Задача A. Красивые числа

Имя входного файла:	`beauty.in`
Имя выходного файла:	`beauty.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Будем называть число *красивым*, если сумма его цифр в десятичной системе счисления делится на количество цифр в нем (в десятичной системе счисления).

Необходимо найти n-ое в порядке возрастания красивое число.

Формат входного файла

Входной файл содержит целое число n ($1 \le n \le 100000$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

beauty.in	beauty.out
1	1
15	20

Задача B. Разбиение на разноцветные циклы

Имя входного файла:	`colorful.in`
Имя выходного файла:	`colorful.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Рассмотрим неориентированный двудольный граф G с n вершинами в каждой доле. Пусть каждое ребро графа раскрашено в один из k цветов.

Цикл в G называется разноцветным, если никакие два соседних ребра в нем не раскрашены в один и тот же цвет. Множество циклов $\{C_1, C_2, \ldots, C_l\}$ называется *разбиением графа G на разноцветные циклы*, если все циклы в множестве — разноцветные, и каждая вершина графа принадлежит ровно одному циклу из этого множества.

По заданному графу G найдите его разбиение на разноцветные циклы, либо выясните, что искомого разбиения не существует.

Формат входного файла

Первая строка входного файла содержит три целых числа: n, m и k — количество вершин в каждой доле графа, количество ребер в графе, и количество цветов ($2 \le n \le 50$, $2 \le k \le 5$). Пусть вершины первой доли пронумерованы от 1 до n, а вершины второй доли — от -1 до $-n$.

Следующие m строк содержат по три целых числа и описывают ребра. Каждое ребро описывается номером вершины в первой доле, номером вершины во второй доле, и цветом. В графе нет параллельных ребер.

Формат выходного файла

Если разбиение графа на разноцветные циклы существует, выведите “`YES`” на первой строке выходного файла. Вторая строка должна содержать l — количество циклов в разбиении. Далее должно сделать l строк, каждая из которых описывает один цикл. Описание каждого цикла состоит из числа a — количества вершин в цикле, после которого следует a чисел — номера вершин цикла в порядке его некоторого обхода.

Если разбиения графа на разноцветные циклы не существует, выведите “`NO`” на первой строке выходного файла.

Пример

colorful.in	colorful.out

`3 7 3`
`1 -1 1`
`1 -2 2`
`2 -1 3`
`2 -2 1`
`2 -3 2`
`3 -2 3`
`3 -3 1`
	`YES`
	`1`
	`6 1 -1 2 -3 3 -2`

Задача C. Числа Фибоначчи

Имя входного файла:	`fib.in`
Имя выходного файла:	`fib.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Числа Фибоначчи — это последовательность целых чисел, заданная рекуррентным соотношением: $F_0 = 0$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$, $n \geq 2$.

Ваша задача — найти наибольший общий делитель двух чисел Фибоначчи.

Формат входного файла

Во входном файле два числа i и j ($1 \leq i, j \leq 10^6$) — номера чисел Фибоначчи.

Формат выходного файла

В выходной файл выведите остаток от деления наибольшего общего делителя чисел F_i и F_j на 10^9.

Примеры

fib.in	fib.out
5 10	5
2 4	1

Задача D. Развлечения с измерителем — 2

Имя входного файла:	`fun2.in`
Имя выходного файла:	`fun2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дима обнаружил у папы на столе специальный чертежный прибор, похожий на циркуль — измеритель. Измеритель отличается от обычного циркуля тем, что в обеих его ножках находятся иголки (у обычного циркуля в одной ножке находится иголка, а в другой — грифель).

Кроме измерителя Дима нашел на столе клетчатый лист бумаги, в углах некоторых клеток которого были нарисованы точки. Так как измеритель служит для измерения расстояний, то Дима решил измерить все попарные расстояния между всеми точками на листе бумаги.

Ваша задача — написать программу, которая по координатам точек определит, сколько различных расстояний встречается среди расстояний, которые измерил Дима.

Формат входного файла

Первая строка входного файла содержит число n — количество точек ($2 \le n \le 50$). Следующие n строк содержат по два целых числа — координаты точек. Координаты не превышают 10^9 по абсолютной величине.

Формат выходного файла

На первой строке выходного файла выведите k — количество различных расстояний, которые измерил Дима. Следующие k строк должны содержать по одному вещественному числу — сами расстояния. Расстояния должны быть выведены в возрастающем порядке. Каждое число должно быть выведено с точностью не менее чем 10^{-5}.

Примеры

fun2.in	fun2.out
4 0 0 1 1 1 0 0 1	2 1.0 1.414213562373

Задача E. Генерация тестов

Имя входного файла:	`gentest.in`
Имя выходного файла:	`gentest.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

При подготовке задач для олимпиад по информатике и программированию часто возникает необходимости подготовки тестов. Поскольку зачастую размеров тестов достаточно велик, то генерацию тестов разумно автоматизировать.

В геометрических задачах часто требуется сгенерировать n точек на плоскости так, чтобы никакие три из них не лежали на одной прямой. В этом и состоит ваша задача. Напишите программу, которая по числу n построит множество из n точек, обладающее указанным свойством.

Формат входного файла

Входной файл содержит целое число n ($1 \le n \le 2\,000$).

Формат выходного файла

Если искомое множество точек можно построить, то выведите в первой строке выходного файла слово “`YES`”, а далее — n строк, каждая из которых должна содержать два числа — координаты соответствующей точки. Среди точек не должно быть совпадающих. Все координаты должны быть целыми числами, не превосходящими 10000 по абсолютному значению. Если искомое множество точек нельзя построить, выведите в выходной файл строку “`NO`”.

Примеры

gentest.in	gentest.out
1	YES 0 0
4	YES 0 0 0 1 1 0 1 1

Задача F. Игра Гранди

Имя входного файла:	`grundy.in`
Имя выходного файла:	`grundy.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мегабайта

Игра Гранди ведется следующим образом. На столе располагается несколько кучек камней, i-я кучка содержит a_i камней. Два игрока делают ходы по очереди. Ход игрока заключается в том, что он выбирает кучку, содержащую более двух камней, и разбивает ее на две неравных кучки. Игрок, который не может сделать ход, поскольку все кучки содержат один или два камня, проигрывает.

По заданной начальной позиции определите, кто из игроков выигрывает при оптимальной игре обоих.

Формат входного файла

Первая строка входного файла содержит число n — количество кучек ($1 \le n \le 1000$). Вторая строка содержит n целых чисел a_i — размеры кучек в начальной позиции ($1 \le a_i \le 100\,000$).

Формат выходного файла

В выходной файл выведите “`first`”, если выигрывает игрок, который ходит первым, и “`second`”, если выигрывает игрок, который ходит вторым.

Примеры

grundy.in	grundy.out
2 3 3	second
2 3 5	first

Задача G. Матрица

Имя входного файла:	`matrix.in`
Имя выходного файла:	`matrix.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Задана матрица K, содержащая n строк и m столбцов. *Седловой точкой* этой матрицы назовем элемент, который одновременно является минимумом в своей строке и максимумом в своем столбце.

Найдите количество седловых точек заданной матрицы.

Формат входного файла

Первая строка входного файла содержит целые числа n и m ($1 \le n, m \le 750$). Далее следуют n строк по m чисел в каждой. j-ое число i-ой строки равно k_{ij}. Все k_{ij} по модулю не превосходят 1000.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

matrix.in	matrix.out
2 2 0 0 0 0	4
2 2 1 2 3 4	1

Задача H. Рекурсия

Имя входного файла:	`recursion.in`
Имя выходного файла:	`recursion.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	64 мегабайта

Рекурсия. *сущ. муж.* см. рекурсия.
– Фольклор.

Петя разрабатывает новую среду разработки для языка Ж±. Сейчас он пишет модуль, который будет отслеживать потенциальную бесконечную рекурсию.

Ж± -– процедурный язык. Все переменные в Ж± являются двоичными, каждая переменная может принимать два значения: "0" и "1". Каждая процедура может иметь несколько аргументов. В Ж± есть следующие операторы:

- увеличить значение на один "`x++`" -– если x равен 0, он становится равен 1;
- уменьшить значение на один "`x--`" -– если x равен 1, он становится равен 0;
- условный оператор "`if (x) S1 else S2`" где S1 и S2 -– произвольные операторы, выполнить S1, если x равен 1, или S2, если x равен 0;
- вызов процедуры "`f(x, y)`";
- составной оператор "`{S1 S2 ...}`", где S1, S2, и т. д. -– произвольные операторы: выполнить S1, S2, и т. д. в этом порядке.

Описание процедуры выглядит так: "`<имя процедуры>(<список аргументов>) S`" где S -– произвольный оператор.

Все аргументы передаются по значению (то есть изменения внутри процедуры не меняют значений соответствующих переменных в вызывающей процедуре).

Например, программа в первом примере содержит процедуру "`inc`", которая получает трех-битное значение в виде трех аргументов `x2`, `x1` и `x0` и рано или поздно вызывает процедуру "`done`" с тремя аргументами, которые задают значение, увеличенное на единицу.

По заданной программе выясните, можно ли так некоторую процедуру с такими аргументами, чтобы результатом вызова стала бесконечная рекурсия.

Формат входного файла

Входной файл содержит программу на языке Ж±. Она содержит не более 20 процедур, каждая процедура имеет не более 15 аргументов. Размер входного файла не превышает 1000 байт. Имена процедур и аргументов состоят из букв и цифр и являются зависимыми от регистра. Все имена не длиннее 40 символов. Никакая процедура не называется "`if`" или "`else`".

Формат выходного файла

Выведите "`YES`", если можно вызвать некоторую процедуру с такими аргументами, что в результате получится бесконечная рекурсия, или "`NO`", если нет.

Пример

<table>
<tr><th>recursion.in</th><th>recursion.out</th></tr>
<tr><td><pre>inc(x2, x1, x0) {
  if (x0) {
    x0--
    inc1(x2, x1, x0)
  } else {
    x0++
    done(x2, x1, x0)
  }
}

inc1(x2, x1, x0) {
  if (x1) {
    x1--
    inc2(x2, x1, x0)
  } else {
    x1++
    done(x2, x1, x0)
  }
}

inc2(x2, x1, x0) {
  if (x2) {
    x2--
    done(x2, x1, x0)
  } else {
    x2++
    done(x2, x1, x0)
   }
}

done(x2, x1, x0) {
}</pre></td><td><pre>NO</pre></td></tr>
<tr><td><pre>a(x0) {
  if (x0) {
  } else {
    a(x0)
  }
}</pre></td><td><pre>YES</pre></td></tr>
</table>

Задача I. Школа Магии

Имя входного файла:	school.in
Имя выходного файла:	school.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Гарри Чайников учится в Школе Магии. Сегодня он планирует начать готовиться к своему первому экзамену. На экзамене будут проверять знания по четырем разделам магии: алхимии, колдовству, левитации и демонологии.

Готовность Гарри по каждому из разделов может быть охарактеризована одним целым числом: оно находится в пределах от 0 до a для алхимии, от 0 до b для колдовства, от 0 до c для левитации и от 0 до d для демонологии. Чтобы сдать экзамен, необходимо чтобы каждый из параметров имел максимально возможное значение. Но Гарри редко ходил на лекции, поэтому сейчас, в начале подготовки, все его параметры равны 0.

Чтобы увеличить параметры Гарри может читать учебники. У него есть четыре учебника: "Абстрактная алхимия", "Основы колдовства", "Левитация не сложнее гидродинамики" и "Демонология для чайников". Чтение одного из учебников в течение часа увеличивает соответствующий параметр на единицу.

Однако, читая книгу, Гарри абстрагируется от остальных тем, поэтому остальные параметры могут тоже измениться. Он может забыть что-нибудь, что только что прочитал, и параметр уменьшится на единицу, может вспомнить что-нибудь, что он обсуждал с друзьями и параметр увеличится на единицу. Формально, если Гарри в течение часа читает "Абстрактную алхимию", его параметр подготовки по алхимии увеличивается на единицу, а каждый из параметров подготовки по колдовству, левитации и демонологии независимо равновероятно либо увеличивается на единицу, либо уменьшается на единицу, либо не меняется. При этом параметр никогда не становится меньше нуля и никогда не становится больше максимального значения (если параметр равен нулю либо своему максимальному значению, то равновероятный выбор делается из двух вариантов). Аналогично влияет на соответствующие параметры Гарри чтение других учебников.

Теперь Гарри хочет узнать, сколько ему придется готовится к экзамену, если он будет придерживаться оптимальной стратегии в области выбора учебника, который следует читать. Помогите ему определить ожидаемое время подготовки.

Формат входного файла

Входной файл содержит четыре целых числа: a, b, c и d ($1 \le a, b, c, d \le 4$).

Формат выходного файла

Выведите одно вещественное число — ожидаемое время подготовки в часах. Ответ должен быть выведен с точностью 10^{-6}.

Пример

school.in	school.out
1 1 1 1	8.0