

Задача 1. Кампус

Имя входного файла: `building.in`
Имя выходного файла: `building.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Новое здание кампуса Университета Байтбурга имеет n этажей, пронумерованных снизу вверх от 1 до n . Комнаты студентов расположены в нескольких подъездах.

В каждом подъезде на этажах, номер которых кратен числу k , расположено по x комнат, а на остальных этажах расположено по y комнат.

Комнаты внутри каждого подъезда пронумерованы последовательными натуральными числами. Номера комнат на первом этаже имеют наименьшие значения в этом подъезде, затем следуют номера комнат на втором этаже, и так далее. Комнаты в первом подъезде пронумерованы, начиная с 1, в каждом следующем подъезде нумерация комнат начинается с числа, следующего после максимального номера комнаты в предыдущем подъезде.

На рис. 1 показаны номера комнат в здании с $n = 7$ этажами, 3 подъездами, и параметрами $k = 3$, $x = 2$, $y = 3$.

	Подъезд 1	Подъезд 2	Подъезд 3
7 этаж	17, 18, 19	36, 37, 38	55, 56, 57
6 этаж	15, 16	34, 35	53, 54
5 этаж	12, 13, 14	31, 32, 33	50, 51, 52
4 этаж	9, 10, 11	28, 29, 30	47, 48, 49
3 этаж	7, 8	26, 27	45, 46
2 этаж	4, 5, 6	23, 24, 25	42, 43, 44
1 этаж	1, 2, 3	20, 21, 22	39, 40, 41

Рис. 1. Пример нумерации комнат в здании

Для организации расселения студентов администрация кампуса должна по номеру комнаты оперативно определять этаж, на котором она находится.

Требуется написать программу, которая по заданным числам n , k , x и y , а также по номерам комнат, определяет для каждой комнаты, на каком этаже она находится.

Формат входного файла

Первая строка входного файла содержит натуральные числа n , k , x и y ($1 \leq n \leq 10^9$, $1 \leq k \leq n$, $1 \leq x, y \leq 10^9$). Соседние числа разделены ровно одним пробелом.

Вторая строка входного файла содержит натуральное число q — количество номеров комнат, для которых требуется определить этаж ($1 \leq q \leq 1000$).

Третья строка содержит q целых чисел a_1, a_2, \dots, a_q — номера комнат ($1 \leq a_i \leq 10^{18}$). Можно считать, что в здании так много подъездов, что все комнаты с заданными номерами существуют.

Формат выходного файла

Требуется вывести q чисел, по одному на строке. Для каждого номера комнаты во входном файле требуется вывести номер этажа, на котором она находится.

Пример входных и выходных файлов

<code>building.in</code>	<code>building.out</code>
7 3 2 3	1
4	7
1 19 20 50	1
	5

Описание подзадач и системы оценивания

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

Подзадача	Баллы	Дополнительные ограничения		Необходимые подзадачи
		n, x, y	q, a_i	
1	31	$1 \leq n \leq 10,$ $1 \leq x, y \leq 10$	$q = 1$ $1 \leq a_i \leq 100$	
2	19	$1 \leq n \leq 10^7,$ $1 \leq x, y \leq 10^9$	$q = 1$ $1 \leq a_i \leq 10^7$	1
3	16	$1 \leq n \leq 10^9,$ $1 \leq x, y \leq 10^9,$ $x = y$	$1 \leq q \leq 1000$ $1 \leq a_i \leq 10^{18}$	
4	34	$1 \leq n \leq 10^9$ $1 \leq x, y \leq 10^9$	$1 \leq q \leq 1000$ $1 \leq a_i \leq 10^{18}$	1, 2, 3

Получение информации о результатах окончательной проверки

По запросу сообщается результат окончательной проверки на каждом тесте.

Задача 2. Калькулятор

Имя входного файла: `calc.in`
Имя выходного файла: `calc.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

В качестве домашнего задания по информатике ученикам предложено разработать специальный калькулятор, который устроен следующим образом.

Сначала пользователь вводит целое положительное число n , которое выводится на экран. Затем пользователь может нажимать на три кнопки: А, В и С.

При нажатии на кнопку А число, которое выведено на экран, делится на 2. Если число на экране нечетное, то остаток отбрасывается. Например, результат этой операции для числа 80 равен 40, а для числа 239 равен 119.

При нажатии на кнопку В к числу, которое выведено на экран, прибавляется 1, и результат делится на 2. Остаток от деления отбрасывается. Например, результат операции для числа 80 равен 40, а для числа 239 равен 120.

При нажатии на кнопку С происходит следующее. Если число, которое выведено на экран, положительное, то из него вычитается 1 и результат делится на 2, остаток отбрасывается. Если же перед нажатием на кнопку С на экран было выведено число 0, то оно остается неизменным. Например, результат операции для числа 80 равен 39, а для числа 239 равен 119.

Пользователь ввел число n и собирается нажать на кнопки операций в некотором порядке. В частности, он планирует нажать на кнопку А суммарно a раз, на кнопку В – b раз и на кнопку С – c раз. Его заинтересовал вопрос, какое минимальное число может получиться в результате выполнения описанных операций.

Требуется написать программу, которая по введенному числу n и числам a , b и c , показывающим количество произведенных на калькуляторе операций разного типа, определяет минимальное число, которое может получиться в результате работы калькулятора.

Формат входного файла

Входной файл содержит четыре целых числа: n , a , b и c ($1 \leq n \leq 10^{18}$, $0 \leq a, b, c \leq 60$). Числа заданы на одной строке, соседние числа разделены одним пробелом.

Формат выходного файла

Требуется вывести одно число — минимальное число, которое может получиться у пользователя в результате работы калькулятора.

Примеры входных и выходных файлов

<code>calc.in</code>	<code>calc.out</code>
72 2 1 1	4

Пояснение к примеру

В примере пользователю необходимо оптимально действовать следующим образом: нажать на кнопку В и получить число 36, затем нажать на кнопку А и получить число 18, затем нажать на кнопку С и получить число 8, затем второй раз нажать на кнопку А и получить число 4.

Описание подзадач и системы оценивания

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

Подзадача	Баллы	Дополнительные ограничения		Необходимые подзадачи
		n	Дополнительные условия на a, b, c	
1	26	$1 \leq n \leq 10^9$	$0 \leq (a + b + c) \leq 7$	
2	23	$1 \leq n \leq 10^{18}$	$c = 0$	
3	24	$1 \leq n \leq 10^{18}$	$b = 0$	
4	27	$1 \leq n \leq 10^{18}$		1, 2, 3

Получение информации о результатах окончательной проверки

По запросу сообщается результат окончательной проверки на каждом тесте.

Задача 3. Размещение данных

Имя входного файла: data.in
Имя выходного файла: data.out
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Телекоммуникационная сеть крупной IT-компании содержит n серверов, пронумерованных от 1 до n . Некоторые пары серверов соединены двусторонними каналами связи, всего в сети m каналов. Гарантируется, что сеть серверов устроена таким образом, что по каналам связи можно передавать данные с любого сервера на любой другой сервер, возможно с использованием одного или нескольких промежуточных серверов.

Множество серверов A называется *отказоустойчивым*, если при недоступности любого канала связи выполнено следующее условие. Для любого не входящего в это множество сервера X существует способ передать данные по остальным каналам на сервер X хотя бы от одного сервера из множества A .

На рис. 1 показан пример сети и отказоустойчивого множества из серверов с номерами 1 и 4. Данные на сервер 2 можно передать следующим образом. При недоступности канала между серверами 1 и 2 — с сервера 4, при недоступности канала между серверами 2 и 3 — с сервера 1. На серверы 3 и 5 при недоступности любого канала связи можно по другим каналам передать данные с сервера 4.

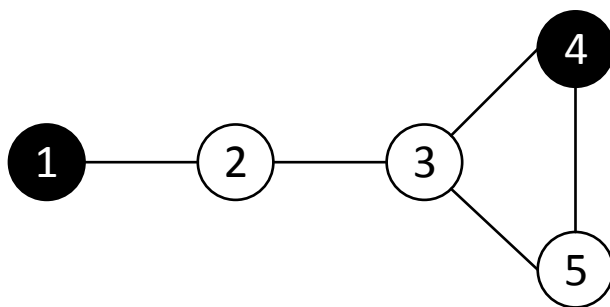


Рис. 1. Пример сети и отказоустойчивого множества серверов.

В рамках проекта группе разработчиков компании необходимо разместить свои данные в сети. Для повышения доступности данных и устойчивости к авариям разработчики хотят продублировать свои данные, разместив их одновременно на нескольких серверах, образующих отказоустойчивое множество. Чтобы минимизировать издержки, необходимо выбрать минимальное по количеству серверов отказоустойчивое множество. Кроме того, чтобы узнать, насколько гибко устроена сеть, необходимо подсчитать количество способов выбора такого множества, и поскольку это количество способов может быть большим, необходимо найти остаток от деления этого количества способов на число $10^9 + 7$.

Требуется написать программу, которая по заданному описанию сети определяет следующие числа: k — минимальное количество серверов в отказоустойчивом множестве серверов, c — остаток от деления количества способов выбора отказоустойчивого множества из k серверов на число $10^9 + 7$

Формат входного файла

Первая строка входного файла содержит целые числа n и m — количество серверов и количество каналов связи соответственно ($2 \leq n \leq 200\,000$, $1 \leq m \leq 200\,000$).

Следующие m строк содержат по два целых числа и описывают каналы связи между серверами. Каждый канал связи задается двумя целыми числами: номерами серверов, которые он соединяет.

Гарантируется, что любые два сервера соединены напрямую не более чем одним каналом связи, никакой канал не соединяет сервер сам с собой, и для любой пары серверов существует способ передачи данных с одного из них на другой, возможно с использованием одного или нескольких промежуточных серверов.

Формат выходного файла

Выведите два целых числа, разделенных пробелом: k — минимальное число серверов в отказоустойчивом множестве серверов, s — количество способов выбора отказоустойчивого множества из k серверов, взятое по модулю $10^9 + 7$

Пример входных и выходных файлов

data.in	data.out
5 5 1 2 2 3 3 4 3 5 4 5	2 3

Пояснение к примеру

В приведенном примере отказоустойчивыми являются следующие множества из двух серверов: $\{1, 3\}$, $\{1, 4\}$, $\{1, 5\}$.

Описание подзадач и системы оценивания

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

Подзадача	Баллы	Дополнительные ограничения		Необходимые подзадачи
		N	m	
1	25	$2 \leq n \leq 10$	$1 \leq m \leq 45$	
2	27	$2 \leq n \leq 200\,000$	$m = n - 1$	
3	28	$2 \leq n \leq 1000$	$1 \leq m \leq 5000$	1
4	21	$2 \leq n \leq 200\,000$	$1 \leq m \leq 200\,000$	1, 2, 3

Получение информации о результатах окончательной проверки

По запросу сообщается результат окончательной проверки на каждом тесте.

Задача 4. Полезные ископаемые

Имя входного файла: mining.in
Имя выходного файла: mining.out
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Ведется проект по освоению планеты соседней звездной системы. Для добычи полезных ископаемых планируется направить на планету несколько партий роботов.

Участок поверхности планеты, на котором планируется добывать полезные ископаемые, представляет собой клетчатый прямоугольник размером w на h , клетки участка имеют координаты от $(1, 1)$ до (w, h) . В некоторых клетках участка находятся базы специалистов, в которые могут быть доставлены партии роботов. Всего на участке размещено s баз, и i -я база находится в клетке с координатами (x_i, y_i) .

Каждая партия роботов характеризуется тремя параметрами: j -я партия доставляется на базу b_j , содержит n_j роботов и каждый робот партии обладает мобильностью m_j .

Когда партия роботов доставляется на соответствующую базу, каждый робот этой партии перемещается по поверхности планеты от базы до некоторой клетки. Если мобильность робота равна m , он может не более m раз переместиться на одну из восьми соседних клеток, как показано на рис. 1.

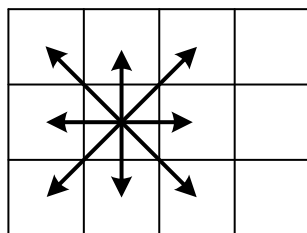


Рис. 1. Возможные перемещения робота в восьми направлениях.

После того как роботы из всех доставленных партий размещаются на участке, они активируются и начинают добычу полезных ископаемых. В процессе перемещения в одной клетке может одновременно находиться произвольное количество роботов. Однако после активации в каждой клетке должно находиться не более q роботов.

Руководством проекта получена информация о t партиях роботов, которые могут быть последовательно отправлены на планету. После доставки всех партий роботов, учитывая их ограниченную мобильность, возможна ситуация, что не удастся разместить роботов на участке так, чтобы в каждой клетке оказалось не больше q роботов. Поэтому руководство должно выбрать k первых партий роботов, где $0 \leq k \leq t$, которые будут полностью доставлены на соответствующие базы. После этого, если $k < t$, следует дополнительно принять z из n_{k+1} роботов следующей, $(k+1)$ -й партии, $0 \leq z < n_{k+1}$.

Все полученные таким образом роботы должны с учетом ограничения на мобильность разместиться на участке таким образом, чтобы в каждой клетке было не более q роботов. После этого они будут активированы и начнут добычу полезных ископаемых. Разумеется, руководство проекта старается максимизировать количество роботов, которые будут доставлены на планету, поэтому, с учетом описанных ограничений, требуется максимизировать k , а затем максимизировать z .

Требуется написать программу, которая по размерам участка, числу q , описанию расположения баз, а также количеству запланированных партий роботов и их описанию определяет максимальное число k — количество партий роботов, и затем — максимальное число z — дополнительное количество роботов из $(k+1)$ -й партии, чтобы, доставив роботов на планету, их можно было разместить на участке таким образом, чтобы в каждой клетке оказалось не более q роботов.

Формат входного файла

Первая строка входного файла содержит числа w , h , s и q ($1 \leq w, h \leq 10^5$, $1 \leq s \leq 4$, $1 \leq q \leq 100$). Последующие s строк содержат по два целых числа x_i , y_i и описывают базы специалистов ($1 \leq x_i \leq w$, $1 \leq y_i \leq h$).

Следующая строка содержит число t — количество партий роботов ($1 \leq t \leq 100$). Последующие t строк описывают партии роботов и содержат по 3 целых числа: b_j , n_j и m_j ($1 \leq b_j \leq s$, $1 \leq n_j \leq w \cdot h \cdot q$, $0 \leq m_j < \max(w, h)$).

Формат выходного файла

Требуется вывести два числа: k и z , $0 \leq k \leq t$. Если $k = t$, то z должно быть равно 0, иначе должно выполняться условие $0 \leq z < n_{k+1}$.

Пример входных и выходных файлов

mining.in	mining.out
4 3 2 1 1 1 3 2 3 1 4 1 2 9 1 1 12 2	1 7

Пояснение к примеру

В приведенном примере описания входных данных следует полностью принять первую партию роботов и дополнительно принять 7 роботов из второй партии. На рис. 2 показано, как можно разместить этих роботов на участке, чтобы в каждой клетке было не более одного робота. Базы специалистов показаны кружками. Клетки, в которых окажутся роботы с базы 1, показаны вертикальной штриховкой, а клетки, в которых окажутся роботы с базы 2, показаны серым цветом.

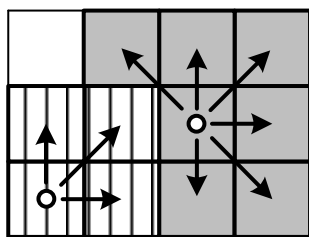


Рис. 2. Возможное размещение роботов на участке в данном примере.

Описание подзадач и системы оценивания

Баллы за каждую из подзадач 1–5 начисляются только в случае, если все тесты этой подзадачи и необходимых подзадач успешно пройдены.

Тесты для подзадачи 6 запускаются только в случае, если все тесты подзадач 1–5 успешно пройдены. Каждый тест в подзадаче 6 оценивается независимо в 1 балл.

Подзадача	Баллы	Дополнительные ограничения			Необходимые подзадачи
		w, h	s	q	
1	18	$1 \leq w, h \leq 20$	$s = 1$	$q = 1$	
2	12	$1 \leq w, h \leq 20$	$1 \leq s \leq 2$	$q = 1$	1
3	9	$1 \leq w, h \leq 20$	$1 \leq s \leq 3$	$q = 1$	1, 2
4	10	$1 \leq w, h \leq 20$	$1 \leq s \leq 3$	$1 \leq q \leq 100$	1, 2, 3
5	15	$1 \leq w, h \leq 10^5$	$s = 1$	$1 \leq q \leq 100$	1
6	36	$1 \leq w, h \leq 10^5$	$1 \leq s \leq 4$	$1 \leq q \leq 100$	1, 2, 3, 4, 5

Получение информации о результатах окончательной проверки

По запросу сообщаются баллы за каждую подзадачу.