Задача A. НВП

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

Числовая последовательность задана рекуррентной формулой: $a_{i+1} = (ka_i + b) \bmod m$. Найдите её наибольшую возрастающую подпоследовательность.

Формат входных данных

Программа получает на вход пять целых чисел: длину последовательности n ($1 \leqslant n \leqslant 10^5$), начальный элемент последовательности a_1, параметры k, b, m для вычисления последующих членов последовательности ($1 \leqslant m \leqslant 10^4$, $0 \leqslant k < m$, $0 \leqslant b < m$, $0 \leqslant a_1 < m$).

Формат выходных данных

Требуется вывести длину НВП.

Примеры

lis.in	lis.out
5 41 2 1 100	3

Замечание

В данном примере последовательность состоит из 5 элементов: $a_1 = 41$, $a_{i+1} = (2a_i + 1) \bmod 100$, то есть последовательность имеет вид 41, 83, 67, 35, 71.

Задача B. Кратчайший путь

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

Дан ориентированный взвешенный ациклический граф. Требуется найти в нем кратчайший путь из вершины s в вершину t.

Формат входных данных

Первая строка входного файла содержит четыре целых числа n, m, s и t — количество вершин, дуг графа, начальная и конечная вершина соответственно. Следующие m строк содержат описания дуг по одной на строке. Ребро номер i описывается тремя целыми числами b_i, e_i и w_i — началом, концом и длиной дуги соответственно ($1 \leqslant b_i, e_i \leqslant n$, $|w_i| \leqslant 1000$).

Входной граф не содержит циклов и петель.

$1 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 200\,000$, $1 \leqslant s, t \leqslant n$.

Формат выходных данных

Первая строка выходного файла должна содержать одно целое число — длину кратчайшего пути из s в t. Если пути из s в t не существует, выведите `Unreachable`.

Примеры

dag-shortpath.in	dag-shortpath.out
2 1 1 2 1 2 -10	-10
2 1 2 1 1 2 -10	Unreachable

Задача C. Распил брусьев

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вам нужно распилить деревянный брус на несколько кусков в заданных местах. Распилочная компания берет k рублей за распил одного бруска длиной k метров на две части.

Понятно, что различные способы распила приводят к различной суммарной стоимости заказа. Например, рассмотрим брус длиной 10 метров, который нужно распилить на расстоянии 2, 4 и 7 м, считая от одного конца. Это можно сделать несколькими способами. Можно распилить сначала на отметке 2 м, потом 4 и, наконец, 7 м. Это приведет к стоимости $10+8+6=24$, потому что сначала длина бруса, который пилили, была 10 м, затем она стала 8 м, и, наконец, 6 м. А можно распилить иначе: сначала на отметке 4 м, затем 2, затем 7м. Это приведет к стоимости $10+4+6=20$, что лучше.

Определите минимальную стоимость распила бруса на заданные части.

Формат входных данных

Первая строка входных данных содержит целое число L ($2 \leqslant L \leqslant 10^6$) - длину бруса и целое число N ($1 \leqslant N \leqslant 100$) - количество распилов. Во второй строке записано N целых чисел C_i ($0 < C_i < L$) в строго возрастающем порядке - места, в которых нужно сделать распилы.

Формат выходных данных

Выведите одно натуральное число - минимальную стоимость распила.

Примеры

stdin	stdout
10 3 2 4 7	20
100 3 15 50 75	200

Задача D. Игра

Имя входного файла:	`game.in`
Имя выходного файла:	`game.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

На листке записано в одну строку N целых положительных чисел. Каждое число не превышает 200. Играют двое. За каждый ход можно зачеркивать крайнее число либо слева, либо справа. Зачеркнутое число добавляется к очкам игрока. N – четное. Игру начинает первый игрок. Необходимо вывести максимально возможную сумму очков для первого игрока при условии, что противник играет наилучшим образом.

Формат входных данных

В первой строке входного файла содержится одно целое число $N(2 \leqslant N \leqslant 10000)$. В следующих N строках записан исходный ряд чисел, по одному числу в строке.

Формат выходных данных

Выходной файл должен содержать единственное число – максимально возможную сумму очков для первого игрока при наилучшей игре второго игрока.

Примеры

game.in	game.out
4 4 7 2 9	16

Задача E. Контест-палиндром

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Как вы знаете, Хет и Юра очень любят готовить контесты и готовы заниматься этим целыми часами. Вот и сейчас Хет перебирает задачи, которые он хочет добавить в предстоящий контест. Он считает, что контест будет **успешным** только в том случае, если первые буквы названий задач, которые в него входят, образуют палиндром (контест, который не содержит ни одной задачи, по логике Хета, тоже успешный!).

Система подготовки контестов «Моногон» содержит в себе N задач, которые пронумерованны последовательно от 1 до N. Все их названия начинаются с заглавной латинской буквы. Хет хочет выбрать некоторые из них. Но «Моногон» требует одно интересное правило: номера задач в контесте должны образовывать возрастающую последовательность.

Хет выписал первые буквы названий задач с номера 1 по N и принёс вам. Помогите Хету определить, сколько всего **успешных** контестов он сможет подготовить.

Так как количество контестов может быть очень большим, ответ требуется вывести по модулю $10^9 + 7$.

Формат входных данных

Единственная строка представляет из себа последовательность длины N ($1 \leqslant N \leqslant 5000$) из латинских заглавных букв.

Формат выходных данных

Выведите кол-во успешных контестов по модулю $10^9 + 7$.

Примеры

стандартный ввод	стандартный вывод
ABACABA	42
ITMO	5

Замечание

В первом примере название задачи №1 начинается с буквы «A», №2 с буквы «B», №3 с буквы «A» и т. д.

Задача F. Транзисторы над Пекином

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Всемирно известный профессор В.В. Адимов продолжает свои разнообразные исследование устойчивости транзисторов. Теперь в голову ему пришла следующая задача: в доме N этажей, профессор хочет выяснить номер максимального этажа, падение с которого оставляет транзистор целым. Поскольку профессор исследует сферичиские транзисторы в вакууме, то можете считать что разбившись при падении с этажа f транзистор обязательно разобьется при падении с этажа $f + 1$. Дополнительно поставлено условие, что разрешено проведение не более чем K испытаний.

Эта задача была поручена именно вам, как самому успешному аспиранту профессора Адимова. Поскольку транзисторы нынче в цене, но наука все-таки дороже, то необходимо выяснить, какое минимальное количество транзисторов необходимо закупить, чтобы успешно провести эксперимент даже если вам будет катастрофически не везти.

Формат входных данных

В первой и единственной строке входного файла содержатся два целых положительных числа N и K не превосходящих 2000.

Формат выходных данных

Выведите единственное число - ответ на поставленную задачу. Если для данных N и K возможна ситуация, при которой мы не сможем получить ответ на вопрос даже имея неограниченный запас бесплатных транзисторов выведите -1.

Примеры

stdin	stdout
4 2	-1
4 3	2

Задача G. Логическое дерево

Имя входного файла:	`boolean.in`
Имя выходного файла:	`boolean.out`
Ограничение по времени:	0.5 секунда
Ограничение по памяти:	256 мегабайт

Рассмотрим разновидность двоичного дерева, которую мы назовем логическим деревом. В этом дереве каждый уровень полностью заполнен, за исключением, возможно, последнего (самого глубокого) уровня. При этом все вершины последнего уровня находятся максимально слева. Дополнительно, каждая вершина дерева имеет ноль или двоих детей.

Каждая вершина дерева имеет связанное с ней логическое значение (1 или 0). Кроме этого, каждая внутренняя вершина имеет связанную с ней логическую операцию („И“ или „ИЛИ“). Значение вершины с операцией „И“ — это логическое „И“ значений её детей. Аналогично, значение вершины с операцией „ИЛИ“ — это логическое „ИЛИ“ значений её детей. Значения всех листьев задаются во входном файле, поэтому значения всех вершин дерева могут быть найдены.

Наибольший интерес для нас представляет корень дерева. Мы хотим, чтобы он имел заданное логическое значение v, которое может отличаться от текущего. К счастью, мы можем изменять логические операции некоторых внутренних вершин (заменить „И“ на „ИЛИ“ и наоборот).

Дано описание логического дерева и набор вершин, операции в которых могут быть изменены. Найдите наименьшее количество вершин, которые следует изменить, чтобы корень дерева принял заданное значение v. Если это невозможно, то выведите строку `«IMPOSSIBLE»` (без кавычек).

Формат входных данных

В первой строке входного файла находятся два числа n и v ($1 \leqslant n \leqslant 10\,000, 0 \leqslant v \leqslant 1$) — количество вершин в дереве и требуемое значение в корне соответственно. Поскольку все вершины имеют ноль или двоих детей, то n нечётно. Следующие n строк описывают вершины дерева. Вершины нумеруются от 1 до n.

Первые $(n-1)/2$ строк описывают внутренние вершины. Каждая из них содержит два числа — g и c, которые принимают значение либо 0, либо 1. Если $g = 1$, то вершина представляет логическую операцию „И“, иначе она представляет логическую операцию „ИЛИ“. Если $c = 1$, то операция в вершине может быть изменена, иначе нет. Внутренняя вершина с номером i имеет детей $2i$ и $2i+1$.

Следующие $(n+1)/2$ строк описывают листья. Каждая строка содержит одно число 0 или 1 — значение листа.

Формат выходных данных

В выходной файл выведите ответ на задачу.

Примеры

boolean.in	boolean.out
9 1 1 0 1 1 1 1 0 0 1 0 1 0 1	1
5 0 1 1 0 0 1 1 0	IMPOSSIBLE

Задача H. Редукция дерева

Имя входного файла:	`tree.in`
Имя выходного файла:	`tree.out`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Задано неориентированное дерево, содержащее n вершин. Можно выбрать некоторое ребро и удалить его, при этом инцидентные ему вершины не удаляются. Таким образом можно удалить из дерева некоторый набор рёбер. В результате дерево распадается на некоторое количество меньших деревьев. Требуется, удалив наименьшее количество рёбер, получить в качестве хотя бы одной из компонент связности дерево, содержащее ровно p вершин.

Формат входных данных

Первая строка входного файла содержит пару натуральных чисел n и p ($1 \leqslant p \leqslant n \leqslant 1000$). Далее в $n-1$ строке содержатся описания рёбер дерева. Каждое описание состоит из пары натуральных чисел a_i, b_i ($1 \leqslant a_i, b_i \leqslant n$) — номеров соединяемых ребром вершин.

Формат выходных данных

В первую строку выведите наименьшее количество рёбер q в искомом наборе.

Примеры

tree.in	tree.out
11 6 1 2 1 3 1 4 2 6 2 7 1 5 2 8 4 9 4 10 4 11	2

Задача I. Шоколадка

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

Команда «Отбой» участвует в очередном марафоне по «Угадай мелодию. Rock version». Чтобы было чем подкрепиться во время игры, команда взяла с собой большую прямоугольную плитку шоколада размерами $w \times h$. У команды есть список из n пар чисел — размеры шоколадок, которые команда считает счастливыми. Прежде чем приступить к поеданию шоколадки, участники команды решили поделить имеющуюся плитку на счастливые шоколадки. Для этого они действуют следующим образом: сначала плитка шоколада ломается на 2 части по линии, строго параллельной одной из своих сторон, после чего каждую из полученных частей они могут продолжить ломать аналогичным образом.

Вам поручили определить, какое максимальное количество счастливых шоколадок команда сможет получить, действуя по данной схеме. Шоколадки, полученные поворотом счастливых, счастливыми не являются.

Формат входных данных

В первой строке входного файла заданы три целых числа w, h, n — размеры плитки шоколада и количество вариантов размера счастливых шоколадок соответственно ($1 \leqslant w, h \leqslant 300, 1 \leqslant n \leqslant w \times h$). В следующих n строках заданы пары целых чисел w_i, h_i — размеры счастливых шоколадок ($1 \leqslant w_i \leqslant w, 1 \leqslant h_i \leqslant h$).

Формат выходных данных

В единственную строку выходного файла выведите максимальное количество счастливых шоколадок, на которые можно разрезать данную плитку.

Примеры

chocolate.in	chocolate.out
21 11 4 10 4 6 2 7 5 15 10	15
9 12 5 1 12 2 6 3 4 4 3 6 2	9

Задача J. Сбалансируй-ка!

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

Лидеру команды «Отбой» на День Рождения подарили подвешенное бинарное дерево. Однако, ему не понравилось, что дерево было несбалансированным. Теперь он хочет удалить минимальное количество вершин в дереве, чтобы оно стало сбалансированным. Перед тем как удалить вершину из дерева, он обязан удалить все вершины из её поддерева. Напомним, что дерево является сбалансированным тогда и только тогда, когда для любой вершины высота её левого и правого поддеревьев отличается не более чем на 1 (высота пустого дерева равна нулю, а высота дерева из одной вершины — единице). Корнем дерева является вершина 1.

Формат входных данных

В первой строке входного файла задано целое число n — количество вершин в дереве ($1 \leqslant n \leqslant 1111$). В следующих n строках заданы по два целых числа $left_i$ и $right_i$ — номера левого и правого ребёнка вершины соответственно или 0, если этого ребёнка не существует.

Формат выходных данных

В единственной строке выходного выведите одно число — искомое минимальное количество удаляемых вершин.

Примеры

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