Задача A. Прямоугольнирование квадрата

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вам дан квадрат размера $N \times N$, который разделён сеткой на N^2 одинаковых маленьких квадратиков размера 1×1.

Любители Квадратных Шаблонов просят вас разбить этот квадрат на несколько различных (больше одного) прямоугольников так, чтобы стороны полученных прямоугольников проходили по сетке исходного квадрата, а разность площадей любых двух прямоугольников не превосходила $1.5 \times N$. Два прямоугольника считаются равными, если один из них можно превратить в другой с помощью перемещений и поворотов.

Формат входных данных

В единственной строке ввода дано одно целое число N ($1 \leqslant N \leqslant 10^6$).

Формат выходных данных

В первой строке выведите S – количество прямоугольников в разбиении.

В последующих S строках выведите описания найденных прямоугольников. Каждая из этих строк должна содержать 4 числа x, y, wx, wy – координаты левого нижнего угла, ширина и высота прямоугольника соответственно.

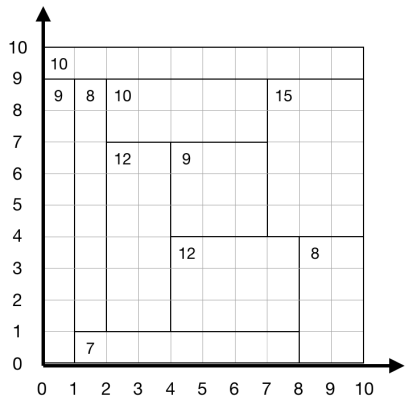
Если искомого разбиения не существует, в единственной строке файла выведите число -1.

Пример

стандартный ввод	стандартный вывод
10	10 0 0 1 9 1 0 7 1 8 0 2 4 1 1 1 8 2 1 2 6 4 1 4 3 4 4 3 3 7 4 3 5 2 7 5 2 0 9 10 1

Замечание

Пояснение к первому примеру:



В первом примере максимальная попарная разность между площадями прямоугольников равна 8.

Задача B. Игра со стеком

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

У Вани есть натуральное число n. Он придумал следующую игру. У него есть стек, в котором изначально лежит единственное число 1. Он может выполнять следующую операцию: пусть сейчас в стеке лежит s целых чисел $a_1, a_2, \ldots, a_s$, в порядке от самого первого элемента стека к самому последнему. Ваня может выбрать два целых числа $1 \leqslant l \leqslant r \leqslant s$ и положить число $a_l + a_{l+1} + \ldots + a_r$ в конец стека.

Ваня хочет сделать последнее число в стеке равным n. Определите минимальное количество операций, которое для этого требуется и постройте любой возможный пример с этим количеством операций.

Формат входных данных

В первой строке находится единственное целое число t ($1 \leqslant t \leqslant 1000$) — количество тестовых случаев.

В следующих t строках находится по одному целому числу n ($1 \leqslant n \leqslant 10^{18}$) — число Вани.

Формат выходных данных

Выведите ответы на тестовые случаи в порядке их следования во входных данных. Для тестового случая выведите ответ в следующем формате:

В первой строке выведите единственное целое число q ($0 \leqslant q \leqslant 1000$) — минимальное количество операций со стеком, которое нужно сделать, чтобы последнее число стало равно n. Гарантируется, что за $\leqslant 1000$ операций возможно сделать последнее число равным n. В следующих строках выведите по два целых числа — описания операций. В i-й строке выведите два целых числа l_i, r_i ($1 \leqslant l_i \leqslant r_i \leqslant i$). Во время i-й операции в конец стека кладется число $a_{l_i} + a_{l_i+1} + \ldots + a_{r_i}$. После выполнения предоставленных операций последнее число стека должно быть равно n.

Все тестовые случаи независимы.

Пример

стандартный ввод	стандартный вывод
3 2 3 7	2 1 1 1 2 3 1 1 2 2 1 3 4 1 1 1 2 2 3 1 4

Задача C. Минимальная уникальная подстрока

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Пусть у вас есть строка s из символов "`0`" и "`1`". Будем называть строку t подстрокой строки s, если существует $1 \leqslant l \leqslant |s| - |t| + 1$, такое что $t = s_l s_{l+1} \dots s_{l+|t|-1}$. Будем называть подстроку t строки s уникальной, если существует единственное такое l.

Например, пусть $s =$"`1010111`". Тогда $t =$"`010`" является уникальной подстрокой s, так как существует единственное подходящее $l = 2$. Заметим, что $t =$"`10`" не является уникальной подстрокой s, так как подходят $l = 1$ и $l = 3$. А, например, $t =$"`00`" вообще не является подстрокой строки s, так как не существует подходящих l.

Сегодня Вася на уроке информатики решал такую задачу: дана строка из символов "`0`" и "`1`", надо найти длину её кратчайшей уникальной подстроки. Написав решение к этой задаче, он решил его протестировать. Он просит помощи у вас.

Вам даны 2 таких целых положительных числа n и k, что $(n \bmod 2) = (k \bmod 2)$, где $(x \bmod 2)$ — это операция взятия остатка числа x при делении на 2. Найдите любую строку s состоящую из n символов "`0`" и "`1`", такую что наименьшая длина её уникальной подстроки равна k.

Формат входных данных

В первой строке даны два целых числа n и k, разделённые пробелом ($1 \leqslant k \leqslant n \leqslant 100\,000$, $(k \bmod 2) = (n \bmod 2)$).

Формат выходных данных

Выведите строку s длины n, состоящую из символов "`0`" и "`1`". Минимальная длина уникальной подстроки s должна равняться k. Среди таких строк разрешается вывести **любую**. Гарантируется, что хотя бы одна подходящая строка существует.

Примеры

стандартный ввод	стандартный вывод
4 4	1111
5 3	01010
7 3	1011011

Замечание

В первом тесте легко видеть, что единственной уникальной подстрокой строки $s =$"`1111`" является вся строка s, длина которой 4.

Во втором тесте у строки $s =$"`01010`" минимальной по длине уникальной подстрокой является строка $t =$"`101`" длина которой 3.

Во третьем тесте у строки $s =$"`1011011`" минимальной по длине уникальной подстрокой является строка $t =$"`110`" длина которой 3.

Задача D. Головоломка из фишек

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Егор придумал новую головоломку из фишек, в которую предлагает сыграть вам.

Головоломка имеет вид таблицы из n строк и m столбцов, в каждой клетке которой могут располагаться несколько черных и белых фишек, положенных в ряд. Таким образом, состояние в клетке можно описать строкой, состоящей из символов «`0`» (белая фишка) и «`1`» (черная фишка), возможно пустой, а вся головоломка — это таблица, в каждой клетке которой стоит строка из нулей и единиц. Задача состоит в том, чтобы из одного состояния таблицы получить другое.

Для этого можно использовать следующую операцию:

- выбрать две различные клетки (x_1, y_1) и (x_2, y_2), при этом клетки должны находиться в одной строке или одном столбце таблицы, и строка в клетке (x_1, y_1) должна быть непустой;
- за операцию можно переместить последний символ строки в клетке (x_1, y_1) в начало строки в клетке (x_2, y_2).

Для вас Егор загадал 2 состояния таблицы — начальное и конечное. Гарантируется, что количества нулей и единиц в таблицах совпадают. Ваша задача — с помощью нескольких операций получить из начального состояния конечное. Конечно, Егор не хочет, чтобы количество операций было очень большим. Обозначим за s количество символов в каждой из таблиц (в таблицах поровну символов). Тогда вы должны использовать не более $4 \cdot s$ операций.

Формат входных данных

В первой строке записаны два целых числа n и m ($2 \leqslant n, m \leqslant 300$) — количества строк и столбцов у таблиц в головоломке, соответственно.

В следующих n строках находится описание начального состояния таблицы в следующем формате: в каждой строке находится m непустых строк, состоящих из нулей и единиц. В i-й из этих строк, j-я строка описывает строку, записанную в клетке (i, j). Строки нумеруются от 1 до n, столбцы нумеруются от 1 до m.

В следующих n строках находится описание конечного состояния таблицы в аналогичном формате.

Обозначим суммарную длину строк в начальном состоянии за s. Гарантируется, что $s \leqslant 100\,000$. Также гарантируется, что количества нулей и единиц совпадают в начальном и конечном состояниях.

Формат выходных данных

В первой строке выведите одно целое число q — количество использованных операций. Необходимо найти решение, для которого $0 \leqslant q \leqslant 4 \cdot s$.

В следующих q строках выведите по 4 целых числа x_1, y_1, x_2, y_2. i-я из этих строк должна описывать i-ю операцию. Необходимо, чтобы было выполнено $1 \leqslant x_1, x_2 \leqslant n$, $1 \leqslant y_1, y_2 \leqslant m$, $(x_1, y_1) \neq (x_2, y_2)$, $x_1 = x_2$ или $y_1 = y_2$. При этом строка в клетке (x_1, y_1) должна быть непустой. Данная последовательность операций должна переводить таблицу из начального состояния в конечное.

Можно показать, что ответ существует. Если есть несколько возможных решений, выведите любое.

Примеры

стандартный ввод	стандартный вывод
2 2 00 10 01 11 10 01 10 01	4 2 1 1 1 1 1 1 2 1 2 2 2 2 2 2 1
2 3 0 0 0 011 1 0 0 0 1 011 0 0	4 2 2 1 2 1 2 2 2 1 2 1 3 1 3 1 2

Замечание

Рассмотрим первый пример.

- Текущее состояние таблицы:

  ```
  00 10
  01 11
  ```

  Первая операция. В клетке $(2, 1)$ записана строка 01. Применяя операцию к двум клеткам $(2, 1)$ и $(1, 1)$, мы переносим 1 из конца строки 01 в начало строки 00, получая строку 100.

- Текущее состояние таблицы:

  ```
  100 10
  0   11
  ```

  Вторая операция. В клетке $(1, 1)$ записана строка 100. Применяя операцию к двум клеткам $(1, 1)$ и $(1, 2)$, мы переносим 0 из конца строки 100 в начало строки 10, получая строку 010.

- Текущее состояние таблицы:

  ```
  10 010
  0  11
  ```

  Третья операция. В клетке $(1, 2)$ записана строка 010. Применяя операцию к двум клеткам $(1, 2)$ и $(2, 2)$, мы переносим 0 из конца строки 010 в начало строки 11, получая строку 011.

- Текущее состояние таблицы:

  ```
  10 01
  0  011
  ```

  Четвертая операция. В клетке $(2, 2)$ записана строка 011. Применяя операцию к двум клеткам $(2, 2)$ и $(2, 1)$, мы переносим 1 из конца строки 011 в начало строки 0, получая строку 10.

- Текущее состояние таблицы:

  ```
  10 01
  10 01
  ```

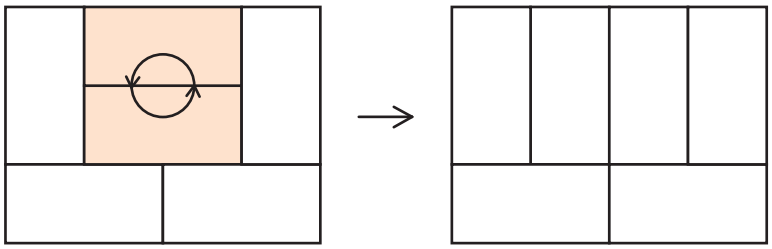
Можно видеть, что мы получили требуемое состояние таблицы.

Задача E. Parquet Re-laying

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Петя решил положить паркет в комнате размером $n \times m$, паркет состоит из досок размером 1×2. Когда строители уложили паркет, выяснилось, что его рисунок выглядит совсем не так, как нравится Пете, и строителям придется его переложить.

Однако строители решили, что снимать паркет целиком и потом раскладывать его заново очень сложно, поэтому каждый час они будут делать такую операцию: вынуть какие-то две доски, образующие квадрат 2×2, повернуть их на 90 градусов и положить на то же место.



При этом у них нет никаких идей, как с помощью таких операций получить требуемое расположение, и возможно ли это вообще.

Помогите Пете составить план для рабочих или скажите, что это невозможно. План должен содержать не более 100 000 команд.

Формат входных данных

Первая строка входного файла содержит числа n и m — размер комнаты ($1 \leqslant n, m \leqslant 50$).

Следующие n строк содержат по m символов — описание текущего положения досок паркета. Каждый символ обозначает положение половинки доски. Символы '`L`', '`R`', '`U`' и '`D`' соответствуют левой, правой, верхней и нижней половинкам соответственно.

Следующие n строк содержат по m символов содержат описание требуемой конфигурации в том же формате.

Формат выходных данных

В первой строке выведите k — число операций в плане для рабочих. В следующих k строках выведите описания операций. Операция задается координатами (строка и столбец) левой верхней половинки доски, над которыми проводится операция.

Если решения нет, выведите в первой строке `-1`.

Система оценки

Подзадача	Баллы	Ограничения	Оценка	Необх. подзадачи
1	13	$n, m \leqslant 4$	подзадача	—
2	28	$n, m \leqslant 20$	подзадача	1
3	59	$n, m \leqslant 50$	подзадача	1, 2

В первой и второй подзадачах сообщаются результаты проверки на каждом тесте подзадачи.

В третьей подзадаче сообщаются баллы за подзадачу и результат проверки первого не пройденного теста.

Пример

стандартный ввод	стандартный вывод
2 3 ULR DLR LRU LRD	2 1 2 1 1

Задача F. Баланс соседей

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Уже совсем скоро начнется отбор в олимпиадную школу, а еще столько всего предстоит сделать! Организаторы долго разбирались с тем, как расселить всех участников, поэтому не успели придумать легенду к этой задаче...

От вас требуется построить связный неориентированный граф с n вершинами, пронумерованными целыми числами от 1 до n. Обозначим за s_v сумму номеров вершин, смежных с вершиной v. Граф должен обладать следующим свойством: значения s_v для всех v должны быть одинаковыми.

Граф не должен содержать петли и кратные ребра.

Формат входных данных

Единственная строка содержит одно целое число n ($3 \leqslant n \leqslant 100$).

Формат выходных данных

В первой строке выведите число ребер m в построенном графе.

В каждой из следующих m строк выведите по два числа — номера вершин, соединенных очередным ребром.

Если решений несколько, выведите любое. Можно показать, что ответ всегда существует.

Пример

стандартный ввод	стандартный вывод
3	2 1 3 2 3

Замечание

В примере из условия вершина 1 соединена только с вершиной 3 (сумма равна 3), вершина 2 соединена только с вершиной 3 (сумма равна 3), а вершина 3 соединена с вершинами 1 и 2 (сумма также равна 3).

Задача G. Тест на терпение

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Как вы знаете, Энакину не хватает терпения, поэтому мастер Оби-Ван решил дать много одинаковых задачек своему ученику. А именно, он дал ему n троек чисел a_i, b_i, c_i и попросил найти для каждой тройки два числа x_i и y_i таких, что побитовое И этих двух чисел равно a_i, побитовое ИЛИ этих чисел равно b_i, а их разность $x_i - y_i$ равна c_i. У Энакина нет времени на решение задач, поэтому он просит вас сделать это задание вместо него. Учтите, что Оби-Ван торопился на совет джедаев, поэтому среди троек могут быть такие, для которых решение не существует.

Формат входных данных

В первой строчке дано целое число n — количество троек ($1 \leqslant n \leqslant 10000$).

В следующих n строчках даны описания троек — три целых числа a_i, b_i, c_i ($0 \leqslant a, b, c \leqslant 10^9$).

Формат выходных данных

Для каждой тройки определите, существует ли два числа, удовлетворяющих условиям.

Если существует, то выведите в одной строчке два целых числа x_i, y_i — ответ для тройки i. Если существует несколько правильных пар чисел, являющихся ответом, разрешается вывести любую.

Если не существует, то выведите в одной строчке -1.

Пример

стандартный ввод	стандартный вывод
3 8 14 2 2 7 3 8 13 4	12 10 6 3 -1

Замечание

Побитовое И двух неотрицательных целых чисел a и b определяется так. Сначала два числа записываются в двоичной системе счисления одно над другим так, чтобы последняя цифра второго числа оказалась точно под последней цифрой первого числа. Далее, если записи чисел оказались неравной длины, то более короткое из двух чисел дополняется слева нулями, пока цифр в числах не станет поровну. Дальше под этими двумя числами записывается третье двоичное число по следующим правилам: если на i-м месте в первом и втором числе стоят `1`, то в третьем числе на i-м месте записывается `1`; в противном случае там записывается `0`. Построенное таким образом третье число и является двоичной записью побитового И данных двух чисел.

Например, рассмотрим числа 17 и 71. В двоичном виде они записываются как `10001` и `1000111`. Дополним первое число двумя нулями слева, чтобы в нём стало семь цифр, как и во втором числе: `0010001`. Напишем под парой цифр `1`, если они обе равны `1`, и `0` иначе:

```
0010001
1000111
0000001
```

Если стереть лидирующие нули, получится `1` — двоичная запись числа 1. Таким образом, 17 AND 71 — побитовое И чисел 17 и 71 — равно 1.

Похожим образом определяется побитовое ИЛИ. Мы так же выписываем числа, дополняем нулями, но под двумя цифрами ставим `1` в случае, если хотя бы одна из цифр равна `1`, а иначе ставим `0`. В частности, для тех же чисел 17 и 71 имеем:

```
0010001
1000111
1010111
```

Полученная двоичная запись представляет число $2^6 + 2^4 + 2^2 + 2^1 + 2^0 = 87$. Таким образом, 17 OR 71 — побитовое ИЛИ чисел 17 и 71 — равно 87.

В языке Pascal побитовые И и ИЛИ вычисляются с помощью команд a `and` b и a `or` b, а в C++, Java, Python — с помощью a `&` b и a `|` b.

Задача H. Микропроцессор

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.25 секунд
Ограничение по памяти:	256 мегабайт

Молодой программист Вася решил изучить устройство микропроцессора. На первом шаге, он разработал процессор с одним целочисленным регистром с начальным значением 0, и двумя инструкциями:

1. Увеличить значение в регистре на 1 (обозначается буквой i);
2. Перейти в начало программы (обозначается буквой j).

Вася быстро заметил, что такой набор инструкций не очень полезен, потому что любая программа с хотя бы одной инструкцией j будет ходить по циклу бесконечно.

Васин друг Петя предложил поменять устройство так, чтобы после выполнения инструкции j процессор заменял бы ее специальной no-op инструкцией, которая ничего не делает.

Теперь, Васин микропроцессор может сделать хоть что-то интересное. Вася хочет поисследовать низкоуровневые оптимизации для своего нового микропроцессора.

Ваша программа должна вывести кратчайшую непустую программу для Васиного микропроцессора, которая поместит заданное значение в регистр.

Формат входных данных

Входной файл содержит единственное целое число N ($1 \leqslant N \leqslant 10^9$) — желаемое значение в регистре.

Формат выходных данных

Вывод должен содержать единственную строку из букв i и j — кратчайшую программу.

Если таких несколько, выведите любую из них.

Примеры

стандартный ввод	стандартный вывод
1	i
17	iijijjjj
2	ij

Задача I. Извилистая ломаная

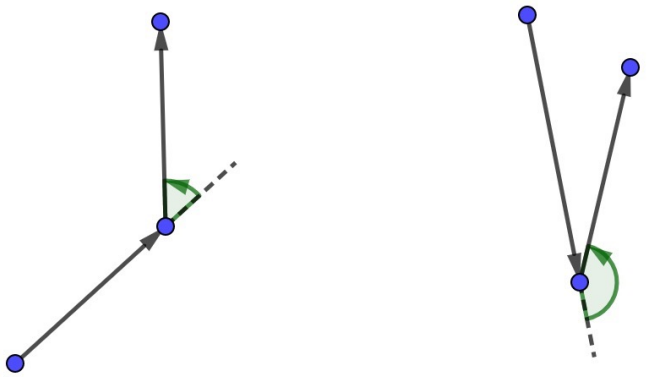
Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

У Васи есть n различных точек $A_1, A_2, \dots A_n$ на плоскости. Никакие три из них не лежат на одной прямой. Он хочет расположить их в некотором порядке $A_{p_1}, A_{p_2}, \dots, A_{p_n}$, где $p_1, p_2, \dots, p_n$ — это некоторая перестановка чисел от 1 до n.

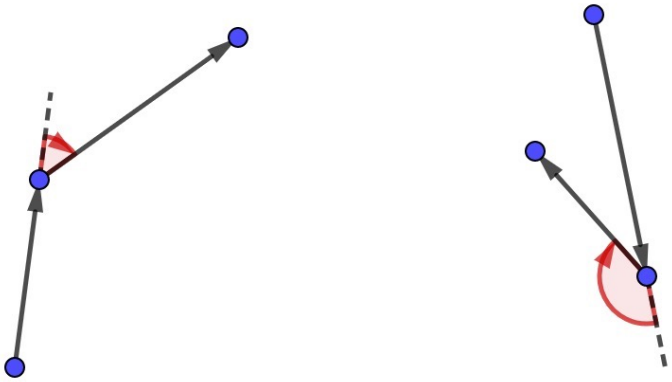
Сделав так, он нарисует ориентированную ломаную на этих вершинах, проведя направленные отрезки из каждой точки в следующую в выбранном порядке точку. То есть для всех $1 \leqslant i \leqslant n-1$ он проведет направленный отрезок из точки A_{p_i} в точку $A_{p_{i+1}}$. Он хочет, чтобы получившаяся ломаная удовлетворяла 2-м условиям:

- она будет несамопересекающейся, то есть любые 2 отрезка, которые не являются соседними, не имеют общих точек.
- она будет извилистой.

У Васи есть строка s, состоящая из $(n-2)$-х символов "`L`" или "`R`". Будем называть направленную ломаную извилистой, если её i-й поворот будет налево, если $s_i =$ "`L`" и направо, если $s_i =$ "`R`". Более формально: i-й поворот ломаной будет в точке $A_{p_{i+1}}$, в ней направленный отрезок из точки A_{p_i} в точку $A_{p_{i+1}}$ поменяется на направленный отрезок из точки $A_{p_{i+1}}$ в точку $A_{p_{i+2}}$. Обозначим вектор $\overrightarrow{v_1} = \overrightarrow{A_{p_i}A_{p_{i+1}}}$ и вектор $\overrightarrow{v_2} = \overrightarrow{A_{p_{i+1}}A_{p_{i+2}}}$. Тогда если для того, чтобы повернуть вектор $\overrightarrow{v_1}$ на наименьший возможный угол, чтобы его направление совпало с направлением вектора $\overrightarrow{v_2}$ надо сделать поворот против часовой стрелки, то будем говорить, что i-й поворот налево, а иначе направо. Для лучшего понимания посмотрите картинки, на которых изображены различные варианты поворотов:



На этой картинке изображены повороты налево



На этой картинке изображены повороты направо

Вам даны координаты точек $A_1, A_2, \dots A_n$ на плоскости и строка s. Найдите перестановку $p_1, p_2, \dots, p_n$ чисел от 1 до n, такую что ломаная, которую нарисует Вася, будет удовлетворять двум заданным условиям.

Формат входных данных

В первой строке написано одно целое число n — количество точек ($3 \leqslant n \leqslant 2000$). В следующих n строках написаны по два целых числа x_i и y_i, разделённые пробелом — координаты точки A_i на плоскости ($-10^9 \leqslant x_i, y_i \leqslant 10^9$). В последней строке написана строка s из символов "L" и "R" длины $(n-2)$. Гарантируется, что все точки различны и и никакие три точки не лежат на одной прямой.

Формат выходных данных

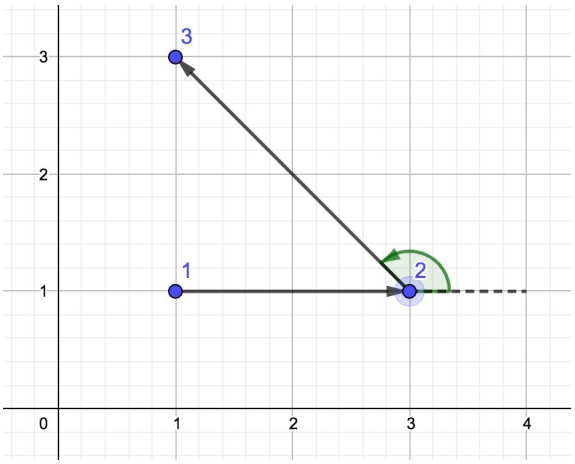
Если подходящей перестановки не существует выведите -1. Иначе выведите n чисел $p_1, p_2, \dots, p_n$ — найденную перестановку ($1 \leqslant p_i \leqslant n$ и все $p_1, p_2, \dots, p_n$ различны). Если подходящих перестановок несколько, выведите любую.

Примеры

стандартный ввод	стандартный вывод
3 1 1 3 1 1 3 L	1 2 3
6 1 0 0 1 0 2 -1 0 -1 -1 2 1 RLLR	5 4 1 6 2 3

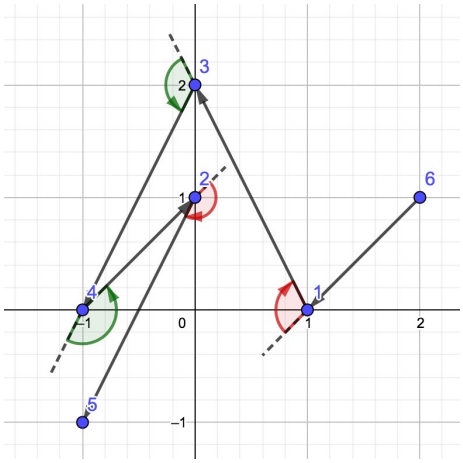
Замечание

Вот картинка, изображающая ломаную из 1 теста:



Как мы видим, эта ломаная несамопересекающаяся, а также извилистая, так как поворот в точке 2 налево.

Вот картинка, изображающая ломаную из 2 теста:


3
2
1
2
6
4
1
−1
0
1
2
5
−1

Задача J. Ферзи

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

На доске $n \times n$ расставьте, пожалуйста, n обычных шахматных ферзей так, чтобы они друг друга не били.

Формат входных данных

В единственной строке входных данных содержится одно целое число n — размер доски ($4 \leqslant n \leqslant 200$).

Формат выходных данных

Для каждой горизонтали исходной доски выведите номер вертикали, на которой стоит ферзь в этой горизонтали, вертикали нумеруются слева направо, начиная с единицы.

Пример

стандартный ввод	стандартный вывод
4	3 1 4 2

Задача K. Макс клика

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.25 секунд
Ограничение по памяти:	256 мегабайт

Макс, опытнейший игрок в доту, постоянно кликал.

Дан случайный неориентированный граф G из n вершин и m ребер.
Подкликой называется такое подмножествао вершин A: $\forall a, b \in A, a \neq b \quad \exists$ ребро (a, b).
Ваша задача — найти подклику A: $|A|$ максимально.

Формат входных данных

На первой строке число вершин $n \geqslant 1$ и число ребер $m \geqslant 1$.
Следующие m строк содержат пары чисел от 1 до n — ребра графа.
В графе нет ни петель, ни кратных ребер.

Формат выходных данных

На первой строке выведите k — количество вершин в максимальной подклике. На следующей строке k целых чисел от 1 до n — номера вершин в подклике. Вершины можно выводить в любом порядке. Если максимальных подклик несколько, выведите любую.

Система оценки

Подзадача 1 (10 баллов) $n \leqslant 20$.
Подзадача 2 (10 баллов) $n \leqslant 25$.
Подзадача 3 (20 баллов) $n \leqslant 40$.
Подзадача 4 (20 баллов) $n \leqslant 60$.
Подзадача 5 (20 баллов) $n \leqslant 70$.
Подзадача 6 (20 баллов) $n \leqslant 80$.

Пример

стандартный ввод	стандартный вывод
5 8 5 4 3 5 1 5 1 3 2 3 1 4 5 2 3 4	4 3 5 1 4