Задача A. Номер по перестановке

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

Дана перестановка из N чисел от 1 до N. Требуется найти её номер в лексикографическом порядке.

Формат входных данных

Во входном файле сначала записано число N ($1 \leqslant N \leqslant 12$). В следующей строке записана сама перестановка — N чисел, разделённых пробелами.

Формат выходных данных

В выходной файл нужно вывести единственное число — номер перестановки в лексикографическом порядке.

Примеры

perm.in	perm.out
3 2 1 3	3

Задача B. Предыдущая перестановка

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

Найдите предыдущую в лексикографическом порядке перестановку. Перестановка вида $N, N-1, ..., 3, 2, 1$ является предыдущей для $1, 2, 3, ..., N-1, N$

Формат входных данных

В первой строке входного файла записано число N ($1 \leqslant N \leqslant 10^5$) количество элементов в перестановке. Во второй строке записана перестановка.

Формат выходных данных

В выходной файл вывести N чисел — искомую перестановку.

Примеры

prev.in	prev.out
3 1 2 3	3 2 1

Задача C. Конфеты Кирилла

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

У Кирилла из параллели С было k конфет, и он захотел их раздать ученикам своей параллели. Однако заметил, что конфет у него меньше чем учеников в параллели. Кирилл сел на скамейку и задумался. Просидев полчаса и доев последнюю конфету он подумал — интересно, а сколько было способов раздать все k конфет n ученикам параллели С, если конфеты нельзя делить, а каждому школьнику можно дать не более одной конфеты.

Формат входных данных

В единственной строке записаны числа n, $k (1 \leqslant k \leqslant n \leqslant 64)$.

Формат выходных данных

Выведите единственное число — ответ на задачу.

Примеры

combination.in	combination.out
5 3	10

Задача D. Восстановление

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

Дарья обнаружила ошибку в своей программе, которая удаляет все символы из строки кроме “(” и “)”. Оказывается, некоторые символы заменяются на что-то нечитаемое.

Теперь её заинтересовал вопрос, сколько различных правильных скобочных последовательностей длины $2n$ могут являться результатом исправленного алгоритма, то есть не будут противоречить данным, которые она таки не потеряла.

Формат входных данных

Единственная строка входного файла содержит строку из круглых скобок и знаков вопроса, где вопросами обозначены утраченные символы. Длина строки не превосходит 10000.

Формат выходных данных

Выведите одно число — количество различных скобочных последовательностей, удовлетворяющих шаблону Дарьи, по модулю $10^9 + 7$.

Примеры

recover.in	recover.out
(??()?	2

Замечание

Вместо `python3` при сдаче этой задачи используйте `pypy3`.

Задача E. Следующее сочетание

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

Дано множество целых чисел от 1 до N. Рассмотрим подмножество этого множества, состоящее из K элементов, в возрастающем порядке.

Выведите следующее в лексикографическом порядке подмножество из K элементов.

Формат входных данных

В первой строке входного файла содержатся целые положительные числа N и K ($1 \leqslant K \leqslant N \leqslant 50$). Во второй строке содержится K целых чисел от 1 до N в возрастающем порядке — подмножество из K элементов.

Формат выходных данных

Выведите следующее в лексикографическом порядке после данного подмножество из K элементов. Если следующего подмножества нет, выведите 0.

Примеры

nextcomb.in	nextcomb.out
6 4 1 4 5 6	2 3 4 5
6 2 5 6	0

Задача F. 30 кресел

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

Потерпев фиаско в погоне за стульями Остап не пал духом, а ввязался в новую авантюру. Услышав, что неподалёку на аукционе распродаются n старинных кресел, не долго думая он решил попытать удачу и проверить: не скрываются ли сокровища в одном из них. Придя на торги, Остап понял, что денег у него хватит на выкуп ровно k из n кресел. Своим самым счастливым числом Остап считает число m, поэтому он снова обращается к Вам за помощью и просит выбрать m-е сочетание k из n кресел.

Формат входных данных

Во входном файле заданы числа n, k и m. $1 \leqslant k \leqslant n \leqslant 30$, $0 \leqslant m \leqslant \binom{n}{k} - 1$.

Формат выходных данных

Выведите в выходной файл в возрастающем порядке номера кресел, входящие в m-е в лексикографическом порядке сочетание по k из чисел от 1 до n. Сочетания занумерованы, начиная с 0.

Примеры

choose.in	choose.out
4 2 3	2 3

Задача G. Лексикографический порядок

Имя входного файла:	`lexsort.in`
Имя выходного файла:	`lexsort.out`
Ограничение по времени:	0.5 секунда
Ограничение по памяти:	64 мегабайта

Будем считать, что одно натуральное число лексикографически меньше другого, если таковы их записи в десятичной системе счисления. Вам необходимо найти k-е по порядку число в лексикографически отсортированном множестве натуральных чисел от 1 до n включительно.

Формат входных данных

Первая строка входного файла содержит два числа n и k ($1 \leqslant k \leqslant n \leqslant 10^9$).

Формат выходных данных

Выведите единственное число — k-й в лексикографическом порядке элемент множества натуральных чисел от 1 до n.

Примеры

lexsort.in	lexsort.out
10 2	10

Задача H. Следующее разбиение на слагаемые

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

Формат входных данных

Входной файл содержит одну строку — разбиение числа n на слагаемые ($1 \leqslant n \leqslant 100\,000$). Слагаемые в разбиении следуют в неубывающем порядке.

Формат выходных данных

Выведите в выходной файл одну строку — разбиение числа n на слагаемые, следующее в лексикографическом порядке после приведенного во входном файле. Если во входном файле приведено последнее разбиение n на слагаемые, выведите `No solution`.

Примеры

nextpartition.in	nextpartition.out
5=1+1+3	5=1+2+2
5=5	No solution