Задача A. Ахо

Имя входного файла:	`aho.in или стандартный ввод`
Имя выходного файла:	`aho.out или стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Это интерактивная задача.

Каждый раз, когда Маргарет и Альфред посещают кафе «У Дональда», они играют в странную игру про угадывание строк.

Её правила таковы:

- Первый игрок пишет строку S фиксированной длины N. Также у первого игрока есть строка T, изначально пустая. Обе строки состоят только из маленьких букв английского алфавита.
- Второй игрок не знает эти строки в течение всей игры. Однако ему разрешено спрашивать про любые две позиции (в обеих строках), правда ли, что символы в них равны. Например, вопрос может выглядеть так "Равны ли второй символ строки S и пятый символ строки T?" Обратите внимание, что можно спрашивать про два символа одной строки.
- Игра состоит из M раундов. В начале каждого раунда первый игрок добавляет один символ в конец строки T.
- После добавления символа второй игрок может задать не более пяти вопросов. После этого он должен сказать, какое число подстрок строки T равно строки S.

Маргарет быстро заметила, что Альфред всегда преуспевает в роли втрого игрока. Она подозревает наличие стратегии, позволяющей второму игроку выигрывать независимо от S и T. А вы так сможете?

Формат входных данных

При запуске ваша программа должна считывать два целых числа N и M ($1 \leqslant N, M \leqslant 20\,000$) из стандартного потока ввода.

Далее следуют M раундов игры. В i-м раунде вы можете задать **не более пяти** вопросов в формате "`<позиция1> <позиция2>`". Описание любой позиции выглядит как "`s` x" где $1 \leqslant x \leqslant N$ (если это x-й символ строки S) или как "`t` y" где $1 \leqslant y \leqslant i$ (если это y-й символ строки T). Ответ программы жюри будет "`Yes`", если символы на этих позициях равны и "`No`" иначе.

Формат выходных данных

В конце любого раунда вы должны вывести ответ в формате "`$` k", где k равно числу вхождений строки S в строки T. После этого в строку T будет автоматически добавлен новый символ (если это не последний раунд).

Не забудьте делать `flush` после каждого вопроса. После того, как вы вывели все m чисел, ваша программа должна автоматически завершиться, иначе ваш вердикт может быть каким угодно.

Примеры

aho.in или стандартный ввод	aho.out или стандартный вывод
3 7 No Yes No Yes Yes Yes No No Yes Yes Yes	s 1 s 2 s 1 s 3 s 2 t 1 s 1 t 1 $ 0 s 2 t 2 $ 0 s 3 t 3 $ 1 s 2 t 4 s 1 t 4 $ 1 s 1 t 5 $ 1 s 2 t 6 $ 1 s 3 t 7 $ 2

Замечание

В примере, строка S изначально равна “aba”, а строка T получается добавлением символов “a”, “b”, “a”, “c”, “a”, “b”, “a”.

Задача B. В поисках мусóрки

Имя входного файла:	`bin.in или стандартный ввод`
Имя выходного файла:	`bin.out или стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Это интерактивная задача.

Робот R8D16 накопил за день довольно много мусора и теперь хочет его выкинуть. К сожалению, он совсем забыл, где находится мусóрка, и теперь хочет это узнать.

Будем считать, что робот находится на плоскости с введенной на ней прямоугольной системой координат. Мусóрка расположена в точке с целыми координатами (x_0, y_0). R8D16 точно знает, что корзина расположена не слишком далеко от центра координат, а именно, выполняется неравенство $-10^9 \leqslant x_0, y_0 \leqslant 10^9$.

Робот может посылать запросы на специальный сервер, отслеживающий положение всех объектов на плоскости. В качестве запроса R8D16 может послать на сервер целочисленные координаты точки на плоскости, а сервер в ответ отправит роботу *манхэттенское* расстояние от этой точки до точки, в которой расположена мусóрка. Манхэттенским расстоянием между двумя точками на плоскости называется сумма абсолютных значений разностей координат этих точек по осям x и y. Иными словами, если R8D16 передает на сервер координаты точки (x, y), в ответ он получит значение величины $|x - x_0| + |y - y_0|$. В силу особенностей работы сервера, для всех запросов должно выполняться неравенство $-10^9 \leqslant x, y \leqslant 10^9$.

R8D16 хочет как можно скорее узнать, где же располагается мусóрка. Помогите ему сделать это, потратив не более трёх запросов!

Протокол взаимодействия

После начала работы программа должна вывести несколько запросов к серверу. Чтобы послать на сервер запрос, содержащий точку (x, y), программа должна вывести строку «`? x y`». После этого ваша программа должна считать ответ на запрос — манхэттенское расстояние до мусóрки от точки, указанной в запросе.

Когда ваша программа определит положение мусóрки, она должна вывести строку «`!` x_0 y_0» — координаты точки, в которой располагается корзина. После этого ваша программа должна немедленно завершиться.

Примеры

`bin.in или стандартный ввод`	`bin.out или стандартный вывод`

5
 	
3
 	
2
	`? 0 0`
	
	`? 1 1`
	
	`? 3 2`
	
	`! 2 3`

Замечание

Вывод **каждой строки** должен завершаться переводом строки и сбросом буфера потока вывода. Для этого используйте `flush(output)` на языке Pascal, `fflush(stdout)` в C/C++ или `cout.flush()` в C++, `sys.stdout.flush()` на языке Python, `System.out.flush()` на языке Java.

Задача C. Ялта

Имя входного файла:	`teleports.in или стандартный ввод`
Имя выходного файла:	`teleports.out или стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Это интерактивная задача.

Вы думаете, товарищ Лиходеев отправился в Ялту в мгновение ока? Для внешнего наблюдателя, возможно, и так. Для самого же Степана Богдановича путешествие было натуральным безумием.

Стартовал он в Москве, но чувствовал себя не человеком на паре ног, а точкой на карте. И было доподлинно известно, что попасть ему следовало в пресловутую Ялту. Находясь в очередном городе, он знал, в какие города можно было отправиться оттуда. Набор городов, куда можно отправиться из города x, всегда был одним и тем же.

Впрочем, пробуя отправиться в некоторые города, он попадал в совершенно другие: некоторые города оказались *телепортами*. Пытаясь отправиться в город-телепорт, Степан Богданович попадал в произвольный город-телепорт: иногда в тот, куда отправлялся, а иногда в другой. Отправляясь в один и тот же город-телепорт, он мог в разных случаях попадать в разные города-телепорты. Москва и Ялта не являлись телепортами.

Совсем заблудившись, Степан Богданович может из любого города усилием воли переместиться в Москву, даже если переход в нее из текущего города явно разрешен не был.

В этой задаче вам нужно будет за Степана Богдановича принимать решения, в какой город отправиться в каждый момент времени, чтобы в итоге оказаться в Ялте.

Формат входных данных

Это интерактивная задача.

Вам предлагается реализовать программу, взаимодействующую с проверяющей программой следующим образом:

- При запуске ваша программа получит на вход количество городов n ($1 \leqslant n \leqslant 50$). Города пронумерованы целыми числами от 0 по $(n-1)$. Москва имеет номер 0, Ялта — номер $(n-1)$.
- Далее проверяющая программа отправляет на стандартный ввод вашей программы описание текущего города. Обратите внимание, что город, в который вы попали, может отличаться от того, в который собирались отправиться, если вы отправлялись в город-телепорт.

  Описание города начинается с его номера id и количества k городов, в которые можно отправиться. После этого в той же строке записаны в произвольном порядке k различных чисел, не совпадающих с id — номера тех городов, в которые можно попасть. Если вы оказались в данном городе не в первый раз, вместо k и набора городов-соседей выводится единственное число -1.
- После этого ваша программа должна выдать номер города, в который нужно отправиться. Если вы попали в Ялту, ваша программа должна корректно завершиться.

Ваша программа может сделать не более 10 000 перемещений между городами.

В этой задаче все тесты таковы, что для проверяющей программы не существует стратегии выбора городов при телепортации, позволяющей не допустить появления Степана Богдановича в Ялте.

Формат выходных данных

Это интерактивная задача.

После получения описания очередного города выведите одно число на отдельной строке — номер города, в который вы собираетесь отправиться.

После вывода очередного хода необходимо сбросить буфер вывода: `fflush(stdout)` или `cout.flush()` в `C/C++`, `flush(output)` в `Pascal`, `sys.stdout.flush()` в `Python`, `System.out.flush()` в `Java`.

Примеры

teleports.in или стандартный ввод	teleports.out или стандартный вывод
4 0 1 1 (ожидание ответа) 1 2 2 3 (ожидание ответа) 2 1 3 (ожидание ответа) 0 -1 (ожидание ответа) 1 -1 (ожидание ответа) 3 0	(ожидание ввода) 1 (ожидание ввода) 2 (ожидание ввода) 0 (ожидание ввода) 1 (ожидание ввода) 3
5 0 3 2 1 3 (ожидание ответа) 2 0 (ожидание ответа) 0 -1 (ожидание ответа) 1 0 (ожидание ответа) 0 -1 (ожидание ответа) 3 1 4 (ожидание ответа) 4 0	(ожидание ввода) 1 (ожидание ввода) 0 (ожидание ввода) 2 (ожидание ввода) 0 (ожидание ввода) 3 (ожидание ввода) 4

Замечание

В первом примере С. Б. последовательно посещает города $0, 1, 2, 0, 1, 3$. Заметим, что он перемещается из города 2 в Москву, несмотря на то, что Москва не перечислена в списке городов, достижимых из второго.

Во втором примере С. Б. пытается отправиться в город 1, но телепортируется в город 2. Это ему не нравится, и он возвращается в Москву, чтобы попробовать отправиться в город 2 напрямую. В результате его телепортирует в город 1, откуда он снова возвращается в Москву. После этого он успешно перемещается в город 3, а затем в Ялту.

Задача D. Демид Сергеевич Кучеренко

Имя входного файла:	`drinking.in` или стандартный ввод
Имя выходного файла:	`drinking.out` или стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

У Демида Сергеевича Кучеренко есть n упаковок кефирчика. Как бы это ни было неправдоподобно, ровно одна из упаковок содержит испортившийся кефирчик, а все остальные упаковки в ближайшие d дней не испортятся.

Но задача не была бы такой непедагогичной, если бы в его домике не было k школьников, которых он может использовать для проверки упаковок. На протяжении d дней он наливает школьникам кефирчик утром, после чего в течение дня наблюдает, кто из них отправился в медпункт. Отправившиеся в медпункт школьники проведут там заметно больше d дней и не смогут быть использованы в экспериментах всех последующих дней. Кефирчики пронумерованы числами от 1 до n, школьники пронумерованы числами от 1 до k.

Помогите Демиду Сергеевичу за d дней гарантированно определить плохую упаковку.

Протокол взаимодействия

Это интерактивная задача.

В начале работы вашей программе в поток стандартного ввода подаётся три числа n, k и d — количество упаковок с кефирчиком, школьников и дней, соответственно. Затем ваша программа может отправлять запросы программе жюри.

Если ваш текущий запрос — это проверка, кто из школьников отправится в медпункт, выведите в стандартный поток вывода слово «`drink`» (без кавычек). Затем в следующих k строках выведите описание того, кефирчик из каких упаковок будет пить каждый школьник. В i-й из этих строк выведите количество кефирчиков c_i, которые Демид Сергеевич любезно предложит i-му школьнику, а затем c_i различных чисел через пробел — номера кефирчиков.

В ответ на этот запрос ваша программа получит в стандартный поток число — сколько школьников пошли в медпункт сегодня, а затем номера этих школьников через пробел. Если школьник в какой-то из дней ранее отправился в медпункт, он больше не пьет кефирчик от Демида; для таких школьников выводите строку с единственным числом 0.

Для того, чтобы сообщить ответ на задачу, выведите слово «`answer`», а на следующей строке номер упаковки с испортившимся кефирчиком. После этого запроса ваша программа должна завершиться.

Вам разрешается сделать не более d запросов «`drink`». Гарантируется, что в данных ограничениях задача разрешима.

Если ваша программа сделает больше, чем d запросов «`drink`», либо сделает запрос в некорректном формате, последующее общение с программой жюри согласно протоколу будет немедленно прервано; с точки зрения вашей программы это будет выглядеть как конец файла (EOF). В этой ситуации ваша программа должна завершиться, в противной ситуации ваше решение может получить неопределенный вердикт вместо ожидаемого `Wrong Answer`.

После каждого запроса, сделанного вашей программой, вызовите функцию сброса буфера вывода:

- `fflush(stdout)` в C или C++
- `System.out.flush()` в Java
- `flush(output)` в Pascal
- `sys.stdout.flush()` в Python

МЯГКАЯ версия: $1 \leqslant k \leqslant 10$, $1 \leqslant n \leqslant 10$; $d = 10$.
ЖЁСТКАЯ версия: $1 \leqslant k \leqslant 10$, $1 \leqslant n \leqslant 239$; $d \leqslant 10$.

Примеры

drinking.in или стандартный ввод	drinking.out или стандартный вывод
5 3 2 0 1 1	 drink 2 1 2 2 1 2 1 3 drink 1 4 0 1 5 answer 4

Замечание

Тест, описанный в условии, может не совпадать с первым тестом в системе.