Задача Peter Parker. Параллельные вселенные

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Берляндия — страна с очень развитой дорожной системой. Всего в Берляндии есть n городов, при том между каждой парой городов есть ровно одна дорога, доступная для движения в обе стороны.

В целях экономии электричества, в Берляндии освещены лишь m_1 дорог, i-я из них соединяет города v_i и u_i. Из соображений безопасности в Берляндии запрещено перемещаться по неосвещённым дорогам.

В параллельной вселенной есть аналогичная страна Черляндия, состоящая из n городов. В ней также между каждой парой городов есть ровно одна дорога. Страны отличаются только экономией электричества: в Черляндии освещены m_2 дорог, i-я из них соединяет города a_i и b_i. Известно, что в Черляндии можно доехать от любого города до любого, используя только освещённые дороги.

Вы владеете секретным заклинанием, позволяющим выбрать любые два различных города x и y и изменить освещённость на дороге между городами x и y в обеих вселенных. То есть в каждой из вселенных если дорога не была освещена, то она становится освещённой и наоборот.

Вы хотите использовать это заклинание не больше чем n раз так, чтобы в Берляндии можно было доехать от любого города до любого другого, используя только освещённые дороги. При этом после применения каждого заклинания Черляндия должна оставаться *связной*, то есть не должно существовать двух городов, между которыми нельзя проехать по освещённым дорогам.

Определите, можно ли этого достигнуть, и если да, то найдите подходящую последовательность заклинаний.

Формат входных данных

Каждый тест состоит из нескольких наборов входных данных. В первой строке находятся два целых числа t и g ($1 \leqslant t \leqslant 60\,000$, $0 \leqslant g \leqslant 10$) — количество наборов входных данных и номер группы тестов. Далее следуют описания наборов входных данных.

В первой строке описания каждого набора входных данных находятся три целых числа n, m_1 и m_2 ($3 \leqslant n \leqslant 300\,000$, $0 \leqslant m_1, m_2 \leqslant 300\,000$, $m_1, m_2 \leqslant \frac{n(n-1)}{2}$) — количество городов, количество освещённых дорог в Берляндии и количество освещённых дорог в Черляндии.

В следующих m_1 строках содержатся описания освещённых дорог в Берляндии. В i-й строке находятся два целых числа v_i и u_i ($1 \leqslant v_i, u_i \leqslant n$) — номера городов, соединённых i-й освещенной дорогой. Гарантируется, что все дороги различны.

В следующих m_2 строках содержатся описания освещённых дорог в Черляндии. В i-й строке находятся два целых числа a_i и b_i ($1 \leqslant a_i, b_i \leqslant n$) — номера городов, соединённых i-й освещенной дорогой. Гарантируется, что все дороги различны, и что в Черляндии между любыми двумя городами существует путь, проходящий только по освещенным дорогам.

Обозначим за N, M_1 и M_2 сумму n, m_1 и m_2 по всем наборам входных данных в одном тесте. Гарантируется, что $N, M_1, M_2 \leqslant 300\,000$.

Формат выходных данных

Для каждого набора входных данных выведите «`No`» (без кавычек), если не существует последовательности заклинаний, удовлетворяющей всем условиям.

В противном случае выведите «`Yes`». Во второй строке выведите целое число k ($0 \leqslant k \leqslant n$) — количество использованных вами заклинаний.

Далее выведите k строк. В i-й из них выведите два целых числа x_i и y_i ($1 \leqslant x_i, y_i \leqslant n$, $x_i \neq y_i$) — номера городов, к которым применяется i-е заклинание. Обратите внимание, что после применения каждого заклинания Черляндия должна оставаться *связной*.

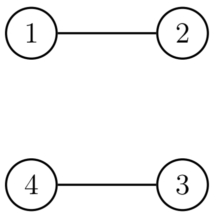
Пример

ввод	вывод
3 0 3 0 3 1 2 2 3 1 3 4 2 3 1 2 3 4 1 3 1 4 2 3 4 3 3 1 2 2 3 1 3 1 4 2 4 3 4	No Yes 1 2 4 Yes 2 1 2 4 2

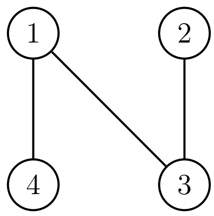
Пояснение

В первом наборе входных данных не существует ни одной подходящей последовательности заклинаний, поэтому ответ «No».

Во втором наборе входных данных освещённые дороги изначально имеют такую структуру:

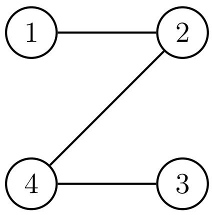


Берляндия

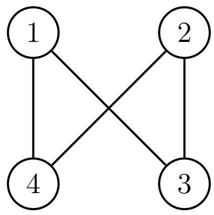


Черляндия

После применения заклинания к городам 2 и 4 и в Берляндии, и в Черляндии эта дорога становится освещённой, так как в обоих странах она была неосвещена. После этого, страны будут иметь следующую структуру:



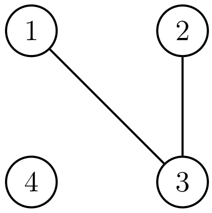
Берляндия



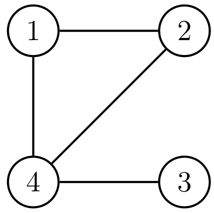
Черляндия

После этой операции, в Берляндии можно доехать от любого города до другого, поэтому такая последовательность заклинания корректна.

В третьем наборе входных данных, после применения заклинания к городам 1 и 2, в Берляндии дорога между этими двумя городами перестаёт быть освещённой, так как до этого она была освещена. В Черляндии наоборот — дорога становится освещённой. После этого, страны будут иметь следующую структуру структуру:

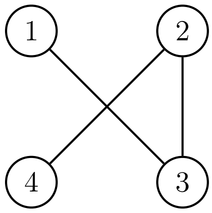


Берляндия

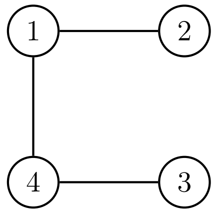


Черляндия

После применения заклинания к городам 2 и 4, старны будут иметь следующую структуру:



Берляндия



Черляндия

Система оценки

Тесты к этой задаче состоят из десяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, прохождение тестов из условия не требуется для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Доп. ограничения N, M_1, M_2	Необх. Группы	Комментарий
0	0	–	–	Тесты из условия.
1	9	$N, M_1, M_2 \leqslant 3000$	–	$n \leqslant 5$
2	7	$N, M_1, M_2 \leqslant 3000$	–	$m_2 = \frac{n(n-1)}{2}$
3	10	$N, M_1, M_2 \leqslant 3000$	–	Берляндия состоит из двух *компонент связности*[1].
4	11	$N, M_1, M_2 \leqslant 3000$	–	В Берляндии нет *изолированных*[2] городов.
5	15	$N, M_1, M_2 \leqslant 3000$	–	$m_2 = n - 1$ $a_i = 1$ и $b_i = i + 1$ для всех $1 \leqslant i \leqslant n - 1$
6	8	$N, M_1, M_2 \leqslant 3000$	5	$m_2 = n - 1$
7	12	$N, M_1, M_2 \leqslant 3000$	–	В обеих странах дорога между городами 1 и 2 освещена.
8	6	$N, M_1, M_2 \leqslant 3000$	0 – 7	
9	8	–	–	$m_2 = n - 1$ $a_i = i$ и $b_i = i + 1$ для всех $1 \leqslant i \leqslant n - 1$
10	14	–	0 – 9	**Offline-проверка.**

[1] *Компонента связности* — это множество городов, что между каждой парой из них можно доехать от одного до другого, используя только освещённые дороги.

[2] Город называется *изолированным*, если не существует освещённой дороги, соединяющей этот город с каким-то другим.

Задача Vito Corleone. Три массива

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Вам даны три массива D, L и R длины n, нумерация элементов которых ведется с 1, а также числа a_0 и b_0. Вы строите два массива A и B длины $n+1$ по следующим правилам:

1. $A_0 = a_0$, $B_0 = b_0$
2. Для всех i от 1 до n вы делаете следующие действия:
   (a) Задать элементы, как $A_i = A_{i-1} + D_i$ и $B_i = B_{i-1} + D_i$.
   (b) Выбрать ровно **одну** из операций ниже и применить её:
   - $A_i = \min(A_i, L_i)$
   - $B_i = \min(B_i, R_i)$

Вы хотите построить массивы A и B так, чтобы максимизировать значение $A_n + B_n$. Найдите максимальное значение $A_n + B_n$, которое можно получить, выполняя описанные выше действия.

Формат входных данных

В первой строке дано одно целое число n ($1 \leqslant n \leqslant 100\,000$) — длина массивов D, L и R.
Во второй строке даны n целых чисел $D_1,\ D_2,\ \ldots,\ D_n$ ($0 \leqslant D_i \leqslant 10^9$) — массив D.
В третьей строке даны n целых чисел $L_1,\ L_2,\ \ldots,\ L_n$ ($0 \leqslant L_i \leqslant 10^9$) — массив L.
В четвертой строке даны n целых чисел $R_1,\ R_2,\ \ldots,\ R_n$ ($0 \leqslant R_i \leqslant 10^9$) — массив R.
В пятой строке даны два целых числа a_0 и b_0 ($0 \leqslant a_0, b_0 \leqslant 10^9$).

Формат выходных данных

Выведите единственное число — максимально возможное значение A_n+B_n среди всех возможных вариантов построить массивы A и B.

Система оценки

Тесты к этой задаче состоят из шести групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, прохождение тестов из условия не требуется для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Доп. ограничения		Необх. группы	Комментарий
		n	D_i		
0	0	–	–	–	Тесты из условия.
1	13	$n \leqslant 15$	–	0	
2	18	$n \leqslant 300$	–	0, 1	
3	14	$n \leqslant 5000$	$D_i = 0$	–	
4	16	$n \leqslant 5000$	–	0–3	
5	19	–	$D_i = 0$	3	
6	20	–	–	0–5	**Offline-проверка.**

Пример

ввод	вывод
5 4 0 7 0 8 10 5 3 7 7 8 5 9 2 23 4 8	34

Пояснение

В первом наборе входных данных следующая последовательность действий приводит к максимальному ответу:

1. $A_0 = 4$, $B_0 = 8$.
2. $A_1 = A_0 + D_1 = 4 + 4 = 8$, $B_1 = B_0 + D_1 = 8 + 4 = 12$.
3. Минимум применяется для $A_1 = \min(A_1, L_1) = \min(10, 8) = 8$, значение $B_1 = 12$ остается прежним.
4. $A_2 = A_1 + D_2 = 8 + 0 = 8$, $B_2 = B_1 + D_2 = 12 + 0 = 12$.
5. Минимум применяется для $A_2 = \min(A_2, L_2) = \min(5, 8) = 5$, значение $B_2 = 12$ остается прежним.
6. $A_3 = A_2 + D_3 = 12$, $B_3 = B_2 + D_3 = 19$.
7. Минимум применяется для $A_3 = \min(A_3, L_3) = 3$, значение $B_3 = 19$ остается прежним.
8. $A_4 = A_4 + D_3 = 3$, $B_4 = B_3 + D_4 = 19$.
9. Минимум применяется для $A_4 = \min(A_4, L_4) = 3$, значение $B_4 = 19$ остается прежним.
10. $A_5 = A_5 + D_4 = 11$, $B_5 = B_4 + D_5 = 27$.
11. Значение $A_5 = 11$ остается прежним, $B_5 = \min(B_5, R_5) = \min(27, 23) = 23$
12. $A_5 + B_5 = 11 + 23 = 34$.

Можно показать, что это является максимальным значением.

Задача Tyler Durden. Бурёнка и Pether

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	3 секунды
Ограничение по памяти:	1024 мегабайта

Однажды принцесса Бурляндии Бурёнка, решила порадовать свою знакомую ReLu. Зная, что ReLu разделяет её интерес к криптовалюте, Бурёнка решила создать собственную блокчейн криптовалюту `Pether`.

Пройдя курсы и тренинги от эксперт-коуча по личностному росту в кибербезопастности, Бурёнка решила, что валюта `Pether` должна быть защищена наилучшим образом. В результате, из-за невероятно сложных и запутанных ограничений, далеко не все пользователи могут обмениваться `Pether` друг с другом.

Устройство блокчейн валюты `Pether` действительно сложное и запутанное. Все пользователи пронумерованы целыми числами от 1 до n. Каждому пользователю присвоен его **уникальный** идентификатор a_i. Также для валюты зафиксировано число d — параметр безопасности.

Пользователь i может напрямую перевести валюту пользователю j, только если $i < j$ и $a_i < a_j$. Но и этого недостаточно! Прямой перевод валюты между пользователями происходит посредством цепочки транзакций через некоторое число промежуточных пользователей. За время каждой транзакции номер каждого следующего промежуточного пользователя (включая последнего пользователя j) должен увеличиваться, но не более, чем на d. Также все промежуточные пользователи кроме i и j должны иметь идентификатор строго меньше a_i.

Более формально, пользователь i может напрямую перевести криптовалюту пользователю j, если выполнены следующие условия:

1. Выполнено, что $i < j$
2. Выполнено, что $a_i < a_j$
3. Существует последовательность промежуточных пользователей x длины k такая что:
   (a) $i = x_1 < x_2 < \ldots < x_{k-1} < x_k = j$
   (b) Для всех $1 \leqslant t \leqslant k-1$ верно, что $x_{t+1} - x_t \leqslant d$
   (c) Для всех $2 \leqslant t \leqslant k-1$ верно, что $a_{x_t} < a_i$

Буренка просит вас, своего знакомого программиста, разобраться в этой системе и выяснить для некоторых пар пользователей, как им передавать `Pether` друг другу.

Вам нужно ответить на q запросов. В каждом запросе вам требуется определить, существует ли последовательность прямых переводов валют (возможно через промежуточных пользователей), позволяющая передать `Pether` от пользователя u_i к пользователю v_i. В некоторых запросах также требуется минимизировать число прямых переводов валют в процессе отправки валюты от u_i к v_i. Обратите внимание, что число транзакций во время каждого прямого перевода минимизировать не требуется.

Формат входных данных

В первой строке даны три целых числа n, d и g ($1 \leqslant n, d \leqslant 300\,000, 0 \leqslant g \leqslant 11$) — количество пользователей, параметр безопасности и номер группы тестов.

Во второй строке даны n целых чисел $a_1,\ a_2,\ \ldots,\ a_n$ ($1 \leqslant a_i \leqslant n$) — идентификаторы пользователей. Гарантируется, что все числа a_i **различны**.

В третьей строке находится единственное целое число q ($1 \leqslant q \leqslant 300\,000$) — количество запросов

В следующих q строках дано по три целых числа $t_i,\ u_i,\ v_i$ ($t_i \in \{1, 2\}, 1 \leqslant u_i < v_i \leqslant n$), где u_i — пользователь, который должен отдать валюту, v_i — пользователь, который должен получить валюту. Если $t_i = 1$, то необходимо выяснить, возможно ли передать валюту, а если $t_i = 2$, то надо дополнительно минимизировать число прямых переводов.

Формат выходных данных

Выведите q строк, в i-й из них должен быть ответ на i-й запрос.

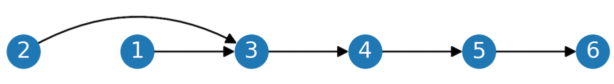
Если передать валюту от пользователя u_i к пользователю v_i нельзя, то в ответ на i-й запрос выведите 0. Иначе, если $t_i = 1$, то выведите 1, а если $t_i = 2$, то выведите минимальное число прямых переводов, требуемых для передачи `Pether` от u_i к v_i.

Примеры

ввод	вывод
6 1 0 2 1 3 4 5 6 6 2 1 3 2 1 2 1 1 4 2 1 5 2 1 6 1 2 6	1 0 1 3 4 1
6 2 0 1 2 3 4 5 6 6 2 1 5 2 2 5 2 1 6 2 2 6 2 1 4 2 2 4	2 2 3 2 2 1
10 2 0 2 1 4 3 5 6 8 7 10 9 10 2 1 5 1 2 5 2 3 5 2 1 9 2 5 8 2 3 9 2 1 8 1 1 2 2 3 8 2 1 9	2 1 1 4 2 3 3 0 2 4

Пояснение

В первом примере возможны следующие прямые переводы между пользователями:



В первом запросе пользователь с индексом 1 может напрямую перевести `Pether` пользователю с индексом 3, сделав 2 транзакции через промежуточного пользователя 2.

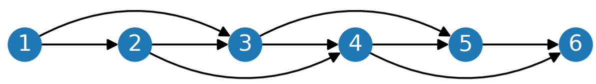
Во втором запросе прямой перевод между пользователями с индексами 1 и 2 невозможен, так как $a_1 = 2 > a_2 = 1$.

В третьем запросе можно перевести валюту от пользователя 1 к пользователю 4 с помощью двух прямых переводов, в начале переведя валюту от пользователя 1 к пользователю 3, а потом от 3 к

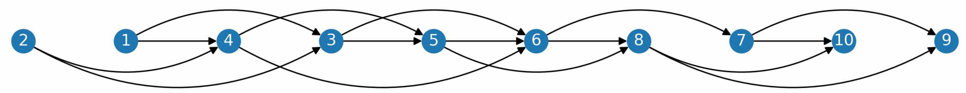
4. Так как $t_3 = 1$, то требуется узнать лишь наличие возможности передачи валюты, поэтому ответ на запрос равен 1.

В четвертом запросе можно обойтись тремя прямыми переводами: от 1 к 3, от 3 к 4 и от 4 к 5.

Во втором примере возможны следующие прямые переводы между пользователями:



В третьем примере возможны следующие прямые переводы между пользователями:



Система оценки

Тесты к этой задаче состоят из двенадцати групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, прохождение тестов из условия не требуется для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Доп. ограничения			Необх. группы	Комментарий
		n	q	v_i, a_n, t_i		
0	0	–	–	–	–	Тесты из условия.
1	10	$n \leqslant 100$	$q \leqslant 100$	–	–	
2	7	$n \leqslant 1000$	–	–	1	
3	14	–	–	$a_n = n, v_i = n$	–	
4	10	–	$q = 1$	$v_i = n$	–	
5	9	–	–	$v_i = n$	3, 4	
6	7	–	–	$t_i = 2$	–	Ответ не превосходит 10
7	7	–	–	$t_i = 2$	1, 6	Ответ не превосходит 150
8	13	–	–	$t_i = 1$	–	
9	10	$n \leqslant 50\,000$	$q \leqslant 50\,000$	–	1	
10	4	$n \leqslant 100\,000$	$q \leqslant 100\,000$	–	1, 9	
11	4	$n \leqslant 200\,000$	$q \leqslant 200\,000$	–	1, 9, 10	
12	5	–	–	–	0 – 11	**Offline-проверка.**

Задача Emmanuel Goldstein. Почти наверное

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Давайте скажем, что два мультимножества равны **почти наверное**, если они равны с точностью до одного элемента. То есть, чтобы можно было изменить не более одного элемента в первом множестве так, чтобы они стали равны. Например, мультимножества $\{1, 1, 2\}$ и $\{1, 2, 3\}$ равны *почти наверное*, $\{1, 1, 1\}$ и $\{1, 1, 1\}$ равны *почти наверное*, а $\{1, 2, 3\}$ и $\{3, 4, 5\}$ не равны *почти наверное*.

Мальчику Васе очень понравилось это определение и он сразу же придумал про него задачу.

У Васи есть два массива a и b, причем $a_i \geqslant b_i$ для всех i от 1 до n. Вася может применять сколько угодно (возможно ноль) раз к массиву a следующую операцию: выбрать любой индекс i ($1 \leqslant i \leqslant n$) и вычесть 1 из a_i. При этом массив b Вася не меняет.

Вася быстро понял, какая последовательность операций нужна, чтобы мультимножество значений массивов a и b стали равны *почти наверное*. Поэтому Вася усложнил задачу — теперь для каждого префикса этих массивов он хочет узнать, какое минимальное количество операций нужно применить, чтобы префиксы массивов стали равны *почти наверное*.

Более формально, для каждого k от 1 до n, Вася хочет взять элементы $a_1, a_2, \ldots, a_k$, а также элементы $b_1, b_2, \ldots, b_k$. Вася хочет знать, какое минимальное количество операций необходимо сделать, чтобы мультимножества этих элементов стали равны *почти наверное*. Обратите внимание, что задача для каждого k решается **независимо**.

Формат входных данных

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

Первая строка каждого набора входных данных содержит единственное целое число n ($1 \leqslant n \leqslant 200\,000$) — размер массивов a и b.

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

Третья строка каждого набора входных данных содержит n целых чисел b_1, b_2, $\ldots$, b_n ($1 \leqslant b_i \leqslant a_i$) — элементы массива b.

Обозначим за N сумму n по всем наборам входных данных в одном тесте. Гарантируется, что $N \leqslant 200\,000$.

Формат выходных данных

Для каждого набора входных данных выведите n чисел, каждое из которых является ответом на задачу для каждой возможной длины префикса. Можно показать, что ответ всегда существует.

Примеры

ввод	вывод
4 2 3 4 1 2 2 3 4 1 3 3 11 17 14 1 13 10 4 100 11 50 42 30 1 20 5	0 1 0 0 0 4 2 0 10 30 48
3 4 2 4 5 12 1 3 4 10 4 3 5 8 20 1 2 6 7 4 4 4 4 4 1 2 3 4	0 1 1 3 0 1 3 6 0 2 3 3

Пояснение

Рассмотрим первый набор входных данных в первом примере.

- Для префикса длины 1 нужно ничего не делать.
- Для префикса длины 2 нужно один раз уменьшить $a_1 = 3$ на единицу, после этого a будет равен $[2, 4]$, b будет равен $[1, 2]$ и они равны *почти наверное*.

Рассмотрим третий набор входных данных в первом примере.

- Для префикса длины 1 нужно ничего не делать.
- Для префикса длины 2 нужно четыре раза уменьшить $a_2 = 17$ на единицу, после этого префикс a будет равен $[11, 13]$, префикс b будет равен $[1, 13]$ и они равны *почти наверное*.
- Для префикса длины 3 нужно один раз уменьшить $a_1 = 11$ на единицу, и один раз уменьшить $a_3 = 14$ на единицу, после этого a будет равен $[10, 17, 13]$, b будет равен $[1, 13, 11]$ и они равны *почти наверное*.

Система оценки

Тесты к этой задаче состоят из шести групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, прохождение тестов из условия не требуется для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Доп. ограничения	Необх. группы	Комментарий
		N		
0	0	–	–	Тесты из условия.
1	16	$N \leqslant 100$	0	–
2	13	$N \leqslant 500$	0, 1	–
3	24	$N \leqslant 3000$	0 – 2	–
4	13	–	–	$a_i < b_{i+1}$
5	14	–	4	$a_i \leqslant a_{i+1}, b_i \leqslant b_{i+1}$
6	20	–	0 – 5	**Offline-проверка.**

Можно показать, что все тесты четвёртой группы удовлетворяют ограничениям пятой группы.