Задача A. Размеры поддеревьев

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

Вам дано дерево в корне с вершиной 1.

Поддеревом вершины v называется такое наибольшее множество вершин, в котором путь от каждой вершины до корня содержит вершину v.

Для каждой вершины выведете размер ее поддерева.

Формат входных данных

В первой строке вам дано число n - размер дерева($2 \leqslant n \leqslant 10^6$).

Затем в следующей строке следует описание массива p размера $n-1$. В нем p_i(где $1 \leqslant i \leqslant n-1$) означает, что в дереве есть ребро между вершинами $i+1$ и p_i($1 \leqslant p_i \leqslant i$).

Формат выходных данных

Для каждой вершины выведете размер ее поддерева.

Примеры

стандартный ввод	стандартный вывод
6 1 1 2 2 3	6 3 2 1 1 1

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. 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 пронумерованы последовательно от 1 до N. Все их названия начинаются с заглавной латинской буквы. Хет хочет выбрать некоторые из них. При этом, «Моногон» требует, чтобы задачи в контесты были отсортированы по порядковому номеру. Т. е. чтобы номера задач в контесте образовывали 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 E. Удаление скобок - 2

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

Дана строка, составленная из круглых, квадратных и фигурных скобок. Определите, какое наименьшее количество символов необходимо удалить из этой строки, чтобы оставшиеся символы образовывали правильную скобочную последовательность.

Формат входных данных

Во входном файле записана строка из круглых, квадратных и фигурных скобок. Длина строки не превосходит 100 символов.

Формат выходных данных

Выведите строку максимальной длины, являющуюся правильной скобочной последовательностью, которую можно получить из исходной строки удалением некоторых символов. Если возможных ответов несколько, выведите любой из них.

Примеры

stdin	stdout
([)]	[]
{([(]{)})]	([]{})
]{)[	

Задача F. Игра

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

На листке записано в одну строку $N(2 \leqslant N \leqslant 100)$ 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 $N(2 \leqslant N \leqslant 100)$. 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 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. План лекций

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

В процессе подготовки ЛКШ перед преподавателями стоит ряд задач: проверка вступительной, распределение школьников по параллелям и многое другое. Одной из таких задач является составление плана лекций на будущую смену.

С каждым годом популярность и применимость на олимпиадах тех или иных тем меняется, поэтому каждый год необходимо заново выбирать самые полезные темы для лекций. Это выглядит простой задачей, но всё осложняется тем, что некоторые темы нельзя рассказывать раньше других. Например, в младших параллелях первой всегда идёт лекция по языку Python, так как это необходимо для понимания всех последующих лекций. Также нельзя читать лекцию по алгоритму Дейкстры раньше, чем будут рассказаны способы хранения графов.

Чтобы сделать задачу более наглядной, преподаватели используют дерево зависимостей — граф, вершинами которого являются темы, а рёбрами — зависимости. Про каждую тему кроме первой известен номер темы p_i, которую необходимо рассказать перед данной (не обязательно именно в предыдущий день). Помимо этого, про каждую тему известна её полезность w_i. Чтение одной и той же более одного раза обладает нулевой полезностью (мы надеемся).

За много лет было установлено, что граф зависимостей всегда является подвешенным деревом, то есть каждая тема кроме корня дерева зависит ровно от одной другой темы. Это сильно упрощает задачу, так как корень дерева — единственная тема, для которой не ничего не требуется знать, а значит, именно её нужно рассказывать в первый день. Осталось только определить остальные лекции.

Вам предстоит написать программу для нахождения максимальной суммарной полезности лекций в наборе, который можно прочитать. Конечно, читать лекцию v_i можно только если все лекции, в которых содержится материал, необходимый для понимания v_i, уже прочитаны.

Формат входных данных

В первой строке ввода записано два числа n и k ($1 \leqslant k < n \leqslant 4000$) — количество вершин в графе зависимостей и количество лекций соответственно.

Во второй строке записано $n - 1$ число, i-е из которых соответствует номеру предка темы $i + 1$ в дереве зависимостей. Корнем дерева является тема номер 1.

В третьей строке находятся n целых чисел w_i ($0 \leqslant w_i \leqslant 100\,000$) — полезности лекций.

Формат выходных данных

Выведите одно целое число — максимальную суммарную полезность k различных лекций, которую можно достичь.

Примеры

стандартный ввод	стандартный вывод
5 3 1 1 2 3 1 1 2 2 1	4
7 3 1 1 2 3 4 4 1 2 1 2 1 2 1	5

Замечание

В первом тесте из условия оптимальное решение — выбрать лекции 1, 2 и 3. В таком случае суммарная полезность лекций составит 4.

Во втором тесте из условия оптимальное решение — выбрать лекции 1, 2 и 4. В таком случае суммарная полезность лекций составит 5.