Задача 1. Выбор зала

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

Для проведения церемонии открытия олимпиады по информатике организаторы осуществляют поиск подходящего зала. Зал должен иметь форму прямоугольника, длина каждой из сторон которого является целым положительным числом.

Чтобы все участники церемонии поместились в зале, и при этом он не выглядел слишком пустым, площадь зала должна находиться в пределах от A до B квадратных метров, включительно.

Чтобы разместить на стенах зала плакаты, рассказывающие об успехах школьников на олимпиадах, но при этом не создать ощущения, что успехов слишком мало, периметр зала должен находиться в пределах от C до D метров, включительно.

Прежде чем сделать окончательный выбор, организаторы олимпиады решили просмотреть по одному залу каждого подходящего размера. Залы с размерами $Y \times Z$ и $Z \times Y$ считаются одинаковыми. Чтобы понять необходимый объем работ по просмотру залов организаторы задались вопросом, сколько различных залов удовлетворяют приведенным выше ограничениям.

Требуется написать программу, которая по заданным A, B, C и D определяет количество различных залов, площадь которых находится в пределах от A до B, а периметр — от C до D, включительно.

Формат входного файла

Входной файл содержит четыре разделенных пробелами целых числа: A, B, C и D ($1 \le A \le B \le 10^9$, $4 \le C \le D \le 10^9$).

Формат выходного файла

Выходной файл должен содержать одно число — искомое количество залов.

Пример входных и выходных файлов

hall.in	hall.out
2 10 4 8	3

Пояснения к примеру

В примере ограничениям удовлетворяют залы следующих размеров: 1×2, 1×3, 2×2.

Система оценки и описание подзадач

Подзадача 1 (50 баллов)

$1 \le A \le B \le 1000$, $4 \le C \le D \le 1000$.

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

Подзадача 2 (50 баллов)

$1 \le A \le B \le 10^9$, $4 \le C \le D \le 10^9$.

В этой подзадаче 25 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 2. Призы

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

Алиса и Боб стали победителями телевикторины, и теперь им предстоит выбрать себе призы. На выбор предлагается n призов, пронумерованных от 1 до n.

Распределение призов происходит следующим образом. Организаторы телевикторины сообщают победителям целое положительное число k ($1 \le k \le n/3$). Сначала Алиса выбирает себе любые k подряд идущих номеров призов. Потом Боб выбирает себе k подряд идущих номеров призов, при этом он не может выбирать номера, которые уже выбрала Алиса. После этого победители забирают выбранные ими призы.

Алиса хорошо знает Боба, и для каждого приза выяснила его ценность для Боба, которая является целым положительным числом. Алиса обижена на Боба и хочет выбрать свои призы так, чтобы суммарная ценность призов, которые достанутся Бобу, была как можно меньше. При этом Алису не волнует, какие призы достанутся ей.

Требуется написать программу, которая по информации о ценности призов и значению k определит, для какого минимального значения x Алиса сможет добиться того, чтобы Боб не смог выбрать призы с суммарной ценностью больше x.

Формат входного файла

Первая строка входного файла содержит два целых числа: n — общее количество призов и k — количество подряд идущих номеров призов, которое должен выбрать каждый из победителей ($3 \le n \le 100\,000$, $1 \le k \le n/3$).

Вторая строка содержит n целых положительных чисел: $a_1, a_2, \ldots, a_n$. Для каждого приза указана его ценность для Боба ($1 \le a_i \le 10^9$).

Формат выходного файла

Выходной файл должен содержать одно число — минимальное значение x, для которого Алиса сможет добиться того, чтобы Боб не смог выбрать призы с суммарной ценностью больше x.

Пример входных и выходных файлов

prizes.in	prizes.out
10 2 1 2 4 5 2 4 2 2 1 6	7

Пояснение к примеру

В приведенном примере Алиса может, например, выбрать 4-й и 5-й призы. После этого для Боба оптимально выбрать 9-й и 10-й призы с суммарной ценностью 7.

Система оценки и описание подзадач

В этой задаче три подзадачи. Баллы за подзадачу начисляются только в случае, если все тесты для данной подзадачи успешно пройдены.

Подзадача 1 (30 баллов)

$3 \le n \le 50$, $1 \le a_i \le 10^5$

Подзадача 2 (30 баллов)

$3 \le n \le 5000$, $1 \le a_i \le 10^5$

Подзадача 3 (40 баллов)

$3 \le n \le 100\,000$, $1 \le a_i \le 10^9$

Получение информации о результатах окончательной проверки

По запросу сообщается результат окончательной проверки на каждом тесте.

Задача 3. Река

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

Во Флатландии протекает богатая рыбой река Большой Флат. Много лет назад река была поделена между n рыболовными предприятиями, каждое из которых получило непрерывный отрезок реки. При этом i-е предприятие, если рассматривать их по порядку, начиная от истока, изначально получило отрезок реки длиной a_i.

С тех пор с рыболовными предприятиями во Флатландии k раз происходили различные события. Каждое из событий было одного из двух типов: банкротство некоторого предприятия или разделение некоторого предприятия на два.

При некоторых событиях отрезок реки, принадлежащий предприятию, с которым это событие происходит, делится на две части. Каждый такой отрезок имеет длину большую или равную 2. Деление происходит по следующему правилу. Если отрезок имеет четную длину, то он делится на две равные части. Иначе он делится на две части, длины которых различаются ровно на единицу, при этом часть, которая ближе к истоку реки, имеет меньшую длину.

При банкротстве предприятия происходит следующее. Отрезок реки, принадлежавший обанкротившемуся предприятию, переходит к его соседям. Если у обанкротившегося предприятия один сосед, то этому соседу целиком передается отрезок реки обанкротившегося предприятия. Если же соседей двое, то отрезок реки делится на две части описанным выше способом, после чего каждый из соседей присоединяет к своему отрезку ближайшую к нему часть.

При разделении предприятия отрезок реки, принадлежавший разделяемому предприятию, всегда делится на две части описанным выше способом. Разделившееся предприятие ликвидируется, и образуются два новых предприятия.

Таким образом, после каждого события каждое предприятие владеет некоторым отрезком реки.

Министерство финансов Флатландии предлагает ввести налог на рыболовные предприятия, пропорциональный квадрату длины отрезка реки, принадлежащего соответствующему предприятию. Чтобы проанализировать, как будет работать этот налог, министр хочет по имеющимся данным узнать, как изменялась величина, равная сумме квадратов длин отрезков реки, принадлежащих предприятиям, после каждого произошедшего события.

Требуется написать программу, которая по заданному начальному разделению реки между предприятиями и списку событий, происходивших с предприятиями, определит, чему равна сумма квадратов длин отрезков реки, принадлежащих предприятиям, в начальный момент времени и после каждого события.

Формат входного файла

Первая строка входного файла содержит два целых числа: n и p — исходное количество предприятий ($2 \le n \le 100\,000$) и номер подзадачи ($0 \le p \le 4$).

Вторая строка входного файла содержит n целых чисел $a_1, a_2, \ldots, a_n$ — длины исходных отрезков реки.

Третья строка входного файла содержит целое число k — количество событий, происходивших с предприятиями ($1 \le k \le 100\,000$).

Последующие k строк содержат описания событий, i-я строка содержит два целых числа: e_i и v_i — тип события и номер предприятия, с которым оно произошло. Значение $e_i = 1$ означает, что предприятие, которое после всех предыдущих событий является v_i-м по

порядку, если считать с единицы от истока реки, обанкротилось, а значение $e_i = 2$ означает, что это предприятие разделилось на два.

Гарантируется, что значение v_i не превышает текущее количество предприятий. Гарантируется, что если отрезок предприятия при банкротстве или разделении требуется поделить на две части, то он имеет длину большую или равную 2. Гарантируется, что если на реке осталось единственное предприятие, оно не банкротится.

Формат выходного файла

Выходной файл должен содержать $(k + 1)$ целых чисел, по одному в строке. Первая строка должна содержать исходную сумму квадратов длин отрезков реки, а каждая из последующих k строк — сумму квадратов длин отрезков реки после очередного события.

Пример входных и выходных файлов

river.in	river.out
4 0 3 5 5 4 5 1 1 2 1 1 3 2 2 1 3	75 105 73 101 83 113

Система оценки и описание подзадач

В этой задаче четыре подзадачи. Баллы за каждую подзадачу начисляются только в случае, если все тесты для данной подзадачи успешно пройдены.

Внимание! Тест из примера не подходит под ограничение подзадачи 3. Тем не менее, решение задачи принимается на проверку только в том случае, если оно выводит правильный ответ на тесте из примера, даже если участник претендует на правильное решение только этой подзадачи.

В первой строке каждого теста после числа n указан номер подзадачи, для теста из примера указано число 0, в тестах первой подзадачи указано число 1, и т. д.

Подзадача 1 (30 баллов)

$2 \le n \le 100$, $1 \le k \le 100$, $1 \le a_i \le 100$, $p = 1$

Подзадача 2 (30 баллов)

$2 \le n \le 100\,000$, $1 \le k \le 100\,000$, $1 \le a_i \le 10^4$, $p = 2$

Для всех i от 1 до $k - 1$ выполнено условие: $|v_i - v_{i+1}| \le 10$

Подзадача 3 (20 баллов)

$2 \le n \le 100\,000$, $1 \le k \le 100\,000$, $1 \le a_i \le 10^4$, $p = 3$

Все события имеют тип $e_i = 1$ (нет разделений предприятий, только банкротства).

Подзадача 4 (20 баллов)

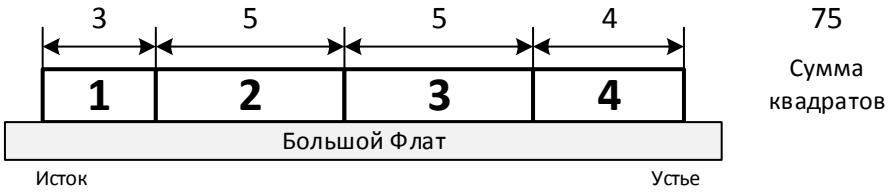
$2 \le n \le 100\,000$, $1 \le k \le 100\,000$, $1 \le a_i \le 10^4$, $p = 4$

Получение информации о результатах окончательной проверки

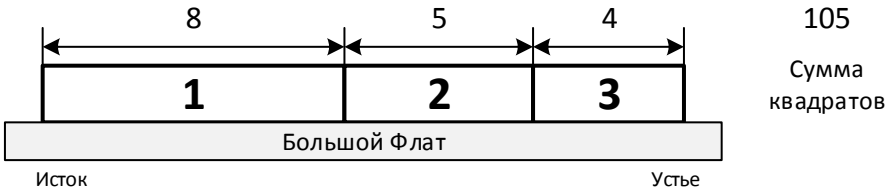
По запросу сообщается результат окончательной проверки на каждом тесте.

Пояснение к примеру

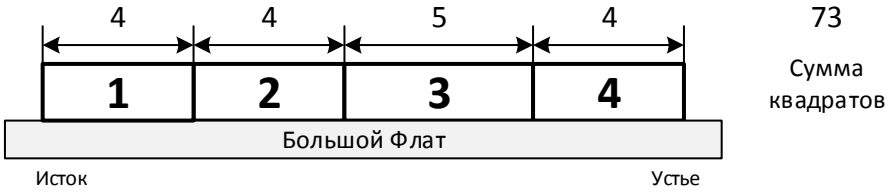
Распределение отрезков реки между предприятиями после каждого события, описанного в примере, приведено на рисунке ниже.



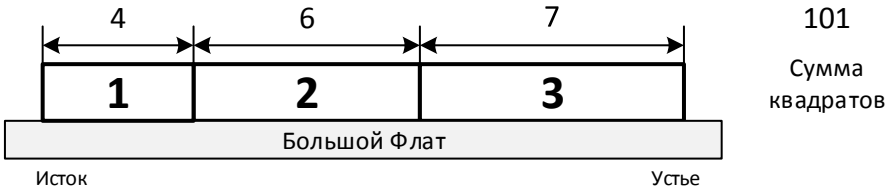
После банкротства предприятия 1



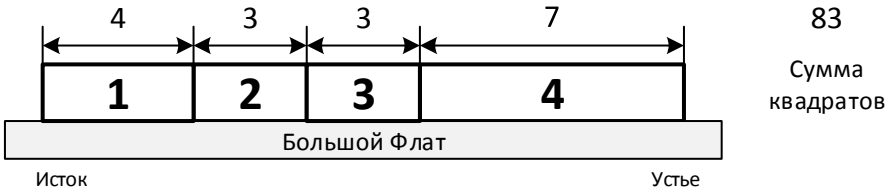
После разделения предприятия 1



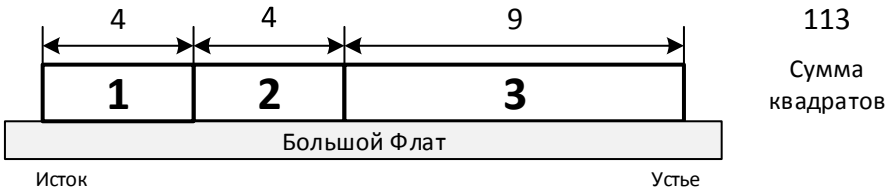
После банкротства предприятия 3



После разделения предприятия 2



После банкроства предприятия 3



Задача 4. Чемпионат по поиску в сети Меганет

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

Для проведения чемпионата мира по поиску в сети Меганет организаторам необходимо ограничить доступ к некоторым адресам. Адрес в сети Меганет представляет собой строку, состоящую из имени сервера и имени раздела.

Имя сервера представляет собой строку, содержащую от одной до пяти частей включительно. Каждая часть представляет собой непустую строку, состоящую из строчных букв латинского алфавита. Части разделены точкой. Примеры корректных имен сервера: «`a`», «`ab.cd`», «`abacaba`», «`a.b.c.d.e`».

Имя раздела представляет собой строку, которая может быть либо пустой, либо содержать от одной до пяти частей включительно. Каждая часть начинается с символа «`/`», после которого следует одна или несколько строчных латинских букв. Примеры корректных имен разделов: «», «`/a`», «`/aba`», «`/a/b/c/d/e`».

Адрес формируется приписыванием имени раздела в конец имени сервера. Например, корректными адресами являются строки: «`a`», «`aba/d/f/g/h`», «`a.b`», «`aba.caba/def/g`», «`c.d.e.f.g/a/b/c/d/e`».

Для ограничения доступа к некоторым адресам сети Меганет организаторы чемпионата подготовили несколько фильтров. Фильтр, как и адрес, состоит из двух частей: фильтра сервера и фильтра раздела.

Фильтр сервера состоит из имени сервера, перед которым может также идти строка «`*.`». Если фильтр сервера представляет собой только имя сервера, то этому фильтру соответствует только сервер, имеющий точно такое же имя. Если фильтр сервера представляет собой строку «`*.`S», где S — имя сервера, то ему соответствуют сервера, удалением нуля или более начальных частей от имени которых можно получить строку S.

Аналогично, *фильтр раздела* представляет собой имя раздела, после которого может идти строка «`/*`». Фильтру раздела, который представляет собой просто имя раздела R, соответствуют только разделы, в точности совпадающие с R. Если фильтр раздела представляет собой строку «R`/*`», то ему соответствуют все разделы, удалением от имен которых нуля или более конечных частей можно получить строку R.

Адрес *соответствует* фильтру, если его имя сервера соответствует фильтру сервера, а его имя раздела соответствует фильтру раздела.

Примеры фильтров и соответствующих им адресов приведены в таблице ниже.

Фильтр	**Примеры подходящих адресов**		
`ab.c/d/e`	`ab.c/d/e`		
`*.a`	`a`	`ax.a`	`efg.a`
`*.a/b/c`	`a/b/c`	`x.a/b/c`	`e.fg.a/b/c`
`x.yz/a/*`	`x.yz/a`	`x.yz/a/b/c`	`x.yz/a/xyz`
`*.a/*`	`a` `a/b/c`	`x.a` `x.a/ddd/c`	`e.fg.a` `e.fg.a/b/c/g/haha/i`
`*.a/b/c/*`	`a/b/c` `a/b/c/xxx`	`x.a/b/c` `e.fg.a/b/c/d/e/f`	`e.fg.a/b/c`

Требуется написать программу, которая по заданному набору фильтров и списку адресов определяет для каждого адреса, какому числу фильтров соответствует этот адрес.

Формат входного файла

Первая строка входного файла содержит два целых числа: n — количество фильтров ($1 \le n \le 50\,000$) и p — номер подзадачи ($0 \le p \le 3$).

Последующие n строк содержат фильтры, по одному на строке. Каждый фильтр удовлетворяет ограничениям, описанным выше.

Следующая строка содержит одно целое число k — количество адресов, которые необходимо проверить на соответствие фильтрам ($1 \le k \le 50\,000$).

Последующие k строк содержат адреса, по одному на строке. Каждый адрес удовлетворяет ограничениям, описанным выше.

Длина каждой строки входного файла не превышает 50 символов.

Общий размер входного файла не превышает 4 мегабайт.

Формат выходного файла

Выходной файл должен содержать k целых чисел, по одному на строке — для каждого адреса необходимо вывести количество фильтров, которым соответствует этот адрес.

Примеры входных и выходных файлов

search.in	search.out
2 0 a.bb/c bb/c/d 4 a.bb bb/c/d a.bb/c/d bb/c	0 1 0 0
4 0 *.bb/c *.bb/c/* bb/c/* bb/c/* 6 bb bb/c bb/c/d a.bb a.bb/c a.bb/c/d	0 4 3 0 2 1

Пояснения к примерам

В первом примере в фильтрах не встречается символ «*», поэтому адрес соответствует фильтру только в случае полного совпадения.

Во втором примере следует обратить внимание на то, что фильтры могут повторяться, а также, что фильтрам вида «*.<сервер>/…» соответствуют также адреса, в которых часть имени сервера полностью совпадает с соответствующей частью фильтра. Аналогично фильтрам «…/<раздел>/*» соответствуют также адреса, в которых часть имени раздела полностью совпадает с соответствующей частью фильтра.

Внимание! Первый тест из примера не подходит под ограничение подзадачи 1, а второй тест из примера не подходит под ограничения подзадач 1 и 2. Тем не менее, решение принимается на проверку только в том случае, если оно выводит правильный ответ на оба теста из примера, даже если участник претендует на правильное решение только одной из указанных подзадач.

Система оценки и описание подзадач

В этой задаче три подзадачи. В первой строке каждого теста после числа n указан номер подзадачи, для теста из примера указано число 0, в тестах первой подзадачи указано число 1, и т. д.

Подзадача 1 (27 баллов)

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

Каждый фильтр начинается с «`*.`» и заканчивается на «`/*`».

В этой подзадаче 9 тестов, каждый тест оценивается в 3 балла. Баллы за каждый тест начисляются независимо.

Подзадача 2 (25 баллов)

$1 \le n \le 50\,000$, $1 \le k \le 50\,000$, $p = 2$

Фильтры не содержат символов «`*`».

В этой подзадаче 5 тестов, каждый тест оценивается в 5 баллов. Баллы за каждый тест начисляются независимо.

Подзадача 3 (48 баллов)

$1 \le n \le 50\,000$, $1 \le k \le 50\,000$, $p = 3$

В этой подзадаче 12 тестов, каждый тест оценивается в 4 балла. Баллы за каждый тест начисляются независимо.

Получение информации о результатах окончательной проверки

По запросу сообщаются баллы за каждую подзадачу.

Задача 5. Кольцевая линия

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

В городе, в котором живут друзья Андрей и Борис, метро состоит из единственной кольцевой линии, вдоль которой на равном расстоянии друг от друга расположены n станций, пронумерованных от 1 до n. Участок линии метро между двумя соседними станциями называется перегоном.

Поезда по кольцевой линии двигаются как по часовой стрелке, так и против часовой стрелки, поэтому чтобы добраться от одной станции до другой, пассажир может выбрать то направление, в котором требуется проехать меньше перегонов. Минимальное число перегонов, которое необходимо проехать, чтобы добраться от одной станции до другой, назовем расстоянием между станциями.

Друзья заметили, что выполняется следующее условие: если загадать некоторую станцию X и выписать для нее два числа: D_a — расстояние от станции, на которой живет Андрей, до станции X и D_b — расстояние от станции, на которой живет Борис, до станции X, то полученная пара чисел $[D_a, D_b]$ будет однозначно задавать станцию X.

Например, если $n = 4$, Андрей живет на станции 1, а Борис живет на станции 2, то станция 1 задается парой [0, 1], станция 2 — парой [1, 0], станция 3 — парой [2, 1] и станция 4 — парой [1, 2].

Их одноклассник Сергей живет в соседнем городе и не знает, на каких станциях живут Андрей и Борис. Чтобы найти друзей, он заинтересовался, сколько существует вариантов пар станций A, B, таких что если Андрей живет на станции A, а Борис — на станции B, то выполняется описанное выше условие.

Требуется написать программу, которая по числу станций n на кольцевой линии определяет искомое количество вариантов.

Формат входного файла

Первая строка входного файла содержит одно целое число n ($3 \le n \le 40\,000$).

Формат выходного файла

Выходной файл должен содержать одно число — искомое количество вариантов.

Примеры входных и выходных файлов

circle.in	circle.out
4	8
5	20

Пояснения к примерам

В первом примере подходят следующие варианты:

- Андрей живет на станции 1, а Борис на станции 2;
- Андрей живет на станции 1, а Борис на станции 4;
- Андрей живет на станции 2, а Борис на станции 1;
- Андрей живет на станции 2, а Борис на станции 3;
- Андрей живет на станции 3, а Борис на станции 2;
- Андрей живет на станции 3, а Борис на станции 4;
- Андрей живет на станции 4, а Борис на станции 1;
- Андрей живет на станции 4, а Борис на станции 3.

Система оценки и описание подзадач

В этой задаче три подзадачи. Баллы за подзадачу начисляются только в случае, если все тесты для данной подзадачи успешно пройдены.

Подзадача 1 (25 баллов)

$3 \le n \le 50$.

Подзадача 2 (25 баллов)

$3 \le n \le 500$.

Подзадача 3 (50 баллов)

$3 \le n \le 40000$.

Получение информации о результатах окончательной проверки

По запросу сообщается результат окончательной проверки на каждом тесте.

Задача 6. Вырубка леса

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

Фермер Николай нанял двух лесорубов: Дмитрия и Федора, чтобы вырубить лес, на месте которого должно быть кукурузное поле. В лесу растут X деревьев.

Дмитрий срубает по A деревьев в день, но каждый K-й день он отдыхает и не срубает ни одного дерева. Таким образом, Дмитрий отдыхает в K-й, $2K$-й, $3K$-й день, и т.д.

Федор срубает по B деревьев в день, но каждый M-й день он отдыхает и не срубает ни одного дерева. Таким образом, Федор отдыхает в M-й, $2M$-й, $3M$-й день, и т.д.

Лесорубы работают параллельно и, таким образом, в дни, когда никто из них не отдыхает, они срубают $A + B$ деревьев, в дни, когда отдыхает только Федор — A деревьев, а в дни, когда отдыхает только Дмитрий — B деревьев. В дни, когда оба лесоруба отдыхают, ни одно дерево не срубается.

Фермер Николай хочет понять, за сколько дней лесорубы срубят все деревья, и он сможет засеять кукурузное поле.

Требуется написать программу, которая по заданным целым числам A, K, B, M и X определяет, за сколько дней все деревья в лесу будут вырублены.

Формат входного файла

Входной файл содержит пять целых чисел, разделенных пробелами: A, K, B, M и X ($1 \le A, B \le 10^9$, $2 \le K, M \le 10^{18}$, $1 \le X \le 10^{18}$).

Формат выходного файла

Выходной файл должен содержать одно целое число — искомое количество дней.

Пример входных и выходных файлов

forest.in	forest.out
2 4 3 3 25	7

Пояснение к примеру

В приведенном примере лесорубы вырубают 25 деревьев за 7 дней следующим образом:

- 1-й день: Дмитрий срубает 2 дерева, Федор срубает 3 дерева, итого 5 деревьев;
- 2-й день: Дмитрий срубает 2 дерева, Федор срубает 3 дерева, итого 10 деревьев;
- 3-й день: Дмитрий срубает 2 дерева, Федор отдыхает, итого 12 деревьев;
- 4-й день: Дмитрий отдыхает, Федор срубает 3 дерева, итого 15 деревьев;
- 5-й день: Дмитрий срубает 2 дерева, Федор срубает 3 дерева, итого 20 деревьев;
- 6-й день: Дмитрий срубает 2 дерева, Федор отдыхает, итого 22 дерева;
- 7-й день: Дмитрий срубает 2 дерева, Федор срубает оставшееся 1 дерево, итого все 25 деревьев срублены.

Внимание! Тест из примера не подходит под ограничения для подзадач 2 и 3, но решение принимается на проверку только в том случае, если оно выводит правильный ответ на тесте из примера. Решение должно выводить правильный ответ на тест даже, если оно рассчитано на решение только каких-либо из подзадач 2 и 3.

Система оценки и описание подзадач

Подзадача 1 (32 балла)

$1 \le X \le 1000$, $1 \le A, B \le 1000$, $2 \le K, M \le 1000$

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

Подзадача 2 (10 баллов)

$1 \le X \le 10^{18}$

$X < K$

$X < M$

При решении этой подзадачи можно считать, что лесорубы не отдыхают.

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

Подзадача 3 (10 баллов)

$1 \le X \le 10^{18}$

Дополнительно к приведенным ограничениям выполняется условие $K = M$.

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

Подзадача 4 (48 баллов)

$1 \le X \le 10^{18}$, $1 \le A, B \le 10^9$, $2 \le K, M \le 10^{18}$

В этой подзадаче 16 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 7. Укладка плитки

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

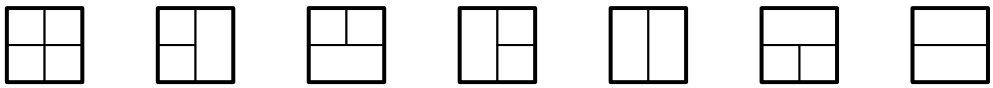


Рисунок 1. Все способы укладки плиток в первом примере

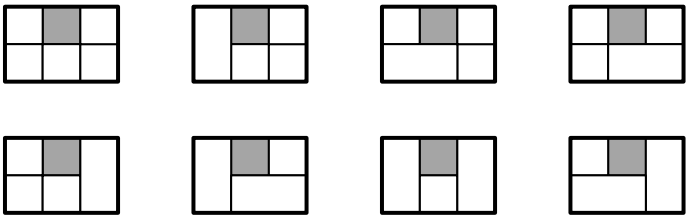


Рисунок 2. Все способы укладки плиток в третьем примере.
Уже уложенная плитка отмечена серым цветом.

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 8. Магические порталы

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

В некотором королевстве есть n городов, соединенных магическими порталами. Каждая пара различных городов соединена ровно одним магическим порталом, позволяющим мгновенно перемещаться из одного города в другой.

Из-за свойств магии, определяющей работу порталов, каждый портал можно использовать только в одну сторону. Для каждой пары городов A и B известно, можно ли воспользоваться порталом для перемещения напрямую из A в B или из B в A.

Из-за особенностей магических порталов иногда при перемещении жителей королевства из одного города в другой приходится использовать несколько порталов. Также могут существовать такие пары городов, что из одного города в другой нельзя добраться, используя только магические порталы.

Жители королевства называют город *совершенным*, если из него можно добраться до любого другого города в королевстве, используя только магические порталы. Пусть изначально количество совершенных городов в королевстве равно k.

Недавно король принял решение выбрать пару городов и изменить разрешенное направление перемещения по порталу, соединяющему их, на противоположное. Для выбора лучшего варианта король хочет понять, какое количество совершенных городов может оказаться в королевстве после перенастройки ровно одного портала.

Для получения этой информации король планирует запросить в министерстве транспорта соответствующий отчет. Король может запросить либо *частичный*, либо *полный* отчет. Содержимое отчета зависит от параметра L, для частичного отчета $L = k + 1$, для полного отчета $L = 1$.

Отчет содержит для каждого целого числа m, такого что $m \geq L$, число таких пар городов A и B, для которых выполняются следующие условия:

- исходно магический портал позволяет перемещаться напрямую из города A в город B;
- если изменить направление перемещения этого магического портала на противоположное, чтобы он позволял напрямую перемещаться из города B в город A, то количество совершенных городов в королевстве станет равным m.

Таким образом, частичный отчет содержит информацию только о тех способах перенастройки одного портала, которые строго увеличивают количество совершенных городов в королевстве. Полный отчет содержит информацию обо всех способах перенастройки одного портала.

Требуется написать программу, которая по информации о разрешенных направлениях перемещения с использованием магических порталов, и информации о том, требуется предоставить частичный или полный отчет, формирует соответствующий отчет и выводит его в описанном ниже формате.

Формат входного файла

Первая строка входного файла содержит два целых числа: n — количество городов в королевстве ($2 \leq n \leq 2000$) и p, равное либо 0, если требуется вывести частичный отчет, либо 1, если требуется вывести полный отчет.

Последующие n строк содержат по n символов, каждый из которых может быть «+», «−» или «`.`», и i-я из этих строк описывает магические порталы, соединяющие i-й город с другими городами.

В i-й строке j-й символ равен «+», если магический портал позволяет напрямую перемещаться из i-го города в j-й, равен «−», если магический портал позволяет напрямую перемещаться из j-го города в i-й, и равен «.», если $i = j$.

Формат выходного файла

Первая строка выходного файла должна содержать одно целое число k — количество совершенных городов в королевстве.

Если требуется частичный отчет (p = 0), то вторая строка выходного файла должна содержать $(n - k)$ целых неотрицательных чисел, разделенных пробелами, где i-е из этих чисел должно быть равно количеству пар городов, изменение направления портала между которыми на противоположное приводит к тому, что количество совершенных городов в королевстве станет равным $(k + i)$. Если при этом $k = n$, то вторая строка может отсутствовать, либо быть пустой.

Если требуется полный отчет (p = 1), то вторая строка должна содержать n целых неотрицательных чисел, разделенных пробелами, где i-е из этих чисел должно быть равно количеству пар городов, изменение направления портала между которыми на противоположное приводит к тому, что количество совершенных городов в королевстве станет равным i.

Примеры входных и выходных файлов

transform.in	transform.out
5 0 .-+++ +.+++ --.+- ---.+ --+-.	1 0 0 0 3
5 1 .-+++ +.+++ --.+- ---.+ --+-.	1 7 0 0 0 3

Пояснение к примерам

В приведенных примерах изначально совершенным является только город 2.

Изменив направление порталов, соединяющих пары городов (2, 3), (2, 4) или (2, 5), можно сделать все города совершенными. Изменение направление любого другого портала делает совершенным один город.

Внимание! Второй 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 второй тест даже, если оно рассчитано на решение только каких-либо подзадач из первых трех.

В других тестах $n \neq 5$.

Система оценки и описание подзадач

В этой задаче четыре подзадачи. Баллы за каждую подзадачу начисляются только в случае, если все тесты подзадачи пройдены.

Подзадача 1 (20 баллов)

$2 \le n \le 50$, $p = 0$

Подзадача 2 (30 баллов)

$2 \le n \le 300$, $p = 0$

Подзадача 3 (20 баллов)

$2 \le n \le 2000$, $p = 0$

Подзадача 4 (30 баллов)

$2 \le n \le 2000$, $p = 1$

Получение информации о результатах окончательной проверки

По запросу сообщается баллы за каждую подзадачу.