Задача A. Курьеры

Имя входного файла:	`couriers.in`
Имя выходного файла:	`couriers.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

В свободное от борьбы с преступностью время, Флэш помогает своей тете. А именно, он развозит пирожки. Как ни странно, он лучший в этом деле. Благодаря своей суперспособности Флэш доставляет пирожки практически моментально. Но, к сожалению, сегодня Флэш приболел и не смог помочь любимой тетушке.

Город, в котором живет Флэш, представляет из себя набор домов, выстроенных в линию. Все дома пронумерованы, начиная с 1. Всего в городе k домов. Тетушка, дабы спасти положение, наняла бригаду из n курьеров. i-й курьер изначально находится около p_i-го дома. Изначально каждый курьер свободен. Если курьер доставил заказ к i-у дому, то следующий заказ он начнет выполнять уже от этого i-го дома.

Известно, что сегодня поступит m заказов на доставку. Каждый заказ характеризуется временем t_i, в которое он поступит, и числом x_i — номером дома, в который нужно привезти заказ. Поскольку клиенты не любят долго ждать, тетушка хочет, чтобы каждый заказ был доставлен в минимально возможный срок. А именно, когда поступает заказ, тетушка рассчитывает для каждого курьера время, к которому он сможет доставить этот заказ, выбирает курьера с минимальным временем и добавляет ему этот заказ. Время для каждого курьера рассчитывается как сумма времен на каждый заказ в его списке заказов. Если несколько курьеров смогут доставить заказ одновременно, выбирается курьер с наименьшим номером. Время которое курьер тратит на заказ рассчитывается следующим образом: если до выполнения заказа курьер находился около дома с номером i, а доставить заказ надо к дому с номером j, то время, которое потратит курьер на этот заказ, будет равно $|i - j|$. Курьер не может выполнить новый заказ пока не выполнит все предыдущие. Тетушка не знает заранее всех заказов, поэтому она не может направить курьера к какому-либо дому до поступления заказа.

Помогите тетушке Флэша автоматизировать процесс. Вам нужно для каждого поступившего заказа вывести номер курьера, который его доставит.

Формат входного файла

В первой строке входного файла даны два числа n и k ($1 \le n \le 10^5, 1 \le k \le 10^5$) — количество курьеров в бригаде и домов в городе. В следующей строке дано n чисел p_i ($1 \le p_i \le k$) — номер дома, рядом с которым изначально стоит i-й курьер. В следующей строке дано число m ($1 \le m \le 10^5$) — число заказов. В следующих m строках дано по два числа t_i и x_i ($0 \le t_i \le 10^5, 1 \le x_i \le k$) — время поступления i-го заказа и номер дома к которому надо его доставить. Гарантируется, что заказы даны в хронологическом порядке, то есть $t_i \le t_{i+1}$. Если два заказа поступают в одно время, то считается что заказ встретившийся во входном файле первым поступает раньше.

Формат выходного файла

Выведите m строк. В i-й строке выведите номер курьера, который доставит i-й заказ.

Пример

couriers.in	couriers.out
3 10 1 5 10 5 1 7 2 4 5 10 6 1 7 7	2 1 3 1 2

Задача B. Шифровка

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

Когда Флэш снимает свой костюм, он предстает перед нами в образе Барри Аллена — полицейского судмедэксперта.

В этот раз на месте преступления Барри обнаружил записку, на которой была записана последовательность латинских букв. Так же он обратил внимание, что на ней часто встречается последовательность букв «fa». Барри выдвинул гипотезу, что эта строка зашифрована с помощью «Кирпичного шифра».

Этот шифр предполагает, что в изначальной строке после каждой гласной буквы вставляется строка «fa». Так например после шифровки строка «hello» будет выглядить как «hefallofa». Барри считает, что гласные буквы в английском языке — это «a», «o», «i», «u», «e» и «y».

Поскольку Барри считает, что эта записка не последняя такого вида, он просит вас написать программу, которая сможет расшифровать строку, зашифрованную «Кирпичным шифром» или сказать, что это невозможно.

Формат входного файла

В первой и единственной строке входного файла задана строка s из строчных латинских букв ($1 \le |s| \le 10^4$).

Формат выходного файла

В единственной строке выходного файла выведите первоначальную строку, если ее можно расшифровать или `impossible`, если это невозможно.

Пример

crypt.in	crypt.out
hefallofa	hello
abc	impossible

Задача C. Флэш на уроке математики

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

Вы можете не верить, но даже супергерои когда-то были маленькими детьми. И даже Флэш. Сегодня у него свободный день, поэтому он лежит и вспоминает свое обучение в школе.

Он вспомнил, как в пятом классе учительница по математике вызвала его к доске, написала арифметическое выражение, состоящее из натуральных чисел, символов «+», «*» и круглых скобок, и сказала ему посчитать его значение. Флэш был способным мальчиком, поэтому быстро справился с этим заданием. Но после этого учительница заменила один из знаков арифметических операций на другой («+» был заменен на «*» или «*» был заменен на «+», будем называть эту замену — заменой на противоположный знак) и сказала: «Чему равно значение выражения теперь?». Тогда Флэш не смог придумать ничего лучше, кроме как посчитать значение выражения заново. Но, так как он любит скорость, теперь он задался вопросом: а можно ли это делать как-то быстрее?

Ваша задача написать программу, которая по данному выражению, сможет вычислять как изменится его значение при изменении какого-то знака.

Формат входного файла

В первой строке входного файла дана непустая строка, описывающая исходное арифметическое выражение, данное Флэшу. Выражение содержит только символы «(» (ASCII-код 40), «)» (ASCII-код 41), «*» (ASCII-код 42), «+» (ASCII-код 43) и цифры от 0 до 9. Гарантируеся, что все числа в выражении натуральные и не содержат лидирующих нулей. Также гарантируется, что выражение корректно. Длина выражения не превосходит 200 000.

В следующей строке дано число n ($0 \le n \le 100\,000$) — количество изменений арифметических знаков. В следующих n строках находится описание запросов. Каждый запрос описывается одним числом i — индекс символа в строке. Гарантируется, что этот символ равен либо «+», либо «*». Ответ на запрос — значение исходного выражения после замены данного символа на противоположный («+» заменяется на «*», «*» на «+»).

Формат выходного файла

В выходной файл выведите n строк, в i-й из которых записано значение выражение при замене арифметической операции в i-м запросе. Ответ следует выводить по модулю $10^9 + 7$.

Пример

expression.in	expression.out
2*3+5 2 2 4	10 30
2+3+4 2 2 4	10 14
(2+3)*4 2 3 6	24 9

Комментарий

Если в тесте из примера поменять третий символ строки (это «*») на противоположный знак — «+», то значение выражения будет равно $2 + 3 + 5 = 10$.

Если поменять седьмой символ строки (это «+») на противоположный знак — «*», то значение выражения будет равно $2 * 3 * 5 = 30$.

Задача D. Головоломка

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

На день рождения Барри получил интересную головоломку. На коробке было написано простое число k, и инструкция: для того, чтобы ее открыть, нужно найти минимальное число n такое, что $k \cdot n$ состоит из одинаковых цифр в десятичной системе счисления, и использовать остаток от его деления на $1\,000\,000\,007$ как пароль к цифровому замку. Барри не стал тратить время, и просто попробовал ввести каждое число. Что неудивительно, замок открылся.

А сможете ли вы, не имея скорость Флэша, решить эту задачу?

Формат входного файла

В первой строке входного файла дано одно целое число T ($1 \le T \le 2\,000$) — число тестов.

В следующих T строках входного файла дано по одному целому простому числу k ($2 \le k < 1\,000\,000\,000$).

Формат выходного файла

Для каждого теста в отдельной строке выведите одно целое число — остаток от деления минимального подходящего числа n на $1\,000\,000\,007$, если оно существует, или -1, если такого числа не существует.

Пример

numbers.in	numbers.out
2 11 13	1 8547

Задача E. Очередь

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

Барри Аллен, также известный как Флэш — самый быстрый человек на земле. Он может разогнаться до такой скорости, что прорвет временной барьер и сможет путешествовать во времени. Барри редко злоупотребляет способностями, но стоять в очередях слишком нудно и долго для столь быстрого парня.

Вот и сегодня Флэшу нужно сдать книгу в библиотеку, но стоять долго в очереди он не хочет. Очередь в библиотеке организована следующим способом: каждую минуту из очереди выходит один человек (если очередь непуста), при этом в какие-то моменты времени в очередь приходят сразу несколько человек. В очередь нельзя вступить позже, чем библиотека закроется, но стоять в ней можно и позже. Если Барри придет в очередь вместе с множеством людей, он успеет встать первым.

У Барри есть план: он возьмет записи камеры наблюдения и вычислит в какое время ему нужно будет перенестись, чтобы сдать книгу с минимальными возможными затратами по времени. При этом, если таких моментов времени несколько, Флэш хочет попасть в самый поздний, чтобы тратить меньше сил на путешествие во времени.

Из-за неожиданно объявившегося в городе Гориллы Гродда, Барри не успел вычислить оптимальное время для сдачи книги, поэтому он просит вас помочь ему с этой несложной задачей.

Формат входного файла

В первой строке входного файла задано три целых числа x, n и c ($0 \le n \le c; 1 \le c \le 10\,000; 0 \le x \le 1000$) — количество людей в очереди в начальный момент времени, количество записей о приходе новых людей и время закрытия библиотеки. В следующих n строках содержатся по два целых числа t_i, a_i ($1 \le t_{i-1} < t_i \le c; 1 \le a_i \le 1000$) означающие, что в момент времени t_i в очередь пришло a_i человек. Начальный момент времени принят за 0 и в него никто не уходит.

Формат выходного файла

В единственной строке выходного файла выведите два числа T, X — самое позднее время, когда Флэш может встать в очередь, чтобы простоять в ней наименьшее возможное количество времени и количество человек, которые стоят раньше него в очереди.

Пример

queue.in	queue.out
1 2 5 2 1 3 1	5 0
3 2 5 2 1 3 10	3 1

Задача F. Веселье со стеком

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

Листая свежую газету, Флэш заметил заголовок: «Только для самых быстрых!». Естественно, это привлекло внимание нашего героя.

Однако впереди его ждало разочарование — редакторы газеты предлагали решить задачу, подвластную, как они утверждали, только самым быстрым умам. В задаче была дана последовательность из n чисел a_i. Требовалось получить лексикографически минимальную перестановку этих чисел из начальной, используя только операции со стеком: каждое число из последовательности по очереди кладется на вершину стека. При этом в любой момент из стека можно вынуть число и записать в конец новой последовательности. Перестановка a лексикографически меньше перестановки b, если существует такое k, что первые k чисел обеих перестановок совпадают, а $a_{k+1} < b_{k+1}$.

Флэш уже которую неделю ходит расстроенный. Помогите ему побороть депрессию — напишите программу, которая решила бы эту задачу!

Формат входного файла

В первой строке входного файла дано число n ($1 \le n \le 10^5$) — количество чисел. В следующей строке даны n чисел a_i ($1 \le a_i \le 10^5$) — описание исходной перестановки.

Формат выходного файла

Выведите лексикографически минимальную перестановку, которую можно получить, используя только операции со стеком.

Пример

stack.in	stack.out
5 1 9 10 8 6	1 6 8 10 9

Комментарий

Последовательность операций в первом примере:

- Положить в стек 1.
- Вынуть из стека 1.
- Положить 9.
- Положить 10.
- Положить 8.
- Положить 6.
- Вынуть из стека 6.
- Вынуть 8.
- Вынуть 10.
- Вынуть 9.

Задача G. Восстановление массива

Имя входного файла:	`subsequence.in`
Имя выходного файла:	`subsequence.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

На день рождения Флэшу подарили массив из n очень длинных чисел. Флэш поставил подарок на полку и заметил, что он обладает следующим свойством: все числа, кроме первого, получались из предыдущего прибавлением единицы в один из разрядов. Более формально, для любого $i \geq 2$ выполнено равенство: $a_i = a_{i-1} + 10^k$, где a — массив, подаренный Флэшу, а k — некоторое целое неотрицательное число.

Сегодня утром Флэш проносился с огромной скоростью мимо полки с массивом и потоками воздуха некоторые элементы вынесло в окно (остальные элементы остались на своих местах). Пропажу Флэш обнаружил только вечером и был очень огорчен, что массив, составленный из оставшихся элементов, не обладает старым свойством. Поэтому он задался следующей задачей: как по оставшимся числам восстановить массив, обладающий изначальным свойством? Но эта задача не показалась Флэшу интересной, поэтому он решил добавить еще одно условие: в восстановленном массиве сумма чисел должна быть минимальной среди всех массивов, обладающих свойством из условия.

Эту задачу Флэш уже не смог решить. Помогите ему!

Формат входного файла

В первой строке входного файла дано число n ($1 \leq n \leq 1000$) — количество оставшихся чисел в массиве.

В следующей строке даны n чисел a_i ($1 \leq |a_i| \leq 1000$, $|a_i|$ — длина i-го числа) — оставшиеся в массиве числа. Гарантируется, что для любого $1 \leq i < n$ выполнено $a_i < a_{i+1}$.

Формат выходного файла

В единственной строке выходного файла выведите минимальную сумму чисел в востановленном массиве, обладающим свойством из условия. Ответ можно вывести как полностью, так и по модулю $1\,000\,000\,007$.

Пример

subsequence.in	subsequence.out
2 1 2	3
3 1 2 5	15

Задача H. Трансформация массива

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

В свободное время Флэш любит трансформировать массивы. Пусть есть массив a с n элементами. Трансформацией массива называется следующая операция:

1. Выбирается два индекса i, j ($1 \le i, j \le n$).
2. Выполняется присвоение $a_i = max(a_i, a_j)$.

Сегодня Флэш заинтересовался, можно ли получить из массива a массив b, применив некоторое количество трансформаций к массиву a. Так как он любит делать все быстро, он хочет, чтобы количество трансформаций не превышало n. Сможете ли вы помочь ему?

Формат входного файла

В первой строке входного файла даны два числа n ($1 \le n \le 10^5$) — размеры массивов. Во второй строке заданы n целых положительных чисел a_i ($1 \le a_i \le 10^9$) — массив a. В третьей строке заданы n целых положительных чисел b_i ($1 \le b_i \le 10^9$) — массив b.

Формат выходного файла

В первой строке выходного файла должно содержаться `YES`, если можно преобразовать массив a в массив b или `NO` в противном случае. В случае положительного ответа, вторая строка должна содержать единственное число k ($0 \le k \le n$) — количество трансформаций. В следующих k строках должны содержаться два числа i, j ($1 \le i, j \le n$), описывающих трансформацию. Трансформации должны идти в порядке их выполнения.

Пример

transformation.in	transformation.out
3 1 1 3 1 3 1	NO
5 1 1 1 3 4 3 3 3 4 4	YES 4 1 4 2 4 3 4 4 5

Задача I. Tree

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

Помогая Флэшу, хакер Циско часто использует командную строку. Так как ему очень важно хорошо понимать, где находятся какие файлы (ведь в его работе недопустимо даже секундное промедление!), то он попросил вас написать программу «tree».

Программа «tree» должна выводить список файлов в виде древовидной структуры. Например если в папке «directory» расположены файлы «foo» и «bar», а также директория «inner», в которой расположены файлы «hello» и «world», то программа «tree» должна вывести:

```
directory
|---------inner
|---------|-----hello
|---------|-----world
|---------bar
|---------foo
```

То есть, каждая строка выходного файла соответствует одному файлу или директории. Для каждой директории перед всеми файлами из этой директории нужно вывести отступ, состоящий из символов «-», равный по длине имени этой директории. Отступы должны разделяться символом «|». Внутри одной папки сначала должны быть выведены внутренние директории, а затем файлы, причем как директории, так и файлы должны быть упорядочены в лексикографическом порядке. Для лучшего понимания смотрите примеры.

Формат входного файла

В первой строке входного файла находится число n ($1 \le n \le 100$). В следующих n строках находятся имена файлов, по одному в строке. Имена директорий разделяются символом «/». Имя файла может состоять из маленьких и больших латинских букв, цифр и символов «.». Все имена непустые и длина каждой строки не превышает 100. Имена «.», «..», в отличие от реальных операционных систем, являются обычными и корректными именами файлов. Ни в одной строке «/» не является ни первым, ни последним символом, и два символа «/» не могут идти подряд.

Формат выходного файла

В выходном файле выведите дерево файлов, отформатированное соответствующим образом.

Пример

tree.in	tree.out
6 directory directory/inner/world directory/inner directory/foo directory/bar directory/inner/hello	directory |---------inner |---------|-----hello |---------|-----world |---------bar |---------foo
12 src/java/Gen.java src/cpp/main.cpp src/java/Main.java src/cpp/test.cpp src/cpp/bin/main src/python/generate.py src/test.in src/test.out tests/01 tests/01.a tests/02 tests/02.a	src |---cpp |---|---bin |---|---|---main |---|---main.cpp |---|---test.cpp |---java |---|----Gen.java |---|----Main.java |---python |---|------generate.py |---test.in |---test.out tests |-----01 |-----01.a |-----02 |-----02.a
1 very/long/path/of/the/file	very |----long |----|----path |----|----|----of |----|----|----|--the |----|----|----|--|---file

Задача J. Дерево Флэша

Имя входного файла:	`treemin.in`
Имя выходного файла:	`treemin.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

У Флэша всегда были математические хобби. Раньше, например, он любил изучать деревья. Одно из его любимых занятий с деревьями было следующее: он брал подвешенное дерево, в вершинах которого записаны некоторые положительные числа — a_v, и записывал в каждую вершину минимум в ее поддереве — b_v. Иначе говоря, $b_v = min(a_v, a_{u_1} \dots a_{u_k})$, где u_i все такие вершины, для которых v является предком.

Сегодня он нашел свои старые записи для некоторого дерева с корнем в вершине 1. Но есть проблема: значения a_v утерялись, а некоторые значения b_v кто-то изменил. Делать нечего, придется исправлять ситуацию. Для начала он хочет выяснить — какое наименьшее количество значений b_v нужно изменить, чтобы значения b_v являлись корректными, то есть существовал некоторый набор a_v, по которому могут быть получены значения b_v. Вам нужно помочь ему.

Формат входного файла

В первой строке входного файла целое число n ($1 \le n \le 10^5$) — количество вершин в дереве. В следующей строке заданы n целых положительных чисел b_v ($1 \le b_v \le 10^9$) — значение минимума в v-й вершине. В следующей $n-1$ строке по два числа u, v ($1 \le u, v \le n, u \ne v$) — ребра дерева.

Формат выходного файла

В выходной файл выведите единственное число — наименьшее количество значений b_v, которые нужно изменить.

Пример

treemin.in	treemin.out
5 7 3 1 2 2 1 2 1 3 3 4 3 5	1
5 3 1 4 5 2 1 2 2 3 1 4 4 5	2

Комментарий

В первом примере нужно изменить значение b_1, например, на 1, тогда существует $a = (1, 3, 1, 2, 2)$, удовлетворяющий таким b_v.

Во втором примере нужно изменить b_2 на 4, b_5 на 6, тогда существует $a = (3, 7, 4, 5, 6)$, удовлетворяющий таким b_v.