Задача A. Степенные числа

Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Если вас интересует математика, то эта задача для вас.

Будем называть целое число n *k-степенным*, если его можно разложить в сумму различных степеней числа k, то есть если n представимо в виде $n = k^{a_1} + k^{a_2} + \ldots + k^{a_d}$, где все a_i целые и $a_i \neq a_j$ для всех $i \neq j$.

Ответьте на множество запросов: какое минимальное целое число, большее либо равное n_i, является k_i-степенным?

Формат входных данных

Первая строка ввода содержит целое число q — количество запросов, на которые вам предстоит ответить ($1 \leqslant q \leqslant 10^5$).

Каждая из следующих q строк содержит два целых числа n_i и k_i, описывающие i-й запрос ($1 \leqslant n_i \leqslant 10^9$; $2 \leqslant k_i \leqslant 10^9$).

Формат выходных данных

Выведите q строк, в i-й из которых выведите минимальное k_i-хорошее число, большее либо равное n_i.

Система оценки

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач успешно пройдены.

Подзадача	**Баллы**	**Ограничения**	**Необходимые подзадачи**	**Информация о проверке**
0	–	примеры из условия		полная
1	6	$n_i \leqslant 10^5$, $k_i = 2$ для всех i		первая ошибка
2	9	$n_i \leqslant k_i$ для всех i		первая ошибка
3	10	$q = 1$; $n_i \leqslant 10^5$ для всех i		полная
4	11	$n_i \leqslant 10^5$, $k_i = 10$ для всех i		первая ошибка
5	13	$n_i \leqslant 10^5$ для всех i; $k_i = k_j$ для всех i, j	1, 4	первая ошибка
6	16	$q \leqslant 500$; $k_i \geqslant 20$ для всех i		первая ошибка
7	35	без дополнительных ограничений	0 – 7	первая ошибка

Пример

стандартный ввод	стандартный вывод
7 1 2 2 3 6 5 13 10 14 3 3620 12 10000 3	1 3 6 100 27 20736 19683

Задача B. Смешивание напитков

Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

После тяжелого контеста бывает полезно выпить немного кофе. У настоящих программистов есть множество различных сортов кофе, которые постепенно добавляются в одну кружку. При этом мы будем рассматривать кофе с определенной особенностью: разные сорта кофе держатся в кружке «слоями» и не перемешиваются.

Напиток из таких сортов кофе можно описать следующим образом: всего в кружку налито n сортов, i-й сорт характеризуется *уровнем крепости* p_i и *высотой слоя*, который он занимает в кружке, h_i. При этом если $i < j$, то слой кофе i-го сорта находится ниже кофе j-го сорта. Также известно, что высота кружки равна $\sum\limits_{i=1}^{n} h_i$, то есть верхний край самого верхнего слоя кофе находится ровно на уровне верхней границы кружки.

Для разнообразия иногда хочется получить из такого «коктейля» напиток определенного суммарного уровня крепости. Суммарный уровень крепости определяется как среднее взвешенное уровней налитых в кружку сортов, то есть как

$$P = \frac{\sum\limits_{i=1}^{n} p_i \cdot h_i}{\sum\limits_{i=1}^{n} h_i}.$$

Чтобы как-то изменять P, можно

1. выбрать трубочку произвольной высоты h;
2. один или более раз выполнить следующее: погрузить ее в напиток на любую глубину от 0 до h включительно относительно верхнего края кружки (не относительно текущего уровня жидкости) и отпить произвольное (не обязательно целое) количество кофе с того уровня, на который попал нижний конец трубочки.

При выпивании какого-то количества кофе из одного слоя высота этого слоя уменьшается на соответствующую величину, а все верхние слои опускаются на ту же величину вниз.

Ваша задача — ответить на запросы вида: можно ли из текущего напитка сделать напиток крепости t_i, и если можно, то какая минимальная высота трубочки для этого понадобится. Поскольку идеально точную необходимую высоту трубочки вычислить может быть сложно, достаточно определить минимальное количество верхних слоев кофе, достаточное, чтобы, отпив какое-то количество кофе из некоторых из них, можно было добиться суммарной крепости напитка t_i.

Формат входных данных

В первой строке ввода даны два целых числа n и q — количество слоев кофе в кружке и количество запросов ($1 \leqslant n, q \leqslant 2 \cdot 10^5$).

Следующие n строк содержат по два целых числа p_i и h_i — уровень крепости и высоту i-го снизу кружки слоя кофе ($1 \leqslant p_i, h_i \leqslant 10^9$). Гарантируется, что сумма $p_i \cdot h_i$ по всем i не превосходит 10^{18}.

В i-й из следующих q строк дано единственное целое число t_i, определяющее i-й запрос ($1 \leqslant t_i \leqslant 10^9$).

Формат выходных данных

Выведите q строк, в i-й из которых содержится единственное целое число от 0 до n — ответ i-й запрос. Если для какого-то запроса ответ такой, что нельзя добиться требуемого уровня крепости, выведите в качестве ответа на этот запрос число -1.

Система оценки

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач успешно пройдены. Обратите внимание, что прохождение тестов из условия не является необходимым для тестирования на некоторых группах тестов.

Подзадача	Баллы	Ограничения	Необходимые подзадачи	Информация о проверке
0	–	примеры из условия		полная
1	5	$q \leqslant 100$; $n \leqslant 15$; $p_i, h_i \leqslant 100$	0	полная
2	5	$n, q \leqslant 500$	0, 1	первая ошибка
3	7	$n, q \leqslant 2000$	0 – 2	первая ошибка
4	8	$n, q \leqslant 5000$	0 – 3	первая ошибка
5	15	$p_i \leqslant p_j$ для $i < j$		первая ошибка
6	10	$h_i = h_j$ для любых i, j; есть не более двух различных p_i		первая ошибка
7	10	$h_i = h_j$ для любых i, j	6	первая ошибка
8	15	$p_i \leqslant 10^5$	0, 1	первая ошибка
9	25	без дополнительных ограничений	0 – 8	первая ошибка

Пример

стандартный ввод	стандартный вывод
3 4 1 1 3 7 2 4 1 2 3 4	2 2 3 -1

Замечание

Для примера из условия:

1. В первом запросе, чтобы получить напиток крепости 1, достаточно выпить верхние два слоя кофе.
2. Во втором запросе достаточно выпить часть кофе со второго сверху слоя.
3. В третьем запросе понадобится выпить первый и третий слой кофе.
4. В четвертом запросе невозможно добиться уровня крепости 4.

Задача C. Нечестная игра

Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

На клетчатой доске размером $n \times m$, состоящей из n строк и m столбцов, в клетке (r, c), то есть на пересечении r-й сверху строки и c-го слева столбца, расположена фишка. На этой доске вам предстоит сыграть против компьютера в игру, в которой можно перемещать фишку и удалять клетки поля.

Каждый ход устроен следующим образом.

1. Компьютер называет целое число $k > 0$.
2. Вы ровно k раз некоторым образом выбираете одну из еще не удаленных клеток, соседних по стороне (имеющих общую сторону) с той, в которой фишка находится в текущий момент, и перемещаете фишку в эту клетку. Вы можете перемещать фишку на клетку, в которой она уже была. Если не существует еще не удаленных клеток, соседних по стороне с текущей, перемещение не производится.
3. Компьютер называет координаты (i, j) произвольной еще не удаленной клетки поля, после чего она сразу же удаляется.

Если компьютер удаляет клетку, на которой находится фишка, игра заканчивается вашей победой. Ваша цель — победить как можно раньше. При этом вы не сообщаете компьютеру свои перемещения, поэтому можете играть нечестно: вместо реального перемещения фишки по полю вы можете следить за всеми возможными ее положениями. Иными словами, если в какой-то момент при удалении клетки (i, j) существует последовательность перемещений фишки, при которой в данный момент фишка находится в точности в клетке (i, j), вы можете сообщить компьютеру, что игра завершена, и вы победили.

Разумеется, компьютер следит за соблюдением правил, поэтому если вы сообщаете ему о своей победе при удалении клетки, на которой фишка не могла в этот момент находиться, вы автоматически проигрываете.

Определите как можно меньший номер хода, после которого вы можете сообщить комьютеру о своей победе, то есть после которого фишка могла оказаться на удаляемой клетке. Чем ближе ваш ответ к минимальному возможному, тем больше баллов вы получите.

Формат входных данных

В единственной строке ввода даны четыре целых числа n, m, r и c — размеры доски и координаты изначального расположения фишки ($1 \leqslant r \leqslant n \leqslant 1000$; $1 \leqslant c \leqslant m \leqslant 1000$).

Во второй строке ввода дано максимальное количество баллов, которое можно получить за соответствующую группу тестов. Ваше решение это значение может игнорировать.

В следующих $n \cdot m$ строках даны ходы, которые последовательно собирается сделать компьютер. Описание t-го хода задается тремя целыми числами k_t, i_t и j_t — количеством перемещений фишки, которые вам понадобится совершить, и координатами клетки поля, которую после этого требуется удалить ($1 \leqslant k_t \leqslant 10^9$; $1 \leqslant i_t \leqslant n$; $1 \leqslant j_t \leqslant m$).

Гарантируется, что все удаляемые клетки различны, то есть никакая клетка не удаляется дважды.

Формат выходных данных

Выведите одно целое число от 1 до $n \cdot m$ — номер хода, после которого вы сообщите компьютеру о своей победе.

Система оценки

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач пройдены с положительным вердиктом.

То есть, если вы проигрываете игру на каком-либо тесте группы, вы получаете вердикт `WA` за соответствующий тест и 0 баллов за всю группу. Иначе, если $R(\texttt{group})$ — максимальное число баллов

за группу, а $j(\texttt{test})$ и $p(\texttt{test})$ — ответы на тест программы жюри и вашей программы, соответственно, вы получаете за группу

$$\left\lfloor R(\texttt{group}) \cdot \min_{\texttt{test} \in \texttt{group}} \frac{j(\texttt{test})}{p(\texttt{test})} \right\rfloor \text{ баллов.}$$

Обратите внимание, что прохождение тестов из условия не является необходимым для тестирования на некоторых группах тестов.

Подзадача	**Баллы**	**Ограничения**	**Необходимые подзадачи**	**Информация о проверке**
0	–	примеры из условия		полная
1	7	$k_1 = n + m$		полная
2	6	$k_t = 1$ для всех t		первая ошибка
3	10	ходы компьютера случайны и равновероятны	0	первая ошибка
4	14	(i_t, j_t) и (i_{t+1}, j_{t+1}) имеют общую сторону для всех t	0	первая ошибка
5	11	$k_t \leqslant 2$ для всех t	2	первая ошибка
6	17	$n = 1$		полная
7	9	$n, m \leqslant 40$	0	первая ошибка
8	9	$r = c = 1$		полная
9	17	без дополнительных ограничений	0 – 8	первая ошибка

Примеры

стандартный ввод	стандартный вывод
2 2 1 1 0 1 1 1 2 2 1 3 2 2 4 1 2	2
3 3 2 2 0 1 2 2 1 1 2 1 1 1 1 2 1 1 3 1 1 3 2 1 3 3 1 2 3 1 1 3	9

Замечание

В первом примере можно, например, первым ходом передвинуть фишку из $(1, 1)$ в $(1, 2)$, а вторым — из $(1, 2)$ в $(2, 2)$ и затем в $(2, 1)$, тем самым поместив ее на удаляемую клетку.

Задача D. Стековое кодирование

Ограничение по времени:	3 секунды
Ограничение по памяти:	1024 мегабайта

Если увлечься темой кодирования информации, можно поймать себя на придумывании совершенно неожиданных алгоритмов. Сегодня вам предстоит разобраться в *кодировании массива стеком*.

Изначально вам дан пустой стек и пустой массив. *Кодом* массива a назовем последовательность действий вида

- `push`(x) — положить число x на вершину стека;
- `pop` — снять число с вершины стека;
- `print` — выписать в конец массива все элементы стека по порядку от нижнего к верхнему,

приводящую к тому, что в изначально пустой массив оказываются выписаны все элементы a по порядку. При выполнении третьей операции стек не очищается.

Например, при выполнении последовательности действий `push`(1), `push`(2), `print`, `print`, `pop` и `print` в массив оказываются выписаны числа $[1, 2, 1, 2, 1]$, а на стеке остается лежать только число 1. То есть такая последовательность является кодом массива $[1, 2, 1, 2, 1]$ длины 6.

Вам дан массив a и q запросов: какой у отрезка массива a с l_i-го по r_i-й элемент включительно минимальный по количеству действий со стеком код?

Найдите для каждого запроса код соответствующего отрезка массива. Не требуется найти код наименьшей возможной длины, но чем меньше найденный код, тем больше баллов получит ваше решение.

Формат входных данных

В первой строке ввода даны четыре целых числа n и q — длина массива a, про отрезки которого спрашивается в запросах, количество запросов, а также максимальный балл за тест и параметр γ, указанный в системе оценивания, которые ваше решение может игнорировать ($1 \leqslant n \leqslant 2000$; $1 \leqslant q \leqslant 10^4$).

Во второй строке перечислены n целых чисел a_i — элементы массива a ($1 \leqslant a_i \leqslant 10^9$).

В i-й из следующих q строк даны два целых числа l_i и r_i — границы отрезка из i-го запроса ($1 \leqslant l_i \leqslant r_i \leqslant n$).

Формат выходных данных

Для каждого запроса выведите в отдельной строке целое число k от 1 до $n + 1$ — количество действий в вашем коде соответствующего отрезка массива, после чего в следующей строке выведите через пробел k целых чисел, описывающих эти действия в порядке их выполнения:

- для действия `push`(x) выведите число x от 1 до 10^9;
- для действия `pop` выведите число -1;
- для действия `print` выведите число 0.

Если в результате выполнения выведенных действий происходит попытка снять число с вершины пустого стека или в конце не получается массив, равный заданному отрезку массива a, ваше решение получает вердикт `Wrong Answer`. Также вы получите вердикт `Wrong Answer`, если в вашем коде будет больше $n + 1$ действия.

Система оценки

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач пройдены с положительным вердиктом. То есть, если вы получаете вердикт `WA` на каком-либо тесте в группе, вы получаете 0 баллов за всю группу. Если все тесты группы пройдены с положительным вердиктом, вы получаете сумму баллов за все тесты в группе.

Для каждой подзадачи даны общие для тестов этой подзадачи ограничения и параметр γ, показывающий, насколько близкое к ответу жюри решение будет засчитываться на ненулевой балл.

Если ваше решение для i-го запроса в конкретном тесте строит код из p_i действий, а решение жюри строит код из j_i действий, то за этот тест ваше решение получает

$$\max\left(0, \texttt{score} - \max\left(0, \max_{i=1}^{q}\left\lfloor\frac{p_i - j_i}{\gamma}\right\rfloor\right)\right) \text{ баллов,}$$

где `score` — максимальный балл за этот тест. Иными словами, за каждые γ действий свыше решения жюри в каком-либо из запросов вы теряете один балл за тест.

Обратите внимание, что прохождение тестов из условия не является необходимым для тестирования на некоторых группах тестов.

Подзадача	**Баллы**	**Ограничения**	γ	**Необходимые подзадачи**	**Информация о проверке**
0	2×0	примеры из условия	100		полная
1	4×1	$n \leqslant 10$	10	0	первая ошибка
2	4×1	$n \leqslant 10$	1	0, 1	первая ошибка
3	4×1	$n \leqslant 200$, $q = 1$	1		первая ошибка
4	5×2	$q = 1$	1	3	первая ошибка
5	5×2	$n \leqslant 200$	1	0 – 3	первая ошибка
6	4×1	$a_i = 1$ для всех i	100		первая ошибка
7	5×2	$a_i = 1$ для всех i	1	6	первая ошибка
8	5×3	$a_i \leqslant 2$ для всех i; $a_i = 2$ для не более чем одного i	50	6, 7	первая ошибка
9	5×2	$a_i \leqslant 2$ для всех i; $a_i = 2$ для не более чем одного i	2	6 – 8	первая ошибка
10	9×2	$q \leqslant 4000$	50	0 – 9	первая ошибка
11	11×1	$q \leqslant 4000$	2	0 – 10	первая ошибка

Примеры

стандартный ввод	стандартный вывод
9 4 0 1 1 2 3 1 2 3 1 2 3 1 9 1 6 4 9 4 6	6 1 2 3 0 0 0 5 1 2 3 0 0 5 1 2 3 0 0 4 1 2 3 0
12 4 0 1 1 2 3 4 1 2 3 1 2 3 4 5 1 12 2 11 3 10 6 7	10 1 2 3 4 0 -1 0 4 5 0 11 2 3 4 1 2 3 1 2 3 4 0 9 3 4 1 2 3 1 2 3 0 3 2 3 0

Задача E. Годовой отчет

Ограничение по времени:	6 секунд
Ограничение по памяти:	1024 мегабайта

Некоторые IT-компании проводят ежегодное глобальное мероприятие для сотрудников, на котором подводятся итоги года и обсуждаются ключевые точки в развитии всех проектов. В одной из компаний, на которой мы сфокусируемся, планируется обсуждение k различных ее проектов.

Для того, чтобы мероприятие прошло интересно, необходимо выбрать хороших спикеров. Есть n кандидатов, i-й из которых характеризуется своей осведомленностью о проектах — целым числом a_i от 0 до $2^k - 1$. При этом некоторые из кандидатов дружат между собой, а именно, есть m пар друзей (f_i, s_i).

Разумеется, всем хочется, чтобы мероприятие прошло гладко, а для этого необходимо, чтобы все спикеры попарно дружили между собой. При этом, чтобы рассказы спикеров не казались однообразными, важно, чтобы количество выбранных спикеров было как можно больше. *Осведомленностью* группы размера s, состоящей из кандидатов $i_1, i_2, \ldots, i_s$, называется $a_{i_1} \oplus a_{i_2} \oplus \ldots a_{i_s}$, где $\oplus$ — операция побитового исключающего «ИЛИ». Соответственно, помимо уже описанных критериев, среди всех подходящих групп кандидатов максимального размера необходимо выбрать группу спикеров с максимальной осведомленностью.

Мероприятие уже скоро, поэтому процесс набора кандидатов сейчас в самом разгаре. Ваша задача — выбрать оптимальную по описанным критериям группу спикеров.

Так как список кандидатов еще не утвержден окончательно и может меняться, вам надо решить эту задачу для t возможных наборов кандидатов.

Формат входных данных

В первой строке ввода находится одно целое число t — количество случаев, для которых надо решить задачу ($1 \leqslant t \leqslant 120$). Далее следует описание t наборов входных данных.

Первая строка каждого набора содержит три целых числа n, m и k — количество кандидатов, количество пар друзей среди кандидатов и количество различных тем, которые будут обсуждаться на конференции, соответственно ($1 \leqslant n \leqslant 40$; $0 \leqslant m \leqslant \frac{n \cdot (n-1)}{2}$; $0 \leqslant k \leqslant 30$).

Во второй строке каждого набора через пробел перечислены n целых чисел a_i — уровни осведомленности о проектах каждого из кандидатов ($0 \leqslant a_i \leqslant 2^k - 1$).

Далее следуют m строк, i-я из которых содержит два целых числа f_i и s_i — номера кандидатов, образующих i-ю пару друзей ($1 \leqslant f_i, s_i \leqslant n$; $f_i \neq s_i$). Гарантируется, что все перечисленные пары друзей различны.

Гарантируется, что сумма n по всем наборам входных данных не превосходит 120.

Формат выходных данных

Для каждого набора входных данных выведите через пробел два числа в отдельной строке — сначала выведите размер максимальной группы спикеров, а затем выведите максимальную возможную осведомленность группы.

Система оценки

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач успешно пройдены. Обратите внимание, что прохождение тестов из условия не является необходимым для тестирования на некоторых группах тестов.

Все указанные ограничения распространяются на все наборы входных данных в каждом тесте соответствующей группы.

Последняя подзадача имеет потестовую оценку и содержит 7 тестов, каждый из которых независимо оценивается в 4 балла.

Подзадача	Баллы	Ограничения	Необходимые подзадачи	Информация о проверке
0	–	примеры из условия		полная
1	7	$n \leqslant 20$	0	полная
2	9	$k = 0$		первая ошибка
3	15	$k \leqslant 3$	2	первая ошибка
4	18	$n \leqslant 30$	0, 1	первая ошибка
5	14	$a_i = 0$ при $2 \leqslant i \leqslant n$	2	первая ошибка
6	9	$k = 2$, n четное, $a_1 = \ldots = a_{\frac{n}{2}} = 1$, $a_{\frac{n}{2}+1} = \ldots = a_n = 2$		первая ошибка
7	7×4	без дополнительных ограничений	0 – 6	первая ошибка

Пример

стандартный ввод	стандартный вывод
3 6 2 4 1 2 3 4 5 6 1 2 3 4 6 8 8 1 2 4 8 16 32 1 2 2 3 3 4 4 5 5 6 6 1 1 3 1 4 5 8 6 1 2 4 8 16 1 2 1 3 1 4 2 3 2 4 2 5 3 5 3 4	2 7 3 13 4 15

Замечание

В примере из условия:

1. в первом наборе входных данных выгодно выбрать спикеров под номерами 3 и 4 с осведомленностью $a_3 \oplus a_4 = 3 \oplus 4 = 7$;
2. во втором наборе входных данных выгодно выбрать спикеров под номерами 1, 3 и 4 с осведомленностью $1 \oplus 4 \oplus 8 = 14$;

3. в третьем наборе входных данных есть единственный способ выбрать четырех попарно дружащих спикеров: выбрать всех, кроме пятого.