,

Задача A. Следующая перестановка

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

Найдите следующую перестановку. Лексикографически первая перестановка является следующей для обратной.

Формат входных данных

В первой строке входного файла записано число N ($1 \leqslant N \leqslant 100\,000$) — количество элементов в перестановке. Во второй строке записана перестановка из N чисел.

Формат выходных данных

В выходной файл вывести N чисел — искомую перестановку.

Примеры

nextperm.in	nextperm.out
3 3 2 1	1 2 3
2 1 2	2 1

,

Задача B. Разбиения на слагаемые

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

Перечислите все разбиения целого положительного числа N на целые положительные слагаемые. Разбиения должны обладать следующими свойствами:

- Слагаемые в разбиениях идут в невозрастающем порядке.
- Разбиения перечисляются в лексикографическом порядке.

Формат входных данных

Во входном файле находится единственное число N ($1 \leqslant N \leqslant 40$).

Формат выходных данных

В выходной файл выведите искомые разбиения по одному на строку.

Пример

partition.in	partition.out
4	1 1 1 1 2 1 1 2 2 3 1 4

Задача C. Без двух единиц подряд

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

По данному натуральному числу n выведите все двоичные последовательности длины n, не содержащие двух единиц подряд, в лексикографическом порядке.

Формат входных данных

Одно натуральное число n ($n \leqslant 20$).

Формат выходных данных

Каждая последовательность должна выводиться в отдельной строке, вывод должен завершаться символом новой строки. Числа, входящие в последовательность, должны быть разделены одним пробелом.

Пример

fibseq.in	fibseq.out
3	0 0 0 0 0 1 0 1 0 1 0 0 1 0 1

,

Задача D. Сочетания-2

Имя входного файла:	`comb2.in`
Имя выходного файла:	`comb2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

По данным натуральным k и n ($1 \leqslant k \leqslant n$) выведите все **убывающие** последовательности длины k состоящие из чисел $1 \dots n$ в лексикографическом порядке.

Формат входных данных

Во входном файле два числа — k и n ($1 \leqslant k \leqslant n \leqslant 1000$).

Формат выходных данных

Каждая последовательность должна выводиться в отдельной строке, вывод должен завершаться символом новой строки. Числа, входящие в последовательность, должны быть разделены одним пробелом. Гарантируется, что количество чисел в выходном файле не превосходит 500 000.

Пример

comb2.in	comb2.out
3 5	3 2 1 4 2 1 4 3 1 4 3 2 5 2 1 5 3 1 5 3 2 5 4 1 5 4 2 5 4 3

,

Задача E. Перестановка по номеру

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

Выведите перестановку по её номеру.

Формат входных данных

В первой строке входного файла записано число N ($1 \leqslant N \leqslant 12$) — количество элементов в перестановке. Во второй строке записано число K ($0 \leqslant K < N!$) — номер перестановки в нумерации с нуля.

Формат выходных данных

В выходной файл выведите N чисел через пробел — искомую перестановку.

Пример

bynumber.in	bynumber.out
3 0	1 2 3

,

Задача F. Генерация правильных скобочных последовательностей - 2

Имя входного файла:	`brackets2.in`
Имя выходного файла:	`brackets2.out`
Ограничение по времени:	6 секунд
Ограничение по памяти:	64 мегабайта

По данному числу n выведите все правильные скобочные последовательности из круглых и квадратных скобок длины n. Придерживайтесь следующего порядка скобок: "([])"(см. тест из условия)

Формат входных данных

Одно целое число n ($0 \leqslant n \leqslant 16$).

Формат выходных данных

Выведите все правильные скобочные последовательности из круглых и квадратных скобок длины n в лексикографическом порядке. Каждая послеждовательность должна выводиться в новой строке.

Пример

brackets2.in	brackets2.out
4	(()) ([]) ()() ()[] [()] [[]] []() [][]

Задача G. ПСП по номеру

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

Определим по индукции множество $\mathcal{R}$ *правильных скобочных последовательностей*:

- $\varepsilon \in \mathcal{R}$ (пустая строка)
- $A \in \mathcal{R} \Rightarrow (A) \in \mathcal{R}$
- $A \in \mathcal{R}$, $B \in \mathcal{R} \Rightarrow AB \in \mathcal{R}$

Пусть теперь $\mathcal{R}_n$ — это множество правильных скобочных последовательностей из $2n$ символов — n открывающих и n закрывающих скобок.

Упорядочим элементы множества $\mathcal{R}_n$ лексикографически с порядком символов '`(`' < '`)`'.

По данным числам n и p найдите p-ый в этом порядке элемент множества $\mathcal{R}_n$.

Формат входных данных

В первой строке входного файла заданы через пробел два целых числа n и p ($0 \leqslant n \leqslant 20$, $0 \leqslant p \leqslant 2 \cdot 10^9$). Скобочные последовательности нумеруются с нуля.

Формат выходных данных

В первой строке выходного файла выведите $2n$ символов без пробелов — p-ю правильную скобочную последовательность длины $2n$.

Если для данного n не существует p-я правильная скобочная последовательность, выведите в первой строке "`N/A`".

Примеры

parens.in	parens.out
3 0	((()))
4 2000000000	N/A
3 4	()()()

Задача H. Номер правильной последовательности

Имя входного файла:	`brackets2num2.in`
Имя выходного файла:	`brackets2num2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Во входном файле задана правильная скобочная последовательность с двумя типами скобок. Выведите в выходной ее номер в лексикографическом порядке среди всех правильных скобочных последовательностей с таким же количеством открывающихся скобок, '(', ')', '[', ']'. Последовательности занумерованы, начиная с 0. Количество открывающихся скобок в последовательности — от 1 до 20.

Формат входных данных

Во входном файле записана строка из круглых и квадратных скобок. Длина строки не превосходит 20 символов.

Формат выходных данных

Выведите одно число — ответ на задачу.

Примеры

brackets2num2.in	brackets2num2.out
`([])()[]`	100
`()[]`	2
`[[[()[]([])]]]`	266079

,

Задача I. Монетки

Имя входного файла:	`coins.in`
Имя выходного файла:	`coins.out`
Ограничение по времени:	2.2 секунд
Ограничение по памяти:	64 мегабайта

64 мегабайта

В Волшебной стране используются монетки достоинством $A_1, A_2, \ldots, A_M$. Волшебный человечек пришел в магазин и обнаружил, что у него есть ровно по две монетки каждого достоинства. Ему нужно заплатить сумму N. Напишите программу, определяющую, сможет ли он расплатиться без сдачи.

Формат входных данных

Сначала вводится целое число N ($1 \leqslant N \leqslant 10^9$), затем — целое число M ($1 \leqslant M \leqslant 10$) и далее M попарно различных целых чисел $A_1, A_2, \ldots, A_M$ ($1 \leqslant A_i \leqslant 10^9$).

Формат выходных данных

Выведите сначала K — количество монет, которое придется отдать Волшебному человечку, если он сможет заплатить указанную сумму без сдачи. Далее выведите K чисел, задающих достоинства монет. Если решений несколько, выведите вариант, в котором Волшебный человек отдаст наименьшее возможное количество монет. Если таких вариантов несколько, выведите любой из них.

Если без сдачи не обойтись, то выведите одно число 0. Если же у Волшебного человечка не хватит денег, чтобы заплатить указанную сумму, выведите одно число –1 (минус один).

Примеры

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

,

Задача J. Новогодняя гирлянда

Имя входного файла:	`garland.in`
Имя выходного файла:	`garland.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

256 мегабайт

Дети в детском саду как-то раз решили повесить к Новому году гирлянду. Но это оказалось для них очень трудной задачей. На помощь пришёл Дед Мороз, который теперь каждый Новый год приносит с собой гирлянду и помогает её повесить.

Гирлянда представляет собой ломаную в плоскости, состоящую из n звеньев. Гирлянда начинается в точке $(0, 0)$, возле электророзетки и должна заканчиваться в точке $(n, 0)$. Число n называется длиной гирлянды. Каждое звено может располагаться либо горизонтально, либо под углом 45° к оси OX. Длина горизонтальной проекции любого звена равна 1. При этом не должно быть вершины ломаной с отрицательной координатой y, а также двух последовательных вершин с нулевой координатой y. Поднимающимся (опускающимся) назовём звено ломаной, у которого координата у правого конца больше (соответственно, меньше) координаты у левого конца. Звено, у которого координаты у концов совпадают, назовём горизонтальным. Обозначим поднимающееся звено буквой `u`, опускающееся — буквой `d`, а горизонтальное — буквой `h`. Тогда гирлянда кодируется строкой из n символов. У Деда Мороза есть волшебная книга, в которой перечислены все гирлянды длины n в виде строк. Хотя книга и волшебная, строки в ней располагаются в обычном лексикографическом порядке, по возрастанию. Дед Мороз отметил на полях книги галочкой гирлянду, которую повесил в прошлый раз. В этот Новый год он хочет повесить следующую в книге гирлянду. Найдите эту гирлянду без использования волшебной книги.

Формат входных данных

В первой строке вводится целое число n ($2 \leqslant n \leqslant 100\,000$). Во второй — строчка из n букв (все буквы: `u`, `d`, либо `h`) — прошлогодняя гирлянда.

Формат выходных данных

Выведите в виде строки гирлянду, которую Дед Мороз Павлович должен прихватить с собой в этот Новый год, либо `No solution`, если такой гирлянды не существует.

Пример

garland.in	garland.out
6 uhduhd	uhhdud