Задача A Конвейер

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

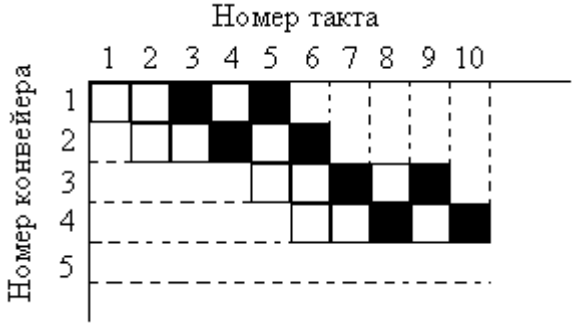
Известно, что в современных процессорах используется конвейерный метод исполнения команд. Рассмотрим некоторый абстрактный аналог современного процессора и процесс выполнения им программы подробнее:

- машинное время измеряется в тактах;
- программа — это набор из N однотипных команд;
- процессор состоит из трех устройств: буфера, распределителя и исполнителя команд;
- для упрощения положим, что буфер команд изначально содержит всю выполняемую программу;
- распределитель берет очередную команду из буфера, определяет самый ранний момент когда она может начать выполняться (не раньше начала выполнения предыдущей команды) и передает ее исполнителю в этот момент времени;
- исполнитель состоит из M конвейеров, работающих параллельно;
- при получении исполнителем команды, он отправляет ее на конвейер, которым она будет обрабатываться;
- обозначим номер конвейера, на котором будет обрабатываться i-я команда как c_i, тогда всегда должно выполняться условие:

$c_i = 1$, если $i = 1$ или $c_{i-1} = M$,
$c_i = c_{i-1} + 1$, если $i > 1$ и $c_{i-1} < M$;
т.е. следующий конвейер по циклу;

- для ускорения работы каждая поступающая на конвейер команда разбиваются на K ($1 \le K \le M$) стадий выполнения (микрокоманд), одинаковых для всех команд;
- на каждую микрокоманду уходит один такт машинного времени;
- каждая команда должна выполняться непрерывно (то есть выполнение команды не может быть приостановлено, а затем продолжено);
- некоторые микрокоманды (как минимум — одна в команде) могут быть исключительными;
- в один момент времени только на одном из конвейеров может выполняться исключительная микрокоманда.

Допустим N = 4, M = 5, K = 5 и исключительными микрокомандами являются 3-я и 5-я. Тогда можно нарисовать такую схему выполнения процессором программы (белые квадраты — обычные микрокоманды, черные — исключительные):



Пояснения. Первая команда начинает выполнение с первого такта; вторая — со второго; третья — с пятого, так как, если она начнет выполняться с 3-го или 4-го такта, возникнет ситуация когда две исключительные микрокоманды выполняются параллельно на разных конвейерах (в первом случае — на 5-м такте, во втором — на 6-м); четвертая команда выполняется с шестого такта. Всего на выполнение программы потрачено 10 тактов машинного времени.

Ваша задача — определить, за сколько времени (в тактах) выполнится данная программа при определенных параметрах процессора.

Формат входных данных

В первой строке входного файла записаны целые числа N, M, K ($1 \le N \le 10^9$, $1 \le M \le 10$, $1 \le K \le M$). Во второй строке записано целое число L ($1 \le L \le K$) — количество исключительных микрокоманд. В третьей строке записана последовательность из L различных целых чисел, разделенных пробелом, a_1, … a_L — номера исключительных микрокоманд ($1 \le a_i \le K$). Все числа в строках разделены пробелами.

Формат выходных данных

В единственной строке выходного файла нужно вывести целое число — количество тактов, потраченных на выполнение программы.

Пример

conveyor.in	conveyor.out
4 5 5 2 3 5	10

Задача B Перестановки

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

Перестановкой называется последовательность из N различных целых чисел от 1 до N, например (3, 5, 1, 2, 4) – перестановка 5 чисел.

Допустим, имеются две перестановки: A = (a_1, a_2, ... a_N) и B = (b_1, b_2, ... b_N). Произведением перестановок A и B назовем перестановку C = A x B = (c_1, c_2, ... c_N), где

$c_i = a_{b_i}$.

Например, произведением перестановки (3, 5, 1, 2, 4) на (2, 3, 5, 4, 1) будет перестановка (5, 1, 4, 2, 3).

Известно, что произведение перестановок в общем случае не обладает свойством коммутативности (то есть A x B ≠ B x A):

(3, 5, 1, 2, 4) x (2, 3, 5, 4, 1) = (5, 1, 4, 2, 3)
(2, 3, 5, 4, 1) x (3, 5, 1, 2, 4) = (5, 1, 2, 3, 4)

Однако для некоторых пар перестановок это свойство соблюдается:

(3, 4, 2, 5, 1) x (4, 1, 5, 3, 2) = (5, 3, 1, 2, 4)
(4, 1, 5, 3, 2) x (3, 4, 2, 5, 1) = (5, 3, 1, 2, 4)

Вам требуется для заданной перестановки A найти количество перестановок B таких, что A x B = B x A и вывести остаток от его деления на P.

Формат входных данных

Первая строка входного файла содержит два целых числа N и P ($1 \le N \le 10^5$, $2 \le P \le 10^9$). На второй строке расположено N целых чисел a_1, a_2, ... a_N – перестановка A ($1 \le a_i \le N$, все a_i различны). Все числа в строках разделены пробелами.

Формат выходных данных

Выходной файл должен содержать одну строку с целым числом K – ответом к задаче ($0 \le K \le P - 1$).

Пример

permutations.in	permutations.out
5 100 3 4 2 5 1	5

Искомые перестановки для примера:
1 2 3 4 5
2 5 4 1 3
3 4 2 5 1
4 1 5 3 2
5 3 1 2 4

Задача C Дороги

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

В некотором государстве для уменьшения заторов на дорогах решили внедрить одностороннее движение по максимально возможному числу городских улиц. Городские власти установили лишь одно условие для плана одностороннего движения в городе – возможность проезда из любой точки в любую другую точку дорог города, это очевидно возможно до и должно быть возможно после внедрения одностороннего движения. Представим себе карту дорог города в виде неориентированного графа дорог, в котором вершинами являются перекрестки и тупики, а ребрами являются дороги. В графе дорог возможны петли и кратные ребра, более того он необязательно является планарным. Граф дорог одного города обязательно является связным. Для каждого ребра графа дорог можно ввести ориентацию, что соответствует внедрению одностороннего проезда по соответствующей дороге. Вы должны предложить план ориентации графов дорог городов, содержащий максимально возможное число ориентированных ребер и оставшийся при этом связным для каждого города. Дороги между городами решено оставить двунаправленными и они отсутствуют во входных данных.

Формат входных данных

Входной файл описывает граф дорог одного и более городов. Первая строка входного файла содержит два целых числа: N – число вершин, E – число ребер ($1 \le N \le 20000$, $0 \le E \le 20000$). Далее следуют E строк – описания ребер, содержащих E пар целых чисел – номера вершин соединенных ребром, вершины нумеруются от 1 до N. Все числа разделены пробелами.

Формат выходных данных

Выходной файл должен повторять данные входного файла точно в том же порядке с дополнительными элементами: описание ребра помимо двух чисел, являющихся левой и правой вершиной ребра, содержит справа символ > если дорога ориентирована от левой вершины к правой, символ < если дорога ориентирована от правой вершины к левой или символ = если дорога остается неориентированной. Не забывайте разделять элементы пробелами и выведите любой понравившийся вам вариант ориентации графа дорог.

Пример

roads.in	roads.out
2 2 1 2 1 2	2 2 1 2 > 1 2 <
2 1 1 2	2 1 1 2 =