсивность дождя будет равна  $p_i$ . Из-за дождей будет собираться дождевая вода; пусть  $a_j$  будет обозначать объем воды, накопившийся в точке  $j$ . Изначально  $a_j$  равно 0, после дождя в  $i$ -й день это значение увеличится на  $\max(0, p_i - |x_i - j|)$ .

Поле затопит, если в некоторый момент будет существовать точка  $j$ , в которой накопится  $a_j > m$  воды.

Вы можете использовать заклинание и отменить дождь в **ровно один** из дней, т. е. установить  $p_i = 0$ . Для каждого  $i$  от 1 до  $n$  определите, если вы отмените дождь в  $i$ -й день, верно ли, что затопления не случится?

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 10^4$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка содержит два целых числа  $n$  и  $m$  ( $1 \leq n \leq 2 \cdot 10^5$ ,  $1 \leq m \leq 10^9$ ) — количество дней и максимальный объем воды, который может находиться в одной точке, не вызывая затопления.

Далее следуют  $n$  строк.  $i$ -я из этих строк содержит два целых числа  $x_i$  и  $p_i$  ( $1 \leq x_i, p_i \leq 10^9$ ) — положение центра и интенсивность дождя в  $i$ -й день.

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

### Выходные данные

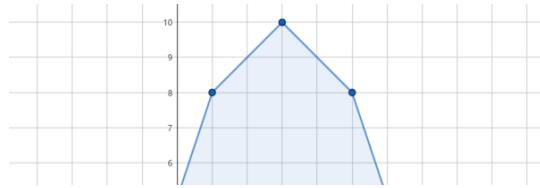
Для каждого набора входных данных выведите бинарную строку  $s$  длины  $n$ .  $i$ -й символ строки  $s$  должен быть равен 1, если в случае отмены дождя в  $i$ -й день затопления **не** случится, и 0, если в случае отмены дождя в  $i$ -й день затопление произойдет.

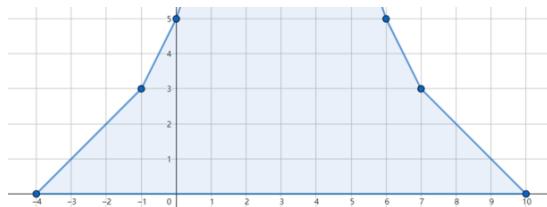
### Пример

входные данные	Скопировать
4	
3 6	
1 5	
5 5	
3 4	
2 3	
1 3	
5 2	
2 5	
1 6	
10 6	
6 12	
4 5	
1 6	
12 5	
5 5	
9 7	
8 3	
выходные данные	Скопировать
001	
11	
00	
100110	

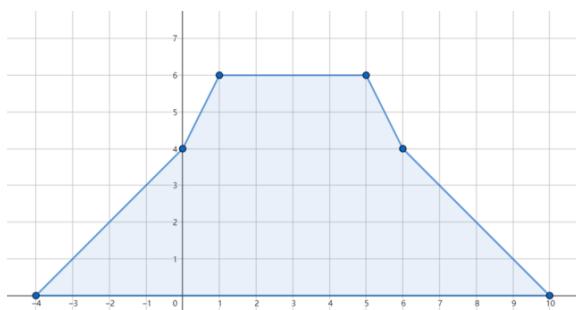
### Примечание

В первом наборе входных данных, если не использовать заклинание, собравшаяся вода будет выглядеть следующим образом:





Если отменить дождь в третий день, затопление будет предотвращено, и собравшаяся вода будет выглядеть следующим образом:



Во втором примере изначально нет затопления, поэтому можно отменить дождь в любой день.

В третьем примере нет способа избежать затопления.

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 08.08.2025 11:20:29 UTC+5 (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО

### C. Вам задана WASD-строка...

ограничение по времени на тест: 2 секунды

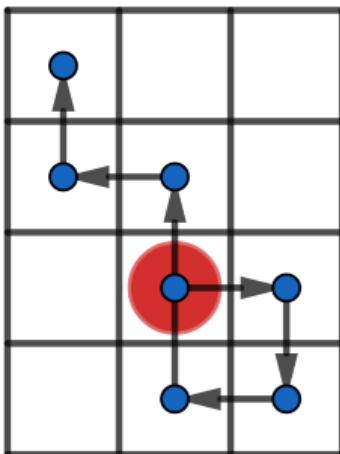
ограничение по памяти на тест: 256 мегабайт

У вас есть строка  $s$  — последовательность команд для робота. Робот находится в одной из клеток клетчатого поля. Он может выполнять следующие команды:

- 'W' — переместиться на одну клетку вверх;
- 'S' — переместиться на одну клетку вниз;
- 'A' — переместиться на одну клетку влево;
- 'D' — переместиться на одну клетку вправо.

$Grid(s)$  — прямоугольное поле минимальной площади, такое, что на нем можно выбрать стартовую позицию робота так, что при выполнении всей последовательности команд  $s$  робот не выйдет за пределы прямоугольника. Например, если  $s = \text{DSAWWAW}$ , то  $Grid(s)$  — это прямоугольник  $4 \times 3$ .

1. вы можете поместить робота в клетку  $(3, 2)$ ;
2. робот выполняет команду 'D' и перемещается в  $(3, 3)$ ;
3. робот выполняет команду 'S' и перемещается в  $(4, 3)$ ;
4. робот выполняет команду 'A' и перемещается в  $(4, 2)$ ;
5. робот выполняет команду 'W' и перемещается в  $(3, 2)$ ;
6. робот выполняет команду 'W' и перемещается в  $(2, 2)$ ;
7. робот выполняет команду 'A' и перемещается в  $(2, 1)$ ;
8. робот выполняет команду 'W' и перемещается в  $(1, 1)$ .



У вас есть 4 дополнительных буквы: одна 'W', одна 'A', одна 'S' и одна 'D'. Вы можете вставить одну из них (либо не вставлять вообще) в любую позицию в строке  $s$  для минимизации площади  $Grid(s)$ .

Какую минимальную площадь  $Grid(s)$  вы можете получить?

#### Входные данные

Первая строка содержит число  $T$  ( $1 \leq T \leq 1000$ ) — количество запросов.

Следующие  $T$  строк содержат запросы. Каждый запрос содержит строку  $s$  ( $1 \leq |s| \leq 2 \cdot 10^5$ ,  $s_i \in \{\text{W}, \text{A}, \text{S}, \text{D}\}$ ) — последовательность команд.

Гарантируется, что суммарная длина строк  $s$  по всем запросам не превосходит  $2 \cdot 10^5$ .

#### Выходные данные

Выведите  $T$  строк, по одному числу в каждой строке.

На каждый запрос выведите минимальное значение  $Grid(s)$ , которое вы можете получить.

#### Пример

входные данные

Скопировать

3  
DSAWWAW  
D  
WA

**ВЫХОДНЫЕ ДАННЫЕ**

8  
2  
4

[Скопировать](#)

**Примечание**

В первом запросе вам нужно получить строку DSAWWDAW.

Во втором и третьем запросах вы не можете уменьшить площадь  $Grid(s)$ .

---

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

Соревнования по программированию 2.0

Время на сервере: 08.08.2025 11:20:35<sub>UTC+5</sub> (h1).

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

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

При поддержке



ИТМО

# ← Технокубок

|    
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Грайм Зоопарк

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Сейчас рэп ХХОСа представляет из себя строку из нулей, единиц и знаков вопроса. К сожалению, хейтеры не дремлют. За каждое вхождение **подпоследовательности 01** в рэп ХХОСа хейтеры напишут  $x$  гневных комментариев, а за каждое вхождение **подпоследовательности 10** будет написано  $y$  гневных комментариев. Вы должны заменить каждый знак вопроса на 0 либо 1, чтобы минимизировать число гневных комментариев, которые получит ХХОС.

Подпоследовательностью строки  $a$  называется строка  $b$ , которая может получиться в результате удаления нескольких символов из строки  $a$ . Два вхождения подпоследовательности считаются разными, если различаются множества позиций оставленных символов.

### Входные данные

В первой строке записан рэп ХХОСа — строка  $s$  ( $1 \leq |s| \leq 10^5$ ). Во второй строке даны два целых числа  $x$  и  $y$  — количество гневных комментариев, которые ХХОС получит за каждую подпоследовательность 01 и 10, соответственно ( $0 \leq x, y \leq 10^6$ ).

### Выходные данные

В единственной строке выведите минимальное число гневных комментариев, которые может получить ХХОС.

### Примеры

входные данные	Скопировать
0?1 2 3	
выходные данные	Скопировать
4	
входные данные	Скопировать
????? 13 37	
выходные данные	Скопировать
0	
входные данные	Скопировать
?10? 239 7	
выходные данные	Скопировать
28	
входные данные	Скопировать
01101001 5 7	
выходные данные	Скопировать
96	

### Примечание

В первом примере одним из оптимальных вариантов замены является 001. Тогда в строке будет 2 подпоследовательности 01 и 0 подпоследовательностей 10. Суммарное количество гневных комментариев равно  $2 \cdot 2 + 0 \cdot 3 = 4$ .

Во втором примере одним из оптимальных вариантов замены является 11111. Тогда в строке будет 0 подпоследовательностей 01 и 0 подпоследовательностей 10. Суммарное количество гневных комментариев равно  $0 \cdot 13 + 0 \cdot 37 = 0$ .

В третьем примере одним из оптимальных вариантов замены является 1100. Тогда в строке будет 0 подпоследовательностей 01 и 4 подпоследовательности 10. Суммарное количество гневных комментариев равно  $0 \cdot 239 + 4 \cdot 7 = 28$ .

В четвёртом примере одним из оптимальных вариантов замены является 01101001. Тогда в строке будет 8 подпоследовательностей 01 и 8 подпоследовательностей 10. Суммарное количество гневных комментариев равно  $8 \cdot 5 + 8 \cdot 7 = 96$ .



|   
[Kap6502](#) | [Выходи](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## B2. Конфетная вечеринка (сложная версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Это **сложная версия задачи**. Единственное отличие состоит в том, что в этой версии каждый должен дать конфеты **не более чем одному** человеку и получить конфеты **не более чем от одного** человека. Обратите внимание, что посылка не может пройти обе версии задачи одновременно. Вы можете делать взломы, только если обе версии задачи решены.

После экзамена Daniel и его друзья собираются устроить вечеринку. Все придут с конфетами.

На вечеринке будут присутствовать  $n$  человек. Изначально у  $i$ -го человека есть  $a_i$  конфет. Во время вечеринки они будут обмениваться конфетами. Для этого они выстроются в **произвольном порядке** и каждый из них сделает следующее **не более одного раза**:

- Выберет целое число  $p$  ( $1 \leq p \leq n$ ) и **неотрицательное** целое число  $x$ , затем даст  $2^x$  конфет  $p$ -му человеку. Заметим, что человек **не может** отдать больше конфет, чем у него есть в данный момент (он мог получить конфеты от кого-то другого), и он **не может** отдать конфеты самому себе.

Daniel любит справедливость, поэтому он будет счастлив тогда и только тогда, когда каждый получит конфеты от **не более чем одного человека**. В то же время его друг Том любит среднее, поэтому он будет счастлив тогда и только тогда, когда у всех людей будет одинаковое количество конфет после всех обменов.

Определите, существует ли способ обмениваться конфетами так, чтобы Daniel и Tom были счастливы после обменов.

### Входные данные

Первая строка входных данных содержит одно 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 человек на вечеринке.

Вторая строка каждого набора входных данных содержит  $n$  целых чисел  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ) — количество конфет у каждого человека.

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

### Выходные данные

Для каждого набора входных данных выведите «Yes» (без кавычек), если существует способ обмениваться конфетами так, чтобы Daniel и Tom были счастливы, и выведите «No» (без кавычек) в противном случае.

Вы можете вывести ответ в любом регистре (верхнем или нижнем). Например, строки «yEs», «yes», «Yes», и «YES» будут распознаны как положительный ответ.

### Пример

входные данные	<a href="#">Скопировать</a>
6	
3	
2 4 3	
5	
1 2 3 4 5	
6	
1 4 7 1 5 4	
2	
20092043 20092043	
12	
9 9 8 2 4 4 3 5 1 1 1 1	
6	
2 12 7 16 11 12	
выходные данные	<a href="#">Скопировать</a>
Yes	
Yes	
No	
Yes	
No	
Yes	

**Примечание**

В первом наборе входных данных второй человек даёт 1 конфету первому человеку, поэтому у всех людей есть по 3 конфеты.

Во втором наборе входных данных четвёртый человек даёт 1 конфету второму человеку, пятый человек даёт 2 конфеты первому человеку, третий человек ничего не делает. После всех обменов у каждого есть по 3 конфеты.

В третьем наборе входных данных невозможно сделать так, чтобы у всех людей было одинаковое количество конфет.

В четвёртом наборе входных данных двум людям не нужно ничего делать.

---

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов  
Соревнования по программированию 2.0  
Время на сервере: 08.08.2025 11:20:39<sub>UTC+5</sub> (h1).  
Мобильная версия, переключиться на [десктопную](#).  
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



| ИТМО

## E. Переставь скобки

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Правильная скобочная последовательность — это скобочная последовательность, которую можно превратить в корректное арифметическое выражение, вставив символы «1» и «+» между исходными символами. Например:

- скобочные последовательности «( ) ( )» и «(( ))» — правильные (из них можно получить выражения «(1)+(1)» и «((1+1)+1)»);
- скобочные последовательности « ) ( », «( » и « ) » — неправильные.

Дана правильная скобочная последовательность. За один ход вы можете удалить пару **соседних** скобок такую, что левая скобка открывающая, а правая — закрывающая. Затем склеить полученные части, не изменяя порядка. Стоимость такого хода равна количеству скобок справа от правой скобки из этой пары.

*Стоимость* правильной скобочной последовательности равна наименьшей суммарной стоимости ходов, необходимых, чтобы сделать последовательность пустой.

На самом деле, никаких скобок вы не удаляете. Вместо этого вам дана правильная скобочная последовательность и целое число  $k$ . Вы можете проделать следующее действие **не больше  $k$  раз**:

- вытащить скобку из последовательности и вставить ее в любую позицию (между двух скобок, в начало или в конец; возможно, туда же, где она и была).

После всех действий скобочная последовательность должна быть правильной. Какая наименьшая *стоимость* полученной правильной скобочной последовательности?

### 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  $k$  ( $0 \leq k \leq 5$ ) — наибольшее количество действий, которые вы можете совершить.

Во второй строке записана непустая скобочная последовательность, она состоит только из символов '(' и ')'.

Суммарная длина скобочных последовательностей по всем наборам входных данных не превосходит  $2 \cdot 10^5$ .

### Выходные данные

На каждый набор входных данных выведите одно целое число — наименьшая *стоимость* правильной скобочной последовательности после того, как вы проделаете над ней не более  $k$  действий.

### Пример

входные данные	Скопировать
7 0 ( 0 (( 1 ()) 5 ( 1 ((())((()) 2 ((())((())((()) 3 ((())((())((())	
выходные данные	Скопировать
0 1 0 0 1 4 2	



|   
[Kap6502](#) | [Выйти](#)

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

## D. Путь домой

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 512 мегабайт

Известный фокусник Боря Будини путешествовал по стране  $X$ , которая состоит из  $n$  городов. Однако случилось несчастье, и его обокрали в городе номер 1. Теперь Будини предстоит нелегкий путь домой в город  $n$ .

Добраться он собирается самолетами. Всего в стране есть  $m$  авиарейсов,  $i$ -й летит из города  $a_i$  в город  $b_i$  и стоит  $s_i$  монет. Обратите внимание, что  $i$ -й рейс односторонний, то есть не может использоваться, чтобы добраться из города  $b_i$  в город  $a_i$ . Чтобы им воспользоваться, Боря должен быть в городе  $a_i$  и иметь на руках хотя бы  $s_i$  монет (которые он потратит на перелет).

После ограбления у него осталось всего  $p$  монет, однако он не отчаивается! Находясь в городе  $i$ , он может хоть каждый день организовывать представления, которые будут приносить ему по  $w_i$  монет каждый.

Помогите фокуснику узнать, сможет ли он добраться до дома, а также какое минимальное количество представлений придется для этого организовать.

### Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число  $t$  ( $1 \leq t \leq 80$ ) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных содержит три целых числа  $n$ ,  $m$  и  $p$  ( $2 \leq n \leq 800$ ,  $1 \leq m \leq 3000$ ,  $0 \leq p \leq 10^9$ ) — количество городов, количество авиарейсов и изначальное количество монет.

Во второй строке каждого набора входных даны  $n$  целых чисел  $w_1, w_2, \dots, w_n$  ( $1 \leq w_i \leq 10^9$ ) — прибыль от представлений.

В следующих  $m$  строках даны по три целых числа  $a_i$ ,  $b_i$  и  $s_i$  ( $1 \leq a_i, b_i \leq n$ ,  $1 \leq s_i \leq 10^9$ ) — начальный и конечный город, а также стоимость  $i$ -го авиарейса.

Гарантируется, что сумма  $n$  по всем наборам входных данных не превосходит 800, а также, что сумма  $m$  по всем наборам входных данных не превосходит 10000.

### Выходные данные

Для каждого теста выведите единственное целое число — минимальное количество представлений, которое придется организовать Боре, чтобы добраться до дома, или  $-1$ , если это сделать невозможно.

### Пример

входные данные	Скопировать
4	
4 4 2	
7 4 3 1	
1 2 21	
3 2 6	
1 3 8	
2 4 11	
4 4 10	
1 2 10 1	
1 2 20	
2 4 30	
1 3 25	
3 4 89	
4 4 7	
5 1 6 2	
1 2 5	
2 3 10	
3 4 50	
3 4 70	
4 1 2	
1 1 1 1	
1 3 2	
выходные данные	Скопировать
4	
24	
10	
-1	

### Примечание

В первом примере Боре оптимально сделать 4 представления в первом городе, имея в итоге  $2 + 7 \cdot 4 = 30$  рублей, а потом