Задача A. Скобки

Имя входного файла:	`brackets.in`
Имя выходного файла:	`brackets.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Рассмотрим последовательность из открывающихся и закрывающихся круглых скобок. Последовательность называется *правильной*, если она может быть построена по следующим правилам:

- пустая строка является правильной скобочной последовательностью;
- если S — правильная скобочная последовательность, то (S) — тоже правильная скобочная последовательность.
- если A и B — правильные скобочные последовательности, то AB — тоже правильная скобочная последовательность.

Примеры правильных скобочных последовательностей — «», «`()`», «`((()))`», «`()()()`», «`(((())())(()))`». Неформально говоря, правильная скобочная последовательность — это последовательность скобок, которая может быть получена из некоторого арифметического выражения удалением из него всего, кроме скобок.

Рассмотрим последовательность скобок, содержащую как круглые, так и квадратные скобки. Пусть разрешается выполнять следующие операции: заменить открывающуюся квадратную скобку на произвольное число открывающихся круглых и заменить закрывающуюся квадратную скобку на произвольное количество закрывающихся круглых. Разрешается при замене создавать ноль скобок, то есть просто удалять соответствующую квадратную скобку.

Требуется с использованием указанных операций получить из заданной строки минимальную по длине правильную скобочную последовательность, состоящую только из круглых скобок.

Например, из строки `[)())(]()]` можно получить правильную скобочную последовательность `(()())()()`.

Формат входного файла

Входной файл содержит одну строку, состоящую только из круглых и квадратных скобок. Длина строки не превышает 2000 символов.

Формат выходного файла

Выведите в выходной файл минимальную по длине правильную скобочную последовательность из круглых скобок, которую можно получить из заданной строки описанными операциями. Если решений несколько, выведите любое. Если из данной строки нельзя получить ни одной правильной скобочной последовательности, выведите в выходной файл слово «`Impossible`».

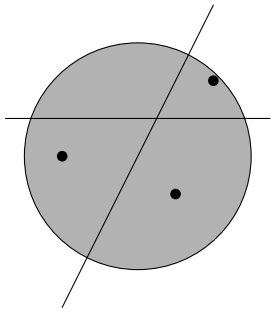
Пример

brackets.in	brackets.out
`[)())(]()]`	`(()())()()`
`[]`	
`[)(][]`	`()()`
`())`	`Impossible`

Задача B. Свечки

Имя входного файла:	`candles.in`
Имя выходного файла:	`candles.out`
Ограничение по времени:	13 секунд
Ограничение по памяти:	64 мегабайта

Мише исполнилось n лет. Праздничный торт, испеченный по этому случаю, имеет форму круга радиуса r с центром в начале координат. На торте стоят n свечек. Мишина мама разделила торт на части, сделав m прямолинейных разрезов. Каждый гость взял один из получившихся кусков.



Миша хочет узнать, не досталось ли кому-нибудь из его гостей более одной свечки. Помогите ему это выяснить.

Формат входного файла

Первая строка входного файла содержит целые числа n, m и r ($1 \le n \le 10000$, $0 \le m \le 1000$, $1 \le r \le 2000$).

Следующие n строк содержат пары целых чисел x_i, y_i — координаты точек, где расположены свечки. Гарантируется, что эти точки лежат внутри круга, размерами свечек следует пренебречь. Никакие две свечки не совпадают.

Последние m строк содержат описание разрезов — тройки целых чисел a_i, b_i, c_i. Такая тройка соответствует разрезу, который задается уравнением $a_i x + b_i y + c_i = 0$. Ни один разрез не проходит через свечку. Никакие два разреза не совпадают. Числа a_i, b_i, c_i не превышают 10000 по модулю.

Формат выходного файла

Если одному из гостей досталось более одной свечки, выведите в выходной файл слово «YES», иначе выведите слово «NO».

Пример

candles.in	candles.out
3 2 3 2 2 1 -1 -2 0 2 -1 0 0 1 -1	NO
3 2 3 2 2 1 -1 -2 0 1 1 -1 0 1 -1	YES
1 0 100 0 0	NO

Задача C. N-угольники

Имя входного файла:	`ngon.in`
Имя выходного файла:	`ngon.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Один из известных производителей товаров для детей во Флатландии собирается выпустить на рынок новую развивающую игру. Набор для игры будет состоять из некоторого количества отрезков, из которых дети смогут складывать различные фигуры.

Однако на презентации нового продукта перед государственной комиссией один из специалистов указал на то, что составление невырожденных n-угольников может крайне негативно сказаться на психическом развитии детей, поэтому следует избегать возможности появления в наборе такого множества из n отрезков, из которых можно составить невырожденный n-угольник.

Вырожденным называется n-угольник, площадь которого равна 0.

Производственная линия сконструирована таким образом, что длины получающихся отрезков могут быть натуральными числами, не превосходящими k. Директор компании хочет, чтобы набор состоял из как можно большего числа отрезков. Ваша задача — построить такой набор.

Формат входного файла

Входной файл содержит два целых числа: n — количество вершин в запрещенных многоугольниках и k — максимальную длину отрезков ($3 \le n \le 10$, $1 \le k \le 10^8$).

Формат выходного файла

На первой строке выходного файла выведите одно число — наибольшее возможное количество отрезков в наборе, которое может быть достигнуто при данных ограничениях.

На второй строке выведите длины этих отрезков в неубывающем порядке. Если решений несколько, выведите любое.

Пример

ngon.in	ngon.out
3 7	5 1 1 2 3 5

Задача D. Нефть

Имя входного файла:	`oil.in`
Имя выходного файла:	`oil.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Президент одной маленькой, но очень гордой страны вдруг узнал, что на дворе двадцать первый век, и на лошадях ездить уже не модно. Однако, как выяснилось, нефти в стране нет, а без бензина автомобили ездить не умеют. Так что придется закупать нефть в других странах.

Исследование внешнего рынка показало, что в мире есть n стран, экспортирующих нефть. При этом i-е государство продает баррель нефти либо за a_i долларов, либо за b_i евро.

У президента есть a долларов и b евро. Главный бухгалтер утверждает, что если попытаться купить нефть у одного государства и за доллары, и за евро, то бюрократия может надолго отложить покупку, чего президент, разумеется, не хочет.

Помогите президенту в таких непростых условиях узнать, сколько баррелей нефти он сможет купить.

Формат входного файла

На первой строке входного файла записаны три целых числа: n, a и b ($1 \le n \le 100$, $0 \le a, b \le 1000$). В последующих n строках содержатся пары чисел a_i, b_i ($1 \le a_i, b_i \le 1000$).

Формат выходного файла

Выведите в выходной файл максимальное количество нефти, которое может купить президент. Выведите ответ не менее чем с двумя знаками после десятичной точки.

Пример

oil.in	oil.out
3 2 5 6 4 3 5 8 7	1.92
4 3 2 1 1 2 2 3 3 4 4	4.00

Задача E. Перестановки

Имя входного файла:	`perms.in`
Имя выходного файла:	`perms.out`
Ограничение по времени:	12 секунд
Ограничение по памяти:	64 мегабайта

Саша и Федя играют в интересную игру. У них есть n кубиков, на которых написаны различные числа от 1 до n. Ребята нарисовали на бумаге n клеточек в ряд и играют по следующим правилам.

Сначала первый игрок выставляет некоторые кубики на клеточки, затем второй игрок выставляет на свободные клетки оставшиеся кубики. После этого первый игрок делает следующие действия: он смотрит, какое число написано на последнем кубике (пусть это число a) и после этого переставляет последние a кубиков в обратном порядке. Эти действия первый игрок повторяет до тех пор, пока последним не станет кубик с числом 1.

Например, пусть у ребят пять кубиков. Если первый игрок поставил второй и третий кубик на третье и пятое место: «`..3.2`», то второй игрок может расставить оставшиеся кубики так: «`41352`». В этом случае первому игроку потребуется сделать пять действий: «`41325`», «`52314`», «`54132`», «`54123`», «`54321`», после чего игра закончится.

Сейчас первым ходил Саша. Помогите Феде расставить кубики так, чтобы Саша сделал максимально возможное количество действий.

Формат входного файла

Во входном файле содержится число n ($1 \le n \le 25$). Следующие n чисел задают расположение кубиков после хода Саши. Число 0 означает, что клетка свободна, число от 1 до n — номер кубика, который стоит в этой клетке. Во входном файле не более 10 нулей.

Формат выходного файла

На первой строке выходного файла выведите максимальное количество действий, которое придется сделать Саше.

На второй строке выведите n чисел от 1 до n, где i-е число означает номер кубика, стоящего в i-ой клетке после хода Феди. Если оптимальных решений несколько, выведите любое.

Пример

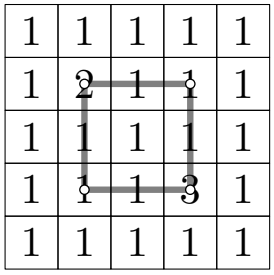
perms.in	perms.out
5 0 0 3 0 2	5 4 1 3 5 2
2 0 0	1 1 2

Задача F. Непростая задача

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

Дана прямоугольная таблица, состоящая из m строк и n столбцов. На пересечении i-й строки и j-го столбца записано целое число a_{ij}.

Требуется найти такие четыре различные ячейки таблицы, чтобы их центры были вершинами прямоугольника со сторонами, параллельными сторонам таблицы, а сумма чисел, записанных в этих ячейках, была максимальна.



Формат входного файла

На первой строке записаны два натуральных числа m и n ($2 \le m, n \le 500$). Далее следует описание таблицы — m строк, каждая из которых содержит по n целых чисел ($-10^7 \le a_{ij} \le 10^7$).

Формат выходного файла

На первой строке выходного файла выведите целое число r — максимальную сумму выбранных элементов, на второй строке выведите 4 натуральных числа x_1, y_1, x_2, y_2 — координаты левой верхней и правой нижней из выбранных ячеек, соответственно ($1 \le x_1 < x_2 \le m$, $1 \le y_1 < y_2 \le n$). Если оптимальных решений несколько, выведите любое.

Пример

problem.in	problem.out
5 5 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1	7 2 2 4 4
5 5 1 -1 -1 -1 -1 -1 -2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -3 -1 -1 -1 -1 1 -1	0 1 1 5 4

Задача G. Головоломка

Имя входного файла:	`puzzle.in`
Имя выходного файла:	`puzzle.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Не так давно были достаточно популярны настольные игры на больших бумажных картах, в которых игроки передвигали фишки по определенным правилам.

Недавно Вася нашел на чердаке целую кипу таких карт, но к ним, к сожалению, не прилагались правила игры. Без описаний он смог только понять, что на каждой из карт нарисовано некоторое количество кружков — возможных позиций для фишки, среди которых выделены начальная и конечная. Какие-то кружки соединены между собой отрезками, по которым разрешается перемещать фишку, при этом между некоторыми парами кружков может быть проведено сразу несколько отрезков. Перемещать фишку по отрезку разрешается в обоих направлениях.

Поскольку правила игры Васе найти не удалось, он придумал свои собственные. В игре участвуют одновременно все карты. На каждой карте Вася использует ровно одну фишку. Изначально каждая фишка расположена в начальной позиции на соответствующей карте. Каждым ходом Вася перемещает фишку на каждой из карт из ее текущей позиции в некоторую другую позицию на этой карте, с которой текущая соединена отрезком. При этом даже если фишка на некоторой карте уже находится в конечной позиции, то Вася, делая очередной ход, все равно должен ее переместить.

Вася заинтересовался, за какое минимальное количество ходов можно добиться того, чтобы фишки на всех картах одновременно оказались в конечных позициях. Помогите ему это выяснить.

Гарантируется, что в пределах каждой карты из любой позиции можно переместить фишку в любую другую, возможно через некоторые промежуточные позиции.

Формат входного файла

Первая строка входного файла содержит число k — количество карт ($1 \le k \le 10$).

Затем следуют k блоков, которые описывают карты. Первая строка каждого блока содержит два целых числа n_i и m_i ($2 \le n_i \le 50$, $1 \le m_i \le 1500$), они задают количество позиций и количество отрезков на i-й карте, соответственно. Будем считать, что позиции пронумерованы числами от 1 до n_i, причем начальная позиция имеет номер 1, а конечная — номер n_i. В следующих m_i строках находятся пары номеров позиций, соединенных соответствующим отрезком.

Формат выходного файла

В случае, если существует последовательность ходов, после которой фишки на каждой карте одновременно окажутся в конечных состояниях, выведите в выходной файл минимальную длину такой последовательности. Если такой последовательности не существует, выведите слово «`Impossible`».

Пример

puzzle.in	puzzle.out
2 5 4 1 2 2 3 3 4 3 5 3 3 1 2 2 3 3 1	3

Задача H. Экспериментатор

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

На старости лет один профессор загорелся идеей исследования на прочность транзисторов «КД521(2)». К сожалению, ему не удалось привлечь на помощь никого из коллег, поэтому проводить измерения придется самостоятельно. Но это не пугает профессора.

В шкафу профессор обнаружил m транзисторов данной модели, оставшихся со старых времен, и решил использовать их для экспериментов.

После некоторых размышлений был выбран следующий способ проведения измерений: профессор собирается, перемещаясь по пожарной лестнице, сбрасывать транзисторы с различных этажей. Таким образом он планирует определить, при падении с какого минимального этажа транзистор разбивается. При этом профессор уверен, что транзистор не может выдержать падение с последнего этажа, однако падение с высоты человеческого роста (то есть когда профессор находится на первом этаже) не причиняет транзистору вреда.

Известно, что все транзисторы абсолютно одинаковые, и если транзистор разбивается при падении с некоторого этажа, то он разбивается и при падении со всех этажей с большим номером.

Разбившиеся транзисторы снова использовать нельзя, а если транзистор остался целым после падения, его можно использовать повторно. Для того, чтобы поднять оставшийся целым транзистор, профессору надо спуститься на первый этаж. Оказавшись на первом этаже, профессор может поднять все лежащие там транзисторы.

Годы профессора уже дают о себе знать, поэтому он хочет минимизировать суммарное расстояние, которое ему придется подниматься по лестнице. Но, возраст дает и определенные преимущества — сняв очки, профессор может с любого этажа определить, разбился транзистор или нет.

Изначально профессор находится на первом этаже, и у него имеется m транзисторов. В доме, в котором живет профессор, n этажей.

Найдите минимальное число этажей, которое профессору в худшем случае придется подниматься вверх по лестнице во время проведения экспериментов.

Формат входного файла

Во входном файле заданы два целых числа — высота дома n ($2 \le n \le 50$) и количество транзисторов m ($1 \le m \le 10$).

Формат выходного файла

В выходной файл выведите единственное число — минимальное расстояние в этажах, которое в худшем случае придется подниматься вверх по лестнице профессору во время эксперимента.

Пример

tester.in	tester.out
5 2	3
2 2	0

В первом из приведенных примеров оптимальное поведение профессора следующее. Сначала следует подняться на два этажа и бросить транзистор с третьего. Если транзистор разобьется, то следует спуститься на второй этаж и попытаться бросить транзистор оттуда — если транзистор разбивается и при бросании со второго этажа, то результат 2, иначе — 3. Если же транзистор не разобьется при падении с третьего этажа, то придется подняться на четвертый и бросить транзистор оттуда. Если он разобьется, то результат 4, если нет — 5. В худшем случае придется подняться на три этажа.

Во втором примере ничего бросать не надо, результат исследования — 2.

Задача I. Перевод времени

Имя входного файла:	`time.in`
Имя выходного файла:	`time.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Как известно, с целью экономии электроэнергии многие страны используют переход на так называемое *летнее время*. Перевод времени осуществляется два раза в год — весной и осенью. Весной осуществляется переход на летнее время: часы переводятся на один час вперед. Осенью летнее время отменяется и часы переводятся на один час назад.

Перевод времени осуществляется ночью. При переходе на летнее время через минуту после 01:59 сразу наступает 03:00. При отмене летнего времени час с 02:00 по 02:59 повторяется два раза. А именно, в день перевода, когда первый раз после 02:59 должно стать 03:00, вместо этого снова становится 02:00.

Как одному из разработчиков новой операционной системы «*Mocrosoft Widows 2006*», вам поручено написать фрагмент ядра операционной системы, который будет осуществлять автоматический перевод системных часов на летнее время и обратно.

По заданным начальному моменту и информации о переводе в текущие и следующие сутки, ваша программа должна вывести показания часов в течение заданного количества минут.

Формат входного файла

Первая строка входного файла содержит целое число m — количество минут, которое прошло от начала текущих суток, до первого момента времени, который следует вывести. Гарантируется, что оно неотрицательно и строго меньше числа минут в текущих сутках.

На второй строке находятся два целых числа d_1 и d_2, которые указывают, какой перевод времени осуществляется в текущие и в следующие сутки. Значение 1 означает, что осуществляется переход на летнее время, -1 означает, что осуществляется отмена летнего времени, а 0 означает, что перевода времени не осуществляется.

На третьей строке записано число k — количество отсчетов времени, которое ваша программа должна вывести ($1 \le k \le 600$).

Формат выходного файла

Выходной файл должен состоять из k строк, i-я из которых должна содержать показания часов через $(i-1)$ минут после начального момента времени. Выводите время в формате «`hh:mm`».

Пример

time.in	time.out
118 1 0 4	01:58 01:59 03:00 03:01
190 -1 1 1	02:10
0 -1 0 3	00:00 00:01 00:02
1438 0 1 4	23:58 23:59 00:00 00:01

Задача J. Стена

Имя входного файла:	`wall.in`
Имя выходного файла:	`wall.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Фирма «Kel-Morian Productions» разрабатывают искусственный интеллект для новой автоматизированной модели промышленного робота SCV-2. На данном этапе создается робот для строительства и ремонта стен, составленных из стандартных строительных блоков.

Для начала было принято решение сделать упрощенную модель робота, который будет работать со стенами, состоящими из блоков одинакового размера. Стена представляет собой последовательность столбиков, составленных из блоков, пример такой стены приведен на рисунке 1.

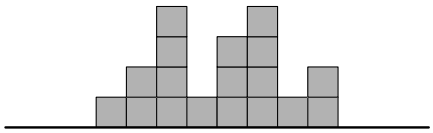


Рисунок 1

Первая модель робота может выполнять ровно одно элементарное действие — взять верхний блок в некотором столбике и положить его на соседний столбик. При этом создавать новый столбик, ставя блок рядом с краем стены, не разрешается.

В качестве тестового задания для искусственного интеллекта была поставлена задача *выравнивания стены*. Стена называется *ровной*, если высота двух любых столбиков различается не более чем на один. В процессе выравнивания робот должен с использованием элементарных действий превратить заданную стену в произвольную ровную. При этом количество выполненных элементарных действий должно быть минимально.

Например, стена, приведенная на рисунке 1, может быть превращена в ровную стену, приведенную на рисунке 2, за четыре элементарных действия, и это число действий является минимальным для данной стены.

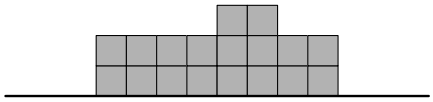


Рисунок 2

Помогите разработчикам искусственного интеллекта проверить разработанный ими алгоритм, найдите минимальное количество действий, которое придется совершить роботу для выравнивания заданной стены.

Формат входного файла

В первой строке входного файла записано целое число n ($1 \le n \le 1000$) — количество вертикальных рядов, из которых состоит стена. Вторая строка содержит числа $a_1, a_2, \ldots, a_n$, где a_i задает количество блоков в i-м столбике ($1 \le a_i \le 10^6$).

Формат выходного файла

В выходной файл выведите единственное целое число — минимальное количество перемещений блоков, необходимое для выравнивания стены. Выровненная стена должна состоять из такого же количества столбиков, как исходная стена.

Пример

wall.in	wall.out
8 1 2 4 1 3 4 1 2	4