Задача A. Произведение матриц

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

В произведении последовательности матриц полностью расставлены скобки, если выполняется один из следующих пунктов:

- Произведение состоит из одной матрицы.
- Оно является заключенным в скобки произведением двух произведений с полностью расставленными скобками.

Полная расстановка скобок называется оптимальной, если количество операций, требуемых для вычисления произведения, минимально.

Требуется найти оптимальную расстановку скобок в произведении последовательности матриц.

Формат входных данных

В первой строке входных данных содержится целое число n — количество матриц ($1 \leqslant n \leqslant 400$). В n следующих строк содержится по два целых числа a_i и b_i — количество строк и столбцов в i-ой матрице соответственно ($1 \leqslant a_i, b_i \leqslant 100$). Гарантируется, что $b_i = a_{i+1}$ для любого $1 \leqslant i \leqslant n - 1$

Формат выходных данных

В выходной файл выведите оптимальную расстановку скобок. Если таких расстановок несколько, выведите любую.

Пример

matrix.in	matrix.out
3 10 50 50 90 90 20	((AA)A)

Замечание

В данном примере возможно две расстановки скобок: $((AA)A)$ и $(A(AA))$. При первой количество операций будет равно $10 \cdot 50 \cdot 90 + 10 \cdot 90 \cdot 20 = 63000$, а при второй — $10 \cdot 50 \cdot 20 + 50 \cdot 90 \cdot 20 = 100000$.

Задача B. Наибольшая возрастающая подпоследовательность

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

Дана последовательность, требуется найти её наибольшую возрастающую подпоследовательность.

Формат входных данных

В первой строке входных данных задано целое число N — длина последовательности ($1 \leqslant N \leqslant 300\,000$).

Во второй строке задается сама последовательность. Числа разделяются пробелом.

Элементы последовательности — целые числа, не превосходящие 10^9 по модулю.

Подзадача 1: $N \leqslant 5000$

Подзадача 2: $N \leqslant 300\,000$

Формат выходных данных

В первой строке выведите длину наибольшей возрастающей подпоследовательности, а во второй строке выведите через пробел саму наибольшую возрастающую подпоследовательность данной последовательности. Если ответов несколько — выведите любой.

Пример

lis.in	lis.out
6 3 29 5 5 28 6	3 3 5 28

Задача C. Оптимальный префиксный код с сохранением порядка

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

Дан алфавит Σ, где каждому символу c_i сопоставим его код p_i. Кодирование называется оптимальным префиксным с сохранением порядка, если соблюдаются:

1. $\forall i, j : c_i < c_j \iff p_i < p_j$.
2. $\nexists i, j : i \neq j, p_i$ — префикс p_j.
3. $\sum_{i=1}^{|\Sigma|} f_i \cdot |p_i|$ — минимально, где f_i — частота встречаемости символа c_i в тексте, а $|p_i|$ — длина его кода.

В алфавите n символов. i-й в лексикографическом порядке символ встречается в тексте f_i раз. Постройте оптимальный префиксный код с сохранением порядка.

Формат входных данных

В первой строке дано целое число n ($1 \leqslant n \leqslant 2000$) — количество символов в алфавите. Далее в n строках записаны целые числа f_i ($1 \leqslant f_i \leqslant 10^6$) обозначающие количество раз, которое встречается соответствующий символ в тексте.

Подзадача 1: $n \leqslant 300$

Подзадача 2: $n \leqslant 2000$

Формат выходных данных

В первой строке выведите значение $\sum_{i=1}^{|\Sigma|} f_i \cdot |p_i|$. В $i+1$-й строке выведите код, который соответствует i-у символу. Если существует несколько решений, выведите любое.

Примеры

optimalcode.in	optimalcode.out
3 10000 1 1	10004 0 10 11
5 15 1 2 3 4	44 0 1000 1001 101 11

Задача D. Рюкзак

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

Дано n предметов массой $m_1, \ldots, m_n$ и стоимостью $c_1, \ldots, c_n$ соответственно.

Ими наполняют рюкзак, который выдерживает вес не более m. Определите набор предметов, который можно унести в рюкзаке, имеющий наибольшую стоимость.

Формат входных данных

В первой строке вводится натуральное число n, не превышающее 1000 и натуральное число m, не превышающее 10000.

Во второй строке вводятся n натуральных чисел m_i, не превышающих 100.

Во третьей строке вводятся n натуральных чисел $_i$, не превышающих 100.

Формат выходных данных

В первой строке выведите количество предметов, которые нужно взять. Во второй строке выведите номера предметов (числа от 1 до n), которые войдут в рюкзак наибольшей стоимости.

Пример

knapsack.in	knapsack.out
4 6 2 4 1 2 7 2 5 1	3 1 3 4

Задача E. Расстояние Левенштейна

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

Дана текстовая строка. С ней можно выполнять следующие операции:

- Заменить один символ строки на другой символ.
- Удалить один произвольный символ.
- Вставить произвольный символ в произвольное место строки.

Например, при помощи первой операции из строки «СОК» можно получить строку «СУК», при помощи второй операции — строку «ОК», при помощи третьей операции — строку «СТОК». Минимальное количество таких операций, при помощи которых можно из одной строки получить другую, называется стоимостью редактирования или расстоянием Левенштейна. Определите расстояние Левенштейна для двух данных строк.

Формат входных данных

Программа получает на вход две строки, длина каждой из которых не превосходит 5000 символов, строки состоят только из заглавных латинских букв.

Формат выходных данных

Требуется вывести одно число — расстояние Левенштейна для данных строк.

Пример

levenshtein.in	levenshtein.out
ABCDEFGH ACDEXGIH	3

Задача F. Шаблоны

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

Многие операционные системы используют шаблоны для ссылки на группы объектов: файлов, пользователей, и т. д. Ваша задача — реализовать простейший алгоритм проверки шаблонов для имен файлов.

В этой задаче алфавит состоит из маленьких букв английского алфавита и точки (‘.’). Шаблоны могут содержать произвольные символы алфавита, а также два специальных символа: ‘?’ и ‘*’. Знак вопроса (‘?’) соответствует ровно одному произвольному символу. Звездочка ‘*’ соответствует подстроке произвольной длины (возможно, нулевой). Символы алфавита, встречающиеся в шаблоне, отображаются на ровно один такой же символ в проверяемой строчке. Строка считается подходящей под шаблон, если символы шаблона можно последовательно отобразить на символы строки таким образом, как описано выше. Например, строчки “ab”, “aab” и “beda.” подходят под шаблон “*a?”, а строчки “bebe”, “a” и “ba” — нет.

Формат входных данных

Первая строка входного файла определяет шаблон P. Вторая строка S состоит только из символов алфавита. Ее необходимо проверить на соответствие шаблону. Длины обеих строк не превосходят 10 000. Строки могут быть пустыми — будьте внимательны!

Формат выходных данных

Если данная строка подходит под шаблон, выведите `YES`. Иначе выведите `NO`.

Примеры

стандартный ввод	стандартный вывод
k?t*n kitten	YES
k?t?n kitten	NO

Задача G. Деловые встречи

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

Алексей — успешный предприниматель, и в течение одного дня у него бывает много встреч с разными деловыми партнёрами. К сожалению, встречи бывают разные и не все приносят ему радость, после других же настроение улучшается. Также, на многие встречи не стоит приходить в слишком плохом или хорошем настроении — результат таких встреч может быть не таким, какой хочется Алексею.

К счастью, недавно Алексей научился оценивать своё настроение с помощью целых чисел. После этого для каждой встречи он оценил, при каком максимальном и минимальном настроении стоит на неё приходить, а также как изменится его настроение после этой встречи. Теперь он хочет распланировать порядок встреч так, чтобы в течение дня совершить максимальное число встреч.

Ваша задача — написать программу, которая по информации о всех встречах и настроении Алексея в начале дня находит порядок проведения встреч такой, что их количество при этом максимально.

Формат входных данных

Первая строка входного файла содержит два целых числа n и k ($1 \leqslant n \leqslant 20$, $-100 \leqslant k \leqslant 100$) — количество встреч и настроение Алексея в начале дня.

Следующие n строк содержат по три целых числа a_i, b_i и c_i ($-100 \leqslant a_i, b_i, c_i \leqslant 100$) — минимальное и максимальное настроение, при котором встреча возможна, и изменение настроения по окончании встречи, соответственно.

Формат выходных данных

В первой строке выходного файла выведите число m — максимально возможно число встреч. В следующей строке выведите m целых чисел — номера встреч в порядке их проведения. Встречи пронумерованы в порядке описания во входном файле.

Если ответов с максимальным числом встреч несколько, выведите любой.

Примеры

meetings.in	meetings.out
3 0 1 3 3 0 1 2 1 3 1	3 2 3 1
3 1 -10 -5 3 -5 5 -2 -3 2 1	2 2 3

Задача H. Перестановки

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

Задано множество из n различных натуральных чисел. Перестановку элементов этого множества назовем *k-перестановкой*, если для любых двух соседних элементов этой перестановки их наибольший общий делитель не менее k. Например, если задано множество элементов $S = \{6, 3, 9, 8\}$, то перестановка $\{8, 6, 3, 9\}$ является *2-перестановкой*, а перестановка $\{6, 8, 3, 9\}$ — нет.

Перестановка $\{p_1, p_2, \ldots, p_n\}$ будет *лексикографически меньше* перестановки $\{q_1, q_2, \ldots, q_n\}$, если существует такое натуральное число i $(1 \leqslant i \leqslant n)$, для которого $p_j = q_j$ при $j < i$ и $p_i < q_i$.

В качестве примера упорядочим все *k-перестановки* заданного выше множества в лексикографическом порядке. Например, существует ровно четыре *2-перестановки* множества S: $\{3, 9, 6, 8\}$, $\{8, 6, 3, 9\}$, $\{8, 6, 9, 3\}$ и $\{9, 3, 6, 8\}$. Соответственно, первой *2-перестановкой* в лексикографическом порядке является множество $\{3, 9, 6, 8\}$, а четвертой — множество $\{9, 3, 6, 8\}$.

Требуется написать программу, позволяющую найти m-ую *k-перестановку* в этом порядке.

Формат входных данных

Входной файл в первой строке содержит три натуральных числа — n $(1 \leqslant n \leqslant 16)$, m и k $(1 \leqslant m, k \leqslant 10^9)$. Вторая строка содержит n различных натуральных чисел, не превосходящих 10^9. Все числа в строках разделены пробелом.

Формат выходных данных

В выходной файл необходимо вывести m-ую *k-перестановку* заданного множества или –1, если такой нет.

Примеры

perm.in	perm.out
4 1 2 6 8 3 9	3 9 6 8
4 4 2 6 8 3 9	9 3 6 8
4 5 2 6 8 3 9	-1

Задача I. Налог на проезд

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

Байтландия состоит из n городов, которые соединены $n-1$ двусторонними дорогами. Известно, что между любыми двумя городами существует путь по дорогам. Президенту байтландии не хватает ровно m байтландских денежных единиц для выполнения всех своих предвыборных обещаний. Чтобы собрать нужную сумму, он решил ввести налог на проезд по дорогам.

После работы специальной комиссии, для каждой дороги было определено минимальное и максимальное количество денег, которые готовы платить жители Байтландии, чтобы воспользоваться этой дорогой. После опроса также было выяснено, что из каждого города Байтландии в каждый другой город Байтландии в этом году собирается поехать ровно один человек.

Каждый житель, который едет из одного города в другой, всегда выбирает самый кратчайший маршрут и двигается по нему. При проезде по дороге житель платит налог, который был определен президентом.

Президента Байтландии очень волнует вопрос, сколько различных способов назначить налоги на проезд по дорогам, чтобы сумма доходов была ровно m. Два способа считаются различными, если существует дорога, для которой различается налог на проезд в этих способах. Выведите ответ по модулю 10^9+7.

Формат входных данных

Первая строка содержит два целых числа n и m ($1 \leqslant n, m \leqslant 5 \cdot 10^5$) количество городов Байтландии и необходимая сумма денег.

В следующих $n-1$ строках записано по четыре числа $a_i, b_i, l_i, r_i (1 \leqslant a_i, b_i \leqslant n; 1 \leqslant l_i, r_i \leqslant 5 \cdot 10^5)$, обозначающие, что между городами a_i и b_i существует дорога на которую можно ввести налог в размере от l_i до r_i денежных единиц включительно.

Формат выходных данных

Выведите единственное число ответ на задачу по модулю 10^9+7.

Пример

стандартный ввод	стандартный вывод
6 152 1 2 3 4 2 3 1 2 2 4 3 5 4 5 1 1 4 6 2 2	2

Задача J. Общая подпоследовательность

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	6 секунд
Ограничение по памяти:	16 мегабайт

Даны две строки, состоящих из маленьких латинских букв. Нужно найти их наибольшую общую подпоследовательность.

Формат входных данных

На первой строке первая строка.
На второй строке вторая строка.
Длины строк от 1 до 10^4.

Формат выходных данных

Максимальную по длине общую подпоследовательность на отдельной строке. Если ответов несколько, выведите любой. Если ответ пуст, перевод строки выводить все равно нужно.

Пример

стандартный ввод	стандартный вывод
abacabadabacaba dbdccdbd	bcbd

Задача K. Подпалиндромы

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

Задана строка из алфавита размера 10^9. Найдите число подпоследовательностей, которые являются палиндромами.

Формат входных данных

Первая строка входного файла содержит число n — размер массива ($1 \leqslant n \leqslant 2000$). Вторая строка содержит n целых чисел, каждое из которых задает элемент массива и находится пределах от 1 до 10^9.

Формат выходных данных

Выведите в выходной файл одно число — искомое количество подпоследовательностей по модулю 10^9.

Примеры

стандартный ввод	стандартный вывод
4 1 2 3 1	7

Задача L. Паросочетание максимального веса в дереве

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

Паросочетанием называется такое подмножество рёбер графа, в котором у любых двух рёбер нет общей вершины. Дан взвешенный неориентированный граф, в котором между каждой парой вершин существует ровно один простой путь. Требуется в заданном графе найти паросочетание максимального веса.

Формат входных данных

В первой строке входных данных содержится число n — количество вершин в графе ($2 \leqslant n \leqslant 10^5$). Следующие $n - 1$ строк содержат описание ребер. Каждое ребро задаётся стартовой вершиной, конечной вершиной и весом ребра w_i ($1 \leqslant w_i \leqslant 10^9$).

Формат выходных данных

Требуется вывести одно число — вес максимального паросочетания.

Пример

matching.in	matching.out
4 1 2 10 1 3 1 3 4 1	11

Задача M. Задача коммивояжера

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

Вам дан неориентированный взвешенный граф без петель и кратных ребер. Необходимо найти в нем путь наименьшего веса, который проходит по всем вершинам ровно один раз.

Формат входных данных

В первой строке находятся два целых числа n и m — количество вершин и ребер в графе ($1 \leqslant n \leqslant 18$, $0 \leqslant m \leqslant \frac{n \cdot (n-1)}{2}$). Следующие m строк содержат описания ребер: три целых числа a_i, b_i, w_i, обозначающих соответственно пару вершин и вес ребра, соединяющего эти вершины ($1 \leqslant a_i, b_i \leqslant n$, $1 \leqslant w_i \leqslant 10^8$).

Формат выходных данных

Выведите в выходной файл одно число — вес искомого пути. Если такого пути не существует, выведите -1.

Примеры

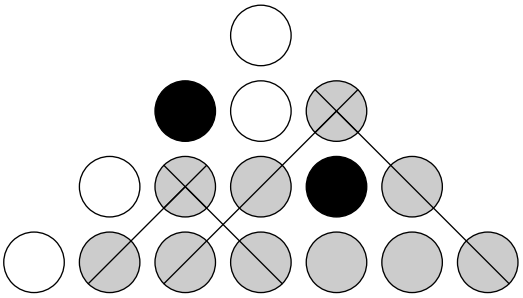
salesman.in	salesman.out
4 6 1 2 20 1 3 42 1 4 35 2 3 30 2 4 34 3 4 12	62
4 3 1 2 1 1 3 1 1 4 1	-1

Задача N. Съёмка

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

До финального матча чемпионата Берляндии остались считанные дни! Но, так как всё делается в последний момент, некоторые организационные моменты ещё не улажены. В том числе нужно организовать съёмку матча. В городе есть несколько компаний, которые готовы организовать съёмку. Но они все конкурируют, и хотят снимать матч, а не операторов конкурентных компаний!

Операторов можно разместить на трибунах стадиона. Трибуны состоят из n рядов: в первом $2n-1$ место, во втором — $2n-3$, ..., в n-м — одно. Однако, не все подходят к матчу безответственно, и некоторые болельщики уже купили себе билеты на определённые места. Соответственно, операторов можно поставить только на свободные.



Пример стадиона. Серым отмечены места, которые снимаются операторами. Чёрным — занятые зрителями места. Операторы отмечены крестиками. Для каждого из них показано, что снимает его камера.

Но конкуренция есть не везде, и все операторы используют камеры одной и той же модели, неизменяемый угол обзора которой позволяет снимать три места в следующем ряду, пять — через ряд, итд. Власти хотят, чтобы всё прошло на высшем уровне, и попросили предоставить им возможные способы расставить операторов, чтобы самим выбрать наиболее оптимальный. Однако, начальство стадиона не считает это хорошей идеей, полагая, что таких способов очень много. Помогите им выяснить, сколько же существует способов.

Формат входных данных

В первой строке заданы числа n и m ($1 \leqslant n \leqslant 2000$, $0 \leqslant m \leqslant 1000$) — число рядов и количество занятых мест.

Далее в m строках заданы занятые места по одному в строке. Место задается двумя числами: r и p ($1 \leqslant r \leqslant n$, $1 \leqslant p \leqslant 2(n-r)+1$) — номер ряда и номер места в этом ряду.

Формат выходных данных

Выведите остаток от деления количества способов на 10^9+7.

Примеры

operators.in	operators.out
2 1 1 2	5

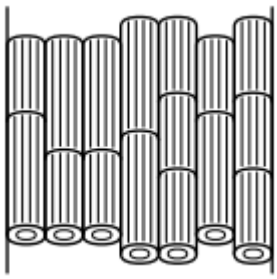
Задача O. Мостостроение

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

В деревне Гадюкино регулярно идут проливные дожди, в результате чего речка Вонючка, которую обычно можно просто перешагнуть, выходит из берегов. Чтобы можно было перейти разлившуюся реку, планируется построить плавучий мост из бревен, оставшихся от строительства бани бизнесмена, поселившегося неподалеку.

Все оставшиеся бревна имеют одинаковую толщину. При этом есть x бревен длины a и y бревен длины b.

Построенный мост должен состоять из l рядов, каждый из которых составлен из одного или нескольких бревен. Пилить бревна нельзя, так как последняя пила утонула при разливе Вонючки.



Главный инженер хочет построить мост максимальной возможной ширины, при этом ширина моста определяется по минимальной ширине ряда бревен.

Например, если нужно построить мост из семи рядов, и при этом есть шесть бревен длины 3 и десять бревен длины 2, то можно построить мост ширины 5.

Формат входных данных

Входной файл содержит пять натуральных чисел: x, a, y, b и l. Все числа не превышают 150. Общее количество бревен не меньше l.

Формат выходных данных

Выведите в выходной файл одно число — максимальную возможную ширину моста.

Примеры

bridge.in	bridge.out
6 3 10 2 7	5
10 7 20 9 25	9

Задача P. Симпатичные узоры 2

Имя входного файла:	`nice2.in`
Имя выходного файла:	`nice2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Компания *BrokenTiles* планирует заняться выкладыванием во дворах у состоятельных клиентов узор из черных и белых плиток, каждая из которых имеет размер 1×1 метр. Известно, что дворы всех состоятельных людей имеют наиболее модную на сегодня форму прямоугольника $n \times m$ метров.

Однако при составлении финансового плана у директора этой организации появилось целых две серьезных проблемы: во первых, каждый новый клиент очевидно захочет, чтобы узор, выложенный у него во дворе, отличался от узоров всех остальных клиентов этой фирмы, а во вторых, этот узор должен быть симпатичным.

Как показало исследование, узор является симпатичным, если в нем нигде не встречается квадрата 2×2 метра, полностью покрытого плитками одного цвета.

Для составления финансового плана директору необходимо узнать, сколько клиентов он сможет обслужить, прежде чем симпатичные узоры данного размера закончатся. Помогите ему!

Формат входных данных

На первой строке входного файла находятся два натуральных числа n и m. $1 \leqslant n \cdot m \leqslant 300$.

Подзадача 1: $n \cdot m \leqslant 30$

Подзадача 2: $n \cdot m \leqslant 300$

Формат выходных данных

Выведите в выходной файл единственное число — количество различных симпатичных узоров, которые можно выложить во дворе размера $n \times m$ по модулю $2^{30} + 1$. Узоры, получающиеся друг из друга сдвигом, поворотом или отражением считаются различными.

Пример

nice2.in	nice2.out
2 2	14
3 3	322

Задача Q. Симпатичные узоры 3

Имя входного файла:	`nice3.in`
Имя выходного файла:	`nice3.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Компания *BrokenTiles* планирует заняться выкладыванием во дворах у состоятельных клиентов узор из черных и белых плиток, каждая из которых имеет размер 1×1 метр. Известно, что дворы всех состоятельных людей имеют наиболее модную на сегодня форму прямоугольника $n \times m$ метров.

Однако при составлении финансового плана у директора этой организации появилось целых две серьезных проблемы: во первых, каждый новый клиент очевидно захочет, чтобы узор, выложенный у него во дворе, отличался от узоров всех остальных клиентов этой фирмы, а во вторых, этот узор должен быть симпатичным.

Как показало исследование, узор является симпатичным, если в нем нигде не встречается квадрата 2×2 метра, полностью покрытого плитками одного цвета.

Для составления финансового плана директору необходимо узнать, сколько клиентов он сможет обслужить, прежде чем симпатичные узоры данного размера закончатся. Помогите ему!

Формат входных данных

На первой строке входного файла находятся три натуральных числа n, m, mod. ($1 \leqslant n \leqslant 10^{100}, 1 \leqslant m \leqslant 6, 1 \leqslant mod \leqslant 10\,000$.

n, m — размеры двора. mod — модуль, по которому нужно посчитать ответ.

Формат выходных данных

Выведите в выходной файл единственное число — количество различных симпатичных узоров, которые можно выложить во дворе размера $n \times m$ по модулю mod. Узоры, получающиеся друг из друга сдвигом, поворотом или отражением считаются различными.

Пример

nice3.in	nice3.out
2 2 5	4
3 3 23	0

Задача R. Интересное число

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

Для заданного числа n найдите наименьшее положительное целое число с суммой цифр n, которое делится на n.

Формат входных данных

Во входном файле содержатся целое число n ($1 \leqslant n \leqslant 1000$).

Формат выходных данных

Выходной файл должен содержать искомое число. Ведущие нули выводить не разрешается.

Пример

number.in	number.out
1	1
10	190

Задача S. ABCDE-Последовательности

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

International Biology Manufacturer (IBM) обнаружили, что органический материал на Марсе имеет ДНК, состоящий из 5 символов(a, b, c, d, e), вместо четырех компонентов ДНК на Земле. Четыре пары cd, ce, ed, и ee никогда не идут подряд в строке Марсианских ДНК.

IBM заинтересовались сколько правильных Марсианских строк ДНК длины n возможно составить?

Формат входных данных

Входные данные содержат несколько тестов. Каждый тест содержит одно число n на отдельной строке. Последняя строка входных файлов содержит ноль.

- Количество тестов не превосходит 100.
- Число n лежит в пределах от 1 до 2^{50} включительно.

Формат выходных данных

Для каждого теста выведите на отдельной строке количество правильных строк по модулю 999 999 937.

Пример

sequences.in	sequences.out
1 2 0	5 21

Задача T. Простые пути в дереве

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

Дан неориентированный связный граф из n вершин и $n-1$ ребер. Требуется для каждого ребра посчитать суммарную длину простых путей, проходящих через это ребро. Длиной пути здесь называется количество ребер в пути.

Формат входных данных

На первой строке целое число n ($2 \leqslant n \leqslant 300\,000$). Следующие $n-1$ строк содержат пары чисел от 1 до n — ребра графа.

Формат выходных данных

$n-1$ строк. i-я строка должна содержать целое число — ответ для i-го ребра.

Пример

treedp.in	treedp.out
5 1 2 2 3 2 4 5 1	13 8 8 9