Задача 1. Взлом пароля

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

На Васином любимом сайте обнаружилась уязвимость: недоброжелатель может узнать, сколько раз встречался каждый символ в пароле. И такая напасть, что сайт позволяет использовать в качестве символов пароля только латинские строчные и заглавные буквы, а также цифры. И никаких спецсимволов!

На сайте отсутствует защита от ботов: ни капч, ни двухфакторной аутентификации. Брутфорсь в удовольствие.

Вася придумал новый пароль, и ему интересно, насколько он надёжен при учёте этой уязвимости. Но сам он только новости про уязвимости читает, а код написать не может – просит Вас.

Нужно найти, с какой попытки зайдёт в его аккаунт взломщик, узнавший набор символов, которые встречаются в пароле, если будет перебирать все подходящие пароли в лексикографическом порядке.

Формат входных данных

В первой строке входного файла содержится натуральное число N – длина пароля ($1 \leqslant N \leqslant 100$).

Во вторую строку записана последовательность символов длины N, состоящая только из цифр и из букв латинского алфавита, как строчных, так и заглавных.

Формат выходных данных

В выходной файл необходимо вывести одно натуральное число – номер попытки, с которой будет введён правильный пароль, если перебирать все пароли в лексикографическом порядке, а попытки считать с единицы.

Так как число может получиться очень большим, то требуется вывести его по модулю $10^9 + 7$.

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
1	0	Тесты из условия	
2	20	$1 \leqslant N \leqslant 12$	1
3	40	$1 \leqslant N \leqslant 100$, каждый символ повторяется не более одного раза	1, 2
4	40	$1 \leqslant N \leqslant 100$, каждый символ может повторяться больше одного раза	1, 2, 3

Примеры

input.txt	output.txt
5 bOjXg	53
10 fg6Fg10000	60030
5 JJJJJ	1

Замечание

Лексикографический порядок – это отношение любых строк такое, что строка α предшествует строке β, если

1. Любые первые m символов совпадают, а $(m+1)$-й символ строки α меньше $(m+1)$-го символа строки β. При этом считается, что буквы сравниваются согласно их кодам в ASCII, то есть цифры меньше заглавных латинских букв, а заглавные латинские буквы меньше строчных латинских букв.

   Пример: `GREW` $<$ `GROW`

2. Строка α является началом строки β

   Пример: `GROW` $<$ `GROWN`

Задача 2. Графляндия

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Правитель государства Графляндия, которая славится односторонним дорожным движением, хочет провести реформу дорог, цель которой — получить «идеальный» город. Идеальным считается такой город X, что если выехать из него, то можно вернуться обратно, посетив по пути непустое множество других городов.

Правитель настолько вдохновился идеей реформации, что решил действовать незамедлительно, предложив следующий метод создания идеального города. Он выбирает произвольные два города U и V в своем государстве и выдвигает приказ об их объединении в один город X. При этом множество дорог, ведущих в X будет состоять из объединения множеств дорог, ведущих в города U и V, а множество выходящих из X дорог — это объединение множеств дорог, выходящих из U и из V, а дороги, соединяющие напрямую города U и V в любую сторону, становятся внутренними дорогами города X.

Так как правитель очень спешит, он просит вас написать программу, которая позволит понять, будет ли город X идеальным, если объединить города U и V.

Формат входных данных

Первая строка входного файла содержит целое число N — количество городов в Графляндии ($2 \leqslant N \leqslant 10^4$). Города пронумерованы числами от 1 до N.

В каждой из следующих N строк содержится целое число M_i — количество дорог, ведущих в город с номером i, а также номера городов p_{i_j}, из которых эти дороги выходят ($0 \leqslant i < N$, $0 \leqslant M_i \leqslant N-1$, $0 \leqslant p_{i_j} < N$).

В следующей строке содержится число Q — количество запросов ($1 \leqslant Q \leqslant 10^6$).

Далее в Q строках записаны запросы. Каждый запрос задается парой чисел U и V — номерами городов, с помощью объединения которых правитель хочет попытаться получить «идеальный» город ($0 \leqslant U, V < N$).

Каждый из запросов является независимым.

Гарантируется, что изначально в государстве нет «идеальных» городов, а также все дороги между городами являются односторонними.

Формат выходных данных

В выходной файл необходимо вывести Q строк — ответ на каждый из запросов. Если возможно получить «идеальный» город, объединив указанные в запросе города, то в соответствующую строку должно быть выведено слово `YES`, иначе нужно вывести слово `NO`.

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
1	0	Тесты из условия	
2	20	$1 \leqslant N \leqslant 10^4$, $Q = 1$	1
3	30	$1 \leqslant N \leqslant 500$, $1 \leqslant Q \leqslant 10^3$	1, 2
4	50	$1 \leqslant N \leqslant 10^4$, $1 \leqslant Q \leqslant 10^6$	1, 2, 3

Примеры

input.txt	output.txt
4 0 1 0 1 0 2 1 2 3 0 2 0 3 1 2	NO YES NO
7 0 1 0 1 0 1 0 2 1 0 1 2 3 4 5 3 8 2 0 6 1 4 6 2 4 5 6 0 6 0 1 3 4	NO YES NO NO NO YES NO NO
11 0 1 0 1 0 1 0 1 0 1 1 4 5 2 3 4 3 5 0 2 2 2 3 2 0 7 2 6 8 7 0 4 4 2 6 2 5 4 0 7 5 10 5 8	NO NO NO NO YES YES NO

Задача 3. Дополнительные секунды

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вы, наверное, замечали, что часы могут спешить или отставать, из-за чего координировать действия становится сложно. В наше время такая проблема встаёт всё реже — на электронных устройствах с доступом в интернет можно просто синхронизировать время.

Существует несколько стандартов времени, но основными являются UT1 и UTC. Универсальное время (UT1) зависит от солнечного времени на нулевой долготе и является максимально точным. Но используется везде Всемирное координированное время (UTC) — приближённое значение UT1 к шкале атомных часов.

Иногда добавляются дополнительные секунды так, чтобы UTC отличалось от UT1 на не более чем на ± 0.9 секунды.

В данное время о добавлении секунд специально сообщает Международная служба вращения Земли 30-го июня или 31-го декабря, но в будущем может потребоваться добавлять секунды чаще, а также может понадобиться их отнимать.

Вы считаете разницу во времени между годами $[s_i, f_i)$, и хотели бы дать наиболее точный результат. Вы уже учли всё-всё-всё, осталось лишь узнать, на сколько секунд надо скорректировать результат.

Формат входных данных

В первой строке входного файла содержится два целых числа N и Q — количество известных вам лет, когда были добавлены или вычтены дополнительные секунды, и количество запросов соответственно ($1 \leqslant N \leqslant 10^6,\ 1 \leqslant Q \leqslant 10^5$).

В следующих N строках записаны пары целых чисел y_i и n_i — год и количество дополнительных секунд ($-10^9 \leqslant y_i,\ n_i \leqslant 10^9$). Если $n_i < 0$, то эти секунды вычитались, иначе — добавлялись.

В следующих Q строках записаны запросы, которые задаются парами чисел s_i и f_i — двумя годами, между которыми требуется посчитать временной интервал ($-10^9 \leqslant s_i,\ f_i \leqslant 10^9$).

Формат выходных данных

Выходной файл должен содержать Q строк, на каждой из которой должно быть одно число — количество секунд, на которое нужно скорректировать ваши расчёты для соответствующего запроса.

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
1	0	Тесты из условия	
2	35	$1 \leqslant N,\ Q \leqslant 10^4$	1
3	65	$1 \leqslant N \leqslant 10^6,\ 1 \leqslant Q \leqslant 10^5$	1, 2

Примеры

input.txt	output.txt
10 5 85 8 68 7 14 10 0 -10 -57 -8 70 -10 87 7 -73 8 -40 10 62 4 -97 50 10 45 47 97 56 68 91 97	10 10 16 4 0
15 6 61 -4 16 -6 23 -7 45 3 30 10 45 -2 25 -8 93 -3 64 1 42 -4 61 -4 41 -8 65 10 14 2 81 -6 12 23 24 97 35 83 49 79 51 69 89 90	-4 -15 -14 3 3 0

Задача 4. Заключительный тур ВсеСиба

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

На финальный тур Всесибирской олимпиады прошло N участников, и всех их надо рассадить в две аудитории.

Умная нейросеть проанализировала социальные сети участников и выделила пары друзей. Организаторы, располагая этой информацией, хотят распределить участников между двумя аудиториями так, чтобы в них сидело равное количество участников и никакие два друга не сидели в одной аудитории.

Это довольно просто сделать, но организаторов очень волнует другой вопрос: сколько существует подходящих распределений участников по аудиториям. Два распределения считаются **различными**, если существует пара участников, которые сидят в одной и той же аудитории в одном распределении, но в разных аудиториях в другом распределении.

Например, по такой логике рассадки варианты $(1, 2)$ / $(3, 4)$ и $(3, 4)$ / $(1, 2)$ считаются одинаковыми.

Организаторы не могут ответить на этот вопрос сами, поэтому просят вашей помощи.

Формат входных данных

В первой строке входного файла записаны два целых числа N и M — количество участников и количество выявленных пар друзей соответственно ($1 \leqslant N \leqslant 5 \cdot 10^3$, $0 \leqslant M \leqslant 10^6$).

Далее на M строках содержится по два различных числа a_i и b_i — номера участников i-й пары друзей ($1 \leqslant a_i,\ b_i \leqslant N$).

Гарантируется, что каждая пара друзей записана ровно один раз.

Формат выходных данных

В выходной файл необходимо вывести только одно целое число — количество различных распределений участников по аудиториям такое, что любые два друга пишут олимпиаду в разных аудиториях, а в каждой из аудиторий сидит равное количество участников.

Система оценки

Подзадача	Баллы	Ограничения	Необходимые подзадачи
1	0	Тесты из условия	
2	20	$N \leqslant 10$, $M \leqslant 20$	1
3	30	$N \leqslant 30$, $M \leqslant 100$	1, 2
4	50	$N \leqslant 5 \cdot 10^3$, $M \leqslant 10^6$	1, 2, 3

Примеры

input.txt	output.txt
6 3 1 2 2 3 4 5	2
5 0	0

Задача 5. Абонементы в парк

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	128 мегабайт

Петр Сергеевич, в прошлом хороший программист, решил сменить область деятельности. Теперь он директор парка аттракционов. Но первая специальность ему очень пригодилась. Он написал программу, которая генерирует коды для абонементов в его парк. Каждый код можно транслировать в последовательность нулей и единиц. Некоторые позиции в этой последовательности занимает служебная информация, которую менять нельзя. На остальные свободные позиции можно ставить либо 0 либо 1. Код для каждого абонемента должен быть уникальным.

Наш директор напечатал первую тестовую партию абонементов. Некоторые из них попали в руки хулиганов, которых больше пускать в парк не хочется. Система безопасности парка записала в свою базу некоторые вырезки из кодов этих нежелательных посетителей. Очередную партию абонементов нужно сделать так, чтобы их коды не содержали подстрок, записанных в базу.

Петр Сергеевич хочет узнать, сколько новых абонементов он может напечатать, но заниматься этим ему некогда, так как у директора много других обязанностей. Он просит решить эту задачу Вас.

Формат входных данных

Первая строка входного файла содержит шаблон для генерации кодов абонементов. Это последовательность, состоящая из символов `0`, `1` и `?`. Ее длина (N) не превосходит 10^4. Символ `?` обозначает пропуск, куда можно поставить либо `0` либо `1`.

В следующей строке записано одно целое число T — количество запрещённых строк ($0 \leqslant T \leqslant 10^4$).

Далее в T строках указаны запрещённые строки, состоящие из `0` и `1`. Суммарная длина запрещенных строк (K) не превосходит 10^4

Формат выходных данных

В выходной файл необходимо вывести одно целое число — количество способов заполнить пропуски в шаблоне так, чтобы запрещенные строки не являлись подстроками генерируемого кода. Это число может быть очень большим, поэтому его нужно вывести по модулю $10^9 + 7$.

Система оценки

Подзадача	Баллы	Ограничения	Необходимые подзадачи
1	0	Тесты из условия	
2	23	количество символов ? в шаблоне не превосходит 10, $N \leqslant 1000, K \leqslant 1000$	1
3	77	$N \leqslant 10^4, K \leqslant 10^4$	1,2

Примеры

input.txt	output.txt
11?101? 1 111	2
11?101? 2 111 110	0

Задача 6. Кубик, соберись!

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Ваня не умеет собирать кубик Рубика, и вместо того, чтобы научиться этому, он написал программу, которая собирает кубик Рубика за него. На вход этой программе подается развертка кубика, а в ответ она выдает последовательность поворотов граней, после выполнения которых кубик должен собраться.

Ваня не уверен, что его программа работает правильно, и поэтому он хочет ее протестировать. Для этого он просит вас написать вспомогательную программу, которая по изначальной развертке и последовательности поворотов вычисляет развертку после выполнения поворотов.

Собранный кубик Рубика состоит из 6 граней, каждая из которых имеет свой цвет: белый (`W`), оранжевый (`O`), желтый (`Y`), зеленый (`G`), синий (`B`) и красный (`R`).

Каждая грань имеет имеет размер 3×3. Одну из граней обозначают как переднюю (`F`), и относительно нее определяют заднюю (`B`), верхнюю (`U`), нижнюю (`D`), правую (`R`) и левую (`L`) грани.

Каждую из граней можно поворачивать по часовой или против часовой стрелки. При этом направление часовой стрелки определяем, представляя, что держим поворачиваемую грань перед собой. Поворот грани по часовой стрелке будем записывать, как условное обозначение поворачиваемой грани. При повороте против часовой стрелки к обозначению поворачиваемой грани будем дописывать штрих (символ 39 в ASCII).

Например, поворот передней грани по часовой стрелке будет записываться как `F`. Поворот левой грани против часовой стрелки будет записываться как `L'`.

Формат входных данных

Для ввода/вывода будем использовать следующий вид развертки:

```
U
F R B L
D
```

Каждая грань задается девятью символами, расположенными в трех строках по три символа в каждой.

Первые девять строк входного файла задают изначальную развертку кубика. Они состоят только из символов `W`, `O`, `Y`, `G`, `B`, `R`, которые обозначают цвета граней. 1-3 и 7-9 строчки состоят из трех символов. 4-6 строчки состоят из двенадцати символов. В строках, задающих развертку, нет пробелов.

В десятой строке входного файла записано число N ($1 \leqslant N \leqslant 10^5$) — количество поворотов граней.

В последней строке через пробел записаны N обозначений поворотов граней, каждое из которых состоит либо из одного символа (обозначение грани), если это поворот по часовой стрелке, либо из двух (обозначение грани и символ штриха ASCII-39), если это поворот против часовой стрелки. Грани обозначаются символами `F`, `B`, `U`, `D`, `R`, `L`.

Формат выходных данных

Выходной файл должен содержать развертку кубика после выполнения заданных поворотов, записанную в том же формате и с тем же расположением граней, что и во входном файле.

Гарантируется, что изначальная развертка корректна, то есть она получена из собранного кубика Рубика определенной последовательностью поворотов граней, но не гарантируется, что расположение цветов граней у собранного кубика совпадало с классическим.

Система оценки

Каждый тест, кроме теста из условия, оценивается по отдельности в 2 балла. Если решение не проходит тест из условия, оно получает 0 баллов.

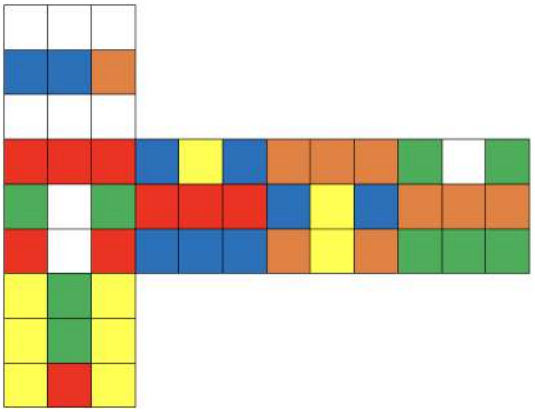
Пример

input.txt	output.txt
WWW BBO WWW RRRBYBOOOGWG GWGRRRBYBOOO RWRBBBOYOGGG YGY YGY YRY 4 U’ L R’ B	BBB BBB BBB WWWRRRYYYOOO WWWRRRYYYOOO WWWRRRYYYOOO GGG GGG GGG

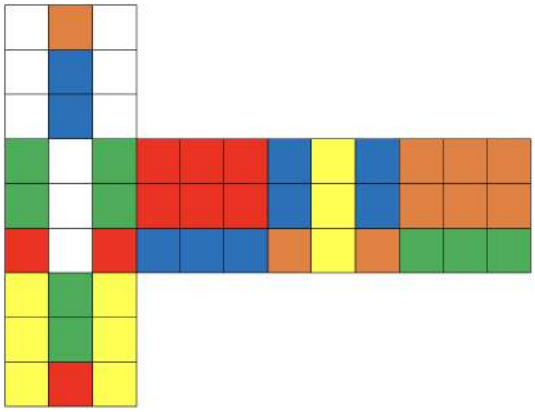
Замечание

Развертки кубика в примере из условия в порядке сборки:

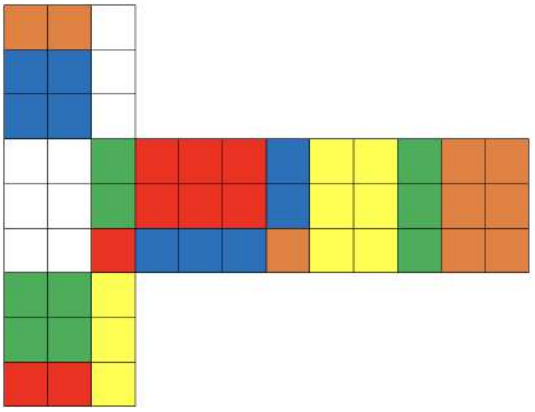
0.



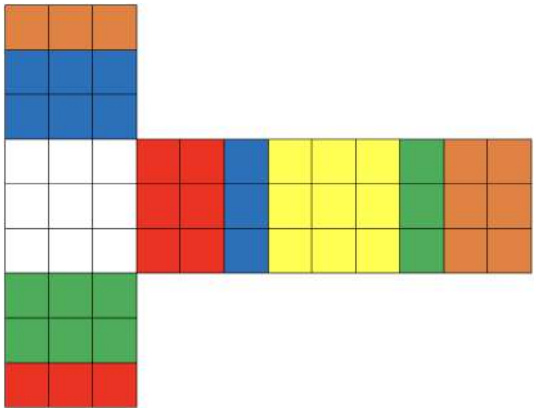
1.



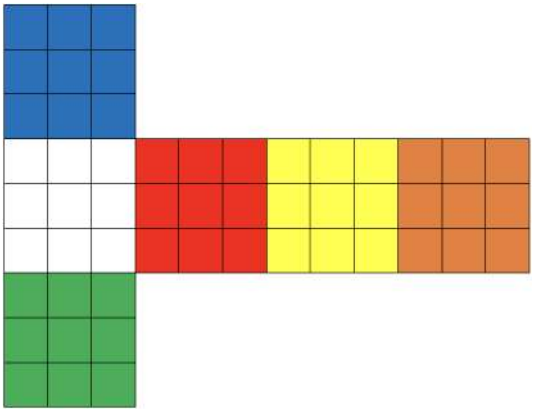
2.



3.



4.



Задача 7. Майнкрафт

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Вася играет в компьютерную игру. Перед ним стоит следующая задача:

Под землей находится ограниченное пространство — параллелепипед, разделенный на $W \cdot H \cdot D$ равных блоков (кубиков). Блоки бывают двух видов: из твердой и мягкой породы. Раскапывать можно блоки только из мягкой породы, при этом только полностью. Существует N входов и N выходов из этого пространства. Входы и выходы — блоки мягкой породы, находящиеся у края пространства. Они имеют не более 5 соседних блоков. Блоки считаются соседними, если у них есть общая сторона. Перемещаться можно только между соседними блоками, если оба они раскопаны.

Необходимо прокопать N проходов таким образом, чтобы ровно один проход соединял один вход и один выход, и ни один проход не пересекался с другим. При этом проходы могут соприкасаться между собой. Вход и выход считаются соединенными проходом только в том случае, если возможно пройти от этого входа до выхода.

Помогите Васе решить эту задачу.

Формат входных данных

В первой строке входного файла записаны три целых числа H — высота, D — глубина и W — ширина пространства ($1 \leqslant H \leqslant 50$, $2 \leqslant D,\ W \leqslant 50$).

Во второй строке записано целое число N — количество входов и выходов ($1 \leqslant N \leqslant 10$).

В следующих $(D+1) \cdot H$ строках записано, из каких блоков состоит пространство. Всего H слоев, после описания каждого слоя вставлена пустая строка. Каждый слой записан в D строках, каждая из которых содержит W символов. Слои расположены друг над другом в порядке ввода.

Символы, задающие карту пространства имеют следующие обозначения:

- `'.'` — блок из мягкой породы;
- `'X'` — блок из твердой породы;
- `'S'` — один из входов в пространство;
- `'F'` — один из выходов из пространства.

Формат выходных данных

Если прокопать требуемое количество проходов невозможно, в выходной файл необходимо вывести слово `NO`.

Если же прокопать такие ходы можно, то в первую строку нужно вывести слово `YES`, а в следующих $(D+1) \cdot H$ строках должно быть расположено описание пространства с отмеченными ходами в том же формате, в котором оно записано во входных данных. Раскопанный блок, принадлежащий проходу с номером i необходимо отметить цифрой i ($0 \leqslant i \leqslant 9$). Вход и выход также необходимо раскопать и пометить номером прохода. Нетронутые блоки должны выводиться в неизменном виде. Если ответов несколько, выведите любой.

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
1	0	Тесты из условия	
2	20	$1 \leqslant H \leqslant 15$; $2 \leqslant D,\ W \leqslant 15$; $N = 1$	1
3	30	$H = 1$; $2 \leqslant D,\ W \leqslant 15$; $1 \leqslant N \leqslant 10$	1
4	30	$1 \leqslant H \leqslant 15$; $2 \leqslant D,\ W \leqslant 15$; $1 \leqslant N \leqslant 10$	1, 2, 3
5	20	$1 \leqslant H \leqslant 50$; $2 \leqslant D,\ W \leqslant 50$; $1 \leqslant N \leqslant 10$	1, 2, 3, 4

Примеры

input.txt	output.txt
1 3 3 1 XXX S.. X.F	YES XXX 00. X00
3 3 3 2 S.F S.. X.. ... ... .XF ..X X..	YES 000 1.. X.. 111 1.. ... .X1 ..X X..

Замечание

На рисунке показано пространство из второго примера. Зелеными кубиками обозначены входы, красными — выходы. Прозрачные кубы — это блоки из мягкой породы, а черные — блоки из твердой породы.