Задача A. Суффиксный путь

Имя входного файла:	`sufpath.in`
Имя выходного файла:	`sufpath.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В одной супер-секретной лаборатории «Кимод йытясед» было разработано супер-секретное устройство. Синдикату «Черное солнце» удалось выяснить, что устройство имело супер-секретное название «Тамотва 173». Лучшие умы пытались расшифровать его, но так ничего и не удавалось. Пробовали и шифр цезаря, и сдвиг, и RSA. В конце концов, синдикат обратился за помощью в Весенний Гуманитарный Детский сад (ВГД). Как оказалось, это — также супер-секретная организация, которая занимается подготовкой элитного отряда дворников. Они профессионально умеют складывать мусор на самых видных местах, подметать так, что просыпается весь район и также немного увлекаются философией. Разумеется, они мгновенно поняли, что в названии зашифровано ни что иное, как основная часть этого устройства! Еще они выяснили, что это устройство умеет обрабатывать только файлы размера не более, чем шестьдесят мегабайт.

После этого синдикат обратился за помощью к лучшим друзьям ВГД, Еженедельному Женевскому Завтраку (ЕЖЗ). Они также специализируется на шифрах: красят заборы, моют полы, а также занимаются модернизацией и инновациями. ЕЖЗ сообщили, что это устройство — их профиль, оно идеально сочетается с их идеологией. Более формально, устройство также умеет выполнять модернизацию. К сожалению, с инновациями может справиться только ЕЖЗ, поэтому устройство не умеет их придумывать, а может лишь проверять, является ли некоторая инновация действительно революционной инновацией. Как оказалось, внутри устройства хранится строка, состоящая из маленьких латинских букв. Модернизация состоит в том, чтобы к хранящейся строке дописать маленькую латинскую букву. ЕЖЗ хотели сообщить более подробную информацию, но после фразы, что ВГД ошиблись в подсчете максимального размера обрабатываемого файла в тридцать раз, они были в непригодном для общения состоянии: у большинства отвалилась челюсть.

Синдикат обратился к своим последним друзьям: команде Инноваций и Культа Лени (ИКЛ). Они сразу объяснили, что строка является для устройства инновационной, если она является частью хранящейся в устройстве строки. Но всем сразу стало ясно: здесь есть какой-то подвох! И снова лучшие умы стали биться над этой задачей. Перепробовали всё: и кричать, и танцевать, и петь песенки, и бить баклуши. Но так им и не удавалось понять, что же происходит на самом деле. Наконец, директор ИКЛ пришёл к директору ЕЖЗ, и они, вместе с директором ВГД и мокренькой кисонькой, поняли, что строка должна быть не просто частью, а, будучи развернутой, должна являться префиксом развернутой строки, хранящейся в устройстве! Более того, она также должна состоять из маленьких латинских букв. После этого все вместе они пошли спать.

Из более достоверных источников (а именно Мадагаскарский Национальный Отряд Профессиональных Супер-агентов - МНОП) стало известно, что тот, у кого окажется данное устройство, получит неограниченную власть над миром. Ваша задача кристально ясна: реализуйте данное устройство.

Формат входных данных

В первой строке входного файла содержится число M — количество выполняемых операций. Далее в M строках содержатся описания операций: либо `+ c` для модернизации, либо `? s` для проверки инновационности. Гарантируется, что устройство сможет обработать входной файл.

Формат выходных данных

Для каждого запроса проверки выведите «YES», если строка является инновационной, и «NO» в противном случае.

Примеры

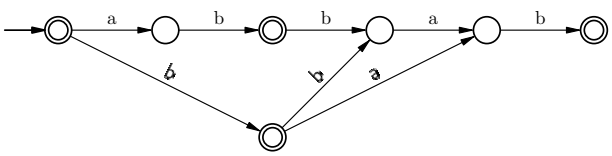
sufpath.in	sufpath.out
2 + a ? a	YES

Задача B. Суффиксный автомат

Имя входного файла:	`suffix.in`
Имя выходного файла:	`suffix.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Суффиксным автоматом для строки w называется детерминированный конечный автомат A, который допускает язык $Suff(w)$ — множество суффиксов слова w. Например, суффиксный автомат для слова *abbab* должен допускать в точности следующие слова: $\{abbab, bbab, bab, ab, b, \varepsilon\}$. Мы также потребуем, чтобы суффиксный автомат не имел недостижимых состояний, и не было состояний, из которых не достижимы допустимые. Других ограничений, например, минимальности, накладывать не будем.

На рисунке показан суффиксный автомат для слова *abbab*.



По заданному *скелету* суффиксному автомату некоторого слова требуется восстановить суффиксный автомат. А именно — вам даны состояния, переходы, начальное состояние и допускающие состояния. Но пометки на ребрах удалены.

Вам следует расставить пометки на ребрах заданного суффиксного автомата, так чтобы он стал суффиксным автоматом некоторого слова w, а также найти это слово. Для простоты будем считать, что размер алфавита ничем не ограничен, вы можете использовать в качестве символов числа от 1 до k (k вы можете выбрать сами).

Формат входных данных

Первая строка входного файла содержит три целых числа: n, m и t — количество состояний, количество переходов, и количество допускающих состояний, соответственно ($2 \leqslant n \leqslant 200, 1 \leqslant m \leqslant 1000$, $1 \leqslant t \leqslant n$). Вторая строка содержит t целых чисел — номера допускающих состояний (состояния пронумерованы с 1, начальное состояние имеет номер 1).

Следующие m строк описывают переходы: каждая строка содержит два целых числа s_i и t_i и описывает переходы из s_i в t_i.

Формат выходных данных

На первой строке выходного файла выведите два целых числа: l и k — длину слова w и размер алфавита. Используйте числа $\{1, \ldots, k\}$ как элементы алфавита. k не должно превышать m.

Вторая строка должна содержать l целых чисел — слово w.

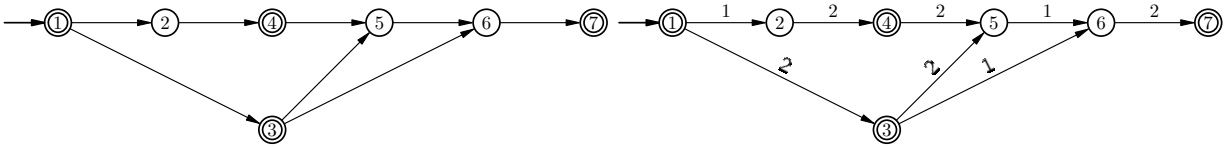
Наконец, третья строка должна содержать m целых чисел — метки на переходах скелета автомата, в том порядке, в котором они описаны во входном файле.

Гарантируется, что ответ всегда существует.

Примеры

suffix.in	suffix.out
7 8 4 1 3 4 7 1 2 1 3 2 4 3 5 3 6 4 5 5 6 6 7	5 2 1 2 2 1 2 1 2 2 2 1 2 1 2

Замечание



Задача C. Суффиксный пулемёт

Имя входного файла:	`suffix2.in`
Имя выходного файла:	`suffix2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Или зачёт, или автомат.

Ганнибал Ректор

Теоретическая подготовка новобранцев армии Поссилтума включала в себя не только занятия по военному праву, но и начала криптографии. Лекции читал майор Мега Байт, не чуждый солдатского юмора. Гвидо и Нунцио, в чьё задание входил развал армии Поссилтума изнутри, решили на этом сыграть, внеся путаницу в терминологию. В начале очередной лекции Нунцио поднял руку и спросил:

— Вот вы на прошлой лекции рассказывали про конечные автоматы. А про конечные пулемёты расскажете?

Мега Байт не растерялся.

— Суффиксный пулемёт — это конечный автомат, принимающий все суффиксы данной строки (от нулевого до L-го включительно, где L — длина строки), и только их. Сержант Гвидо!

— Я, господин майор!

— Вы сможете отличить автомат от пулемёта?

— Так точно, господин майор!

— Вам дан конечный автомат. Требуется проверить, является ли он суффиксным пулемётом данной строки.

К сожалению, написание программ такого типа не входило в обязанности Гвидо и Нунцио как в Синдикате, так и в корпорации М. И. Ф. Так что соответствующую программу придётся писать Вам.

Формат входных данных

Во входном файле задан один или несколько тестовых наборов. В первой строке каждого набора заданы количество состояний автомата N, количество переходов M, а также количество принимающих состояний T ($1 \leqslant T \leqslant N \leqslant 50\,000$, $1 \leqslant M \leqslant 100\,000$). Во второй строке через пробел заданы T различных чисел в пределах от 1 до N — принимающие состояния автомата, в возрастающем порядке. В последующих M строках заданы переходы в виде a_i b_i c_i, где $1 \leqslant a_i, b_i \leqslant n$, а c_i — маленькая буква латинского алфавита. Переход производится из состояния a_i в состояние b_i по букве c_i. Из каждого состояния a_i есть не более одного перехода по символу c_i. Последняя строка описания набора — это строка S, для которой автомат должен являться пулемётом. Она состоит только из маленьких латинских букв, и ее длина лежит в пределах от 1 до 50 000 включительно. Кроме того, сумма всех N и суммарная длина всех строк, для которых необходимо произвести проверку, не превосходит 50 000, а сумма всех M не превосходит 100 000.

Файл заканчивается фиктивным набором, в котором $N = M = T = 0$.

Начальным состоянием автомата является первое. Если при интерпретации какой-то строки в автомате отсутствует соответствующий переход, то автомат вываливается по ошибке и строку не принимает. Таким образом, строка принимается, только если при её интерпретации были найдены все переходы, и по их завершении автомат оказался в принимающем состоянии (при этом неважно, были по пути принимающие состояния, или нет).

Формат выходных данных

Выведите в выходной файл, является ли данный автомат пулемётом, следуя формату примера.

Примеры

suffix2.in	suffix2.out
2 1 2 1 2 1 2 a a 2 2 2 1 2 1 1 a 1 2 b ab 0 0 0	Automaton 1 is a machinegun. Automaton 2 is not a machinegun.

Задача D. Рефрен

Имя входного файла:	`refrain.in`
Имя выходного файла:	`refrain.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Рассмотрим последовательность n целых чисел от 1 до m. Подпоследовательность подряд идущих чисел называется *рефреном*, если произведение ее длины на количество вхождений в последовательность максимально.

По заданной последовательности требуется найти ее рефрен.

Формат входных данных

Первая строка входного файла содержит два целых числа: n и m ($1 \leqslant n \leqslant 150\,000$, $1 \leqslant m \leqslant 10$).

Вторая строка содержит n целых чисел от 1 до m.

Формат выходных данных

Первая строка выходного файла должна содержать произведение длины рефрена на количество ее вхождений. Вторая строка должна содержать длину рефрена. Третья строка должна содержать последовательность которая является рефреном.

Примеры

refrain.in	refrain.out
9 3 1 2 1 2 1 3 1 2 1	9 3 1 2 1

Задача E. Ненокку

Имя входного файла:	`nenokku.in`
Имя выходного файла:	`nenokku.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Очень известный автор не менее известной книги решил написать продолжение своего произведения. Он писал все свои книги на компьютере, подключенном к интернету. Из-за такой неосторожности мальчику Ненокку удалось получить доступ к еще ненаписанной книге. Каждый вечер мальчик залазил на компьютер писателя и записывал на свой компьютер новые записи. Ненокку, записав на свой компьютер очередную главу, заинтересовался, а использовал ли хоть раз писатель слово "книга". Но он не любит читать книги (он лучше полазает в интернете), и поэтому он просит вас узнать есть ли то или иное слово в тексте произведения. Но естественно его интересует не только одно слово, а достаточно много.

Формат входных данных

В каждой строчке входного файла записано одна из двух записей.

1. `? <слово>` (<слово> - это набор не более 50 латинских символов);
2. `A <текст>` (<текст> - это набор не более 10^5 латинских символов).

1 означает просьбу проверить существование подстроки <слово> в произведение.
2 означает добавление в произведение <текст>.
Писатель только начал работать над произведением, поэтому он не мог написать более 10^5 символов. А входной файл содержит не более 15 мегабайт информации.

Формат выходных данных

Выведите на каждую строчку типа 1 "`YES`", если существует подстрока <слово>, и "`NO`" в противном случае. Не следует различать регистр букв.

Примеры

nenokku.in	nenokku.out
? love ? is A Loveis ? love ? WHO A Whoareyou ? is	NO NO YES NO YES