Задача A. Калькулятор

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 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: A, B и C.

При нажатии на кнопку A 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 B 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 C 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 C 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 A суммарно a раз, на кнопку B – b раз и на кнопку C – 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 ($1 \le n \le 10^{18}$, $0 \le a, b, c \le 60$). 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

calc.in	calc.out
72 2 1 1	4

Пояснение к примеру

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

Задача B. Интервальные тренировки

В академии физической культуры разработали новый метод интервальных тренировок спортсменов. В соответствии с этим методом спортсмен должен тренироваться каждый день, однако рост нагрузки должен постоянно сменяться её снижением и наоборот.

План тренировки представляет собой набор целых положительных чисел $a_1, a_2, \ldots, a_m$, где a_i описывает нагрузку спортсмена в i-й день. Любые два соседних дня должны иметь различную нагрузку: $a_i \neq a_{i+1}$. Чтобы рост нагрузки и её снижение чередовались, для i от 1 до $m-2$ должно выполняться следующее условие: если $a_i < a_{i+1}$, то $a_{i+1} > a_{i+2}$, если же $a_i > a_{i+1}$, то $a_{i+1} < a_{i+2}$.

Суммарная нагрузка в процессе выполнения плана должна составлять n, то есть $a_1 + a_2 + \ldots + a_m = n$. Ограничения на количество дней в плане нет, m может быть любым, но нагрузка в первый день тренировок зафиксирована: $a_1 = k$.

Прежде чем приступить к тестированию нового метода, руководство академии хочет выяснить, сколько различных планов тренировок удовлетворяет описанным ограничениям.

Требуется написать программу, которая по заданным n и k определяет, сколько различных планов тренировок удовлетворяют описанным ограничениям, и выводит остаток от деления количества таких планов на число $10^9 + 7$.

Формат входных данных

В первой строке входных данных находятся целые числа n, k ($1 \leqslant n \leqslant 5000$, $1 \leqslant k \leqslant n$).

Формат выходных данных

Выведите одно число: остаток от деления количества планов тренировок на $10^9 + 7$.

Примеры

стандартный ввод	стандартный вывод
6 2	4
3 3	1

Пояснения к примерам

В первом примере подходят следующие планы: $[2, 1, 2, 1]$, $[2, 1, 3]$, $[2, 3, 1]$, $[2, 4]$.

Во втором примере единственный подходящий план $[3]$.

Система оценивания

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
1	23	$1 \leqslant n \leqslant 10$		полная
2	20	$1 \leqslant n \leqslant 30$	1	первая ошибка
3	23	$1 \leqslant n \leqslant 500$	1, 2	первая ошибка
4	34	$1 \leqslant n \leqslant 5000$	1, 2, 3	только баллы

Задача C. Укладка плитки

Имя входного файла: `stdin`
Имя выходного файла: `stdout`
Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

В процессе ремонта в Лаборатории Информационных Технологий строителям необходимо заменить поврежденные напольные плитки в коридоре лаборатории, который имеет размер $2 \times n$ метров. В распоряжении строителей есть неограниченный запас плиток двух размеров: 1×2 метра и 1×1 метр. При этом плитки размером 1×2 метра перед укладкой разрешается поворачивать на 90 градусов и размещать как вдоль, так и поперек коридора.

Строители уже начали ремонт и уложили в некоторых местах пола коридора k плиток размером 1×1. Для завершения ремонта прорабу необходимо подготовить план дальнейших работ. Для этого ему надо решить, каким образом уложить плитки на места, где они еще не уложены. Это можно сделать различными способами и прораб хочет перебрать все варианты и выбрать самый удачный. Перед тем как это сделать, прораб хочет знать, какое количество вариантов ему придется рассмотреть. Это число требуется найти по модулю $10^9 + 7$.

Требуется написать программу, которая по заданной длине коридора n и расположению плиток, которые уже уложены, определяет количество способов укладки плиток на оставшиеся места. Ответ необходимо вывести по модулю $10^9 + 7$.

Формат входного файла

Первая строка входного файла содержит два целых числа: n — длину коридора и k — количество уже уложенных единичных плиток ($1 \le n \le 100\,000$, $0 \le k < 2n$).

Последующие k строк содержат по два целых числа x_i и y_i, которые задают позиции уже уложенных единичных плиток, i-я плитка уложена на x_i-м метре коридора в y_i-м ряду ($1 \le x_i \le n$, $1 \le y_i \le 2$).

Формат выходного файла

Выходной файл должен содержать одно целое число — количество способов укладки плиток в коридоре, взятое по модулю $10^9 + 7$.

Примеры входных и выходных файлов

tiling.in	tiling.out
2 0	7
3 0	22
3 1 2 1	8

Пояснение к примерам

Внимание! Третий тест не подходит под ограничения для первых трех подзадач, но решение принимается на проверку только в том случае, если оно выводит правильный ответ на все тесты из примера. Решение должно выводить правильный ответ на третий тест даже, если оно рассчитано на решение только каких-либо подзадач из первых трех.

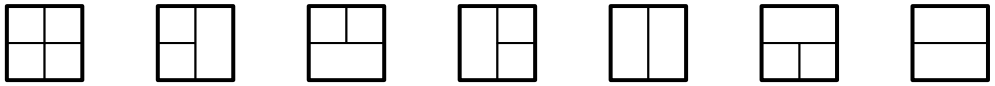

Рисунок 1. Все способы укладки плиток в первом примере

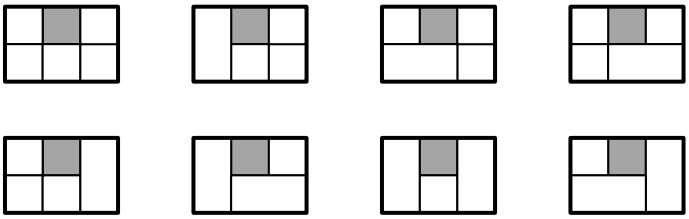

Рисунок 2. Все способы укладки плиток в третьем примере.
Уже уложенная плитка отмечена серым цветом.

Система оценки и описание подзадач

Подзадача 1 (20 баллов)

$1 \le n \le 8$, $k = 0$

Баллы за подзадачу начисляются только в случае, если все тесты подзадачи пройдены.

Подзадача 2 (20 баллов)

$1 \le n \le 1000$, $k = 0$

Баллы за подзадачу начисляются только в случае, если все тесты подзадачи пройдены.

Подзадача 3 (20 баллов)

$1 \le n \le 100\,000$, $k = 0$

Баллы за подзадачу начисляются только в случае, если все тесты подзадачи пройдены.

Подзадача 4 (40 баллов)

$1 \le n \le 100\,000$, $1 \le k \le 2n$

В этой подзадаче 20 тестов, каждый тест оценивается в 2 балла. Баллы за каждый тест начисляются независимо.

Получение информации о результатах окончательной проверки

По запросу сообщается результат окончательной проверки на каждом тесте.

Задача D. Интересные числа

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Софья считает число интересным, если его цифры идут в неубывающем порядке. Например, числа 123, 1111 или 888999 – интересные.

Софья заинтересовалась, сколько существует интересных положительных чисел, лежащих в диапазоне от L до R включительно. Это число может оказаться довольно большим для больших L и R, поэтому Софья хочет найти остаток от деления этого числа на $10^9 + 7$.

Требуется написать программу, которая по заданным L и R определяет количество интересных чисел, лежащих в диапазоне от L до R включительно, и выводит остаток от деления этого числа на $10^9 + 7$.

Формат входного файла

Входной файл содержит две строки. Первая строка содержит число L, вторая строка содержит число R ($1 \le L \le R \le 10^{100}$).

Формат выходного файла

Выходной файл должен одно целое число – остаток от деления количества интересных чисел, лежащих в диапазоне от L до R включительно, на $10^9 + 7$.

Примеры входных и выходных файлов

numbers.in	numbers.out
1 100	54

Описание подзадач и системы оценивания

Подзадача 1 (21 балл)

$L = 1$, $R \le 1000$

Баллы за подзадачу начисляются только в случае, если все тесты подзадачи пройдены.

Подзадача 2 (до 22 баллов)

$1 \le L \le R \le 10^{18}$

В этой подзадаче 11 тестов, каждый тест оценивается в 2 балла. Баллы за каждый тест начисляются независимо.

Подзадача 3 (до 24 баллов)

$L = 1$, $R = 10^k$ для некоторого целого k, $2 \le k \le 100$.

В этой подзадаче 8 тестов, каждый тест оценивается в 3 балла. Баллы за каждый тест начисляются независимо.

Подзадача 4 (до 33 баллов)

$1 \le L \le R \le 10^{100}$

В этой подзадаче 11 тестов, каждый тест оценивается в 3 балла. Баллы за каждый тест начисляются независимо.

Получение информации о результатах окончательной проверки

По запросу сообщается результат окончательной проверки на каждом тесте.

Задача E. Антивещество

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	10 секунд
Ограничение по памяти:	512 мегабайт

Компания тестирует технологию получения антивещества, используемого в качестве топлива в межпланетном звездолёте. Антивещество получается в результате специальных экспериментов в реакторе.

Известно n типов экспериментов, приводящих к получению антивещества. В результате проведения эксперимента i-го типа в выходной контейнер реактора добавляется от l_i до r_i граммов антивещества. Из соображений безопасности запрещается накапливать в контейнере более a граммов антивещества.

Затраты на проведение эксперимента i-го типа составляют c_i, а стоимость одного грамма полученного антивещества составляет 10^9.

Если после проведения экспериментов в контейнере образовалось t граммов антивещества, а суммарные затраты на проведение экспериментов в реакторе составили s, то прибыль определяется по формуле $(t \cdot 10^9 - s)$. Компании необходимо разработать стратегию проведения экспериментов, позволяющую максимизировать прибыль, которую можно гарантированно получить.

В зависимости от результатов предыдущих экспериментов стратегия определяет, эксперимент какого типа следует провести, или решает прекратить дальнейшее выполнение экспериментов. Стратегия позволяет *гарантированно получить* прибыль x, если при любых результатах проведения экспериментов: во-первых, в контейнере реактора оказывается не более a граммов антивещества, во-вторых, прибыль составит не менее x.

Например, пусть возможен только один тип эксперимента, порождающий от 4 до 6 граммов антивещества, затраты на его проведение равны 10, а вместимость контейнера составляет 17 граммов. Тогда после двукратного проведения эксперимента в контейнере может оказаться от 8 до 12 граммов антивещества. Если получилось 12 граммов, то больше проводить эксперимент нельзя, так как в случае получения 6 граммов антивещества контейнер может переполниться. В остальных случаях можно провести эксперимент в третий раз и получить от 12 до 17 граммов антивещества. В худшем случае придётся провести эксперимент трижды, затратив в сумме 30, прибыль составит $(12 \cdot 10^9 - 30) = 11\,999\,999\,970$.

Требуется написать программу, которая определяет максимальную прибыль x, которую гарантированно можно получить.

Формат входных данных

Первая строка входных данных содержит два целых числа: n — количество типов экспериментов и a — максимально допустимое количество антивещества в контейнере ($1 \leqslant n \leqslant 100$, $1 \leqslant a \leqslant 500\,000$).

Следующие n строк содержат по три целых числа l_i, r_i и c_i — минимальное и максимальное количество антивещества, получаемое в результате эксперимента типа i, и затраты на эксперимент этого типа, соответственно ($1 \leqslant l_i \leqslant r_i \leqslant a$, $1 \leqslant c_i \leqslant 100$).

Формат выходных данных

Выходные данные должны содержать одно целое число — максимальную прибыль x, которую гарантированно можно получить.

Примеры

anti.in	anti.out
1 17 4 6 10	11999999970
2 11 2 2 100 3 5 5	9999999890

Задача F. Робогольф

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

На всемирной олимпиаде роботов проводятся соревнования по робогольфу. Поле, на котором происходит игра, имеет вид прямоугольника, состоящего из $n \times m$ единичных квадратов. Строки поля пронумерованы числами от 1 до n, а столбцы — от 1 до m. Таким образом, каждый квадрат характеризуется двумя положительными целыми числами r и c — номерами строки и столбца, на пересечении которых он находится.

Два робота по очереди перемещают специальную фишку по полю, в некоторых квадратах которого находятся ловушки. Если фишка находится в квадрате (r, c), то робот, выполняющий очередной ход, может переместить её в квадрат $(r + 1, c)$ или в квадрат $(r, c + 1)$. Если соответствующего квадрата не оказалось, поскольку фишка находится в последней строке или в последнем столбце поля, робот может переместить её за границу поля. Игра заканчивается в том случае, когда фишка оказывается либо за границей поля, либо в квадрате с ловушкой.

Каждой ловушке соответствует некоторое целое число v_i — её *цена*. *Результат игры* равен значению цены ловушки, в которой закончилась игра, или равен 0, если фишка оказалась за границей поля. Робот, который делает ход первым, стремится минимизировать результат игры, а робот, который делает ход вторым — максимизировать.

Пусть фишка вначале находится в квадрате (r, c). *Гарантированный* результат игры $g(r, c)$ для этого квадрата равен минимальному возможному результату игры, которого может добиться делающий первый ход робот вне зависимости от действий соперника. Поскольку стартовый квадрат до начала матча неизвестен, разработчики роботов хотят вычислить сумму значений $g(r, c)$ для всех квадратов поля.

Требуется написать программу, которая по заданным расположениям и ценам ловушек определит сумму гарантированных результатов игры по всем квадратам поля.

Формат входных данных

Первая строка входных данных содержит три целых числа n, m и k ($1 \leqslant n, m \leqslant 10^9$; $1 \leqslant k \leqslant \min(n \cdot m, 10^5)$) — количество строк, количество столбцов и количество ловушек, соответственно.

Следующие k строк содержат по три целых числа r_i, c_i и v_i ($1 \leqslant r_i \leqslant n$, $1 \leqslant c_i \leqslant m$, $-10^9 \leqslant v_i \leqslant 10^9$) — номер строки, номер столбца и цену i-й ловушки соответственно. Никакие две ловушки не находятся в одном и том же квадрате.

Формат выходных данных

Выведите одно целое число — остаток от деления суммы гарантированных результатов игры по всем квадратам поля на число 998 244 353.

Остаток от деления a на b, где a может быть отрицательным, равен числу $r = a \bmod b$, лежащему в диапазоне от 0 до $b-1$, такому что $a = qb+r$, где q — целое. Например, $13 \bmod 4 = 1$, $-13 \bmod 4 = 3$, $12 \bmod 4 = 0$, $-12 \bmod 4 = 0$.

Примеры

стандартный ввод	стандартный вывод
3 3 3 2 3 -2 3 1 3 1 2 1	998244352
2 4 3 1 2 2 2 4 -3 2 1 1	998244348

Пояснение к примеру

В первом примере гарантированные результаты игр для всех квадратов выглядят так (квадраты с ловушками выделены):

	1	2	3
1	1	1	-2
2	0	-2	-2
3	3	0	0

Сумма результатов равна: $1 + 1 - 2 + 0 - 2 - 2 + 3 + 0 + 0 = -1$. Ответ равен $(-1) \bmod 998\,244\,353 = -1 + 998\,244\,353 = 998\,144\,352$.

В втором примере гарантированные результаты игр для всех квадратов выглядят так (квадраты с ловушками выделены):

	1	2	3	4
1	1	2	0	-3
2	1	0	-3	-3

Сумма результатов равна: $1 + 2 + 0 - 3 + 1 + 0 - 3 - 3 = -5$. Ответ равен $(-5) \bmod 998\,244\,353 = 998\,244\,348$.

Система оценки

Подз.	Баллы	Ограничения			Необх. подзадачи	Результаты во время тура
		n, m	k	дополнительно		
1	20	$n, m \leqslant 1\,000$			У	первая ошибка
2	14	$n \leqslant 5$, $m \leqslant 10^9$			У	первая ошибка
3	14	$n, m \leqslant 100\,000$	$k = n + m - 1$	для любого i, $r_i = n$ или $c_i = m$	—	первая ошибка
4	10			для любого i, $r_i \geqslant n - 1\,000$ и $c_i \geqslant m - 1\,000$	У, 1	первая ошибка
5	6	$n, m \leqslant 100\,000$		для любого i, $v_i = 1$	—	первая ошибка
6	6			для любого i, $v_i = 1$	5	первая ошибка
7	10	$n, m \leqslant 100\,000$			У, 1, 3, 5	первая ошибка
8	10		$k \leqslant 1\,000$		У	первая ошибка
9	10				У, 1–8	первая ошибка

Задача G. Древние династии

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

История Татаро-монгольского ханства богата на правителей. Каждый из N правителей принадлежал к одной из двух династий, причём власть часто переходила от одной династии к другой. Каждое восхождение правителя на престол отмечалось праздником, проводимым 26 марта. В летописях зафиксированы годы проведения этих праздников, причем известно, что правители первой династии устраивали для народа праздник кумыса, а второй — праздник мёда.

На конференции по истории Татаро-монгольского ханства каждый из S учёных предложил свою версию толкования летописи. А именно, i-й историк утверждал, что от каждого праздника кумыса до следующего праздника кумыса проходило не менее KL_i лет, но не более KR_i лет, в то время как от каждого праздника мёда до следующего праздника мёда проходило не менее ML_i лет, но не более MR_i лет.

Каждой предложенной версии может соответствовать несколько распределений правителей по династиям. Ученые договорились считать *показателем сомнительности* распределения число переходов власти к представителю той же самой династии.

Требуется написать программу, которая найдёт распределение, соответствующее хотя бы одной из версий и имеющее наименьший показатель сомнительности, а также версию, которой оно соответствует.

Формат входных данных

В первой строке входного файла записано число N ($2 \leqslant N \leqslant 200\,000$) — количество праздников в летописи. Следующая строка содержит целые числа $X_1, X_2, \ldots, X_N$ ($1 \leqslant X_1 < X_2 < \ldots < X_N \leqslant 10^9$) — годы проведения праздников.

В третьей строке записано число учёных S ($1 \leqslant S \leqslant 50$). В каждой из последующих S строк записаны четыре натуральных числа KL_i, KR_i, ML_i, MR_i ($1 \leqslant KL_i \leqslant KR_i \leqslant 10^9$), ($1 \leqslant ML_i \leqslant MR_i \leqslant 10^9$).

Формат выходных данных

Первая строка выходного файла должна содержать числа P и Q, где P — номер учёного, версии которого соответствует распределение с наименьшим показателем сомнительности, а Q — показатель сомнительности этого распределения.

Вторая строка должна состоять из N цифр 1 и 2, записанных без пробелов, означающих приход к власти представителя первой или второй династии соответственно. Если существует несколько решений с наименьшим показателем сомнительности Q, выведите любое из них.

В случае, если ни в одной из версий учёных не существует способа распределения периодов правления между династиями так, чтобы не нарушались ограничения на промежутки времени между праздниками, выходной файл должен содержать единственное число 0.

Система оценивания

Данная задача содержит пять подзадач. Для оценки каждой подзадачи используется своя группа тестов. Баллы за подзадачу начисляются только в том случае, если все тесты из этой группы пройдены.

Подзадача 1

$2 \leqslant N \leqslant 15$, $1 \leqslant S \leqslant 10$. Подзадача оценивается в 20 баллов.

Подзадача 2

$2 \leqslant N \leqslant 2000$, $1 \leqslant S \leqslant 50$, $N \times S \leqslant 2000$. Подзадача оценивается в 20 баллов.

Подзадача 3

$2 \leqslant N \leqslant 10\,000$, $1 \leqslant S \leqslant 50$, $N \times S \leqslant 10\,000$. Подзадача оценивается в 20 баллов.

Подзадача 4

$2 \leqslant N \leqslant 200\,000$, $1 \leqslant S \leqslant 50$, $N \times S \leqslant 200\,000$. Подзадача оценивается в 20 баллов.

Подзадача 5

$2 \leqslant N \leqslant 200\,000$, $1 \leqslant S \leqslant 50$. Подзадача оценивается в 20 баллов.

Примеры

dynasties.in	dynasties.out
3 1 2 3 1 1 1 1 1	1 1 122
4 1 6 9 13 2 1 2 2 3 6 7 3 3	0
5 3 6 8 9 10 2 2 3 1 1 1 4 1 10	2 0 21212