Задача Cheesecake. Кукушки

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

Британские учёные решили заняться орнитологией и понаблюдать за жизнью необычных кукушек. Для этого они вырастили дерево и построили на нём n гнёзд, в каждом из которых живёт кукушка. Наблюдение за деревом состоит в том, что в некоторые моменты времени учёные оценивают, можно ли подложить определённое яйцо в гнездо к некоторой кукушке или нет.

Каждое яйцо может вынашиваться только в двух определённых гнёздах. Каждое яйцо задаётся неупорядоченной парой различных чисел (x, y). Яйцо (x, y) может вынашиваться в любом из гнёзд x и y и не может вынашиваться в других гнёздах. Обратите внимание, яйцо (x, y) не отличается от яйца (y, x).

Теперь опишем процесс подкладывания яйца в имеющиеся гнезда: пусть учёные хотят подложить яйцо (x, y) в гнездо x. Если в гнезде x нет яйца, то яйцо (x, y) просто остаётся в этом гнезде, и процесс на данном шаге завершается. Если же в гнезде x лежит какое-то яйцо (x, p), то кукушка кладёт яйцо (x, y) в данное гнездо, а яйцо (x, p) пытается подложить в гнездо p аналогичным образом, и процесс продолжается.

Вам предлагается отвечать на вопросы учёных. Всего есть три типа вопросов:

1. (Теоретический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x? Так как вопрос чисто теоретический, оно **не добавляется** на самом деле, и состояние гнёзд не меняется.

2. (Практический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x? Если процесс закончится, то яйцо **добавляется** в реальности согласно описанному процессу.

3. (Теоретический) Сколько существует **упорядоченных** пар различных чисел (x, y), таких что яйцо (x, y) можно подложить в гнездо x с учётом имеющихся в гнёздах яиц? При этом для каждого яйца ответ определяется независимо от других добавляемых яиц.

Формат входных данных

В первой строке вводятся три целых числа n, m, q, ($2 \leqslant n \leqslant 200\,000$, $0 \leqslant m \leqslant n$, $1 \leqslant q \leqslant 600\,000$), где n — количество гнёзд на дереве, m —количество яиц, которые учёные уже положили, q — количество вопросов, которые задают учёные.

В каждой из m последующих строк следуют по два числа x_i, y_i, означающих, что в гнезде x_i лежит яйцо (x_i, y_i). Гарантируется, что все x_i различны и что $x_i \neq y_i$ для всех i.

В следующих q строках описаны вопросы учёных. Вопросы даны в том порядке, в котором на них требуется отвечать. Первое число t_j в строке описывает тип вопроса.

Если $t_j = 1$ или $t_j = 2$, то далее идут два различных числа x_j и y_j, описывающих яйцо, которое фигурирует в соответствующем вопросе.

Если $t_j = 1$, то яйцо не требуется добавлять в текущую расстановку.

Если $t_j = 2$, то яйцо требуется добавить, если процесс добавления потребует конечного числа перекладываний.

Если $t_j = 3$, то требуется определить количество упорядоченных пар (x, y), таких что яйцо (x, y) можно добавить в гнездо x с тем, чтобы процесс когда-нибудь завершился. В реальности никакие яйца в расстановку не добавляются.

Формат выходных данных

Для каждого вопроса первого и второго типа выведите единственное слово «`Yes`» или «`No`» в зависимости от того, закончится ли процесс перекладывания.

Для каждого запроса третьего типа выведите количество искомых упорядоченных пар.

Пример

ввод	вывод
5 3 8 1 2 5 1 2 4 1 1 2 3 2 1 2 3 2 4 2 2 5 3 3 1 4 5	Yes 20 Yes 8 No Yes 0 No

Пояснение

Изначальное расположение яиц в тесте из условия такое: в первом гнезде лежит яйцо $(1, 2)$, во втором — $(2, 4)$, в пятом — $(5, 1)$, а в третьем и четвёртом яиц нет.

Яйцо $(1, 2)$ добавить можно, несмотря на то что подобное яйцо на дереве уже есть, это приведёт к перекладыванию имеющегося яйца $(1, 2)$ в другое гнездо.

Также в начальную конфигурацию можно добавить любое из 10 яиц, существующих для дерева с пятью гнёздами, и каждое яйцо можно положить в любое из двух гнёзд, ему отвечающих, и для любого из добавляемых яиц и гнёзд это потребует конечное количество шагов. Таким образом, ответ на второй запрос — 20.

В результате следующего запроса яйцо $(1, 2)$ будет добавлено реально, и распределение яиц будет таким: в первом гнезде лежит яйцо $(1, 2)$, во втором — также $(1, 2)$, в четвёртом — $(2, 4)$, в пятом $(5, 1)$.

Теперь уже можно добавить только яйца $(1, 3)$, $(2, 3)$, $(4, 3)$ и $(5, 3)$, причём по-прежнему любое яйцо можно положить в каждое из двух упомянутых на нём гнёзд, поэтому ответ на запрос — 8.

Яйцо $(4, 2)$ добавить на дерево нельзя, поэтому состояние гнёзд не изменится.

Для добавления яйца $(5, 3)$ понадобится 5 перекладываний яиц, а после этого никакое новое яйцо за конечное количество шагов добавить уже нельзя.

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 других групп (смотрите таблицу ниже). Пусть t_1 — количество запросов первого типа, t_2 — количество запросов второго типа, t_3 — количество запросов третьего типа. **Offline** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Дополнительные ограничения				Необх. группы	Комментарий
		n	t_1	t_2	t_3		
0	0	–	–	–	–	–	Примеры
1	13	$n \leqslant 2000$	$t_1 \leqslant 2000$	$t_2 = 0$	$t_3 = 0$	–	
2	14	$n \leqslant 2000$	$t_1 \leqslant 2000$	$t_2 = 0$	$t_3 \leqslant 1$	1	
3	12	$n \leqslant 2000$	$t_1 \leqslant 2000$	$t_2 \leqslant 2000$	$t_3 \leqslant 2000$	0 – 2	
4	12	–	$t_1 \leqslant 2 \cdot 10^5$	$t_2 = 0$	$t_3 = 0$	1	
5	18	–	$t_1 \leqslant 2 \cdot 10^5$	$t_2 = 0$	$t_3 \leqslant 1$	1 – 2, 4	
6	31	–	$t_1 \leqslant 2 \cdot 10^5$	$t_2 \leqslant 2 \cdot 10^5$	$t_3 \leqslant 2 \cdot 10^5$	0 – 5	**Offline-проверка**

Задача Halva. Глеб и два числа

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

В свободное от написания длинных легенд к задачам время Глеб развлекает себя игрой с числами. Он выбирает два целых числа l и r, после чего пробует подобрать такие целые числа a и b, что $l \leqslant a \leqslant b \leqslant r$, и расстояние Хэмминга между числами a и b максимально.

Расстоянием Хэмминга между двумя целыми числами x и y назовём количество десятичных разрядов, в которых они различаются. Если числа имеют разную длину, то более короткое дополняется слева ведущими нулями.

Формат входных данных

Первая строка входных данных содержит целое число l, а вторая — целое число r ($1 \leqslant l \leqslant r \leqslant 10^{1\,000\,000}$).

Формат выходных данных

Выведите максимально возможное расстояние Хэмминга на отрезке чисел от l до r.

Примеры

ввод	вывод
11 17	1
1 11	2

Пояснение

В первом примере можно выбрать числа 12 и 16, во втором, например, 1 и 10.

Система оценки

Тесты к этой задаче состоят из четырёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов предыдущих групп.

Группа	Баллы	Дополнительные ограничения	Комментарий
		l, r	
0	0	–	Тесты из условия
1	19	$l, r \leqslant 1000$	
2	21	$l, r \leqslant 1\,000\,000$	
3	32	$l, r \leqslant 10^{18}$	
4	28	–	

Задача Strudel. Эффективное тестирование

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

Начиная с $20xx$ года все организаторы всех школьных олимпиад по программированию договорились проводить соревнования исключительно по интернету, для чего было создано общество с ограниченной ответственностью «Организация онлайн-олимпиад» (ООО «ООО»). Разумеется, такая серьёзная организация не может обойтись без собственной тестирующей системы, поэтому для её создания были наняты эффективные менеджеры, закуплены доски и подготовлена синяя изолента.

Для повышения эффективности процесса тестирования была разработана следующая архитектура. Сначала все m тестов задачи располагаются в порядке от 1 к m в очереди тестирования. Затем модуль планирования последовательно выполняет n действий. Действие i состоит в том, чтобы выбрать отрезок очереди с позиции l_i по r_i включительно (в нумерации с единицы) и проверить решение на каждом втором тесте на этом отрезке, а именно на тестах на позициях $l_i, l_i+2, l_i+4, \ldots, r_i$ очереди (при этом гарантируется, что l_i и r_i имеют одинаковую чётность). После этого те тесты, на которых было проведено тестирование, удаляются из очереди, а все оставшиеся тесты сдвигаются по очереди таким образом, чтобы пустых мест не осталось. Например, если в очереди находились тесты с исходными номерами $2, 3, 4, 5, 10, 12, 13, 20$ и была применена операция с $l_i = 3$, $r_i = 7$, то посылка будет протестирована на тестах с позиций 3, 5 и 7, которые исходно имели номера 4, 10 и 13. После выполнения данной операции очередь тестирования будет состоять из тестов с исходными номерами $2, 3, 5, 12, 20$.

Вам поручено реализовать модуль, который для каждого из n описанных выше действий будет определять минимальный и максимальный номер теста в изначальной нумерации из тех, на которых на этом шаге проверялось решение.

Формат входных данных

В первой строке входных данных находятся два числа n и m ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 10^{18}$) — количество действий модуля планирования и количество тестов в задаче.

В каждой из последующих n строк записаны два целых числа l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant m$) — параметры i-го действия модуля планирования. Гарантируется, что перед началом выполнения действия i в очереди тестирования находятся хотя бы r_i тестов и что числа l_i и r_i имеют одинаковую чётность.

Формат выходных данных

Для каждого из n действий модуля планирования выведите два целых числа — минимальный и максимальный номер теста в исходной нумерации из тех, на которых проверялось решение на соответствующем шаге.

Примеры

ввод	вывод
2 10 2 8 1 3	2 8 1 5
4 6 1 1 1 1 1 1 2 2	1 1 2 2 3 3 5 5

Пояснение

Рассмотрим, как изменяется очередь тестирования в первом примере.

0. Изначально в очереди тестирования находятся все тесты от 1 до 10, то есть очередь имеет вид $1, 2, 3, 4, 5, 6, 7, 8, 9, 10$.

1. При выполнении первого запроса будут удалены тесты $2, 4, 6, 8$, и очередь примет вид $1, 3, 5, 7, 9, 10$.

2. При выполнении второго запроса будут удалены тесты 1 и 5, очередь примет вид $3, 7, 9, 10$.

Система оценки

Тесты к этой задаче состоят из семи групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех групп, от которых зависит данная группа. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Дополнительные ограничения		Необх. группы	Комментарий
		n	m		
0	0	–	–	–	Тесты из условия
1	10	$n \leqslant 100$	$m \leqslant 100$	0	
2	9	$n \leqslant 10\,000$	$m \leqslant 10\,000$	0, 1	
3	13	–	$m \leqslant 1\,000\,000$	0 – 2	
4	15	$n \leqslant 1000$	–	0, 1	
5	17	$n \leqslant 10\,000$	–	0 – 2, 4	
6	36	–	–	0 – 5	**Offline-проверка**

Задача Tiramisu. Ваня и куртки

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

Ваня в очередной раз собрался изменить что-нибудь в своей жизни и решил начать с того, чтобы заранее составить график ношения курток в ближайшие n дней.

Он прочитал несколько учебных пособий по эксплуатации курток, поэтому он знает, что разные куртки предназначены для разных температурных диапазонов. Для каждой из своих m курток он определил l_i и r_i — минимальное и максимальное значение температуры, при которых допускается использование i-й куртки.

Ваня знает прогноз погоды на ближайшие n дней — так, в j-й день ожидается температура a_j. Поскольку Ваня считает себя здравомыслящим человеком, то он будет одеваться по погоде, то есть в день j он может надеть любую куртку i, для которой $l_i \leqslant a_j \leqslant r_i$. Помимо этого, Ваня считает себя модным и стильным, поэтому ни при каких обстоятельствах не наденет одну и ту же куртку два дня подряд.

Учитывая, что мама Вани не позволит ему выйти на улицу без куртки или сразу в нескольких куртках, составьте график их ношения на ближайшие n дней, удовлетворяющий всем пожеланиям Вани.

Формат входных данных

В первой строке входных данных записаны два числа n и m ($1 \leqslant n, m \leqslant 100\,000$) — количество дней и число курток в гардеробе Вани соответственно.

Во второй строке записаны n чисел a_i ($0 \leqslant a_i \leqslant 10^9$) — температура в i-й день.

Затем следуют m строк, i-я из которых содержит два числа l_i и r_i ($0 \leqslant l_i \leqslant r_i \leqslant 10^9$) — температурные ограничения на использование i-й куртки.

Все числа во входных данных целые.

Формат выходных данных

Если существует способ подобрать куртку на каждый из n дней, то в первой строке выведите слово «`Yes`» (без кавычек), а во второй строке выведите n чисел b_i — номер куртки, которую Ване следует надеть в i-й день. В противном случае в единственной строке выведите слово «`No`» (без кавычек). Куртки нумеруются с единицы в том порядке, в котором они даны во входных данных.

Если подходящих расписаний ношения курток несколько, разрешается вывести любое из них.

Примеры

ввод	вывод
4 4 25 25 30 50 10 40 20 30 70 100 50 50	Yes 2 1 2 4
4 2 30 40 50 60 30 40 50 60	No

Пояснение

В первом примере, помимо ответа «`2 1 2 4`», правильным также является ответ «`1 2 1 4`».

Во втором примере ответ отсутствует, так как Ване пришлось бы надеть первую куртку и в первый, и во второй день.

Система оценки

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

Группа	Баллы	Дополнительные ограничения	Комментарий
		n, m	
0	0	–	Тесты из условия
1	20	$n, m \leqslant 7$	
2	19	$n, m \leqslant 100$	
3	21	$n, m \leqslant 4000$	
4	40	–	**Offline-проверка**