Задача A. Стандарты времени

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

Каждая операционная система (в том числе и та, которую пишет Вася) должна корректно работать с различными стандартами времени.

Например, во время Великой французской революции на территории Франции действовало десятичное время. Это означало, что каждые сутки содержали 10 десятичных часов, каждый десятичный час содержал 100 десятичных минут, которые в свою очередь были поделены на 100 десятичных секунд.

Никто не знает, какие форматы времени появятся в будущем, потому Вася хочет реализовать поддержку различных стандартов времени. Стандарт времени задается тремя целыми числами a, b и c — соответственно количество «часов» в одних сутках, количество «минут» в каждом «часе» и количество секунд в каждой «минуте». При этом длительность секунд и суток во всех стандартах времени одинаковая, а длительность «минут» и «часов» может различаться.

Заданы два стандарта времени. Необходимо написать программу для перевода времени из одного стандарта в другой.

Формат входного файла

Первая строка входного файла содержит три целых числа a_1, b_1 и c_1 — описание первого стандарта времени. Во второй строке входного файла в аналогичном формате задан второй стандарт времени (a_2, b_2 и c_2)

Третья строка входного файла содержит три числа h_1, m_1 и s_1 — некоторый момент времени в первом стандарте, где h_1 — число «часов», m_1 — число «минут», s_1 — число секунд ($0 \le h_1 < a_1$, $0 \le m_1 < b_1$, $0 \le s_1 < c_1$).

Все числа во входном файле неотрицательные, целые и не превосходят 1000.

Известно, что $a_1 \cdot b_1 \cdot c_1 = a_2 \cdot b_2 \cdot c_2$.

Формат выходного файла

В выходной файл выведите три числа h_2, m_2 и s_2 — заданный момент времени во втором стандарте, где h_2 — число «часов», m_2 — число «минут», s_2 — число секунд ($0 \le h_2 < a_2$, $0 \le m_2 < b_2$, $0 \le s_2 < c_2$).

Примеры

time.in	time.out
24 60 60 12 72 100 11 11 11	5 42 71
2 2 2 1 1 8 1 0 1	0 0 5

Задача B. Файловая система

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

Вася хочет написать новый драйвер для файловой системы, для этого ему надо преобразовывать имена файлов в формат 8.3.

Будем говорить, что имя файла находится в формате 8.3, если выполняются следующие условия:

- Имя файла содержит только заглавные латинские буквы, цифры или символы «`'-!#$%&()@^_{}.~`».
- В имени файла встречается не более одной точки.
- Число символов слева от точки (число символов в имени файла при отсутствии точки) не превосходит восемь.
- Число символов справа от точки не превосходит три.

Для преобразования имени файла к формату 8.3 выполняется следующая последовательность действий:

- Имя файла переводится в верхний регистр. Если после этого имя файла находится в формате 8.3, то преобразование завершается.
- Все пробелы удаляются.
- Все символы из набора «`"/\[]:;=,`» заменяются на подчеркивания.
- Удаляются все подряд идущие точки в конце имени файла.
- Удаляются все точки, кроме последней.
- Пусть X — первые шесть символов имени файла, Y — первые три символа после точки в имени файла. Если в имени файла есть точка, то имя в формате 8.3 будет иметь вид «X~1.Y», а если точки нет, то «X~1».

Формат входного файла

В первой строке входного файла находится имя файла — строка длиной не более 255 символов. Имя файла содержит только латинские буквы, цифры, пробелы и символы из набора «`'-!#$%&()@^_{}.~"/\[]:;=,`».

Формат выходного файла

В выходной файл выведите имя файла в формате 8.3.

Примеры

file.in	file.out
`Windows`	`WINDOWS`
`Program Files`	`PROGRA~1`
`a=b.txt`	`A_B~1.TXT`

Задача C. Игра с шариками

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

Какая же операционная система без встроенных игр?

Вася уже почти написал игру и ему осталось лишь решить небольшую подзадачу — необходимо посчитать число возможных состояний в конце игры.

В процессе игры N одинаковых шаров раскладываются по M одинаковым ящикам. Причем, в каждом ящике не может оказаться более K шаров. Шары при этом никак не различаются, как и ящики. Таким образом, два случая, когда в первом ящике один шар, во втором — два и, когда в первом два шара, а во втором — один, не различаются.

Так как это число может быть очень большим, достаточно найти ответ по модулю R.

Формат входного файла

Во входном файле задано четыре числа N — количество шаров, M — количество ящиков, K — максимальное количество шаров в каждом ящике ($0 \le N, M, K \le 1000$) и R ($2 \le R \le 10^9$).

Формат выходного файла

В выходной файл выведите ответ на задачу — остаток от деления числа различных размещений шаров на R.

Примеры

game.in	game.out
2 2 2 1000	2
4 3 2 1000	2

Задача D. Трамваи

Имя входного файла:	`tram.in`
Имя выходного файла:	`tram.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мегабайт

Правительство небольшого города Мухоловска решило улучшить транспортную ситуацию в своем городе. Для этого была построена сеть трамвайных путей, соединяющая n трамвайных остановок. Для удобства пассажиров между каждой парой остановок можно было проехать на трамвае. С другой стороны, в целях экономии, проехать между двумя остановками можно было единственным образом. Формально говоря, трамвайная сеть представляет собой дерево с n вершинами. При этом вершины дерева соответствуют остановкам, а ребра — путям.

Изначально по каждому трамвайному пути проходил хотя бы один трамвайный маршрут. Однако со временем некоторые маршруты оказались отменены, а, следовательно, и некоторые трамвайные пути стали невостребованными. Путь считается невостребованным, если ни один трамвайный маршрут по нему не проходит. С целью экономии средств невостребованные трамвайные пути Мухоловска было решено разобрать.

Ваша задача — написать программу для определения числа невостребованных путей.

Формат входного файла

Первая строка входного файла содержит единственное число n — количество трамвайных остановок города ($2 \le n \le 100000$). Каждая из следующих $(n-1)$-ой строки содержит описание одного трамвайного пути (ребра дерева). Описание состоит из двух чисел b и e — номеров остановок, соединенных соответствующим путем. Остановки пронумерованы целыми числами от 1 до n.

В следующей строке содержится число m — количество трамвайных маршрутов ($0 \le m \le 100000$). В каждой из следующих m строк содержится описание трамвайного маршрута. Описание состоит из двух чисел x и y — трамвайный маршрут имеет конечные остановки с номерами x и y и проходит по кратчайшему пути между ними ($x \ne y$).

Формат выходного файла

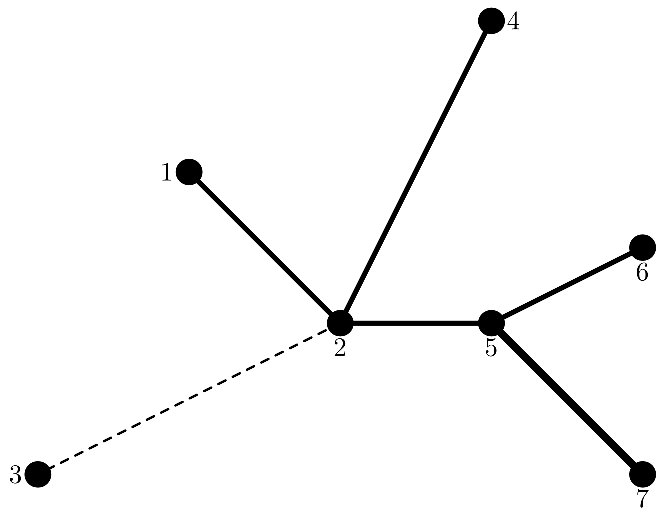
В выходной файл выведите количество невостребованных трамвайных путей Мухоловска.

Примеры

tram.in	tram.out
4 1 2 1 3 1 4 0	3
7 1 2 2 3 2 4 5 2 5 6 7 5 3 1 7 2 4 7 6	1

Решения, правильно работающие для $n \le 100$, будут оцениваться из 60 баллов.
Решения, правильно работающие для $n \le 1000$, будут оцениваться из 80 баллов.

Иллюстрация ко второму примеру.



Пунктирной линией обозначен невостребованный путь.

Задача E. Задача о рюкзаке

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

Задача о рюкзаке формулируется следующим образом: дано n предметов, i-й из которых имеет вес w_i и стоимость v_i, требуется построить набор предметов максимальной стоимости, чтобы их суммарный вес не превосходил числа c (размеров рюкзака). Известно, что задача о рюкзаке является NP-трудной, за время, пропорциональное суммарному весу предметов, задачу можно решить с использованием динамического программирования.

Однако оказывается, что в некоторых случаях задачу о рюкзаке можно решить жадно. Один из таких случаев возникает, когда экземпляр задачи о рюкзаке представляет собой матроид.

Матроидом называется пара $\langle X, \mathcal{I} \rangle$, где X — некоторое конечное множество, а $\mathcal{I}$ — некоторое семейство подмножеств X, элементы $\mathcal{I}$ называют независимыми. Множество $\mathcal{I}$ должно удовлетворять следующим аксиомам:

1. $\mathcal{I} \neq \varnothing$;
2. Если $A \in \mathcal{I}$ и $B \subset A$, то $B \in \mathcal{I}$;
3. Если $A, B \in \mathcal{I}$ и $|A| > |B|$, то найдется $x \in A \setminus B$, такой что $B \cup \{x\} \in \mathcal{I}$.

Здесь как $|A|$ обозначено количество предметов в множестве A.

В качестве примера матроида можно превести множество ребер неориентированного графа, где множество ребер является независимым, если оно является ациклическим.

Рассмотрим множество предметов с весами $w_1, w_2, \ldots, w_n$. В качестве X выберем эти предметы. Будем называть множество предметов $\{i_1, i_2, \ldots, i_k\}$ независимым, если $w_{i_1} + w_{i_2} + \ldots + w_{i_k} \le c$. Требуется выяснить, образует ли описанная конструкция для заданных $w_1, w_2, \ldots, w_n$ и c матроид.

Формат входного файла

Первая строка входного файла содержит число n ($1 \le n \le 50$). Вторая строка содержит n целых чисел $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le 100$). Третья строка содержит число c ($\max w_i \le c \le \sum w_i$).

Формат выходного файла

Выведите “`YES`”, если множество решений задачи о рюкзаке образует матроид. В противном случае выведите “`NO`”.

Во втором случае на второй строке выведите одно целое число — номер аксиомы, которая нарушается. Если нарушается вторая или третья аксиома, то на следующих двух строках выведите описания множеств A и B, для которых утверждение аксиомы не выполняется. Каждое множество описывается количеством предметов в нем, после чего должны следовать номера этих предметов.

Примеры

knapsack.in	knapsack.out
3 1 2 3 4	YES
3 3 4 5 7	NO 3 2 1 2 1 3