Задача A. Непрерывный алкоголик

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

> Водка (от «вести») — управляющая конструкция, эмулирующая ГСЧ
>
> Russian-Geek dictionary

Однажды Великий Скив стоял у окна штаб-квартиры корпорации «МИФ» и задумчиво смотрел на привычный для всех обитателей Базара-на-Деве переполох.

Внимание его привлёк некий странный персонаж, который явно что-то искал, и ещё более явно был нетрезв. Приняв его начальные координаты за точку $(0, 0)$, Скив решил пронаблюдать, что будет дальше. Алкоголик привычно занялся непрерывным случайным блужданием, то есть в каждую бесконечно малую долю времени (скажем, 10^{-100} секунды) он смещался на бесконечно малое расстояние (скажем, 10^{-100}), выбирая каждый раз одно из четырёх направлений (вперёд, назад, вправо или влево) равновероятно.

При этом Скив заметил, что вокруг алкоголика на единичной окружности с центром в точке $(0, 0)$ непрерывным образом расположена водка. Количество водки в точке определяется некоторым многочленом от координат точки. Как только алкоголик достиг единичной окружности в некоторой точке, он выпил всю водку, находящуюся в этой точке (обратите внимание, что количество выпитой водки может быть и отрицательным), и заснул.

Удивлённый необычной картиной, Скив рассказал об увиденном Аазу. «Понятно. А ещё там рядом с окружностью обычно стоит незаметный человечек и принимает ставки на количество выпитой алкоголиком водки. Спасибо за информацию, я думаю, что в следующем раунде я его обыграю».

Для того, чтобы выиграть пари, Аазу требуется определить математическое ожидание выпитой алкоголиком водки.

Формат входного файла

Входной файл содержит не более 1000 тестовых примеров. Каждый пример содержится целиком на одной строке и задаёт многочлен от координат, не более 10 степени, состоящий не более чем из 50 одночленов.

Каждый одночлен описывается непустой строкой в формате $[c][\mathtt{x}[\mathtt{\hat{}}d]][\mathtt{y}[\mathtt{\hat{}}d]]$, где коэффициенты и показатели степеней, равные по модулю единице, могут быть опущены. В произвольные места, кроме как внутри числа, могут вставляться пробельные символы. Одночлены соединяются знаком '+' или '-', в зависимости от коэффициента последующего одночлена. Перед первым одночленом также может стоять знак '-'. Все коэффициенты целые и не превосходят по модулю 1000, все показатели степеней целые и неотрицательные.

Входной файл завершается строкой `FINISH.`

Формат выходного файла

Следуйте формату вывода максимально точно. Ваш результат будет проверен автоматически на нахождение в границах абсолютной или относительной погрешности 10^{-6}.

Пример

alcohol.in

`x ^ 2 + y ^ 2`
`2x-y^3+1`
`FINISH.`

alcohol.out
`Case 1: 1.00000000000 vodka expected` `Case 2: 1.00000000000 vodka expected`

Задача B. Бароны

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

Есть на проходе позволяют себе только пешки

Из «Правил хорошего тона для офицеров, епископов и слонов»

Через полчаса Ааз вернулся в сопровождении гостя. Подпольный букмекер сильно проигрался. К его чести, он сразу же отдал проигрыш, более того, узнав Ааза, он спросил, можно ли получить аудиенцию у Великого Скива. Почувствовав, что дело может обернуться прибылью, Ааз согласился.

Оказывается, в своём измерении подпольный букмекер занимался приёмом ставок на различные соревнования. Но в последнее время случился кризис. Появление новых хитрых устройств привело к тому, что жульничать стало легче, а честно соревноваться — сложнее. Ничейная смерть уже постигла очень многие игры, а в случае, когда всё же кто-то побеждал, общественное мнение сходилось на том, что это неспроста, и наверное, что-то тут нечисто. Организаторы соревнований тратили большую часть своего времени на построение систем отлова жуликов, в связи с чем интерес упал ещё ниже, и случился неизбежный в такой ситуации раскол.

— . . . сейчас происходящее определяется множеством сообществ, возглавляемых баронами, которые пытаются пожрать друг друга. Соревнования ушли в тень. И на что нам принимать ставки?

— Например, на результат взаимопожирания баронов. А точнее — допустим, что N баронов едят друг друга. Если барон A съедает барона B, то вес барона A увеличивается на число баронов, съеденных лично бароном B. И ставки делаются на итоговые веса баронов в некий момент времени, — Ааз с ходу придумал правила.

Для того, чтобы определить величину ставки, букмекеру нужно определить, могли ли из заданных начальных весов баронов получиться заданные конечные, и если да, то как.

Формат входного файла

Во входном файле заданы один или несколько тестов. В первой строке каждого теста указано начальное число баронов N ($1 \le N \le 100\,000$), во второй — начальные целые веса баронов ($1 \le a_i \le 10^9$), в третьей — конечное число баронов M ($1 \le M \le N$), в четвёртой — конечные целые веса баронов в некотором произвольном порядке ($1 \le b_j \le 10^9$). Сумма всех N не превосходит 100 000. Файл завершается нулём.

Формат выходного файла

Если в некотором тесте решение существует, выведите после заголовка теста в $N - M$ строках последовательность поедания баронов в хронологическом порядке. Если барон был съеден, его нельзя выводить повторно. Бароны пронумерованы от 1 до N. Разделяйте ответы на тесты пустой строкой.

Следуйте формату вывода максимально точно, проверка производится автоматически!

Пример

barons.in	barons.out
3 2 3 9 2 1 7 7 2 3 9 1 7 1 7 2 9 3 0	Case 1: IMPOSSIBLE Case 2: 5 eats 2 5 eats 7 4 eats 5 1 eats 4 1 eats 6

Задача C. Бумеранги

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

Новейшая граната «Банзай-1» была выполнена в форме бумеранга.

Из книги «Воспоминания старого камикадзе»

— Кроме того, вы легко можете придумывать новые виды спорта. Пока участники и зрители осознают правила, а жулики ищут дырки в этих правилах, вы сможете сделать неплохие деньги, — вдохновившись, продолжал Ааз.

— Легко сказать... Вот скоро будет чемпионат по лёгкой атлетике. Допинг-тесты, допинг-чекеры, допинг-тестеры, допинг-сливы... И как тут придумать такой вид спорта, чтобы и смотрелся традиционно, и допинг там не помогал? За одну идею наша ассоциация букмекеров готова заплатить немалую сумму.

— Пожалуйста. Фигурное метание бумерангов. Правила просты: двое бросают бумеранги. Каждый бумеранг — это дуга некоторой фиксированной окружности, и он летит по этой окружности с постоянной скоростью до бесконечности (тот, кто бросил, уходит). Высший пилотаж — это пустить бумеранги так, чтобы они не столкнулись, — тут же откликнулся Ааз.

Для разработки системы тренировки атлетов Вам по заданным параметрам бумерангов требуется определить, столкнутся ли бумеранги, и если да, то когда. Бумеранги замкнутые.

Формат входного файла

Во входном файле заданы не более 10 000 тестов. Каждый тест состоит из двух бумерангов. Каждый бумеранг представлен отдельной строкой вида $x\ y\ r\ \alpha\ \beta\ o\ d$. Здесь x и y — координаты центра окружности, содержащей бумеранг ($|x|, |y| \le 100$), r — её радиус ($0 < r \le 100$), α и β — углы в градусах против часовой стрелки от оси OX, задающие начальное положение концов бумеранга на окружности ($0 \le \alpha, \beta < 360, \alpha \ne \beta$), o — ориентация дуги ('+' или '-'), а d — угловая скорость дуги в градусах в секунду ($|d| \le 360$). Ориентация '+' означает, что бумеранг — это дуга окружности от угла α до угла β против часовой стрелки, а ориентация '-' — по часовой стрелке. Знак числа d имеет аналогичный ориентации смысл — направление движения против часовой стрелки или по часовой стрелке. Файл завершается строкой из семи нулей. Все числа во входном файле целые. Изначально бумеранги не имеют общих точек.

Формат выходного файла

Для каждого из тестов выведите одну строку, содержащую ответ. Следуйте формату примера максимально точно. Ваш результат будет проверен автоматически на нахождение в границах абсолютной или относительной погрешности 10^{-6}.

Пример

boomerangs.in
0 0 1 0 90 + 10 1 0 1 0 90 + -13 0 0 1 0 90 + 10 3 0 1 0 90 + -13 0 0 0 0 0 0 0
boomerangs.out
Case 1: Collision after 64.61538461538461 seconds Case 2: They will fly infinitely! Great!

Задача D. Искусственные мозги

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

Чаще появляйтесь в обществе блондинок

Советы диетолога для Ктулху с избыточным весом

— Но главный вопрос — что делать с шахматами? Техника стала играть настолько сильно, что соревнования живых игроков превратились в выставку секретных средств связи. А соревнования машин никому не интересны — какая разница, какой из двух железных ящиков выиграет?

— А вы делайте думающие машины похожими на живых игроков. Или вырастите, например, искусственные мозги, и пересадите их живым игрокам, — Ааз не так давно договорился с производителями искусственных мозгов в измерении Тлейлакс о продвижении их продукции, и решил не упускать свой шанс.

Довольно быстро дело дошло до сделки, и остался вопрос по поводу объёмов поставок. На заводе есть N заготовок искусственных мозгов, каждая из которых представляет собой шар радиуса r_2 с вырезанным внутри шаром радиуса $r_1 < r_2$. Правильный мозг состоит из двух заготовок, которые вкладываются друг в друга (то есть r_2 одной строго меньше r_1 другой). Сколько правильных мозгов можно собрать одновременно из этих заготовок?

Формат входного файла

Во входном файле содержатся не более 100 тестов. Каждый тест состоит из строки, содержащей число N ($1 \le N \le 20$), за которой следуют N строк, содержащих описания мозгов ($1 \le r_{i,1} < r_{i,2} \le 10^9$), все числа целые. Входной файл завершается нулём.

Формат выходного файла

Следуйте формату примера максимально точно, проверка производится автоматически. Пустая строка между ответами на последовательные тесты не требуется. Слово "`brains`" необходимо всегда выводить во множественном числе.

Пример

brains.in	brains.out
6 1 2 3 4 5 6 7 8 1 9 5 10 0	Case 1: We can assemble 2 brains.

Задача E. Кратчайшие расстояния

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

Все дороги ведут в Рим

Варварская народная мудрость

Скив, прислушивавшийся к разговору Ааза с букмекером, неожиданно вмешался.

— Мне кажется, что нам лучше прибыть на место и разобраться с ситуацией прямо там.

— Я не уверен, что это так уж необходимо. Как я вижу, практически все проблемы заказчика можно решить прямо здесь, — Ааз то ли не хотел перебираться в очередное измерение, то ли просто набивал цену.

— Великий Скив, как всегда, прав. Возможно, вы сумеете прекратить всю эту неразбериху. Мы готовы оплатить Ваши транспортные расходы, а также каждый день Вашего пребывания в нашем измерении, — тут же откликнулся заказчик. — Надеюсь, такие опытные маги, как Великий Скив и Ааз, легко доберутся до нашего измерения сами, а далее Вам помогут сделанные мной записи.

Записи оказались матрицей кратчайших расстояний вдоль существующих дорог между городами. Схема дорог в этом измерении представляла собой некоторый неориентированный граф, вершинами которого были города, а рёбра имели неотрицательный вес.

Требуется восстановить карту — то есть найти граф с минимальным числом рёбер, который мог бы порождать такую матрицу. Гарантируется, что хотя бы один такой граф существует.

Формат входного файла

Во входном файле даны один или несколько тестов. В первой строке каждого теста записано число N ($1 \le N \le 100$). Далее следуют N строк по N чисел — кратчайшие расстояния $a_{(i,j)}$ — неотрицательные целые числа, не превосходящие 10 000. Всегда выполнено $a_{(i,j)} = a_{(j,i)}$, $a_{(i,i)} = 0$. Файл завершается нулём. Сумма всех значений N по всем тестам не превосходит 1000.

Формат выходного файла

Следуйте формату примера максимально точно — проверка производится автоматически.

Пример

`distances.in`

```
3
0 1 1
1 0 1
1 1 0
3
0 1 2
1 0 1
2 1 0
0
```

`distances.out`

```
Graph 1: The minimal number of edges is 3:
1<->2
3<->2
3<->1

Graph 2: The minimal number of edges is 2:
1<->2
2<->3
```

Задача F. Игры на графе

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

Противник начал e2–e4. Я проанализировал его архитектуру и сдался

Из мемуаров 20-го чемпиона мира Фрица Рыбкина

Прибыв на место, Ааз тут же потребовал организовать совещание букмекеров, на котором он изложит свой план.

— Главная задача, — начал Ааз своё выступление перед букмекерами, — научиться использовать достижения прогресса. Мы планируем запуск множества новых видов соревнований, что — вполне возможно — приведёт к тому, что появятся какие-то игры по правилам, придуманным не нами. А значит, необходимо уметь быстро выяснять, насколько эти правила могут быть нам полезны.

— А можно ли хотя бы в общем пояснить, как это будет делаться? — последовал вопрос из зала.

— Вот пример задачи, решив которую, мы сможем разобраться с целым классом игр. Дан ориентированный граф некоторой игры для двух игроков и начальная позиция в ней. Напомним, что в игре на графе игрок имеет право походить из позиции в любую позицию, в которую есть ребро из текущей. Игроки ходят по очереди; проигрывает тот, кто не может сделать ход. Требуется проверить, верно ли, что при любой игре сторон всегда выигрывает первый игрок.

Формат входного файла

Во входном файле содержится описание одного или нескольких тестов. В первой строке каждого теста заданы число вершин V и число рёбер E ($1 \le V \le 100\,000$, $1 \le E \le 100\,000$), а также номер начальной позиции a ($1 \le a \le V$). Далее следуют E строк — описания рёбер в формате u_i v_i ($1 \le u_i, v_i \le V$), что означает наличие ребра, направленного из вершины u_i в вершину v_i. Файл завершается тремя нулями. Сумма всех E по всем тестам не превосходит $100\,000$, количество тестов в файле не превосходит 1000.

Формат выходного файла

Следуйте формату примера максимально точно — проверка производится автоматически.

Пример

gg.in
3 2 1 1 2 1 3 1 1 1 1 1 0 0 0

gg.out
First player always wins in game 1. Players can avoid first player winning in game 2.

Задача G. Число

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

В математике «u с волной» встречается довольно редко

Частотный словарь математических терминов

После заседания Ааза и Скиву предложили окончательно договориться об оплате их услуг. Переговоры шли по принятым в этом измерении правилам:

Изначально на доске написано число x. Сторонам разрешается каждую секунду либо вычитать из него 1, либо умножать на 2.

Неожиданно Скив заметил, что Ааз отвлёкся от переговоров, азартно обсуждая что-то с одним из букмекеров. Оказывается, он быстро успел заключить пари, возможно ли в процессе переговоров увидеть на доске число y, и насколько быстро.

Вам требуется выяснить, за какое наименьшее число секунд можно получить число y.

Формат входного файла

Во входном файле содержатся не более 50 000 тестов, каждый описывается одной строкой из двух целых чисел, не превосходящих 10^{18} по абсолютной величине — x и y. Гарантируется, что $|x| + |y| > 0$. Входной файл завершается двумя нулями.

Формат выходного файла

Следуйте формату примера максимально точно — проверка производится автоматически.

Пример

number.in	number.out
2 5 0 1 -2 -3 0 0	Case 1: 4 seconds required Case 2: IMPOSSIBLE Case 3: 1 seconds required

Задача N. Псевдострока

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

Тогда я взял себе псевдо-«Ним»...

из воспоминаний игрока в псевдоигры

— И всё же я бы помог им разобраться собственно с проблемами жульничества, — Скив был немного недоволен тем, как Ааз вёл дело. — По сути, ты просто подменяешь проблему, выигрывая у жуликов немного времени. А надо хотя бы где-то их победить.

В некоторых, довольно популярных среди интеллектуалов, играх ключевая информация передаётся в виде текста, и всегда имеется вероятность, что текст окажется не только у жюри, но и у отдельных участников ещё до соревнований. И тогда турнир неизбежно превращается в псевдотурнир по параинтеллектуальным псевдоиграм...

— Возможно, в таком случае здесь помогут псевдостроки, — заметил Ааз.

— А что это?

— *Псевдострокой* (или *квантовой строкой*) называется строка, у которой на каждой позиции стоит некоторое непустое множество символов. *Реализацией* псевдостроки называется строка, получающаяся выбором какого-то одного элемента из каждого множества. Мы отправляем задания в виде псевдострок, а потом в последний момент сообщаем, какую реализацию выбрать. Даже если произойдёт перехват, шансов на то, что будут выбраны правильные реализации, практически нет.

— То есть это не так просто?

— Да. Попробуй, например, по заданным псевдостроке и обычной строке найти реализацию псевдостроки, в которую данная строка входит максимально возможное число раз как подстрока.

Скив задумался, и понял, что Ааз прав: задача не самая простая. Тем не менее, попробуйте всё же с ней справиться. Помните, что вхождения могут перекрываться (см. пример).

Формат входного файла

Во входном файле даны один или несколько тестов. В первой строке каждого теста дано число N ($1 \le N \le 10\,000$). Далее следуют N строк, каждая из которых описывает соответствующую позицию псевдостроки и содержит одну или несколько различных строчных латинских букв — те буквы, которые могут оказаться на данной позиции в реализации псевдостроки. В последней строке теста задана непустая строка длины не более 1000, состоящая из строчных латинских букв — та строка, количество вхождений которой надо максимизировать. Файл завершается одним нулём. Сумма N по всем тестам файла не превосходит $10\,000$, всего в файле не более 1000 тестов.

Формат выходного файла

Для каждого теста выведите какую-либо оптимальную реализацию, как показано в примере.

Пример

pseudostring.in	pseudostring.out
3 ab ba ac aa 2 agcb gbc cd 0	String 1: aaa String 2: gg

Задача I. Быстрая сортировка

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

qsort: все остальные методы сортировки

Словарь компьютерных терминов планеты Плюк

— А соревнования по программированию у вас тоже проходят? — спросил Скив, вспомнив прошлогодний опыт участия в аналогичном турнире.

— Да, но к сожалению, жуликов приходится ловить и там. Команды умудряются прятать куски решений, написанных на пробном туре, в самые разные места. Одна команда даже спрятала qsort в собственный пароль. Естественно, что это ей не помогло, но сам факт... И как отлавливать использование найденных процедур, непонятно — всё же не хотелось бы дисквалифицировать без повода...

— Нужно использовать «меченые» тесты, — Ааз предложил знакомую идею.

— Например, мы нашли вот такой qsort.

```
procedure qsort (l, r : integer);
var i, j, x, y : integer;
begin
  if l >= r then exit;
  i := l; j := r; x := a[(l + r) div 2];
  repeat
    while a[i] < x do inc (i);
    while x < a[j] do dec (j);
    if i <= j then begin
      y := a[i]; a[i] := a[j]; a[j] := y;
      inc (i); dec (j)
    end
  until i > j;
  qsort (l, j);
  qsort (i, r)
end;
```

```
void qsort (int l, int r) {
  int i, j, x, y;

  if (l >= r) return;
  i = l; j = r; x = a[(l + r) / 2];
  do {
    while (a[i] < x) ++i;
    while (x < a[j]) --j;
    if (i <= j) {
      y = a[i]; a[i] = a[j]; a[j] = y;
      i++; j--;
    }
  } while (i <= j);
  qsort (l, j);
  qsort (i, r);
}
```

В качестве метки мы используем такую перестановку из n элементов, на которой эта процедура qsort запустится заданное количество раз. После этого уже можно считать читерство доказанным и дисквалифицировать.

Требуется построить перестановку с указанными свойствами или выяснить, что это невозможно.

Формат входного файла

Во входном файле заданы один или несколько тестов. Каждый тест состоит из одной строки, содержащей два целых числа n и k ($1 \le n \le 50\,000$, $1 \le k \le 200\,000$). Сумма всех n по всем тестам не превосходит 50 000. Файл завершается двумя нулями.

Формат выходного файла

Следуйте формату примера максимально точно — проверка производится автоматически!

Пример

qsort.in	qsort.out
2 3 3 5 4 10 0 0	Case 1: 1 2 Case 2: 2 1 3 Case 3: IMPOSSIBLE

Задача J. Своппер

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

Современные компьютеры зацикливаются в десятки раз эффективнее человека

Рекламный проспект OS Vista-N

Перед возвращением в штаб-квартиру корпорации Аазу и Скиву пришлось заполнить на местной таможне декларацию о доходах за время визита. Получилась довольно внушительная последовательность чисел. Обработка этой последовательности заняла весьма долгое время.

— Своппер кривой, — со знанием дела сказал таможенник.

— А что такое своппер? — спросил любопытный Скив.

Ааз объяснил, что своппер — это структура данных, которая умеет делать следующее.

- Взять отрезок чётной длины от x до y и поменять местами число x с $x+1$, $x+2$ с $x+3$, и т.д.
- Посчитать сумму чисел на произвольном отрезке от a до b.

Учитывая, что обсчёт может затянуться надолго, корпорация «МИФ» попросила Вас решить проблему со своппером и промоделировать ЭТО эффективно.

Формат входного файла

Во входном файле заданы один или несколько тестов. В первой строке каждого теста записаны число N — длина последовательности и число M — число операций ($1 \le N,\ M \le 100\,000$). Во второй строке теста содержится N целых чисел, не превосходящих 10^6 по модулю — сама последовательность. Далее следуют M строк — запросы в формате `1` x_i y_i — запрос первого типа, и `2` a_i b_i — запрос второго типа. Сумма всех N и M по всему файлу не превосходит 200 000. Файл завершается строкой из двух нулей. Гарантируется, что $x_i < y_i$, а $a_i \le b_i$.

Формат выходного файла

Для каждого теста выведите ответы на запросы второго типа, как показано в примере. Разделяйте ответы на тесты пустой строкой.

Пример

swapper.in	swapper.out
5 5 1 2 3 4 5 1 2 5 2 2 4 1 1 4 2 1 3 2 4 4 0 0	Swapper 1: 10 9 2