Общая информация по задачам второго тура

Задача	**Тип задачи**	**Ограничения**
5. Улитка на склоне	стандартная	1 с, 512 МБ
6. Конференция	стандартная	1 с, 512 МБ
7. Яблоки по корзинам	стандартная	2 с, 512 МБ
8. Выполнить план, но не перевыполнить	интерактивная	3 с, 512 МБ

Необходимо считывать данные из стандартного потока ввода. Выходные данные необходимо выводить в стандартный поток вывода.

Баллы за подзадачу начисляются только если все тесты этой подзадачи пройдены. Решение запускается на тестах для определенной подзадачи, если все тесты всех необходимых подзадач пройдены.

Для некоторых подзадач может также требоваться, чтобы были пройдены все тесты из условия. Для таких подзадач указана дополнительно буква У.

Во всех задачах во всех подзадачах во время тура вам показываются баллы за подзадачу, если все тесты пройдены, либо первая ошибка и номер теста.

Если вы решите перезагрузиться в другую операционную систему во время тура, для входа используйте логин «`roi`» и пароль «`rus172`».

Задача 5. Улитка на склоне

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Тихо взобравшись на вершину горы Фудзи, улитка хочет спуститься вниз. На склоне горы находится система тропинок, образующая подвешенное двоичное дерево.

Дерево содержит n вершин, соединенных $n-1$ тропинками. На вершине горы находится корень дерева. В некоторых вершинах тропинка заканчивается, они являются листами дерева. От каждой вершины, кроме листьев, вниз по склону отходят ровно две тропинки, одна ведет налево, а другая — направо.

Улитка хочет, начав в корне, спуститься по дереву и добраться до одного из листьев. Она будет спускаться, перемещаясь вниз по тропинкам. В каждой вершине по пути улитка может выбрать одно из двух направлений дальнейшего спуска: налево или направо.

Улитка может начать спуск в корне в любом из двух направлений. В каждой из последующих вершин улитка делает *поворот*, если она выбрала направление, отличающееся от выбранного в предыдущей вершине.

Улитке неудобно поворачивать, поэтому на всем пути от корня до листа улитка готова сделать не более k поворотов.

Пронумеруем вершины дерева от 1 до n, при этом корень получит номер 1. Вам дано q запросов. Каждый запрос описывается одной вершиной u_i. Требуется найти количество листьев, в которых улитка сможет завершить свой спуск, если она будет спускаться из корня, сделает не более k поворотов, и по пути пройдет через вершину u_i.

Формат входных данных

В первой строке даны три целых числа n, k и q — количество вершин в дереве, максимальное количество поворотов, которое улитка готова сделать, и количество запросов ($3 \leqslant n \leqslant 200\,000$; $0 \leqslant k \leqslant n$; $1 \leqslant q \leqslant 200\,000$).

В следующих n строках дано описание дерева. Первым в i-й строке дано целое число t_i — количество тропинок, выходящих из i-й вершины ($t_i = 0$ или $t_i = 2$). Если $t_i = 2$, то далее в той же строке даны два целых числа l_i и r_i — номера вершин, в которые ведёт соответственно левая и правая тропинка из вершины i ($1 \leqslant l_i, r_i \leqslant n$). Гарантируется, что это описание соответствует подвешенному двоичному дереву с корнем в вершине 1.

В следующих q строках даны запросы. В i-й строке дано одно целое число u_i — номер вершины, через которую должна пройти улитка на своем пути ($1 \leqslant u_i \leqslant n$).

Формат выходных данных

Для каждого запроса, выведите ответ на него в новой строке — количество листьев, в которых улитка может завершить свой маршрут, если она начинает в корне, спускается вниз, на своем пути совершает не больше k поворотов и проходит через вершину u_i.

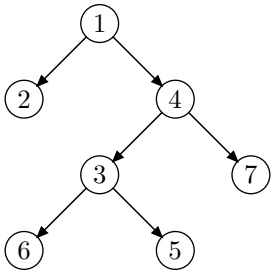
Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необх. подзадачи
1	11	$n \leqslant 500$, $q \leqslant 500$ Во всех запросах u_i является листом	
2	12	$n \leqslant 500$, $q \leqslant 500$	У, 1
3	10	$k = n$	
4	14	$k = 0$	
5	19	Во всех запросах u_i является листом	1
6	34	Без дополнительных ограничений	У, 1–5

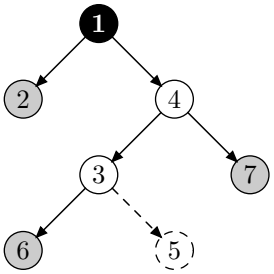
Пример

стандартный ввод	стандартный вывод
7 1 4 2 2 4 0 2 6 5 2 3 7 0 0 0 1 4 3 5	3 2 1 0

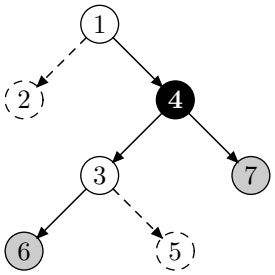
Пояснение к примеру



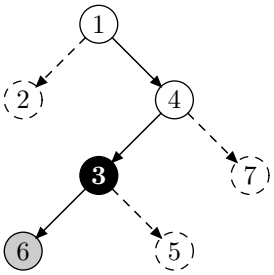
(a) Структура дерева в тесте из примера.



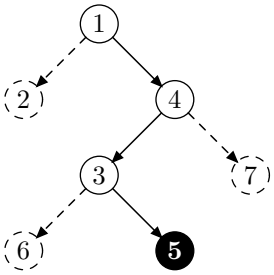
(b) Улитка должна пройти через вершину 1, она может завершить путь в листах 2, 6 и 7.



(c) Улитка должна пройти через 4, она может завершить путь в 6 и 7.



(d) Улитка должна пройти через 3, она может завершить путь только в листе 6.



(e) Улитка должна пройти через 5, однако путь до этой вершины уже содержит больше одного поворота. Поэтому не существует листа, в котором улитка могла бы завершить путь, выполнив все ограничения.

Задача 6. Конференция

Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Тюменская Ассоциация Научных и Образовательных Сообществ организует конференцию, в рамках которой планировалось провести n мероприятий, пронумерованных от 1 до n. При этом i-е мероприятие задаётся двумя целыми числами l_i и r_i — временем начала и окончания мероприятия.

Поскольку некоторые мероприятия могут перекрываться или даже полностью совпадать по времени, один человек не всегда может посетить все мероприятия конференции. Будем считать, что мероприятия i и j не пересекаются, если $r_i < l_j$ или $r_j < l_i$.

Будем называть множество мероприятий *совместным*, если любые два различных мероприятия в этом множестве не пересекаются. Пусть максимальный размер совместного множества мероприятий на конференции равен m. Будем называть *насыщенностью конференции* отношение n/m.

В связи с сокращением финансирования, организаторы конференции приняли решение, что число мероприятий на конференции будет уменьшено ровно в два раза. При этом они хотят сохранить насыщенность конференции неизменной, поэтому максимальный размер совместного множества мероприятий в рамках конференции также должен уменьшиться ровно в два раза. Удачным образом оказалось, что в исходном плане конференции как количество мероприятий n, так и максимальное возможное количество мероприятий в совместном множестве m — чётные числа.

Помогите организаторам выбрать множество из $n/2$ исходно запланированных мероприятий, которые необходимо провести, чтобы при этом размер максимального совместного множества выбранных мероприятий оказался равен $m/2$.

Формат входных данных

Один тест содержит несколько наборов входных данных.

В первой строке дано одно целое число t — количество наборов входных данных ($1 \leqslant t \leqslant 50\,000$).

В первой строке каждого описания набора дано одно целое число n — количество мероприятий в исходном плане ($2 \leqslant n \leqslant 100\,000$, n — чётное).

В следующих n строках каждого описания набора дано описание мероприятий. В i-й строке даны два целых числа l_i и r_i — время начала и конца i-го мероприятия ($1 \leqslant l_i < r_i \leqslant 10^9$).

Гарантируется, что m — размер максимального совместного множества мероприятий для исходного плана, чётно.

Формат выходных данных

Для каждого набора входных данных выведите в новой строке $n/2$ различных номеров мероприятий, которые необходимо провести. Если существует несколько подходящих ответов, вы можете вывести любой из них. Для проведенных мероприятий размер максимального совместного множества мероприятий должен быть равен $m/2$.

Система оценки

Обозначим сумму n по всем наборам входных данных в одном тесте как N.
Будем говорить, что мероприятие i накрывает мероприятие j, если $l_i \leqslant l_j < r_j \leqslant r_i$.

Подз.	Баллы	Ограничения		Необх. подзадачи
		N	Дополнительные ограничения	
1	5	$N \leqslant 100\,000$	Любые два мероприятия не пересекаются	
2	20	$N \leqslant 20$		У
3	7	$N \leqslant 30$		У, 2
4	15	$N \leqslant 500$	В каждой паре мероприятий либо одно мероприятие накрывает другое, либо они не пересекаются, существует мероприятие, которое накрывает все остальные	
5	15	$N \leqslant 100\,000$	В каждой паре мероприятий либо одно мероприятие накрывает другое, либо они не пересекаются	1, 4
6	13	$N \leqslant 500$		У, 2–4
7	13	$N \leqslant 5\,000$		У, 2–4, 6
8	12	$N \leqslant 100\,000$		У, 1-7

Пример

стандартный ввод	стандартный вывод
2 8 12 14 1 3 2 4 1 10 5 6 7 9 8 10 11 13 6 1 2 2 4 1 2 1 4 5 7 6 8	2 5 3 4 1 2 3

Пояснение к примеру

Рисунки визуализируют мероприятия. Мероприятие с началом в момент l_i и концом в момент r_i изображено в виде отрезка $[l_i, r_i]$.

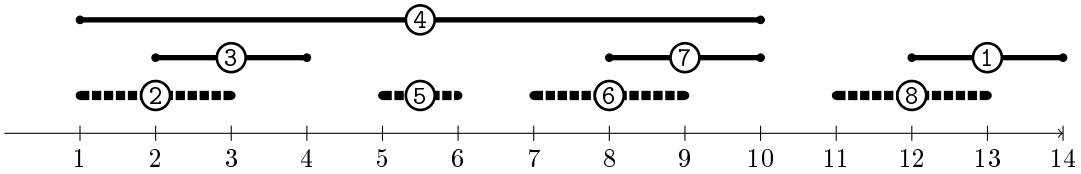


Рис. 1: Исходное множество мероприятий в первом наборе входных данных в примере. Одно из возможных максимальных совместных множеств выделено жирным пунктиром.

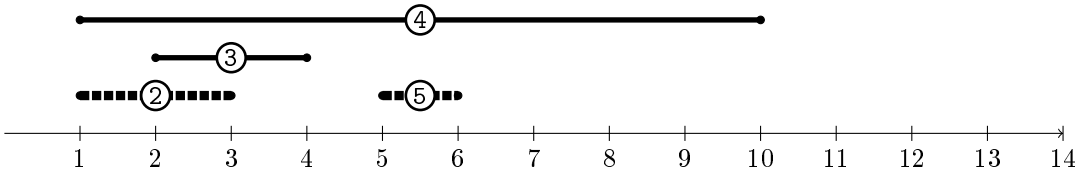


Рис. 2: Множество мероприятий, соответствующее ответу на первый набор входных данных в примере. Одно из возможных максимальных совместных множеств выделено жирным пунктиром.

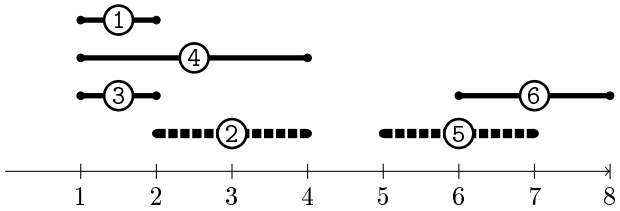


Рис. 3: Исходное множество мероприятий во втором наборе входных данных в примере. Одно из возможных максимальных совместных множеств выделено жирным пунктиром.

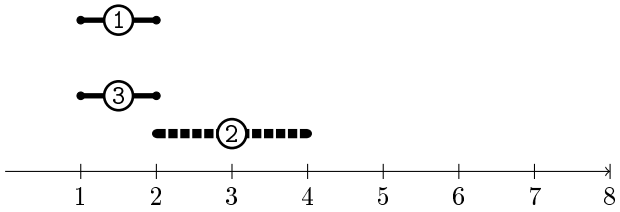


Рис. 4: Множество мероприятий, соответствующее ответу на второй набор входных данных в примере. Одно из возможных максимальных совместных множеств выделено жирным пунктиром.

Задача 7. Яблоки по корзинам

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

У Саши есть n яблок с целыми весами $w_1, w_2, \ldots, w_n$, которые лежат на столе, а также две вместительные корзины.

Саша выбирает целое число k и рассматривает яблоки с весом не больше k. После этого она может положить каждое яблоко с весом $w_i \leqslant k$ в одну из двух корзин, либо оставить его на столе. Яблоки с весом $w_i > k$ в любом случае остаются на столе.

Назовем пару чисел (x, y) *k-достижимой*, если Саша может положить некоторые яблоки с весом не больше k в корзины так, чтобы сумма весов яблок в первой корзине оказалась равна x, а сумма весов яблок во второй корзине оказалась равна y. Назовем пару чисел (a, b) *k-идеальной*, если для всех x и y, где $0 \leqslant x \leqslant a$ и $0 \leqslant y \leqslant b$, пара (x, y) является k-достижимой.

Саша рассматривает q троек чисел k, a, b и для каждой из них хочет понять, является ли k-идеальной пара (a, b).

Формат входных данных

В первой строке даны два целых числа n и q — количество яблок, которые есть у Саши, и количество запросов, которые вам надо обработать ($1 \leqslant n, q \leqslant 300\,000$).

Во второй строке даны n целых чисел w_1, w_2, …, w_n — веса яблок, которые есть у Саши ($1 \leqslant w_i \leqslant 10^{12}$).

В третьей строке находится целое число z, которое используется для формирования запросов, на которые необходимо ответить ($0 \leqslant z \leqslant 10^6$).

В следующих q строках даны описания запросов. Запросы пронумерованы от 1 до q. Каждая строка содержит три целых числа j, c и d ($0 \leqslant j, c, d \leqslant 10^{18}$). Запрос формируется из чисел в этой строке по следующим правилам. Вычислим v, как сумму номеров запросов, сделанных до текущего, для которых заданная в запросе пара (a, b) оказалась k-идеальной. Тогда в текущем запросе $k = j - v \cdot z$; $a = c - v \cdot z$; $b = d - v \cdot z$. Гарантируется, что $k, a, b \geqslant 0$.

Обратите внимание, что при $z = 0$ (что верно для большинства подзадач), значения k, a и b равны j, c и d соответственно. То есть параметры запроса не зависят от ответов на предыдущие запросы и даны во входных данных в явном виде.

Формат выходных данных

На каждый запрос выведите «`Yes`», если пара (a, b) в данном запросе является k-идеальной, иначе выведите «`No`».

Система оценки

Подз.	Баллы	Дополнительные ограничения				Необх. подзадачи
		n, q	a, b	k	z	
1	9	$n, q \leqslant 10$			$z = 0$	
2	6	$n \leqslant 100$	$a \leqslant 100\,000$, $b = 0$	$k = 10^{18}$	$z = 0$	
3	3		$b = 0$	$k = 10^{18}$	$z = 0$	2
4	6	$n, q \leqslant 100$	$a, b \leqslant 300$	$k = 10^{18}$	$z = 0$	
5	6	$n \leqslant 100$	$a, b \leqslant 300$	$k = 10^{18}$	$z = 0$	4
6	2	$n \leqslant 1\,500$	$a, b \leqslant 1\,500$	$k = 10^{18}$	$z = 0$	4–5
7	6	$n \leqslant 5\,000$	$a, b \leqslant 5\,000$	$k = 10^{18}$	$z = 0$	4–6
8	2		$a, b \leqslant 200\,000$	$k = 10^{18}$	$z = 0$	2, 4–7
9	9			$k = 10^{18}$	$z = 0$	2–8
10	3		$b = 0$		$z = 0$	2–3
11	6	$n, q \leqslant 100$	$a, b \leqslant 300$		$z = 0$	4
12	6	$n \leqslant 100$	$a, b \leqslant 300$		$z = 0$	4–5, 11
13	2	$n, q \leqslant 1\,500$	$a, b \leqslant 1\,500$		$z = 0$	4, 11
14	2	$n \leqslant 1\,500$	$a, b \leqslant 1\,500$		$z = 0$	4–6, 11-13
15	6	$n \leqslant 5\,000$	$a, b \leqslant 5\,000$		$z = 0$	4–7, 11-14
16	2		$a, b \leqslant 200\,000$		$z = 0$	4–8, 11–15
17	6				$z = 0$	1–16
18	18					У, 1–17

Примеры

стандартный ввод	стандартный вывод
8 5 17 1 3 2 100 5 6 1 0 6 15 3 9 4 4 5 15 3 17 34 1 16 33 2	Yes No No Yes No
8 5 17 1 3 2 100 5 6 1 1 6 15 3 10 5 5 6 16 4 18 35 2 21 38 7	Yes No No Yes No

Задача 8. Выполнить план, но не перевыполнить

Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Это интерактивная задача.

Требуется разработать планы производства автомобилей и запасных частей в некоторой стране.

Всего в стране n городов, в которых расположены заводы, задействованные в производстве. В i-м городе на заводе могут работать от l_i до r_i человек включительно.

Некоторые города соединены двусторонними дорогами, при этом дорожная сеть имеет форму дерева: от каждого города можно единственным способом добраться до любого другого города, не проезжая через один город дважды.

Планом производства будем называть последовательность целых чисел $[a_1, a_2, \ldots, a_n]$, где a_i — количество человек, которые будут работать на i-м заводе ($l_i \leqslant a_i \leqslant r_i$). После формирования плана производства некоторые заводы будут выбраны как *сборочные*, они будут производить автомобили, а остальные будут производить запчасти. Выбор считается *рациональным*, если никакие два сборочных завода не соединены дорогой напрямую. Среди всех возможных рациональных выборов будет выбран тот, для которого суммарное количество работников сборочных заводов будет максимально. Это число называется *эффективностью* плана $[a_1, a_2, \ldots, a_n]$.

В этой задаче для некоторых значений $v_1, v_2, \ldots, v_q$ вам необходимо выяснить, существует ли план с эффективностью v_i. Если такой план существует, вам может потребоваться предъявить такой план.

Зафиксированы целочисленные параметры x, y и m. Рассмотрим план $[a_1, a_2, \ldots, a_n]$. *Сертификатом* этого плана назовем число $k = \bigoplus_{j=1}^{n} \left((x \cdot j + y \cdot a_j^2) \bmod m \right)$, где $\bigoplus$ — это операция «побитового исключающего или».

Напомним, что эта операция обозначается «`xor`» в Паскале и Python, «`^`» в C++ и Java; для двух целых чисел она определена следующим образом: i-й бит результата равен 1 тогда и только тогда, когда в одном из чисел этот бит 1, а в другом 0. Например, $6 \oplus 10 = 110_2 \oplus 1010_2 = 1100_2 = 12$.

Процесс составления планов будет состоять из двух этапов.

На первом этапе вам будут даны значения $v_1, v_2, \ldots, v_q$. Для каждого из них вам необходимо выяснить, существует ли план с эффективностью v_i и, если его не существует, вывести для этого запроса -1, а если существует, то неотрицательное целое число k_i.

На втором этапе некоторые планы будут проверены: c раз вам будет дано целое число i ($1 \leqslant i \leqslant q$). В ответ на такой запрос требуется либо вывести -1, если плана с эффективностью v_i не существует, либо предоставить план $[a_1, a_2, \ldots, a_n]$, сертификат которого равен k_i, а эффективность равна v_i.

Предоставление сертификатов составленных планов и последующая проверка будут реализованы в интерактивном режиме. До того, как вы предоставите сертификаты, вы не будете знать, какие планы будут проверены. Поэтому в тех подзадачах, в которых $c > 0$, необходимо еще на первом этапе подготовиться к проверке, выведя для значений эффективности v_i, где искомый план существует, такие значения k_i, для которых вы сможете на втором этапе предъявить план с сертификатом, равным k_i.

Протокол взаимодействия

Это интерактивная задача. Ваша программа будет взаимодействовать с программой жюри с использованием стандартных потоков ввода и вывода. Каждое взаимодействие будет состоять из решения задачи для нескольких наборов входных данных.

Сначала ваша программа должна считать целое число t ($1 \leqslant t \leqslant 10^4$) — количество наборов входных данных в рамках одного взаимодействия с программой жюри. Затем t раз необходимо выполнить взаимодействие по решению задачи для набора входных данных.

Рассмотрим протокол взаимодействия для одного набора входных данных.

Сначала ваша программа должна считать данные в следующем формате.

В первой строке находится два целых числа n, q ($2 \leqslant n \leqslant 2 \cdot 10^5$, $1 \leqslant q \leqslant 2 \cdot 10^5$) — количество городов и количество планов, которые надо составить.

Во второй строке находится три целых числа x, y, m ($11 \leqslant m \leqslant 2^{30}$; $0 \leqslant x, y < m$) — параметры вычисления сертификатов планов.

В следующих $n - 1$ строках находится описание дерева дорожной сети между городами. В i-й из этих строк находятся два целых числа s_i, f_i ($1 \leqslant s_i, f_i \leqslant n$; $s_i \neq f_i$) — описание двусторонней дороги между городами s_i и f_i. Гарантируется, что данные дороги образуют дерево.

i-я из следующих n строк содержит два целых числа l_i, r_i ($0 \leqslant l_i \leqslant r_i \leqslant 10^9$) — ограничения на количество работников в i-м городе.

В следующей строке находятся q целых чисел $v_1, v_2, \ldots, v_q$ ($0 \leqslant v_i \leqslant \sum\limits_{i=1}^{n} r_i$) — значения эффективности планов, которые нужно составить. Гарантируется, что все числа v_i различны.

После того как вы считали описание набора входных данных, вы должны вывести q целых чисел $k_1, k_2, \ldots, k_q$ ($-1 \leqslant k_i < 2^{30}$), где $k_i = -1$, если план с эффективностью v_i невозможно составить, иначе $0 \leqslant k_i < 2^{30}$.

После этого может быть выполнена проверка некоторых из составленных планов.

Проверка выполняется в интерактивном режиме следующим образом. Программа жюри выводит одну строку, содержащую целое число i ($1 \leqslant i \leqslant q$) — номер плана для проверки. Гарантируется, что все номера запрошенных планов будут различными.

В ответ вы должны вывести -1, если i-й план составить невозможно. В этом случае ранее должно быть выведено $k_i = -1$. Иначе необходимо вывести n целых чисел $a_1, a_2, \ldots, a_n$ ($l_i \leqslant a_i \leqslant r_i$) — составленный план. Сертификат этого плана должен быть равен k_i, эффективность должна быть равна v_i.

После $c \geqslant 0$ проверок программа жюри передаст на вход вашей программе число $i = 0$, что означает, что работа с этим набором входных данных закончена, и ваше решение должно начать решать следующий набор входных данных или завершить работу, если этот набор был последним.

Обозначим сумму значений n в одном тесте как N. Обозначим сумму значений q в одном тесте как Q. Гарантируется, что $N \leqslant 2 \cdot 10^5$; $Q \leqslant 2 \cdot 10^5$, сумма c по всем наборам входных данных не превосходит 10^4, сумма $n \cdot c$ по всем наборам входных данных не превосходит 10^6.

Поскольку задача интерактивная, после каждого вывода строки вашей программой выводите символ перевода строки и делайте сброс буфера потока вывода.

Если вы используете «`cout << ... << endl`» в C++, «`System.out.println`» в Java, «`print`» в Python, «`writeln`» в Паскале, то сброс потока вывода у вас происходит автоматически, дополнительно ничего делать не требуется. Если вы используете другой способ вывода, рекомендуется делать сброс буфера потока вывода. Обратите внимание, что перевод строки надо выводить в любом случае. Для сброса буфера потока вывода можно использовать «`fflush(stdout)`» в C++, «`flush(output)`» в Паскале, «`System.out.flush()`» в Java, «`sys.stdout.flush()`» в Python.

Система оценки

Подз.	Баллы	Ограничения			Необх. подзадачи
		n, N	Q	дополнительно	
1	11			$l_i = r_i$	
2	9			$c = 0$	
3	12			$l_i = 0$, $r_i \leqslant 1$	
4	4			$l_i = 0$	3
5	8			$s_i = 1$, $f_i = i + 1$	
6	5	$n \leqslant 10$, $N \leqslant 1000$	$Q \leqslant 1000$	$c = q$, $r_i \leqslant 10$, $r_i - l_i \leqslant 2$	
7	2	$n \leqslant 10$, $N \leqslant 1000$	$Q \leqslant 1000$	$c = q$, $r_i \leqslant 10$	6
8	13	$N \leqslant 1000$	$Q \leqslant 1000$	$c = q$, $\sum_{i=1}^{n} r_i - l_i \leqslant 10^4$	6–7
9	11	$N \leqslant 1000$	$Q \leqslant 1000$	$c = q$	6–8
10	6			$c = q$	6–9
11	5	$N \leqslant 1000$			У, 6–9
12	5	$N \leqslant 8000$			У, 6–9, 11
13	9				У, 1–12

Замечание

В примерах сообщения программы участника и программы жюри разделены пустыми строками. Это сделано для облегчения восприятия, чтобы было понятно, какое сообщение является ответом на какое. В реальном взаимодействии в тестирующей системе пустых строк не будет.

Примеры

стандартный ввод	стандартный вывод
1	
9 3	
4 7 15	
1 2	
2 4	
2 5	
1 3	
3 6	
3 7	
6 8	
6 9	
4 4	
2 2	
5 5	
3 3	
2 2	
6 6	
3 3	
4 4	
3 3	
18 19 20	
	-1 10 -1
1	
	-1
2	
	4 2 5 3 2 6 3 4 3
3	
	-1
0	

стандартный ввод	стандартный вывод
3 3 4 3 4 11 1 2 2 3 0 1 0 1 0 1 0 1 2 3 2 0 4 6 1 2 11 1 2 2 3 3 4 0 2 1 1 1 1 1 2 0 1 2 3 4 5 5 2 3 0 5 7 11 31 101 1 2 2 3 2 4 3 5 1 2 1 5 0 4 1 4 4 6 13 12 11 10 9 8 6 3 5 7 0	12 8 3 -1 1 0 0 -1 -1 4 14 9 -1 2 1 1 2 -1 1 1 1 1 -1 127 23 58 13 90 91 2 5 4 1 6 2 4 4 3 4 1 1 0 1 4

Пояснение к примеру

Первый тест подходит под ограничения подзадачи 1. В единственном наборе входных данных есть единственный способ составить план, это $a = [4, 2, 5, 3, 2, 6, 3, 4, 3]$. Его эффективность равна 19, а сертификат этого плана равен 10.

Во втором тесте три набора входных данных.

В первом наборе входных данных:

- Существует единственный план с эффективностью 0, для которого $a = [0, 0, 0]$. Сертификат этого плана равен 12.
- Существуют планы с эффективностью 1, например, $a = [1, 0, 0]$. Сертификат этого плана равен 8.
- Существуют планы с эффективностью 2, например, $a = [1, 0, 1]$. Сертификат этого плана равен 3.
- Не существует плана с эффективностью 3.

Среди четырех запрошенных планов был проверен план с номером $i = 2$ ($v_2 = 1$).

Во втором наборе входных данных:

- Не существует плана с эффективностью 0.
- Не существует плана с эффективностью 1.
- Существуют планы с эффективностью 2, например $a = [1, 1, 1, 1]$. Сертификат этого плана равен 4.
- Существуют планы с эффективностью 3, например $a = [2, 1, 1, 1]$. Сертификат этого плана равен 14.
- Существует единственный план $a = [2, 1, 1, 2]$ с эффективностью 4. Сертификат этого плана равен 9.
- Не существует плана с эффективностью 5.

Среди шести запрошенных планов были проверены планы с номерами $i = 5$ ($v_5 = 4$), $i = 2$ ($v_2 = 1$), $i = 3$ ($v_3 = 2$).

В третьем наборе входных данных были составлены следующие планы (начиная со второго, так как плана с эффективностью $v_1 = 13$ не существует): $[2, 5, 4, 4, 6]$; $[2, 5, 4, 1, 6]$; $[2, 4, 4, 4, 4]$; $[2, 4, 4, 3, 4]$; $[2, 4, 4, 2, 4]$; $[1, 1, 0, 1, 4]$.