

C. Восстанови ПСП

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Скобочная последовательность — это строка, содержащая только символы «« и »»». Правильная скобочная последовательность (или, коротко говоря, ПСП) — это скобочная последовательность, которая может быть преобразована в правильное арифметическое выражение путем вставки символов «1» и «+» между исходными символами последовательности. Например:

- скобочные последовательности «() ()» и «(())» являются правильными (возможные выражения: «(1)+(1)» и «((1+1)+1)»);
- скобочные последовательности «) (», «(» и «)» не являются правильными.

В начале была некоторая ПСП. Некоторые скобки заменили на знаки вопроса. Верно ли, что существует **единственный** способ заменить знаки вопроса на скобки так, чтобы получилась ПСП?

Входные данные

В первой строке записано одно целое число t ($1 \leq t \leq 5 \cdot 10^4$) — количество наборов входных данных.

В единственной строке каждого набора входных данных записана ПСП с некоторыми скобками замененными на знаки вопроса. Каждый символ — это '(', ')' или '?'. Из данной последовательности можно восстановить хотя бы одну ПСП.

Суммарная длина последовательностей по всем наборам входных данных не превосходит $2 \cdot 10^5$.

Выходные данные

На каждый набор входных данных выведите «YES», если способ заменить знаки вопроса на скобки так, чтобы получилась ПСП, **единственный**. Если существует больше одного способа, то выведите «NO».

Пример

входные данные	Скопировать
5 (?)) ?????? () ?? ?(?)()?)	
выходные данные	Скопировать
YES NO YES YES NO	

Примечание

В первом наборе входных данных единственная возможная оригинальная ПСП — это «()».

Во втором наборе существует несколько способов восстановить ПСП.

В третьем и четвертом наборах единственная возможная ПСП — это «()».

В пятом наборе оригинальная ПСП может быть «((()))» или «(() ())».



A. Dreamoon любит красить

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Dreamoon очень любит красить клетки.

Есть ряд из n клеток. Исходно, все клетки пустые (не содержат никакой цвет). Клетки пронумерованы от 1 до n .

Вам дано целое число m и m целых чисел l_1, l_2, \dots, l_m ($1 \leq l_i \leq n$)

Dreamoon совершил m операций.

В i -й операции, Dreamoon выберет число p_i из отрезка $[1, n - l_i + 1]$ (включительно) и покрасит все клетки от p_i до $p_i + l_i - 1$ (включительно) в i -й цвет. Обратите внимание, что клетки могут быть покрашены несколько раз, в таком случае, клетка будет покрашена в цвет из самой последней операции.

Dreamoon надеется, что после m операций, все цвета будут встречаться хотя бы один раз и все клетки будут покрашены.
Пожалуйста, помогите Dreamoon выбрать p_i в каждой операции, чтобы удовлетворить всем ограничениям.

Входные данные

В первой строке записаны два целых числа n, m ($1 \leq m \leq n \leq 100\,000$).

Во второй строке записано m целых чисел l_1, l_2, \dots, l_m ($1 \leq l_i \leq n$).

Выходные данные

Если невозможно совершить m операций, чтобы удовлетворить всем ограничениям, выведите «-1» (без кавычек).

Иначе, выведите m целых чисел p_1, p_2, \dots, p_m ($1 \leq p_i \leq n - l_i + 1$), после этих m операций, все цвета должны встречаться хотя бы один раз и все клетки должны быть покрашены.

Если есть несколько возможных решений, вы можете вывести любое.

Примеры

входные данные	выходные данные
5 3 3 2 2	
	2 4 1
входные данные	выходные данные
10 1 1	
	-1



C. Микаса

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

Вам даны два целых числа n и m . Найдите MEX последовательности $n \oplus 0, n \oplus 1, \dots, n \oplus m$. Здесь \oplus обозначает операцию [побитового исключающего ИЛИ](#).

MEX последовательности неотрицательных целых чисел — это наименьшее неотрицательное целое число, которое не встречается в этой последовательности. Например, $\text{MEX}(0, 1, 2, 4) = 3$, а $\text{MEX}(1, 2021) = 0$.

Входные данные

Первая строка содержит одно целое число t ($1 \leq t \leq 30\,000$) — количество наборов входных данных.

Первая и единственная строка каждого набора входных данных содержит два целых числа n и m ($0 \leq n, m \leq 10^9$).

Выходные данные

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

Пример

входные данные	Скопировать
5 3 5 4 6 3 2 69 696 123456 654321	
выходные данные	Скопировать
4 3 0 640 530866	

Примечание

В первом наборе входных данных последовательность равна $3 \oplus 0, 3 \oplus 1, 3 \oplus 2, 3 \oplus 3, 3 \oplus 4, 3 \oplus 5$, или $3, 2, 1, 0, 7, 6$.

Наименьшее неотрицательное целое число, которое не присутствует в последовательности, т. е. MEX последовательности — 4.

Во втором наборе входных данных последовательность равна $4 \oplus 0, 4 \oplus 1, 4 \oplus 2, 4 \oplus 3, 4 \oplus 4, 4 \oplus 5, 4 \oplus 6$, или $4, 5, 6, 7, 0, 1, 2$. Наименьшее неотрицательное целое число, которое не присутствует в последовательности, т. е. MEX последовательности — 3.

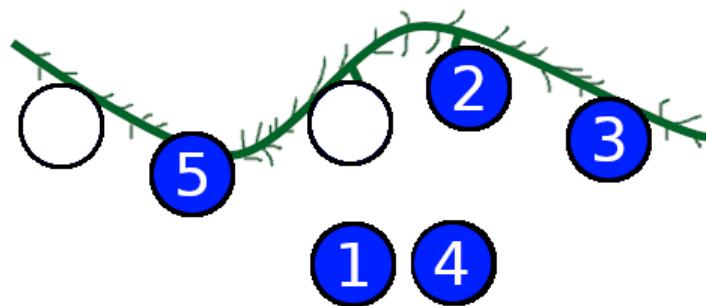
В третьем наборе входных данных последовательность равна $3 \oplus 0, 3 \oplus 1, 3 \oplus 2$, или $3, 2, 1$. Наименьшее неотрицательное целое число, которое не присутствует в последовательности, т. е. MEX последовательности — 0.



A. Гирлянда

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

Вадим очень любит наряжать новогоднюю ёлку, поэтому ему подарили невероятно красивую гирлянду. Она состоит из n лампочек, расположенных в ряд. На каждой лампочке написано число от 1 до n , причем все числа различны и шли в произвольном порядке. Пока Вадим решал задачи, его домашний Карась снял с гирлянды некоторые лампочки. Теперь Вадим хочет вернуть их на место.



Вадим хочет повесить все лампочки обратно на гирлянду. Он считает *сложностью* гирлянды количество пар соседних лампочек, на которых написаны числа разной четности (то есть имеющие разные остатки при делении на 2). Например, сложность гирлянды 1 4 2 3 5 равна 2, а сложность гирлянды 1 3 5 7 6 4 2, равна 1.

Никто не любит сложности, поэтому Вадим хочет минимизировать количество таких пар. Найдите способ повесить снятые лампочки на гирлянду, чтобы минимизировать ее сложность.

Входные данные

Первая строка содержит одно целое число n ($1 \leq n \leq 100$) — количество лампочек в гирлянде.

Вторая строка содержит n целых чисел p_1, p_2, \dots, p_n ($0 \leq p_i \leq n$) — номер i -й лампочки или 0, если Карась ее снял.

Выходные данные

Выведите единственное число — минимальную сложность гирлянды.

Примеры

входные данные	<input type="button" value="Скопировать"/>
5 0 5 0 2 3	
выходные данные	<input type="button" value="Скопировать"/>
2	
входные данные	<input type="button" value="Скопировать"/>
7 1 0 0 5 0 0 2	
выходные данные	<input type="button" value="Скопировать"/>
1	

Примечание

В первом примере надо расставить лампочки как 1 5 4 2 3. В этом случае сложность будет равна 2, так как рядом расположены пары лампочек (5, 4) и (2, 3).

Во втором примере один из правильных ответов 1 7 3 5 6 4 2.

A. Экстремальное вычитание

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Вам дан массив a из n положительных чисел.

Вы можете применять следующую операцию, сколько угодно раз: выбрать любое целое число $1 \leq k \leq n$ и выполнить одно из двух действий:

- отнять от k первых элементов массива единицу.
- отнять от k последних элементов массива единицу.

Например, если $n = 5$ и $a = [3, 2, 2, 1, 4]$, то вы можете применить к нему одну из следующих операций (ниже перечислены не все возможные варианты):

- отнять от первых двух элементов массива единицу. После этой операции $a = [2, 1, 2, 1, 4]$;
- отнять от трех последних элементов массива единицу. После этой операции $a = [3, 2, 1, 0, 3]$;
- отнять от пяти первых элементов массива единицу. После этой операции $a = [2, 1, 1, 0, 3]$;

Определите, возможно ли сделать все элементы массива равными нулю применив некоторое количество операций.

Входные данные

В первой строке находится одно целое положительное число t ($1 \leq t \leq 30000$) — количество наборов тестовых данных.
Далее следуют t наборов тестовых данных.

Каждый набор начинается со строки в которой записано одно целое число n ($1 \leq n \leq 30000$) — количество элементов в массиве.

Во второй строке каждого набора тестовых данных записаны n целых чисел $a_1 \dots a_n$ ($1 \leq a_i \leq 10^6$).

Сумма n по всем наборам тестовых данных не превосходит 30000.

Выходные данные

Для каждого набора тестовых данных в отдельной строке выведите:

- YES, если возможно сделать все элементы массива равными нулю применив некоторое количество операций.
- NO, иначе.

Буквы в словах YES и NO можно выводить в любом регистре.

Пример

Входные данные	<button>Скопировать</button>
4 3 1 2 1 5 11 7 9 6 8 5 1 3 1 3 1 4 5 2 1 10	
Выходные данные	<button>Скопировать</button>
YES YES NO YES	

С. Мессенджер в ЦПМ

ограничение по времени на тест: 3 секунды
ограничение по памяти на тест: 256 мегабайт

В новом мессенджере для учащихся Центра Помощи Магистрам, Кефтемерум, планируется обновление, в котором разработчики хотят оптимизировать набор сообщений, показываемых пользователю. Всего есть n сообщений, каждое сообщение характеризуется двумя целыми числами a_i и b_i . Время, потраченное на прочтение набора сообщений с номерами p_1, p_2, \dots, p_k ($1 \leq p_i \leq n$, все p_i **различны**) рассчитывается по формуле:

$$\sum_{i=1}^k a_{p_i} + \sum_{i=1}^{k-1} |b_{p_i} - b_{p_{i+1}}|$$

Обратите внимание, что время прочтения набора сообщений, состоящего из **одного** сообщения с номером p_1 , равно a_{p_1} . Также время прочтения пустого набора сообщений считается равным 0.

Пользователь может сам определить время l , которое он готов провести в мессенджере. Мессенджер же должен сообщить пользователю максимально возможный размер набора сообщений, время прочтения которого не превышает l . Обратите внимание, что максимальный размер набора сообщений может быть равен 0.

Разработчики популярного мессенджера не справились внедрить данную функцию, поэтому попросили вас решить эту задачу.

Входные данные

Каждый тест состоит из нескольких наборов входных данных. Первая строка содержит одно целое число t ($1 \leq t \leq 5 \cdot 10^4$) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа n и l ($1 \leq n \leq 2000$, $1 \leq l \leq 10^9$) — количество сообщений и время, которое пользователь готов провести в мессенджере.

i -я из следующих n строк содержит два целых числа a_i и b_i ($1 \leq a_i, b_i \leq 10^9$) — характеристики i -го сообщения.

Гарантируется, что сумма n^2 по всем наборам входных данных не превосходит $4 \cdot 10^6$.

Выходные данные

Для каждого набора входных данных выведите одно целое число — максимально возможный размер набора сообщений, время прочтения которого не превышает l .

Пример

входные данные	Скопировать
5	
5 8	
4 3	
1 5	
2 4	
4 3	
2 3	
1 6	
4 10	
3 12	
4 8	
2 1	
2 12	
5 26	
24 7	
8 28	
30 22	
3 8	
17 17	
5 14	
15 3	
1000000000 998244353	
179 239	
228 1337	
993 1007	
выходные данные	Скопировать
3	
1	

Примечание

В первом наборе входных данных, можно взять набор из трёх сообщений с номерами $p_1 = 3$, $p_2 = 2$ и $p_3 = 5$. Время, затрачиваемое на прочтение данного набора, равно

$$a_3 + a_2 + a_5 + |b_3 - b_2| + |b_2 - b_5| = 2 + 1 + 2 + |4 - 5| + |5 - 3| = 8.$$

Во втором наборе входных данных, можно взять набор из одного сообщения со номером $p_1 = 1$. Время, затрачиваемое на прочтение данного набора, равно $a_1 = 4$.

В пятом наборе входных данных можно показать, что не существует такого непустого набора сообщений, затрачиваемое время на прочтение которого не превышает l .



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

В. Назначение весов ребрам

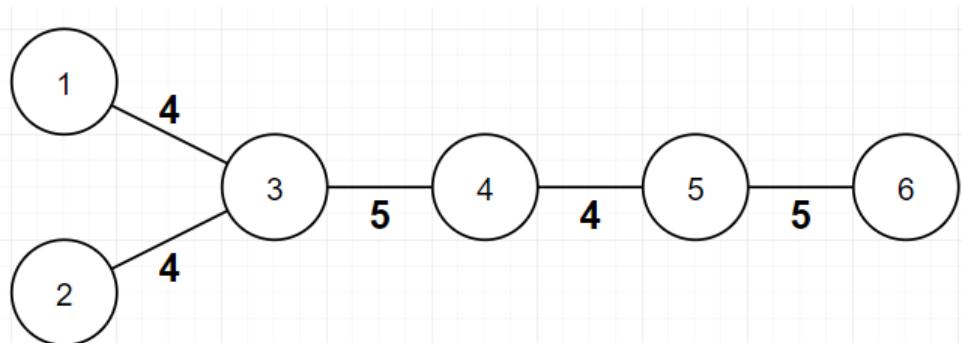
ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 256 мегабайт

У вас есть невзвешенное дерево на n вершинах. Вы должны назначить **положительный** вес каждому ребру, чтобы выполнялось следующее условие:

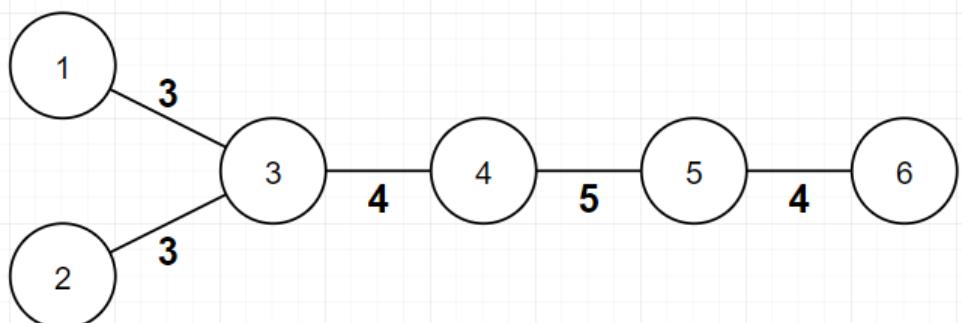
- Для каждого двух разных листов v_1 и v_2 этого дерева, побитовое исключающее ИЛИ весов всех ребер на простом пути между v_1 и v_2 должно быть равно 0.

Обратите внимание, что вы можете назначать **очень большие** натуральные числа (такие как $10^{(10^{10})}$).

Гарантируется, что такое назначение всегда существует при данных ограничениях. Теперь определим f как **количество различных весов** среди назначенных весов.



В этом примере назначение верно, потому что побитовое исключающее ИЛИ всех весов ребер между каждой парой листьев равно 0. Значение f здесь равно 2, потому что есть 2 различных веса ребер (4 и 5).



В этом примере назначение не удовлетворяет условию, поскольку побитовое исключающее ИЛИ всех весов ребер между вершинами 1 и 6 (3, 4, 5, 4) не равно 0.

Чему равны минимальное и максимальное возможные значения f для данного дерева? Найдите и выведите оба.

Входные данные

Первая строка содержит целое число n ($3 \leq n \leq 10^5$) — количество вершин в данном дереве.

i -я из следующих $n - 1$ строк содержит два целых числа a_i и b_i ($1 \leq a_i < b_i \leq n$) — это означает, что существует ребро между a_i и b_i . Гарантируется, что данный граф образует дерево из n вершин.

Выходные данные

Выполните два целых числа — минимальное и максимальное возможные значения f , которые могут быть получены из правильного назначения данного дерева. Обратите внимание, что такое назначение всегда существует при данных ограничениях.

Примеры

входные данные

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

4 5
5 6

входные данные

Скопировать

1 4

входные данные

Скопировать

6
1 3
2 3
3 4
4 5
4 6

входные данные

Скопировать

3 3

входные данные

Скопировать

7
1 2
2 7
3 4
4 7
5 6
6 7

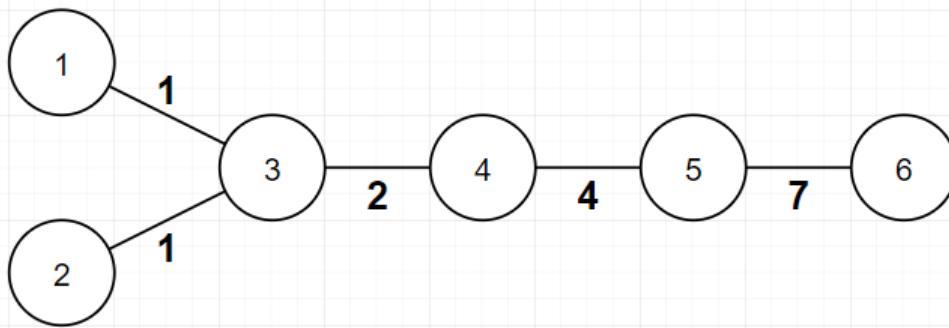
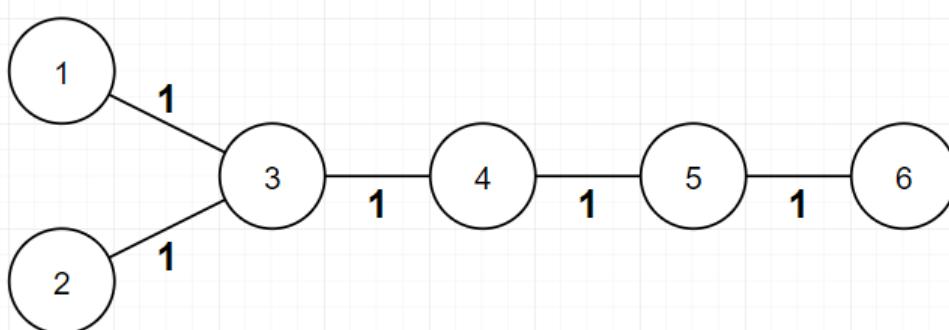
входные данные

Скопировать

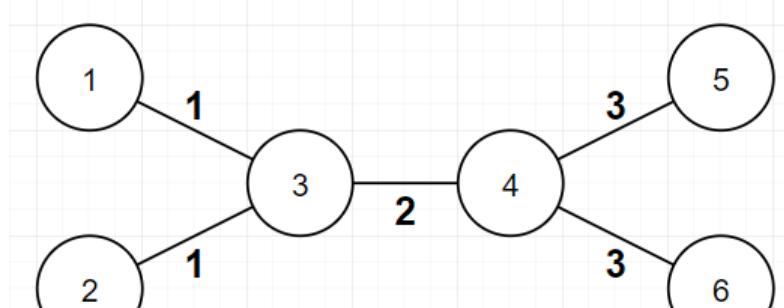
1 6

Примечание

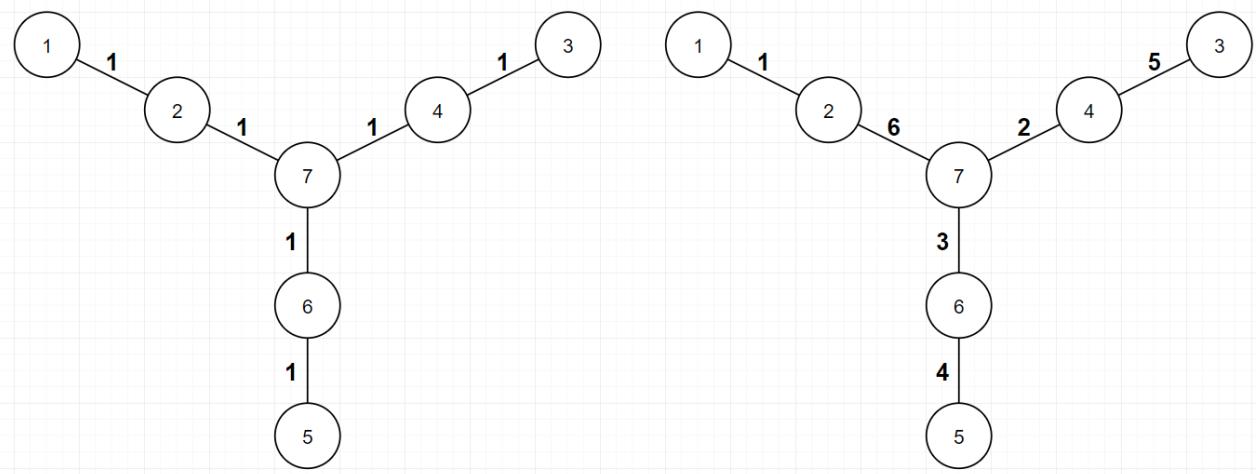
В первом примере возможные назначения для минимума и максимума показаны на рисунке ниже. Конечно, есть и другие назначения.



Во втором примере возможные назначения для минимума и максимума показаны на рисунке ниже. Значение f для правильного назначения этого дерева всегда равно 3.



В третьем примере возможные назначения для минимума и максимума показаны на рисунке ниже. Конечно, есть и другие назначения.



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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:37:01 UTC+5 (k2).

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

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

При поддержке



ИТМО

D. Слизняки

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Есть ряд из n слизней. У каждого слизня есть некоторая целочисленная ценность, связанная с ним (ценность может быть также меньше нуля или равна нулю).

Любой из слизняков может съесть любого из своих соседей (ближайшего к нему слизня слева или справа, при условии что такой слизень есть).

Когда слизень с ценностью x съедает слизня со значением y , то съеденный слизень исчезает, а ценность оставшегося слизня становится равной $x - y$.

Слизни будут есть друг друга, пока не останется ровно один слизень.

Выясните какая наибольшая возможная ценность может оказаться у последнего слизня.

Входные данные

Первая строка входных данных содержит одно целое число n ($1 \leq n \leq 500\,000$) — количество слизней в ряду.

Вторая строка содержит n целых чисел a_i ($-10^9 \leq a_i \leq 10^9$), где a_i — это ценность i -го слизня.

Выходные данные

Выведите одно число — наибольшую возможную ценность у последнего слизня.

Примеры

входные данные	Скопировать
4 2 1 2 1	
выходные данные	Скопировать
4	
входные данные	Скопировать
5 0 -1 -1 -1 -1	
выходные данные	Скопировать
4	

Примечание

В первом примере одним из способов получить последнего слизня с ценностью 4 следующий:

- Второй слизень съедает третьего, ряд теперь содержит слизней 2, -1 , 1
- Второй слизень съедает третьего, ряд теперь содержит слизней 2, -2
- Первый слизень съедает второго, ряд теперь содержит одного слизня 4

Во втором примере первый слизень может есть слизней слева направо, тем самым получая итоговую ценность 4.



D. Задача на доске

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Поликарп написал на доске некоторую строку s из строчных букв латинского алфавита ('a'-'z'). Эта строка вам известна и задана во входных данных.

После этого он стёр какие-то буквы из строки s , а оставшиеся буквы он переписал в **произвольном** порядке. В результате он получил некоторую новую строку t . Её вам и предстоит найти по некоторой дополнительной информации.

Предположим, что длина строки t равна m , а символы пронумерованы слева направо от 1 до m . В таком случае вам задана последовательность из m целых чисел: b_1, b_2, \dots, b_m , где b_i равно сумме расстояний $|i - j|$ от индекса i до всех таких индексов j , что $t_j > t_i$ (считайте, что 'a' < 'b' < ... < 'z'). Иными словами, для вычисления b_i Поликарп находит все такие индексы j , что в индексе j находится буква, которая стоит позже в алфавите чем t_i , и суммирует все значения $|i - j|$.

Например, если $t=«abzb»$, то:

- так как $t_1='a'$, то все остальные индексы содержат буквы, которые позже в алфавите, то есть:
 $b_1 = |1 - 2| + |1 - 3| + |1 - 4| = 1 + 2 + 3 = 6$;
- так как $t_2='b'$, то только индекс $j = 3$ содержит букву, которая позже в алфавите, то есть: $b_2 = |2 - 3| = 1$;
- так как $t_3='z'$, то индексов j , что $t_j > t_i$ не существует: $b_3 = 0$;
- так как $t_4='b'$, то только индекс $j = 3$ содержит букву, которая позже в алфавите, то есть: $b_4 = |4 - 3| = 1$.

Таким образом, если $t=«abzb»$, то $b = [6, 1, 0, 1]$.

По заданной строке s и массиву b найдите любую возможную строку t , для которой выполняются следующие два требования одновременно:

- t получается из s путём стирания некоторых букв (возможно, нуля) и потом записи оставшихся в **произвольном** порядке;
- по строке t получается заданный во входных данных массив b , если его построить по правилам, которые описаны выше.

Входные данные

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

Каждый набор входных данных состоит из трех строк:

- строки s , которая имеет длину от 1 до 50 и состоит из строчных букв латинского алфавита;
- строки, которая содержит целое число m ($1 \leq m \leq |s|$), где $|s|$ — длина строки s , а m — длина массива b ;
- строки, которая содержит целые числа b_1, b_2, \dots, b_m ($0 \leq b_i \leq 1225$).

Гарантируется, что в каждом наборе данных входные данные таковы, что ответ существует.

Выходные данные

Выведите q строк: k -я из них должна содержать ответ (строку t) на k -й набор входных данных. Гарантируется, что ответ на каждый набор входных данных существует. Если ответов несколько, то выведите любой.

Пример

входные данные	Скопировать
4 abac 3 2 1 0 abc 1 0 abba 3 1 0 1 ecoosdcefr 10 38 13 24 14 11 5 3 24 17 0	
выходные данные	Скопировать
aac b	

Примечание

В первом наборе входных данных подходят такие строки t : «aac», «aab».

Во втором наборе входных данных подходят такие строки t : «a», «b», «c».

В третьем наборе входных данных подходит только строка t равная «aba», но символ 'b' может быть со второй или с третьей позиции.

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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:37:06_{UTC+5} (k2).

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

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

При поддержке



ITMO

D. Рейтинговая система

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вы разрабатываете рейтинговую систему для онлайн-игры. Каждый раз, когда игрок участвует в матче в этой игре, его рейтинг меняется в зависимости от результатов.

Изначально рейтинг игрока равен 0. Он будет участвовать в n матчах; после i -го матча рейтинг изменяется на a_i (рейтинг увеличивается на a_i , если a_i положительное, или уменьшается на $|a_i|$, если отрицательное. В последовательности a нет нулей).

В системе есть дополнительное правило: для заданного целого числа k , если рейтинг игрока достиг значения k , он никогда не опустится ниже него. Формально, если рейтинг игрока хотя бы k , а изменение рейтинга должно сделать его меньше k , то рейтинг снизится **ровно до** k .

Ваша задача состоит в том, чтобы определить значение k таким образом, чтобы рейтинг игрока после всех n матчей был максимально возможным (среди всех возможных целочисленных значений k). Если существует несколько возможных ответов, выведите любой из них.

Входные данные

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

Первая строка каждого набора содержит одно целое число n ($1 \leq n \leq 3 \cdot 10^5$) — количество матчей.

Вторая строка содержит n целых чисел a_1, a_2, \dots, a_n ($-10^9 \leq a_i \leq 10^9$; $a_i \neq 0$) — изменение рейтинга после i -го матча.

Сумма n по всем наборам входных данных не превосходит $3 \cdot 10^5$.

Выходные данные

Для каждого набора входных данных выведите одно целое число m ($-10^{18} \leq m \leq 10^{18}$) — такое значение k , при котором рейтинг игрока будет максимально возможным. Можно показать, что существует хотя бы одно такое значение, которое удовлетворяет ограничениям $-10^{18} \leq m \leq 10^{18}$.

Пример

входные данные	Скопировать
4 4 3 -2 1 2 3 -1 -2 -1 2 4 2 7 5 1 -3 2 -1 -2 2	
выходные данные	Скопировать
3 0 25 6	

Примечание

В первом примере, если $k = 3$, то рейтинг изменяется следующим образом: $0 \rightarrow 3 \rightarrow 3 \rightarrow 4 \rightarrow 6$.

Во втором примере, если $k = 0$, то рейтинг изменяется следующим образом: $0 \rightarrow 0 \rightarrow 0 \rightarrow 0$.

В третьем примере, если $k = 25$, то рейтинг изменяется следующим образом: $0 \rightarrow 4 \rightarrow 6$.

В четвертом примере, если $k = 6$, то рейтинг изменяется следующим образом: $0 \rightarrow 5 \rightarrow 6 \rightarrow 6 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 8$.

C. Мирские числа

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Мирские цифры обозначаются заглавными латинскими буквами от A до E. Кроме того, значение буквы A равно 1, B равно 10, C равно 100, D равно 1000, E равно 10000.

Мирское число — это последовательность мирских цифр. Значение мирского числа вычисляется следующим образом: суммируются значения всех цифр, но некоторые цифры берутся со знаком минус; цифра берется со знаком минус, если справа от нее есть цифра со значением **строго больше** (не обязательно сразу после нее); в противном случае цифра берется со знаком плюс.

Например, значение мирского числа DAAABDCA равно $1000 - 1 - 1 - 1 - 10 + 1000 + 100 + 1 = 2088$.

Вам дано мирское число. Вы можете изменить не более одной цифры в нем. Вычислите максимально возможное значение полученного числа.

Входные данные

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

Единственная строка каждого набора входных данных содержит строку s ($1 \leq |s| \leq 2 \cdot 10^5$), состоящую из заглавных латинских букв от A до E — заданное мирское число.

Сумма длин строк по всем наборам входных данных не превышает $2 \cdot 10^5$.

Выходные данные

Для каждого набора входных данных выведите одно целое число — максимально возможное значение числа, если вы можете изменить в нем не более одной цифры.

Пример

входные данные	Скопировать
4 DAAABDCA AB ABCDEEDCBA DDDDAAADABECED	
выходные данные	Скопировать
11088 10010 31000 15886	

Примечание

В первом примере можно получить EAAABDCA со значением $10000 - 1 - 1 - 1 - 10 + 1000 + 100 + 1 = 11088$.

Во втором примере можно получить EB со значением $10000 + 10 = 10010$.



D. Рождественские деревья

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

На бесконечной числовой прямой находятся n рождественских деревьев. i -е дерево растет на позиции x_i . Гарантируется, что все значения x_i различны.

Каждая **целочисленная** точка может быть либо занята рождественским деревом, либо занята человеком, либо не занята вообще. Нечелые точки не могут быть заняты ничем.

Есть m человек, которые хотят отпраздновать Рождество. Пусть y_1, y_2, \dots, y_m — это позиции людей (заметьте, что все значения $x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_m$ должны быть **различны** и все y_j должны быть **целыми**). Вы хотите найти такое расположение людей, что значение $\sum_{j=1}^m \min_{i=1}^n |x_i - y_j|$ минимально возможное (другими словами, сумма расстояний до ближайшего рождественского дерева по всем людям должна быть минимизирована).

Другими словами, пусть d_j равно дистанции от j -го человека до ближайшего к нему рождественского дерева ($d_j = \min_{i=1}^n |y_j - x_i|$). Тогда вам надо выбрать такие позиции y_1, y_2, \dots, y_m , что $\sum_{j=1}^m d_j$ минимально возможна.

Входные данные

Первая строка входных данных содержит два целых числа n и m ($1 \leq n, m \leq 2 \cdot 10^5$) — количество рождественских деревьев и количество людей.

Вторая строка входных данных содержит n целых чисел x_1, x_2, \dots, x_n ($-10^9 \leq x_i \leq 10^9$), где x_i равно позиции i -го рождественского дерева. Гарантируется, что все x_i различны.

Выходные данные

В первой строке выведите одно целое число res — минимально возможное значение $\sum_{j=1}^m \min_{i=1}^n |x_i - y_j|$ (другими словами, сумму расстояний до ближайшего рождественского дерева по всем людям).

Во второй строке выведите m целых чисел y_1, y_2, \dots, y_m ($-2 \cdot 10^9 \leq y_j \leq 2 \cdot 10^9$), где y_j равно позиции j -го человека. Все y_j должны быть различными, а еще все значения $x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_m$ должны быть **различными**.

Если существует несколько ответов, выведите любой.

Примеры

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



E. Возрастающие подпоследовательности

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Напомним, что возрастающая подпоследовательность массива a — это последовательность, которую можно получить из массива, удалив некоторые элементы, не изменяя порядок оставшихся элементов, и оставшиеся элементы строго возрастают (т.е. $a_{b_1} < a_{b_2} < \dots < a_{b_k}$ и $b_1 < b_2 < \dots < b_k$). Обратите внимание, что пустая подпоследовательность также является возрастающей.

Вам дано положительное целое число X . Ваша задача — найти массив целых чисел длиной **не более** 200, такой, что у него ровно X возрастающих подпоследовательностей, или сообщить, что такого массива нет. Если есть несколько ответов, вы можете вывести любой из них.

Если две подпоследовательности состоят из одинаковых чисел, но позиции элементов исходного массива, входящих в подпоследовательность, отличаются, то эти подпоследовательности считаются различными (например, у массива [2, 2] две разных подпоследовательности, равных [2]).

Входные данные

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

Единственная строка каждого набора содержит одно целое число X ($2 \leq X \leq 10^{18}$).

Выходные данные

Для каждого набора входных данных выведите ответ для него. Если невозможно найти требуемый массив, выведите **-1** в первой строке. В противном случае выведите положительное целое число n в первой строке — длину массива. Во второй строке выведите n целых чисел — элементы искомого массива. Если есть несколько ответов, вы можете вывести любой из них. Все элементы массива должны находиться в диапазоне $[-10^9; 10^9]$.

Пример

входные данные	Скопировать
4 2 5 13 37	
выходные данные	Скопировать
1 0 3 0 1 0 5 2 2 3 4 2 7 -1 -1 0 0 2 3 -1	



D. Очередная задача на минимизацию

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Даны два массива a и b длины n .

Вы можете любое количество раз (возможно, ноль) выполнить следующую операцию: выбрать индекс i ($1 \leq i \leq n$) и поменять местами a_i и b_i .

Назовем *стоимостью* массива a величину, равную $\sum_{i=1}^n \sum_{j=i+1}^n (a_i + a_j)^2$. Аналогично, стоимость массива b будет равна $\sum_{i=1}^n \sum_{j=i+1}^n (b_i + b_j)^2$.

Вам необходимо минимизировать суммарную стоимость двух массивов.

Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится единственное целое число t ($1 \leq t \leq 40$) — количество наборов входных данных. Далее следует описание наборов входных данных.

В первой строке каждого набора входных данных содержится целое число n ($1 \leq n \leq 100$) — длина обоих массивов.

Во второй строке каждого набора входных данных содержатся n целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 100$) — элементы первого массива.

В третьей строке каждого набора входных данных содержатся n целых чисел b_1, b_2, \dots, b_n ($1 \leq b_i \leq 100$) — элементы второго массива.

Гарантируется, что сумма n по всем наборам входных данных не превосходит 100.

Выходные данные

Для каждого набора входных данных выведите минимальную возможную суммарную стоимость двух массивов.

Пример

входные данные	Скопировать
3 1 3 6 4 3 6 6 6 2 7 4 1 4 6 7 2 4 2 5 3 5	
выходные данные	Скопировать
0 987 914	

Примечание

Во втором наборе входных данных в одном из оптимальных ответов после выполнения операций $a = [2, 6, 4, 6]$, $b = [3, 7, 6, 1]$.

Стоимость массива a равна $(2 + 6)^2 + (2 + 4)^2 + (2 + 6)^2 + (6 + 4)^2 + (6 + 6)^2 + (4 + 6)^2 = 508$.

Стоимость массива b равна $(3 + 7)^2 + (3 + 6)^2 + (3 + 1)^2 + (7 + 6)^2 + (7 + 1)^2 + (6 + 1)^2 = 479$.

Суммарная стоимость массивов равна $508 + 479 = 987$.

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

D. Новогодняя задача

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

У Влада есть n друзей, для каждого из которых он хочет купить **один** подарок на Новый год.

Всего в городе есть m магазинов, в каждом из которых он может купить подарок любому из друзей. Если j -й друг ($1 \leq j \leq n$) получает подарок, купленный в магазине с номером i ($1 \leq i \leq m$), то друг получает p_{ij} единиц радости. Прямоугольная таблица p_{ij} задана во входных данных.

Влад успевает посетить не более $n - 1$ магазина (n — количество **друзей**). Он сам может выбрать, какие именно магазины посетит, и для каких друзей он будет покупать подарки в каждом из них.

Пусть j -й друг получил a_j единиц радости от подарка Влада. Найдем величину $\alpha = \min\{a_1, a_2, \dots, a_n\}$. Цель Влада — купить подарки так, чтобы значение α было как можно больше. Иными словами, Влад хочет максимизировать минимальную из радостей друзей.

Например, пусть $m = 2$, $n = 2$. Пусть радости от подарков, которые мы можем приобрести в первом магазине: $p_{11} = 1$, $p_{12} = 2$, во втором магазине: $p_{21} = 3$, $p_{22} = 4$.

Тогда Владу достаточно зайти только во второй магазин и купить в нем для первого друга подарок, приносящий радость 3, а для второго — приносящий радость 4. При этом величина α будет равна $\min\{3, 4\} = 3$.

Помогите Владу выбрать подарки для своих друзей так, чтобы значение α было максимально возможным. Обратите внимание на то, что каждый друг должен получить один подарок. Влад может посетить не более $n - 1$ магазина (n — количество **друзей**). В магазине он может купить произвольное количество подарков.

Входные данные

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

Перед каждым набором в тесте записана пустая строка. Далее идет строка, которая содержит целые числа m и n ($2 \leq n$, $2 \leq n \cdot m \leq 10^5$) через пробел — количество магазинов и количество друзей, где $n \cdot m$ — это произведение n на m .

Затем следуют m строк, содержащих по n чисел. Число в i -й строке и j -м столбце p_{ij} ($1 \leq p_{ij} \leq 10^9$) — радость от подарка, предназначенного для друга с номером j в магазине с номером i .

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

Выходные данные

Выполните t строк, каждая из которых должна содержать ответ на соответствующий набор входных данных — максимальное возможное значение α , где α — минимальная из радостей от подарка по всем друзьям Влада.

Пример

входные данные	Скопировать

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

```
4 2  
7 9  
8 1  
9 6  
10 8
```

```
2 4  
6 5 2 1  
7 9 7 2
```

выходные данные**Скопировать**

```
3  
2  
4  
8  
2
```

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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:37:17_{UTC+5} (k2).

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

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

При поддержке



ИТМО

D. Удачная перестановка

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

Вам дана перестановка[†] p длины n .

За одну операцию вы можете выбрать два индекса $1 \leq i < j \leq n$ и поменять местами p_i и p_j .

Найдите минимальное количество операций, которое необходимо сделать, чтобы в итоговой перестановке была **ровно одна инверсия**[‡].

[†] Перестановкой называется массив, состоящий из n различных целых чисел от 1 до n в произвольном порядке. Например, $[2, 3, 1, 5, 4]$ является перестановкой, но $[1, 2, 2]$ не является перестановкой (2 встречается дважды в массиве) и $[1, 3, 4]$ тоже не является перестановкой ($n = 3$, но 4 присутствует в массиве).

[‡] Количество инверсий в перестановке p называется количеством пар индексов (i, j) таких, что $1 \leq i < j \leq n$ и $p_i > p_j$.

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.

В первой строке каждого набора входных данных содержится одно целое число n ($2 \leq n \leq 2 \cdot 10^5$) — длина перестановки.

Вторая строка каждого набора содержит n чисел p_1, p_2, \dots, p_n ($1 \leq p_i \leq n$). Гарантируется, что p — перестановка.

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

Выходные данные

Для каждого набора входных данных выведите одно число — минимальное количество операций, которое необходимо сделать, чтобы в итоговом массиве была ровно одна инверсия. Можно показать, что это всегда возможно.

Пример

входные данные	<button>Скопировать</button>
4 2 2 1 2 1 2 4 3 4 1 2 4 2 4 3 1	
выходные данные	<button>Скопировать</button>
0 1 3 1	

Примечание

В первом наборе входных перестановка уже удовлетворяет условию.

Во втором наборе вы можете выполнить операцию с $(i, j) = (1, 2)$, после этого в перестановке $[2, 1]$ будет ровно одна инверсия.

В третьем наборе нельзя выполнить условие меньше чем за 3 операции. А за 3 операции можно добиться выполнения условия так: (i, j) должны равняться $(1, 3), (2, 4), (3, 4)$ в соответствующих операциях. Итоговой перестановкой будет $[1, 2, 4, 3]$, в ней ровно одна инверсия.

В четвёртом наборе нужно сделать обмен, в котором $(i, j) = (2, 4)$, в результате чего перестановка будет равняться $[2, 1, 3, 4]$.

F. Нечестная игра

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Алиса и Боб собирались вечером, чтобы поиграть в увлекательную игру на последовательности из n натуральных чисел, **не превосходящих** 4. Правила игры слишком сложны для их описания, поэтому просто опишем критерий победы — Алиса побеждает, если **исключающее ИЛИ** всех чисел последовательности отлично от нуля, в противном случае же побеждает Боб.

Ребята позвали Еву, чтобы она выступала в качестве судьи. Изначально Алиса и Боб играют с n числами. После одной игры Ева удаляет одно из чисел из последовательности, затем Алиса и Боб играют с $n - 1$ числами, Ева снова удаляет одно из чисел, после чего Алиса и Боб играют с $n - 2$ числами. Так продолжается до тех пор, пока последовательность чисел не пуста.

Еве кажется, что в такой игре Алиса почти всегда побеждает, поэтому она хочет, чтобы Боб выиграл как можно больше раз. Определите, какое максимальное количество раз Боб сможет победить Алису, если Ева будет удалять числа оптимально.

Входные данные

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

Первая и единственная строка каждого набора содержит четыре целых числа p_i ($0 \leq p_i \leq 200$) — количество единиц, двоек, троек и чётвёрок в последовательности в начале игры.

Выходные данные

Для каждого набора входных данных в отдельной строке выведите максимальное количество раз, когда выиграет Боб, если Ева будет удалять числа оптимально.

Пример

входные данные	Скопировать
5 1 1 1 0 1 0 1 2 2 2 2 0 3 3 2 0 0 9 9 9	
выходные данные	Скопировать
1 1 3 3 12	

Примечание

В первом примере Боб выигрывает, когда Ева ещё не убрала ни одного числа.

Во втором примере Боб выигрывает, если Ева уберёт одну единицу и одну тройку.



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

E1. Покраска строки (простая версия)

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Это простая версия задачи. Задачи отличаются друг от друга, но простая версия почти является подзадачей сложной версии. Обратите внимание на то, что ограничения и формат вывода различаются.

Дана строка s , состоящая из n строчных латинских букв.

Вам нужно раскрасить **все** её символы **в один из двух цветов** (каждый символ покрашен ровно в один цвет, одинаковые буквы могут быть покрашены как одним цветом, так и разными, т.е. вам нужно выбрать ровно один цвет для каждого индекса в s).

После покраски вы можете поменять местами два **любых** соседних символа строки, которые раскрашены в **разные** цвета. Вы можете применить эту операцию произвольное (возможно, нулевое) количество раз.

Цель — сделать строку отсортированной, т.е. все символы должны быть расположены в алфавитном порядке.

Ваша задача — определить, возможно ли покрасить заданную строку так, что после покраски она может быть отсортирована с помощью **какой-либо** последовательности шагов. Обратите внимание: вам нужно восстановить только раскраску, но не последовательность шагов.

Входные данные

Первая строка входных данных содержит одно целое число n ($1 \leq n \leq 200$) — длина строки s .

Вторая строка входных данных содержит строку s , состоящую ровно из n строчных латинских букв.

Выходные данные

Если невозможно раскрасить заданную строку так, что после покраски она может быть отсортирована после **какой-либо** последовательности шагов, выведите «NO» (без кавычек) первой строкой.

В противном случае выведите «YES» первой строкой и **любую** корректную раскраску второй строкой (раскраской считается строка, состоящая из n символов, где i -й символ должен быть '0', если i -й символ покрашен в первый цвет и '1' иначе).

Примеры

входные данные	<input type="button" value="Скопировать"/>
9 abacbecfd	
выходные данные	<input type="button" value="Скопировать"/>
YES 001010101	
входные данные	<input type="button" value="Скопировать"/>
8 aaabbcbba	
выходные данные	<input type="button" value="Скопировать"/>
YES 010110101	
входные данные	<input type="button" value="Скопировать"/>
7 abcdedc	
выходные данные	<input type="button" value="Скопировать"/>
NO	
входные данные	<input type="button" value="Скопировать"/>
5 abcde	
выходные данные	<input type="button" value="Скопировать"/>
YES 00000	

G. Велосипеды

ограничение по времени на тест: 4 секунды
ограничение по памяти на тест: 256 мегабайт

Все друзья Славика планируют отправиться на вечеринку на своих велосипедах из места, где они живут. У всех есть велосипеды, кроме Славика. Есть n городов, через которые они могут путешествовать. Все живут в городе 1 и хотят пойти на вечеринку, которая находится в городе n . Карту городов можно рассматривать как ненаправленный граф с n узлами и m ребрами. Ребро i соединяет города u_i и v_i и имеет длину w_i .

У Славика нет велосипеда, но у него есть деньги. В каждом городе есть ровно один велосипед, который можно купить. У велосипеда в i -м городе есть коэффициент медлительности s_i . Как только Славик купит велосипед, он может использовать его, чтобы путешествовать из города, в котором он находится, в **любой** город. Проезд по ребру i с использованием велосипеда j занимает время $w_i \cdot s_j$.

Славик может купить столько велосипедов, сколько захочет, так как деньги для него не проблема. Славик ненавидит путешествовать на велосипеде, поэтому он хочет добраться на вечеринку за минимальное время. И поскольку он уже подзабыл информатику, он просит вас ему помочь.

Какое минимальное количество времени потребуется Славику, чтобы добраться из города 1 в город n ? Славик не может путешествовать без велосипеда. Гарантируется, что возможно добраться от города 1 до любого другого города.

Входные данные

Первая строка содержит одно целое число t ($1 \leq t \leq 100$) — количество наборов входных данных. Далее следует описание самих наборов.

Первая строка каждого набора содержит два целых числа, разделенных пробелом: n и m ($2 \leq n \leq 1000$; $n - 1 \leq m \leq 1000$) — количество городов и количество дорог, соответственно.

i -я из следующих m строк содержит три целых числа u_i , v_i , w_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$; $1 \leq w_i \leq 10^5$), обозначающих, что есть дорога между городами u_i и v_i длиной w_i .

Следующая строка содержит n целых чисел s_1, \dots, s_n ($1 \leq s_i \leq 1000$) — коэффициент медлительности каждого велосипеда. Одну и ту же пару городов может соединять больше, чем одна дорога.

Сумма n по всем наборам не превышает 1000, и сумма m по всем наборам не превышает 1000.

Дополнительное ограничение на ввод: возможно путешествовать от города 1 до любого другого города.

Выходные данные

Для каждого набора входных данных выведите одно целое число, обозначающее минимальное количество времени, за которое Славик может добраться из города 1 в город n .

Пример

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

```
7 1 2
4 3 5
6 4 2
6 7 1
6 7 4
4 5 9
7 6 5 4 3 2 1
```

ВХОДНЫЕ ДАННЫЕ**Скопировать**

```
19
36
14
```

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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:37:23^{UTC+5} (k2).

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

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

При поддержке



ITMO

E. Камень, ножницы, бумага

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

Аня и Боря решили сыграть в игру «Камень, ножницы, бумага».

Игра состоит из раундов, каждый из раундов проводится независимо от остальных. В каждом раунде оба игрока одновременно показывают один из трех предметов: камень, ножницы или бумагу. При этом, если оба игрока показали одинаковые предметы, то в раунде объявляется ничья. В противном случае, действуют следующие правила:

- если один игрок показал камень, а другой ножницы, то игрок, показавший камень, объявляется победителем раунда, а другой игрок — проигравшим;
- если один игрок показал ножницы, а другой бумагу, то игрок, показавший ножницы, объявляется победителем раунда, а другой игрок — проигравшим;
- если один игрок показал бумагу, а другой камень, то игрок, показавший бумагу, объявляется победителем раунда, а другой игрок — проигравшим.

Аня и Боря решили, что они сыграют ровно n раундов описанной игры. Аня решила, что она a_1 раз покажет камень, a_2 раз покажет ножницы и a_3 раз покажет бумагу. Боря решил, что он b_1 раз покажет камень, b_2 раз покажет ножницы и b_3 раз покажет бумагу. При этом ни Аня, ни Боря еще **не выбрали** последовательность, в которой будут показывать предметы. Гарантируется, что $a_1 + a_2 + a_3 = n$ и $b_1 + b_2 + b_3 = n$.

Перед вами стоит задача определить два числа:

1. минимальное количество раундов, которое может выиграть Аня;
2. максимальное количество раундов, которое может выиграть Аня.

Входные данные

В первой строке следует целое число n ($1 \leq n \leq 10^9$) — количество раундов.

Во второй строке следуют три целых числа a_1, a_2, a_3 ($0 \leq a_i \leq n$) — количество раз, которое Аня покажет камень, ножницы и бумагу, соответственно. Гарантируется, что $a_1 + a_2 + a_3 = n$.

В третьей строке следуют три целых числа b_1, b_2, b_3 ($0 \leq b_j \leq n$) — количество раз, которое Боря покажет камень, ножницы и бумагу, соответственно. Гарантируется, что $b_1 + b_2 + b_3 = n$.

Выходные данные

Выведите два целых числа — минимальное и максимальное количество раундов, которое может выиграть Аня.

Примеры

входные данные	Скопировать
2 0 1 1 1 1 0	
выходные данные	Скопировать
0 1	
входные данные	Скопировать
15 5 5 5 5 5 5	
выходные данные	Скопировать
0 15	
входные данные	Скопировать
3 0 0 3 3 0 0	
выходные данные	Скопировать
3 3	

входные данные	<input type="button" value="Скопировать"/>
686 479 178 29 11 145 530	
выходные данные	<input type="button" value="Скопировать"/>
22 334	
входные данные	<input type="button" value="Скопировать"/>
319 10 53 256 182 103 34	
выходные данные	<input type="button" value="Скопировать"/>
119 226	

Примечание

В первом примере Аня может не выиграть ни одного раунда, если покажет, например, сначала ножницы, а потом бумагу, а Боря покажет камень, а потом ножницы. В лучшем случае Аня выиграет один раунд, если, например, сначала покажет бумагу, а потом ножницы, а Боря покажет камень, а потом ножницы.

Во втором примере Аня может не выиграть ни одного раунда, если, например, Боря в каждом раунде будет показывать те же предметы, что и Аня.

В третьем примере Аня всегда показывает бумагу, а Боря всегда показывает камень, поэтому Аня в любом случае выиграет все три раунда.

[Codeforces](#) (с) Copyright 2010-2025 Михаил Мирзаянов
Соревнования по программированию 2.0
Время на сервере: 05.07.2025 09:37:25 UTC+5 (k2).
Мобильная версия, переключиться на [десктопную](#).
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО

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

C. Спортивная рыбалка

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

Алиса и Боб участвуют в соревновании по ловле рыбы! В общей сложности они поймали n рыб, пронумерованных от 1 до n (чем больше рыба, тем больше ее индекс). Некоторые из этих рыб были пойманы Алисой, другие — Бобом.

Их результаты будут оцениваться следующим образом. Сначала будет выбрано целое число m , и все рыбы будут разделены на m **непустых** групп. Первая группа должна содержать несколько (по крайней мере одну) самых маленьких рыб, вторая группа — несколько (по крайней мере одну) следующих по размеру рыб и так далее. Каждая рыба должна принадлежать ровно одной группе, и каждая группа должна являться непрерывным подотрезком рыб. Обратите внимание, что нумерацию групп менять нельзя: например, рыбы в первой группе не могут быть больше рыб во второй группе, потому что первая группа содержит несколько самых маленьких рыб.

Затем каждой рыбке будет присвоено значение в зависимости от индекса ее группы: каждая рыба в первой группе получает значение, равное 0, каждая рыба во второй группе получает значение, равное 1, и так далее. Таким образом, каждая рыба в i -й группе получает значение, равное $(i - 1)$.

Счет каждого участника равен суммарному значению всех рыб, которые он поймал.

Вы хотите, чтобы счет Боба превышал счет Алисы **не менее** чем на k очков. Какое минимальное количество групп (m) необходимо создать для разделения рыб? Если это невозможно, сообщите об этом.

Входные данные

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

Первая строка каждого набора содержит два целых числа n и k ($2 \leq n \leq 2 \cdot 10^5$; $1 \leq k \leq 10^9$).

Вторая строка содержит строку, состоящую ровно из n символов. i -й символ равен 0 (обозначает, что i -я рыба была поймана Алисой) или 1 (обозначает, что i -я рыба была поймана Бобом).

Дополнительное ограничение на входные данные: сумма n по всем наборам входных данных не превышает $2 \cdot 10^5$.

Выходные данные

Для каждого набора входных данных выведите одно целое число — минимальное количество групп, на которые нужно разделить рыб; или -1, если это невозможно.

Пример

входные данные	Скопировать
7 4 1 1001 4 1 1010 4 1 0110 4 2 0110 6 3 001110 10 20 1111111111 5 11 11111	
выходные данные	Скопировать
2 -1 2 -1 3 4 -1	

Примечание

В первом примере вы можете разделить рыб на группы следующим образом: первые три рыб образуют 1-ю группу, последняя рыба образует 2-ю группу. Тогда счет Боба будет 1, а счет Алисы будет 0.

A. Ключи от офиса

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

На прямой есть n человек и k ключей. Каждый из людей хочет попасть в офис, который также расположен на прямой. Для этого каждому человеку нужно сначала дойти до точки, в которой находится ключ, взять его, а затем отправиться в офис. После того как человек взял ключ, больше этот ключ взять никто не сможет.

Перед вами стоит задача определить минимальное время по истечении которого все n людей смогут добраться до офиса с ключами. Считайте, что каждый из людей преодолевает единицу расстояния за 1 секунду. Если в точку с ключом одновременно придут два человека, то ключ может взять только один из них. Человек не обязан брать ключ, мимо которого проходит.

Входные данные

В первой строке следуют три целых числа n , k и p ($1 \leq n \leq 1\,000$, $n \leq k \leq 2\,000$, $1 \leq p \leq 10^9$) — количество людей, количество ключей и позиция офиса.

Во второй строке следуют n различных целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$) — позиции, в которых изначально находятся люди. Позиции заданы в произвольном порядке.

В третьей строке следуют k различных целых чисел b_1, b_2, \dots, b_k ($1 \leq b_j \leq 10^9$) — позиции, в которых изначально находятся ключи. Позиции заданы в произвольном порядке.

Обратите внимание, что изначально в каждой точке может находиться не более одного человека и не более одного ключа. Человек и ключ могут находиться в одной точке.

Выходные данные

Выведите минимальное время в секундах по истечении которого все n людей смогут добраться до офиса с ключами.

Примеры

входные данные	Скопировать
2 4 50 20 100 60 10 40 80	
выходные данные	Скопировать
50	
входные данные	Скопировать
1 2 10 11 15 7	
выходные данные	Скопировать
7	

Примечание

В первом примере человек, находящийся в точке 20 должен взять ключ, находящийся в точке 40 и дойти с ним до офиса, который находится в точке 50. На это он потратит 30 секунд. Человек, находящийся в точке 100 может взять ключ, находящийся в точке 80 и дойти с ним до офиса. На это он потратит 50 секунд. Таким образом, через 50 секунд все люди могут оказаться в офисе с ключами.

D. Слизни

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

В линию расположено n слизней. Слизни нумеруются от 1 до n слева направо. Размер i -го слизня равен a_i .

Каждую секунду происходит следующее: **ровно один** слизень съедает одного из своих соседей и увеличивает свой размер на значение размера соседа. Слизень может съесть своего соседа, только если он строго больше этого соседа. Если нет такого слизня, который больше хотя бы одного из его соседей — процесс завершается.

Например, предположим, что $n = 5$, $a = [2, 2, 3, 1, 4]$. Процесс может пройти следующим образом:

- сначала 3-й слизень съедает 2-го. Размер 3-го слизня становится равным 5, 2-й слизень съеден.
- затем 3-й слизень съедает 1-го (они соседи, так как 2-й уже съеден). Размер 3-го слизня становится равным 7, 1-й слизень съеден.
- затем 5-й слизень съедает 4-го. Размер 5-го слизня становится равным 5, 4-й слизень съеден.
- затем 3-й слизень съедает 5-го (они соседи, так как 4-й уже съеден). Размер 3-го слизня становится равным 12, 5-й слизень съеден.

Для каждого слизня вычислите минимальное количество секунд, за которое он может оказаться съеден каким-то другим слизнем (среди всех возможных вариантов, как может происходить процесс), или сообщите, что это невозможно.

Входные данные

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

Первая строка каждого набора содержит одно целое число n ($1 \leq n \leq 3 \cdot 10^5$) — количество слизней.

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

Сумма n по всем наборам входных данных не превышает $3 \cdot 10^5$.

Выходные данные

Для каждого набора входных данных выведите n целых чисел, где i -е число должно быть равно минимальному количеству секунд, необходимому для того, чтобы i -й слизень был съеден другим слизнем, или -1, если это невозможно.

Пример

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

B. Кобб

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

Вам даны n целых чисел a_1, a_2, \dots, a_n и целое число k . Найдите максимальное значение $i \cdot j - k \cdot (a_i | a_j)$ среди всех пар (i, j) целых чисел таких, что $1 \leq i < j \leq n$. Здесь $|$ обозначает операцию [побитового ИЛИ](#).

Входные данные

Первая строка содержит одно целое число t ($1 \leq t \leq 10\,000$) — количество наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа n ($2 \leq n \leq 10^5$) и k ($1 \leq k \leq \min(n, 100)$).

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

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

Выходные данные

Для каждого тестового случая выведите одно целое число — максимально возможное значение $i \cdot j - k \cdot (a_i | a_j)$.

Пример

входные данные	Скопировать
4 3 3 1 1 3 2 2 1 2 4 3 0 1 2 3 6 6 3 2 0 0 5 6	
выходные данные	Скопировать
-1 -4 3 12	

Примечание

Пусть $f(i, j) = i \cdot j - k \cdot (a_i | a_j)$.

В первом наборе входных данных,

- $f(1, 2) = 1 \cdot 2 - k \cdot (a_1 | a_2) = 2 - 3 \cdot (1|1) = -1$.
- $f(1, 3) = 1 \cdot 3 - k \cdot (a_1 | a_3) = 3 - 3 \cdot (1|3) = -6$.
- $f(2, 3) = 2 \cdot 3 - k \cdot (a_2 | a_3) = 6 - 3 \cdot (1|3) = -3$.

Таким образом, максимум равен $f(1, 2) = -1$.

В четвертом наборе входных данных максимум равен $f(3, 4) = 12$.



C. Нулевой путь

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

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

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

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

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

Входные данные

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

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

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

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

Выходные данные

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

Пример

входные данные	Скопировать
5 1 1 1 1 2 1 -1 1 4 1 -1 1 -1 3 4 1 -1 -1 -1 -1 1 1 -1 1 1 1 -1 3 4 1 -1 1 1 -1 1 -1 1 1 -1 1 1	
выходные данные	Скопировать
NO YES YES YES NO	

Примечание

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

C. Монстры и заклинания

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

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

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

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

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

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

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

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

Входные данные

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

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

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

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

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

Выходные данные

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

Пример

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

Примечание

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

Во втором наборе входных данных Монокарп может использовать заклинание через 3, 4 и 5 секунд с начала уровня с уроном

D. Ловушки

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

Как известно, в мире всегда есть недобросовестные люди. К сожалению, иногда такие люди делают ревью, и самый умный человек на земле (естественно, Тётя Люсине) придумала хитрейший способ с ними бороться: ханипоты! Но Тётя Люсине пошла ещё дальше: она замаскировала все ханипоты под «ловушки»! Теперь вам, как самому лучшему ревьюверу, нужно через них пробраться.

Вам нужно преодолеть n ловушек, пронумерованных от 1 до n . Вы будете проходить через ловушки по очереди. i -я ловушка наносит вам a_i базового урона.

Вместо того, чтобы проходить через ловушку, вы можете перепрыгнуть ее. Всего вы можете перепрыгнуть не более k ловушек. В случае, если вы перепрыгиваете ловушку, вы не получаете от неё урона. Однако если вы перепрыгиваете какую-то ловушку, то это увеличивает урон каждой следующей ловушки на 1 (это бонусный урон).

Обратите внимание, что если вы перепрыгиваете ловушку, то вы не получаете от неё урона (ни базовый, ни бонусный), а что также бонусные уроны от прыжков складываются. Например, если вы проходите через i -ю ловушку с базовым уроном a_i , а до этого вы перепрыгнули 3 другие ловушки, то вы получите от неё $(a_i + 3)$ урона.

Определите, какое минимальный суммарный урон вы можете получить, если разрешается перепрыгнуть не больше k любых ловушек.

Входные данные

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

Первая строка каждого набора входных данных содержит два целых числа n и k ($1 \leq n \leq 2 \cdot 10^5$, $1 \leq k \leq n$) — количество ловушек и максимальное количество ловушек, которые вы можете перепрыгнуть.

Следующая строка содержит n целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$) — базовые уроны ловушек.

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

Выходные данные

Для каждого набора входных данных выведите одно число — минимальный суммарный урон, который можно получить, если можно перепрыгнуть не больше k любых ловушек.

Пример

входные данные	Скопировать
5 4 4 8 7 1 4 4 1 5 10 11 5 7 5 8 2 5 15 11 2 8 6 3 1 2 3 4 5 6 1 1 7	
выходные данные	Скопировать
0 21 9 6 0	

Примечание

В первом наборе входных данных можно перепрыгнуть все ловушки и получить 0 урона.

Во втором наборе входных данных есть 5 способов перепрыгнуть ловушки:

1. Не перепрыгивать никакие ловушки.
Суммарный урон: $5 + 10 + 11 + 5 = 31$.

2. Перепрыгнуть 1-ю ловушку.

Суммарный урон: $\underline{0} + (10 + 1) + (11 + 1) + (5 + 1) = 29$.

3. Перепрыгнуть 2-ю ловушку.

Суммарный урон: $5 + \underline{0} + (11 + 1) + (5 + 1) = 23$.

4. Перепрыгнуть 3-ю ловушку.

Суммарный урон: $5 + 10 + \underline{0} + (5 + 1) = 21$.

5. Перепрыгнуть 4-ю ловушку.

Суммарный урон: $5 + 10 + 11 + \underline{0} = 26$.

Чтобы получить минимальный урон, необходимо перепрыгнуть 3-ю ловушку, тогда ответ будет 21.

В третьем наборе входных данных оптимально перепрыгнуть через ловушки с номерами 1, 3, 4, 5, 7:

Суммарный урон: $0 + (2 + 1) + 0 + 0 + 0 + (2 + 4) + 0 = 9$.

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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:30:01 UTC+5 (j2).

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

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

При поддержке



C. Очередной Турнир

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вы участвуете в турнире под названием «Очередной Турнир». В турнире участвуют $n + 1$ человек: вы и n ваших соперников, пронумерованных от 1 по n .

Каждый участник сыграет против каждого ровно один раз. Если соперник i состязается с соперником j , он выигрывает тогда и только тогда, когда $i > j$.

Если же оппонент i играет против вас, все становится немного сложнее. Для того чтобы победить оппонента i , вам нужно готовиться к матчу на протяжении a_i минут. В противном случае вы проиграете данному сопернику.

У вас есть m свободных минут для подготовки к матчам, но вы можете готовиться одновременно только к одному матчу.

Другими словами, если вы хотите выиграть у оппонентов p_1, p_2, \dots, p_k , вы должны потратить на подготовку

$a_{p_1} + a_{p_2} + \dots + a_{p_k}$ минут. Если же это число больше m , вы не сможете их всех победить.

Финальное место каждого участника равно количеству участников со строго большим количеством побед + 1. Например, если 3 участника выиграли по 5 раз, 1 участник — 3 раза, и 2 участника — по 1 разу, то первые 3 займут 1-е место, четвертый — 4-е место, и два последних — 5-е место.

Посчитайте минимально возможное место (меньше — лучше), которое вы сможете занять, если на подготовку к матчам у вас всего m минут.

Входные данные

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

В первой строке каждого набора входных данных заданы два целых числа n и m ($1 \leq n \leq 5 \cdot 10^5$; $0 \leq m \leq \sum_{i=1}^n a_i$) — количество ваших оппонентов и общее время, которое у вас есть на подготовку к матчам.

Во второй строке каждого набора заданы n целых чисел a_1, a_2, \dots, a_n ($0 \leq a_i \leq 1000$), где a_i — это количество минут, необходимое на подготовку к матчу с i -м оппонентом.

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

Выходные данные

Для каждого набора входных данных выведите наименьшее возможное место, которое вы сможете занять, если на подготовку у вас есть только m минут.

Пример

входные данные	Скопировать
5 4 401 100 100 200 1 3 2 1 2 3 5 0 1 1 1 1 1 4 0 0 1 1 1 4 4 1 2 2 1	
выходные данные	Скопировать
1 2 6 4 1	

Примечание

В первом наборе входных данных у вас есть время на подготовку ко всем оппонентам, а потому вы выигрываете все 4 матча и займете 1-е место, так как все ваши оппоненты выигрывают не более 3 матчей.

Во втором наборе вы можете подготовиться ко второму оппоненту и выиграть у него. В результате вы выигрываете 1 матч, оппонент 1 — 1 матч, оппонент 2 — 1 матч, оппонент 3 — 3 матча. А потому оппонент 3 займет 1-е место, и все остальные,

включая вас, — 2-е место.

В третьем наборе входных у вас нет времени на подготовку, а потому вы проиграете все матчи. Так как у каждого из оппонентов будет хотя бы 1 победа, вы займете последнее место (место 6).

В четвертом наборе у вас нет времени на подготовку, но вы все равно выиграете у первого оппонента. В результате оппонент 1 не выиграл ни одного матча, у вас 1 победа, и у всех остальных хотя бы 2 победы. А потому ваше место 4.

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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:30:04^{UTC+5} (j2).

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

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

При поддержке



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

B. Нейтральная тональность

ограничение по времени на тест: 3 секунды

ограничение по памяти на тест: 512 мегабайт

Вам дан массив a из n целых положительных чисел, а также массив b из m целых положительных чисел.

Пусть $LIS(c)$ обозначает длину самой длинной строго возрастающей подпоследовательности массива c . Например, $LIS([2, 1, 1, 3]) = 2$, $LIS([1, 7, 9]) = 3$, $LIS([3, 1, 2, 4]) = 3$.

Вам нужно вставить числа b_1, b_2, \dots, b_m в массив a , в любые места, в любом порядке. Пусть после вставки массив будет равен c_1, c_2, \dots, c_{n+m} . Вам нужно выбрать места для вставки так, чтобы **минимизировать** $LIS(c)$.

Формально, вам нужно найти такой массив c_1, c_2, \dots, c_{n+m} , для которого одновременно выполняются следующие условия:

- Массив a_1, a_2, \dots, a_n является подпоследовательностью массива c_1, c_2, \dots, c_{n+m} .
- Массив c_1, c_2, \dots, c_{n+m} состоит из чисел $a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_m$, возможно, переставленных в другом порядке.
- Значение $LIS(c)$ **минимально** возможное среди всех подходящих массивов c .

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 целые числа n, m ($1 \leq n \leq 2 \cdot 10^5, 1 \leq m \leq 2 \cdot 10^5$) — длину массива a и длину массива b .

Вторая строка каждого набора входных данных содержит n целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$) — элементы массива a .

Третья строка каждого набора входных данных содержит m целых чисел b_1, b_2, \dots, b_m ($1 \leq b_i \leq 10^9$) — элементы массива b .

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

Выходные данные

Для каждого набора входных данных выведите $n + m$ чисел — элементы итогового массива c_1, c_2, \dots, c_{n+m} , для которого значение $LIS(c)$ является минимальным из возможных. Если подходящих массивов несколько, вы можете вывести любой из них.

Пример

входные данные	Скопировать
7 2 1 6 4 5 5 5 1 7 2 4 5 5 4 1 2 7 1 9 7 1 2 3 4 5 6 7 8 9 3 2 1 3 5 2 4 10 5 1 9 2 3 8 1 4 7 2 9 7 8 5 4 6 2 1 2 2 1 6 1 1 1 1 1 1 777	
выходные данные	Скопировать
6 5 4 1 1 7 7 2 2 4 4 5 5 9 8 7 7 6 5 4 3 2 1 1 3 5 2 4	

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

Примечание

В первом наборе входных данных $\text{LIS}(a) = \text{LIS}([6, 4]) = 1$. Можно вставить число 5 между 6 и 4, тогда $\text{LIS}(c) = \text{LIS}([6, 5, 4]) = 1$.

Во втором наборе входных данных $\text{LIS}([1, 7, 2, 4, 5]) = 4$. После вставки, $c = [1, 1, 7, 7, 2, 2, 4, 4, 5, 5]$. Несложно видеть, что $\text{LIS}(c) = 4$. Можно показать, что значения $\text{LIS}(c)$, меньшего чем 4, добиться невозможно.

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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:30:06^{UTC+5} (j2).

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

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

При поддержке



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

E. Красивый массив

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Вам дан массив целых чисел a_1, a_2, \dots, a_n и целое число k . Вам нужно сделать его красивым за наименьшее количество операций.

До применения операций вы можете перемешать элементы массива как угодно. За одну операцию вы можете сделать следующее:

- Выбрать индекс $1 \leq i \leq n$,
- Сделать $a_i = a_i + k$.

Массив b_1, b_2, \dots, b_n является красивым, если $b_i = b_{n-i+1}$ для всех $1 \leq i \leq n$.

Найдите наименьшее количество операций, за которое можно сделать массив красивым, или сообщите, что это невозможно.

Входные данные

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

Первая строка каждого набора входных данных содержит два целых числа n и k ($1 \leq n \leq 10^5$, $1 \leq k \leq 10^9$) — размер массива a и число k из условия задачи.

Вторая строка каждого набора входных данных содержит n целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$) — элементы массива a .

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

Выходные данные

Для каждого набора входных данных выведите наименьшее количество операций, которое нужно сделать, чтобы массив стал красивым, или -1 , если это невозможно.

Пример

входные данные	Скопировать
11 1 1000000000 1 2 1 624323799 708290323 3 1 3 2 1 4 1 7 1 5 3 5 1 11 2 15 7 10 7 1 1 8 2 16 8 16 31 13 1 2 1 1 3 3 11 12 22 45 777 777 1500 74 10 2 1 2 1 2 1 2 1 2 1 2 11 2 1 2 1 2 1 2 1 2 1 2 1 13 3 2 3 9 14 17 10 22 20 18 30 1 4 28 5 1 2 3 5 3 5	

выходные данные

[Скопировать](#)

```
0
83966524
1
4
6
1
48
-1
0
14
0
```

Примечание

В первом наборе входных данных массив уже является красивым.

Во втором наборе входных данных можно не перемешивать массив до операций и 83966524 раз сделать операцию с индексом $i = 1$.

В третьем наборе входных данных можно перемешать массив a и сделать его равным $[2, 3, 1]$. После этого применить операцию с индексом $i = 3$, чтобы получился массив $[2, 3, 2]$, который является красивым.

В восьмом наборе входных данных не существует набора операций и способа перемешать элементы, чтобы сделать массив красивым.

В девятом наборе входных данных массив уже является красивым.

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов
Соревнования по программированию 2.0
Время на сервере: 05.07.2025 09:30:07^{UTC+5} (j2).
Мобильная версия, переключиться на [десктопную](#).
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



ИТМО

С. Полей деревья

ограничение по времени на тест: 3 секунды
ограничение по памяти на тест: 256 мегабайт

В парке находятся n деревьев, пронумерованных от 1 до n . Изначальная высота i -го дерева равна h_i .

Вы хотите полить эти деревья таким образом, чтобы они все выросли до **одинаковой** высоты.

Процесс полива происходит следующим образом. Вы начинаете поливать деревья в день 1. В течение j -го дня вы можете:

- Выбрать дерево и полить его. Если день нечетный (например, 1, 3, 5, 7, ...), то высота дерева увеличится на 1. Если день четный (например, 2, 4, 6, 8, ...), то высота дерева увеличится на 2.
- Или пропустить день без полива какого-либо дерева.

Заметьте, что вы не можете поливать больше одного дерева в день.

Ваша задача — найти **минимальное** количество дней, необходимое для того, чтобы все деревья выросли до одинаковой высоты.

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.

Первая строка набора тестовых данных содержит одно целое число n ($1 \leq n \leq 3 \cdot 10^5$) — количество деревьев.

Вторая строка набора тестовых данных содержит n целых чисел h_1, h_2, \dots, h_n ($1 \leq h_i \leq 10^9$), где h_i равно высоте i -го дерева.

Гарантируется, что сумма n по всем наборам тестовых данных не превосходит $3 \cdot 10^5$ ($\sum n \leq 3 \cdot 10^5$).

Выходные данные

Для каждого набора тестовых данных выведите одно целое число — **минимальное** количество дней, необходимое для того, чтобы все деревья выросли до одинаковой высоты.

Пример

входные данные	Скопировать
3 3 1 2 4 5 4 4 3 5 5 7 2 5 4 8 3 7 4	
выходные данные	Скопировать
4 3 16	

Примечание

Рассмотрим первый набор тестовых данных примера. Изначальное состояние деревьев выглядит так $[1, 2, 4]$.

- В течение первого дня польем первое дерево, таким образом последовательность высот станет равна $[2, 2, 4]$;
- в течение второго дня польем второе дерево, таким образом последовательность высот станет равна $[2, 4, 4]$;
- пропустим третий день;
- в течение четвертого дня давайте польем первое дерево, таким образом последовательность высот станет равна $[4, 4, 4]$.

Таким образом, ответ равен 4.

E. MEX и инкременты

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

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

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

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

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

Входные данные

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

Далее следуют описания наборов входных данных.

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

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

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

Выходные данные

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

Пример

входные данные	Скопировать
5 3 0 1 3 7 0 1 2 3 4 3 2 4 3 0 0 0 7 4 6 2 3 5 0 5 5 4 0 1 0 4	
выходные данные	Скопировать
1 1 0 -1 1 1 2 2 1 0 2 6 3 0 1 4 3 1 0 -1 -1 -1 -1 -1 -1 2 1 0 2 -1 -1	

Примечание

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

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

C. Функция Айоуба

ограничение по времени на тест: 1 секунда

ограничение по памяти на тест: 256 мегабайт

Айоуб думает, что он очень умный человек, поэтому он придумал функцию $f(s)$, где s это бинарная строка (строка, содержащая только символы «0» и «1»). Функция $f(s)$ равна количеству подстрок строки s , которые содержат хотя бы один символ, равный «1».

Более формально, $f(s)$ равно количеству пар целых чисел (l, r) , таких что $1 \leq l \leq r \leq |s|$ (где $|s|$ равно длине строки s), таких что хотя бы один из символов s_l, s_{l+1}, \dots, s_r равен «1».

Например, если $s = <01010>$, то $f(s) = 12$, потому что есть 12 таких пар (l, r) :
 $(1, 2), (1, 3), (1, 4), (1, 5), (2, 2), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 4), (4, 5)$.

Айоуб также думает, что он умнее Махмуда, поэтому дал ему два целых числа n и m и следующую задачу. По всем бинарным строкам s длины n , которые содержат ровно m символов, равных «1», найдите максимальное значение $f(s)$.

У Махмуда не получилось решить эту задачу, поэтому он попросил вас о помощи. Можете ли вы помочь ему?

Входные данные

Входные данные состоят из нескольких тестовых случаев. Первая строка содержит единственное целое число t ($1 \leq t \leq 10^5$) — количество тестовых случаев. Далее следует описание тестовых случаев в следующем формате.

Единственная строка для каждого тестового случая содержит два целых числа n и m ($1 \leq n \leq 10^9$, $0 \leq m \leq n$) — длина строки и количество символов, равных «1» в ней.

Выходные данные

Для каждого тестового случая выведите единственное целое число — максимальное значение $f(s)$ по всем строкам s длины n , которые содержат ровно m символов, равных «1».

Пример

входные данные	Скопировать
5 3 1 3 2 3 3 4 0 5 2	
выходные данные	Скопировать
4 5 6 0 12	

Примечание

В первом тестовом случае, существует только 3 строки длины 3, которые содержат ровно 1 символ, равный «1». Эти строки это: $s_1 = <100>$, $s_2 = <010>$, $s_3 = <001>$. Значения f для них это: $f(s_1) = 3$, $f(s_2) = 4$, $f(s_3) = 3$, поэтому максимальное значение это 4 и ответ равен 4.

Во втором тестовом случае, строка s для которой максимальное значение функции это «101».

Во третьем тестовом случае, строка s для которой максимальное значение функции это «111».

В четвертом тестовом случае, единственная строка s длины 4, которая содержит ровно 0 символов, равных «1» это «0000» и значение функции f для этой строки это 0, поэтому ответ равен 0.

В пятом тестовом случае, строка s с максимальным значением функции это «01010» и она была подробно разобрана в качестве примера в тексте условия задачи.

D. Дилемма обмена

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

Даны два массива a и b длины n , состоящих из различных целых положительных чисел, и мы хотим сделать массивы равными. Два массива x и y длины k называются равными, если для всех $1 \leq i \leq k$ выполняется $x_i = y_i$.

За один ход можно выбрать некоторые индексы l и r в a ($l \leq r$) и поменять местами a_l и a_r , затем выбрать некоторые p и q ($p \leq q$) в b такие, что $r - l = q - p$, и поменять местами b_p и b_q .

Можно ли сделать массивы одинаковыми?

Входные данные

Каждый тест состоит из нескольких наборов входных данных. Первая строка содержит одно целое число t ($1 \leq t \leq 2 \cdot 10^4$) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит одно целое число n ($1 \leq n \leq 10^5$) — длину массивов a и b .

Вторая строка каждого набора входных данных содержит n различных целых чисел $a_1, a_2, a_3, \dots, a_n$ ($1 \leq a_i \leq 2 \cdot 10^5$) — массив a .

Третья строка каждого набора входных данных содержит n различных целых чисел $b_1, b_2, b_3, \dots, b_n$ ($1 \leq b_i \leq 2 \cdot 10^5$) — массив b .

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

Выходные данные

Для каждого набора входных данных выведите «YES», если массивы a и b можно сделать одинаковыми. В противном случае выведите «NO».

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
6	
4	
1 2 3 4	
1 2 3 4	
5	
1 3 4 2 5	
7 1 2 5 4	
4	
1 2 3 4	
4 3 2 1	
3	
1 2 3	
1 3 2	
5	
1 5 7 1000 4	
4 1 7 5 1000	
3	
1 4 2	
1 3 2	
выходные данные	Скопировать
YES	
NO	
YES	
NO	
NO	
NO	

Примечание

В первом наборе входных данных не нужно выполнять никаких операций, поскольку массивы уже равны.

Для второго набора входных данных можно доказать, что не существует способа сделать массивы одинаковыми.

В третьем наборе входных данных один из способов сделать массивы равными такой: на первом ходу выбрать $l = 1, r = 3$,

E. Длинное инвертирование

ограничение по времени на тест: 3 секунды
ограничение по памяти на тест: 256 мегабайт

Дана бинарная строка s из n символов. Бинарной строкой называется строка, состоящая только из символов '1' и '0'.

Вы можете выбрать целое число k ($1 \leq k \leq n$), после чего применить любое раз следующую операцию: выбрать k подряд идущих символов строки и инвертировать их, то есть заменить все '0' на '1' и наоборот.

С помощью операций нужно сделать все символы в строке равными '1'.

Например, если $n = 5$, $s = 00100$, то можно выбрать $k = 3$ и действовать следующим образом:

- выбрать отрезок от 1-го до 3-го символа и получить $s = 11000$;
- выбрать отрезок от 3-го до 5-го символа и получить $s = 11111$;

Найдите максимальное значение k , при котором можно с помощью операций сделать все символы строки равными '1'.

Обратите внимание, что количество операций, которые для этого нужно совершить, не имеет значения.

Входные данные

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

Первая строка каждого набора содержит одно целое число n ($1 \leq n \leq 5000$) — длину строки s .

Вторая строка каждого набора содержит строку s длины n , состоящую из символов '1' и '0'.

Гарантируется, что сумма значений n^2 по всем наборам входных данных в тесте не превосходит $25 \cdot 10^6$.

Выходные данные

Для каждого набора входных данных выведите максимальное целое число k ($1 \leq k \leq n$), при котором возможно с помощью описанных операций получить строку s , состоящую только из символов '1'.

Пример

входные данные	Скопировать
5 5 00100 5 01000 7 1011101 3 000 2 10	
выходные данные	Скопировать
3 2 4 3 1	



B2. Букет (сложная версия)

ограничение по времени на тест: 1.5 секунд

ограничение по памяти на тест: 256 мегабайт

Это сложная версия задачи. Единственное отличие в том, что в этой версии вместо перечисления количеств лепестков у каждого цветка задано для всех типов цветов (по количеству лепестков) их количество в магазине.

Девочка готовится к своему дню рождения и хочет составить невероятной красоты букет. В магазине представлено в общей сложности n различных типов цветов, для каждого из перечисленных типов цветов указано их количество в магазине. Цветок с k лепестками стоит k монет. Девочка решила, что разница в количестве лепестков между любыми двумя цветами, которые она будет использовать для составления букета, не должна превышать единицы. В то же время девочка хочет собрать букет с максимально возможным количеством лепестков. К сожалению, у неё есть только m монет, и она не может потратить больше. С каким максимальным количеством лепестков она может собрать букет?

Входные данные

Каждый тест состоит из нескольких тестовых наборов. В первой строке содержится одно целое число t ($1 \leq t \leq 10\,000$) — количество тестовых наборов. Далее следует описание тестовых наборов.

Первая строка каждого тестового примера содержит два целых числа n, m ($1 \leq n \leq 2 \cdot 10^5, 1 \leq m \leq 10^{18}$) — количество типов цветов в магазине и количество монет у девочки. Вторая строка каждого тестового примера содержит n целых **различных** чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$), где a_i — это количество лепестков у i -го типа цветов в магазине (для различных индексов $i \neq j$ гарантируется, что количество лепестков у соответствующих типов цветов различное, то есть $a_i \neq a_j$). Третья строка каждого тестового примера содержит n целых чисел c_1, c_2, \dots, c_n ($1 \leq c_i \leq 10^9$), где c_i — количество цветов i -го типа в магазине.

Сумма n по всем тестовым случаям не превышает $2 \cdot 10^5$.

Выходные данные

Для каждого набора входных данных выведите одно целое число — максимально возможное количество лепестков в букете, который может собрать девочка, соблюдая все перечисленные выше условия.

Пример

входные данные	Скопировать
<pre>7 3 10 1 2 3 2 2 1 3 1033 206 207 1000 3 4 1 6 20 4 2 7 5 6 1 1 2 1 3 1 7 8 100000 239 30 610 122 24 40 8 2 12 13123 112 1456 124 100 123 10982 6 13 2 4 11 1 3 5 2 2 1 2 2 1 8 10330 206 210 200 201 198 199 222 1000 9 10 11 12 13 14 15 16 2 10000000000 11 12 87312315 753297050</pre>	
выходные данные	Скопировать
<pre>7 1033 19 99990 13 10000 9999999999</pre>	

Примечание

В первом наборе входных данных некоторые разрешенные букеты — это $(1, 1, 2, 2)$, $(2, 2, 3)$, $(1, 1)$, $(2, 2)$. Максимум из разрешенных букетов, не превышающий 10, составляет 7 для $(2, 2, 3)$. Во втором наборе входных данных вы можете собрать

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

D. Покраска массива

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Дан массив из n целых чисел, каждое из которых — 0, 1 или 2. Изначально каждый элемент массива покрашен в синий цвет.

Ваша цель — покрасить все элементы в красный цвет. Для этого вы можете выполнять операции двух типов:

- заплатить одну монету, выбрать синий элемент и покрасить его в красный цвет;
- выбрать красный элемент, не равный 0, и **соседний** с ним синий элемент, уменьшить выбранный красный элемент на 1, а выбранный синий элемент покрасить в красный цвет.

Какое минимальное количество монет вам придется потратить, чтобы покрасить все элементы в красный?

Входные данные

В первой строке задано одно целое число n ($1 \leq n \leq 2 \cdot 10^5$).

Во второй строке заданы n целых чисел a_1, a_2, \dots, a_n ($0 \leq a_i \leq 2$).

Выходные данные

Выведите одно целое число — минимальное количество монет, которое вам придется потратить, чтобы покрасить все элементы массива в красный цвет.

Примеры

входные данные	Скопировать
3 0 2 0	
выходные данные	Скопировать
1	
входные данные	Скопировать
4 0 0 1 1	
выходные данные	Скопировать
2	
входные данные	Скопировать
7 0 1 0 0 1 0 2	
выходные данные	Скопировать
4	

Примечание

В первом примере из условия можно покрасить все элементы в красный за одну монету следующим образом:

1. покрасить 2-й элемент в красный за одну монету;
2. уменьшить 2-й элемент на 1 и покрасить 1-й элемент в красный;
3. уменьшить 2-й элемент на 1 и покрасить 3-й элемент в красный.

Во втором примере из условия можно покрасить все элементы в красный за две монеты следующим образом:

1. покрасить 4-й элемент в красный за одну монету;
2. уменьшить 4-й элемент на 1 и покрасить 3-й элемент в красный;
3. покрасить 1-й элемент в красный за одну монету;
4. уменьшить 3-й элемент на 1 и покрасить 2-й элемент в красный.

D. Омкар и Война кроватей

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

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

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

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

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

Входные данные

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

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

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

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

Выходные данные

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

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

Пример

входные данные	выходные данные
5 4 RLRL 6 LRRRRL 8 RLLRRRL 12 LLLLRRLRRRL 5 RRRRR	0 1 1 3 2

Примечание

В первом наборе входных данных игроки 1 и 2 атакуют друг друга, а игроки 3 и 4 — друг друга. Каждый игрок атакует ровно 1 другого игрока, и каждый игрок атакует игрока, который атакует их, так что все игроки уже действуют логично в соответствии со стратегией Войны кроватей, и Омкару не нужно разговаривать ни с одним из них, следовательно, ответ 0.

C. Кот, Лиса и Двойной максимум

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Лиса любит перестановки! Она придумала следующую задачу, и предложила Коту её решить:

Вам дано **четное** целое положительное число n и перестановка[†] p длины n .

Счет другой перестановки q длины n определим как количество **локальных максимумов** в массиве a длины n , где $a_i = p_i + q_i$ для всех индексов i ($1 \leq i \leq n$). Другими словами, счет q — это количество i таких, что $1 < i < n$ (обратите внимание на **строгие неравенства**), $a_{i-1} < a_i$ и $a_i > a_{i+1}$ (еще раз обратите внимание на строгие неравенства).

Найдите перестановку q , которая достигает максимального счета при заданных n и p . Если таких перестановок несколько, вы можете вывести любую из них.

[†] Перестановкой длины n является массив, состоящий из n различных целых чисел от 1 до n в произвольном порядке. Например, $[2, 3, 1, 5, 4]$ — перестановка, но $[1, 2, 2]$ — не перестановка (2 встречается в массиве дважды) и $[1, 3, 4]$ тоже не перестановка ($n = 3$, но в массиве встречается 4).

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 одно **четное** целое число n ($4 \leq n \leq 10^5$, n — четное) — длину перестановки p .

Вторая строка каждого набора входных данных содержит n целых чисел p_1, p_2, \dots, p_n ($1 \leq p_i \leq n$). Гарантируется, что p действительно является перестановкой длины n .

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

Выходные данные

Для каждого набора входных данных выведите одну строку, содержащую любую перестановку чисел длины n (массив q) такую, что счёт q максимальен при заданных ограничениях.

Пример

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

Примечание

В первом примере $a = [3, 6, 4, 7]$. Массив имеет только один локальный максимум (на второй позиции), поэтому счет выбранной перестановки q равен 1. Можно доказать, что этот счет оптимальен при заданных ограничениях.

В последнем примере полученный массив $a = [6, 6, 12, 7, 14, 7, 14, 6]$ имеет 3 максимума — на третьей, пятой и седьмой позициях.

D. Цветные прямоугольники

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Даны три мультимножества пар цветных палок:

- R пар красных палок, у первой пары длины равны r_1 , у второй пары длины равны r_2, \dots , у R -й пары длины равны r_R ;
- G пар зеленых палок, у первой пары длины равны g_1 , у второй пары длины равны g_2, \dots , у G -й пары длины равны g_G ;
- B пар синих палок, у первой пары длины равны b_1 , у второй пары длины равны b_2, \dots , у B -й пары длины равны b_B .

Вы собираете прямоугольники из этих пар палок следующим образом:

1. взять пару палок одного цвета;
2. взять пару палок другого цвета, отличного от первого;
3. прибавить площадь полученного прямоугольника к суммарной площади.

В итоге получаются такие прямоугольники, что противоположные стороны у них одного цвета, а соседние стороны различных цветов.

Каждая пара палок может быть использована не более одного раза, некоторые пары можно не использовать. Не разрешается разбивать пару палок на отдельные палки.

Какую максимальную суммарную площадь можно получить?

Входные данные

В первой строке записаны три целых числа R, G, B ($1 \leq R, G, B \leq 200$) — количество пар красных палок, количество пар зеленых палок и количество пар синих палок.

Во второй строке записаны R целых чисел r_1, r_2, \dots, r_R ($1 \leq r_i \leq 2000$) — длины палок в каждой паре красных палок.

В третьей строке записаны G целых чисел g_1, g_2, \dots, g_G ($1 \leq g_i \leq 2000$) — длины палок в каждой паре зеленых палок.

В четвертой строке записаны B целых чисел b_1, b_2, \dots, b_B ($1 \leq b_i \leq 2000$) — длины палок в каждой паре синих палок.

Выходные данные

Выведите максимально возможную суммарную площадь построенных прямоугольников.

Примеры

входные данные	Скопировать
1 1 1 3 5 4	
выходные данные	Скопировать
20	
входные данные	Скопировать
2 1 3 9 5 1 2 8 5	
выходные данные	Скопировать
99	
входные данные	Скопировать
10 1 1 11 7 20 15 19 14 2 4 13 14 8 11	
выходные данные	Скопировать
372	

Примечание

В первом примере можно построить один из следующих прямоугольников: красный и зеленый со сторонами 3 и 5, красный и синий со сторонами 3 и 4 и зеленый и синий со сторонами 5 и 4. Лучшая площадь из них равна $4 \times 5 = 20$.

Во втором примере лучшие прямоугольники: красный/синий 9×8 , красный/синий 5×5 , зеленый/синий 2×1 . Суммарная площадь равна $72 + 25 + 2 = 99$.

В третьем примере лучшие прямоугольники: красный/зеленый 19×8 и красный/синий 20×11 . Суммарная площадь равна $152 + 220 = 372$. Обратите внимание, что больше прямоугольников нельзя построить, потому что не разрешается иметь обе взятые пары одного цвета.

[Codeforces](#) (c) Copyright 2010-2025 Михаил Мирзаянов
Соревнования по программированию 2.0
Время на сервере: 05.07.2025 09:36:35_{UTC+5} (k2).
Мобильная версия, переключиться на [десктопную](#).
[Privacy Policy](#) | [Terms and Conditions](#)

При поддержке



B. Каменная игра

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

Т играет в игру со своим другом HL.

Есть n кучек с камнями, в i -й из них исходно содержится a_i камней.

Т и HL будут ходить чередуясь, и Т ходит первым. В каждом ходу, игрок выбирает непустую кучу и удаляет из нее один камень. Однако, нельзя выбирать кучку, которая была выбрана на прошлом ходу (кучку которая была выбрана другим игроком, или если текущий ход это первый ход первого игрока, то можно выбрать любую кучу). Игрок, который не может выбрать кучу на своем ходу, проигрывает.

Считая, что оба игрока играют оптимально, для заданных стартовых конфигураций t игр, определите победителя.

Входные данные

В первой строке записано одно целое число t ($1 \leq t \leq 100$) — количество игр. Далее следуют описания игр, описание каждой игры состоит из двух строк:

В первой строке записано одно целое число n ($1 \leq n \leq 100$) — количество куч.

Во второй строке записаны n целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 100$).

Выходные данные

Для каждой игры, выведите в отдельной строке имя победителя, «T» или «HL» (без кавычек).

Пример

входные данные	Скопировать
2 1 2 2 1 1	
выходные данные	Скопировать
T HL	

Примечание

В первой игре, Т убирает один камень из единственной кучи. После этого, несмотря на то, что в куче еще остался 1 камень, HL не можем сходить, из-за того что Т использовал эту кучу на прошлом ходу. Таким образом, Т побеждает.



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

С. Шеф Монокарп

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт

Шеф Монокарп только что поставил n блюд на плиту. Он знает, что оптимальное время готовки i -го блюда равно t_i минут.

В любую **положительную целую** минуту T Монокарп может снять **не более одного** блюда с плиты. Если достать i -е блюдо в некоторую минуту T , то его испорченность будет равна $|T - t_i|$ — модуль значения разности между T и t_i . Как только блюдо снято с плиты, его уже нельзя поставить обратно.

Монокарп должен снять все блюда с плиты. Какую минимальную суммарную испорченность он может получить?

Входные данные

В первой строке записано одно целое число q ($1 \leq q \leq 200$) — количество наборов входных данных.

Затем идут q наборов входных данных.

В первой строке набора входных данных записано одно целое число n ($1 \leq n \leq 200$) — количество блюд на плите.

Во второй строке набора входных данных записаны n целых чисел t_1, t_2, \dots, t_n ($1 \leq t_i \leq n$) — оптимальное время готовки каждого блюда.

Сумма n по всем q наборам входных данных не превосходит 200.

Выходные данные

На каждый набор входных данных выведите одно целое число — минимальную суммарную испорченность блюд, которую может получить Монокарп, когда снимет все блюда с плиты. Помните, что Монокарп может снимать блюда только в положительные целые минуты и не более одного в минуту.

Пример

входные данные	Скопировать
6 6 4 2 4 4 5 2 7 7 7 7 7 7 7 7 1 1 5 5 1 2 4 3 4 1 4 4 4 21 21 8 1 4 1 5 21 1 8 21 11 21 11 3 12 8 19 15 9 11 13	
выходные данные	Скопировать
4 12 0 0 2 21	

Примечание

В первом наборе входных данных Монокарп может снять блюда в минуты 3, 1, 5, 4, 6, 2. Тогда суммарная испорченность будет равна $|4 - 3| + |2 - 1| + |4 - 5| + |4 - 4| + |6 - 5| + |2 - 2| = 4$.

Во втором наборе входных данных Монокарп может снять блюда в минуты 4, 5, 6, 7, 8, 9, 10.

В третьем наборе входных данных Монокарп может снять блюдо в минуте 1.

В четвертом наборе входных данных Монокарп может снять блюда в минуты 5, 1, 2, 4, 3.

В пятом наборе входных данных Монокарп может снять блюда в минуты 1, 3, 4, 5.

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

D2. Префиксно-суффиксный палиндром (усложненная версия)

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Это усложненная версия этой задачи. Различия в ограничениях на сумму длин строк и на количество тестовых случаев. Вы можете делать взломы, только если обе версии этой задачи решены.

Вам дана строка s , состоящая из строчных букв латинского алфавита. Вы должны найти строку t максимальной длины, которая удовлетворяет следующим условиям:

- Длина строки t не превосходит длину строки s .
- t является палиндромом, то есть она равна себе перевернутой.
- Существует две строки a и b (возможно пустые), такие что $t = a + b$ (здесь операция «+» это конкатенация строк) и a является префиксом строки s и b является суффиксом строки s .

Входные данные

Входные данные содержат несколько тестовых случаев. В первой строке находится единственное целое число t ($1 \leq t \leq 10^5$) — количество тестовых случаев. Следующие t строк описывают тестовые случаи.

Для каждого тестового случая, единственная строка содержит непустую строку s , состоящую из строчных символов латинского алфавита.

Гарантируется, что сумма длин строк по всем тестовым случаям не превосходит 10^6 .

Выходные данные

Для каждого тестового случая выведите строку максимальной длины, которая удовлетворяет описанным условиям. Если существует несколько решений, вы можете вывести любое.

Пример

входные данные	Скопировать
5 a abcdfdcecba abbaxyzyx codeforces acbba	
выходные данные	Скопировать
a abcdfdcba xyzyx c abba	

Примечание

В первом тестовом случае, вся строка $s = «a»$ удовлетворяет всем условиям.

Во втором тестовом случае, строка «abcdfdcba» удовлетворяет всем условиям, потому что:

- ее длина равна 9, что не превосходит длину строки s , которая равна 11;
- она палиндром;
- «abcdfdcba» = «abcdfdc» + «ba», где строка «abcdfdc» является префиксом строки s и строка «ba» является суффиксом строки s .

Можно доказать, что невозможно найти более длинную такую строку.

В четвертом тестовом случае, строка «C» это правильный ответ, потому что «C» = «C» + «» и это допустимо, потому что по условию строки a и b могут быть пустыми. Другой возможный ответ на этот тест это строка «S».

D. Кресла

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 512 мегабайт

В ряд стоят n кресел, пронумерованных от 1 до n слева направо. Некоторые кресла заняты людьми (не более одного человека на кресло), остальные свободны. Количество занятых мест не превосходит $\frac{n}{2}$.

По некоторым причинам вы хотите пересадить людей. Если i -е кресло занято, а j -е — нет, вы можете попросить человека, сидящего в i -м кресле, пересесть в j -е кресло. Время, которое потребуется на перемещение из i -го кресла в j -е, равно $|i - j|$ минутам. Эту операцию можно проводить любое количество раз, но операции должны выполняться последовательно, то есть вы не можете попросить человека пересесть, пока человек, которого вы попросили в предыдущей операции, еще не закончил перемещение.

Вы бы хотели добиться следующей ситуации: все кресла, которые были заняты в самом начале, должны оказаться свободными. За какое минимальное время вы сможете это сделать?

Входные данные

В первой строке задано одно целое число n ($2 \leq n \leq 5000$) — количество кресел.

Во второй строке заданы n целых чисел a_1, a_2, \dots, a_n ($0 \leq a_i \leq 1$). $a_i = 1$ означает, что i -е кресло изначально занято, $a_i = 0$ означает, что это кресло свободно. Количество занятых кресел не превосходит $\frac{n}{2}$.

Выходные данные

Выведите одно целое число — минимальное количество минут, за которое вы можете добиться следующей ситуации: все кресла, которые были заняты в самом начале, должны оказаться свободными.

Примеры

входные данные	Скопировать
7 1 0 0 1 0 0 1	
выходные данные	Скопировать
3	
входные данные	Скопировать
6 1 1 1 0 0 0	
выходные данные	Скопировать
9	
входные данные	Скопировать
5 0 0 0 0 0	
выходные данные	Скопировать
0	

Примечание

В первом примере возможна следующая последовательность действий:

1. попросить человека пересесть из кресла 1 в кресло 2 за 1 минуту;
2. попросить человека пересесть из кресла 7 в кресло 6 за 1 минуту;
3. попросить человека пересесть из кресла 4 в кресло 5 за 1 минуту.

Во втором примере возможна следующая последовательность действий:

1. попросить человека пересесть из кресла 1 в кресло 4 за 3 минуты;
2. попросить человека пересесть из кресла 2 в кресло 6 за 4 минуты;
3. попросить человека пересесть из кресла 4 в кресло 5 за 1 минуту;
4. попросить человека пересесть из кресла 3 в кресло 4 за 1 минуту.

В третьем примере ни одно место не занято, поэтому вам не надо тратить время.

C. Дядя Богдан и настроение страны

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Дядя Богдан, будучи матросом на корабле капитана Флинта, порой скучает по родине. Сегодня он рассказал о том, как в его государстве вводили индекс счастья населения.

Всего в стране n городов и $n - 1$ двусторонняя дорога, соединяющая некоторые пары городов. Из любого города можно попасть в любой другой, пройдя по некоторым дорогам. Города пронумерованы от 1 до n и город 1 является столицей. Таким образом, структура государства является корневым деревом.

Всего в стране проживает m человек. В i -м городе проживает p_i человек, но все работают в столице. После напряженного рабочего дня, каждый из жителей возвращается в свой город по кратчайшему пути.

У каждого жителя есть свое настроение: кто-то уходит с рабочего места в хорошем настроении, а у кого-то настроение уже испорчено. Более того, и по пути домой настроение жителя может испортиться. **Если настроение человека испортилось, то оно уже не может улучшиться.**

В каждом городе установлен детектор настроения — он отслеживает настроение **каждого**, кто побывал в этом городе. Детектор настроения i -го города считает индекс счастья h_i как количество человек в хорошем настроении минус количество в плохом. Для простоты будем считать, что *настроение жителя внутри города не меняется*.

Детектор настроения — это новая разработка и нет полной уверенности, что все приборы отработают должным образом. После того, как все жители страны добрались до своих городов, власти обратились к Дяде Богдану, лучшему программисту государства, с просьбой определить корректны ли данные индекса счастья всех городов или где-то закралась ошибка.

Дядя Богдан успешно справился с задачей. А удастся ли это Вам?

Более формально, Вам требуется определить: «*Возможно ли, что, после возвращения всех жителей по домам, для каждого города i будет верно, что индекс счастья в этом городе в точности равен h_i* ».

Входные данные

Первая строка содержит одно целое число t ($1 \leq t \leq 10000$) — количество наборов входных данных.

В первой строке каждого набора заданы два целых числа n и m ($1 \leq n \leq 10^5$; $0 \leq m \leq 10^9$) — количество городов и жителей в стране соответственно.

Во второй строке каждого набора заданы n целых чисел p_1, p_2, \dots, p_n ($0 \leq p_i \leq m$; $p_1 + p_2 + \dots + p_n = m$), где p_i — численность населения города i .

В третьей строке заданы n целых чисел h_1, h_2, \dots, h_n ($-10^9 \leq h_i \leq 10^9$), где h_i — индекс счастья города i .

В следующих $n - 1$ строках заданы описания дорог, по одному в строке. В каждой строке заданы два целых числа 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 городов, которые соединены i -й дорогой.

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 все детекторы настроения исправны или NO — иначе. Буквы в словах YES и NO можно выводить в любом регистре.

Примеры

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

1 3
1 4
3 5

входные данные

Скопировать

YES
YES

входные данные

Скопировать

2
4 4
1 1 1 1
4 1 -3 -1
1 2
1 3
1 4
3 13
3 3 7
13 1 4
1 2
1 3

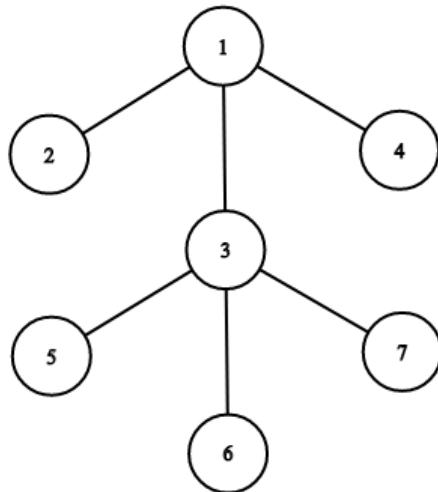
входные данные

Скопировать

NO
NO

Примечание

Рассмотрим первый набор входных данных первого теста:



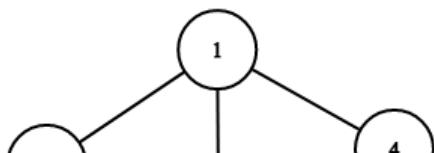
Под конец рабочего дня все жители страны находятся в столице. Рассмотрим один из возможных вариантов развития событий:

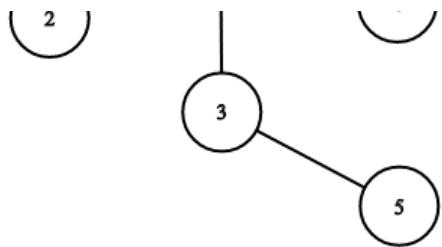
- житель города 1: живет в столице и его настроение не ухудшалось;
- житель города 4: посетил города 1 и 4, настроение ухудшилось между городами 1 и 4;
- житель города 3: посетил города 1 и 3 в хорошем настроении;
- житель города 6: посетил города 1, 3 и 6, настроение ухудшилось между городами 1 и 3;

Таким образом,

- $h_1 = 4 - 0 = 4$,
- $h_2 = 0$,
- $h_3 = 1 - 1 = 0$,
- $h_4 = 0 - 1 = -1$,
- $h_5 = 0$,
- $h_6 = 0 - 1 = -1$,
- $h_7 = 0$.

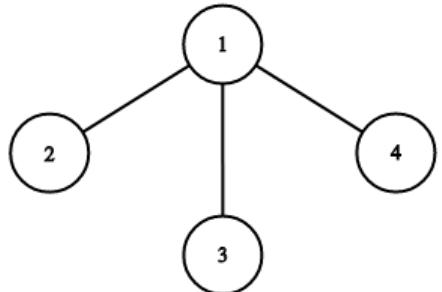
Второй набор первого теста:



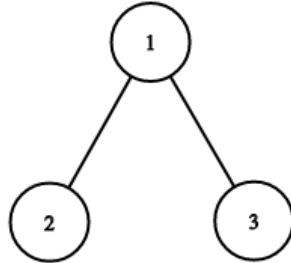


У всех жителей страны уже испортилось настроение в столице. Это единственный возможный вариант развития событий.

Первый набор второго теста:



Второй набор второго теста:



Можно показать, что требуемые значения индексов счастья недостижимы в обоих наборах второго теста.

D. Диана

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

Вам дано целое число n . Найдите любую строку s длины n , состоящую только из латинских строчных букв, такую, что каждая непустая подстрока s встречается в s нечетное количество раз. Если таких строк несколько, выведите любую. Можно показать, что такая строка всегда существует при заданных ограничениях.

Строка a является подстрокой b , если a может быть получена из b удалением нескольких (возможно, ни одного или всех) символов из начала и нескольких (возможно, ни одного или всех) символов из конца.

Входные данные

Первая строка содержит одно целое число t ($1 \leq t \leq 500$) — количество наборов входных данных.

Первая строка каждого набора входных данных содержит одно целое число n ($1 \leq n \leq 10^5$).

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

Выходные данные

Для каждого наборов входных данных выведите одну строку, содержащую строку s . Если таких строк несколько, выведите любую. Можно показать, что такая строка всегда существует при заданных ограничениях.

Пример

входные данные	Скопировать
4 3 5 9 19	
выходные данные	Скопировать
abc diane bbcaabbba youarethecutestuwuu	

Примечание

В первом наборе входных данных каждая подстрока «abc» встречается ровно один раз.

В третьем наборе входных данных каждая подстрока «bbcaabbba» встречается нечетное число раз. В частности, «b» встречается 5 раз, «a» и «bb» встречаются по 3 раза, а каждая из оставшихся подстрок встречается ровно один раз.



D. Максимально распределенное дерево

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

Вам задано дерево, состоящее из n вершин. Вы должны сопоставить каждому из $n - 1$ ребер этого дерева целое число таким образом, чтобы выполнялись следующие условия:

- каждое число должно быть целым и строго больше 0;
- произведение всех $n - 1$ чисел должно быть равно k ;
- количество 1-ц среди всех $n - 1$ чисел должно быть минимально возможным.

Назовем $f(u, v)$ сумму чисел на простом пути из вершины u в вершину v . Также, назовем $\sum_{i=1}^{n-1} \sum_{j=i+1}^n f(i, j)$ как индекс распределения дерева.

Определите максимально возможный индекс распределения, который можно получить. Так как ответ может быть слишком большим, выведите его по модулю $10^9 + 7$.

В данной задаче, так как число k может быть слишком большим, задана факторизация k на простые числа.

Входные данные

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

В первой строке каждого набора задано одно целое число n ($2 \leq n \leq 10^5$) — количество вершин в дереве.

В каждой из следующих $n - 1$ строк заданы ребра: в i -й строке заданы два целых числа u_i и v_i ($1 \leq u_i, v_i \leq n; u_i \neq v_i$) — номера вершин, соединенных i -м ребром.

В следующей строке задано одно число m ($1 \leq m \leq 6 \cdot 10^4$) — количество простых в факторизации k .

В следующей строке заданы m простых чисел p_1, p_2, \dots, p_m ($2 \leq p_i < 6 \cdot 10^4$) такие, что $k = p_1 \cdot p_2 \cdot \dots \cdot p_m$.

Гарантируется, что сумма n по всем наборам входных данных не превосходит 10^5 , сумма m по всем наборам не превосходит $6 \cdot 10^4$, и что заданные ребра в каждом наборе образуют дерево.

Выходные данные

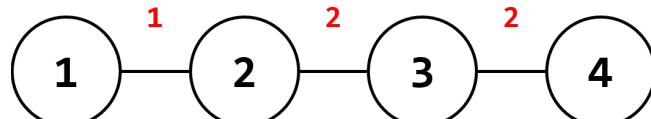
Выполните максимально возможный индекс распределения. Так как ответ может быть слишком большим, выведите его по модулю $10^9 + 7$.

Пример

входные данные	Скопировать
<pre>3 4 1 2 2 3 3 4 2 2 2 4 3 4 1 3 3 2 2 3 2 7 6 1 2 3 4 6 7 3 5 1 3 6 4 7 5 13 3</pre>	
выходные данные	Скопировать

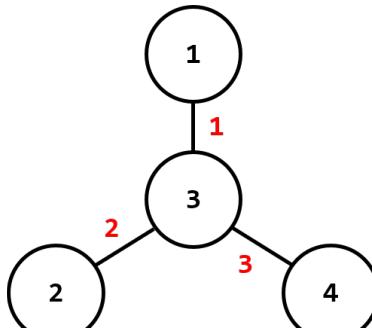
Примечание

В первом наборе входных данных, один из возможных ответов изображен на рисунке ниже:



В данном случае, $f(1, 2) = 1$, $f(1, 3) = 3$, $f(1, 4) = 5$, $f(2, 3) = 2$, $f(2, 4) = 4$, $f(3, 4) = 2$, и сумма этих 6 чисел равна 17.

Во втором наборе входных данных, один из возможных ответов изображен ниже:



В этом случае, $f(1, 2) = 3$, $f(1, 3) = 1$, $f(1, 4) = 4$, $f(2, 3) = 2$, $f(2, 4) = 5$, $f(3, 4) = 3$, и сумма этих 6 чисел равна 18.

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

Соревнования по программированию 2.0

Время на сервере: 05.07.2025 09:36:47 UTC+5 (k2).

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

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

При поддержке



ИТМО

C. Квадратирование

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

ikrpprpp нашел массив a , состоящий из целых чисел. Ему нравится справедливость, поэтому он хочет сделать a честным — то есть сделать его неубывающим. Для этого он может выполнить акт справедливости на индексе $1 \leq i \leq n$ массива, который заменит a_i на a_i^2 (элемент на позиции i на его квадрат). Например, если $a = [2, 4, 3, 3, 5, 3]$ и ikrpprpp решает выполнить акт справедливости на $i = 4$, то a становится $[2, 4, 3, 9, 5, 3]$.

Каково минимальное количество актов справедливости, необходимых для того, чтобы сделать массив неубывающим?

Входные данные

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

Для каждого набора входных данных первая строка содержит целое число n — размер массива a . Вторая строка содержит n ($1 \leq n \leq 2 \cdot 10^5$) целых числа a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^6$).

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

Выходные данные

Для каждого набора входных данных выведите целое число — минимальное количество актов справедливости, необходимых для того, чтобы сделать массив a неубывающим. Если это невозможно, выведите -1 .

Пример

входные данные	Скопировать
7 3 1 2 3 2 3 2 3 3 1 5 4 1 1 2 3 3 4 3 2 9 16 2 4 2 256 2 4 2 8 11 10010 10009 10008 10007 10006 10005 10004 10003 10002 10001 10000	
выходные данные	Скопировать
0 1 -1 0 3 15 55	

Примечание

В первом наборе входных данных нет необходимости выполнять акты справедливости. Массив сам по себе честен!

В третьем наборе входных данных можно доказать, что массив не может стать неубывающим.

В пятом наборе входных данных ikrpprpp может выполнить акт справедливости на индексе 3, затем акт справедливости на индексе 2 и, наконец, еще один акт справедливости на индексе 3. После этого a станет $[4, 9, 16]$.