Задача A. Оплата парковки

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

На каникулах Роман решил отдохнуть во Флатландии и арендовал себе апартаменты и роскошный автомобиль. Разумеется, такой автомобиль нельзя оставлять во дворе, поэтому Роман хочет также арендовать место на ближайшей охраняемой парковке. Поскольку он уже поиздержался с жильём и машиной, он хочет потратить на парковку как можно меньше бурлей.

На парковке доступны три тарифа аренды:

1. Заплатив a бурлей, можно использовать парковку в течение 1 дня.
2. Заплатив b бурлей, можно использовать парковку в течение одной недели, то есть 7 дней.
3. Заплатив c бурлей, можно использовать парковку в течение четырёх недель, то есть 28 дней.

Роман планирует отдыхать во Флатландии n дней. Любой тариф можно использовать произвольное количество раз, также можно арендовать парковку на суммарно больший срок, чем нужно. Какое минимальное количество бурлей придётся заплатить Роману, чтобы иметь возможность использовать стоянку все n дней?

Формат входных данных

Первая строка входных данных содержит три целых числа a, b, c ($1 \leqslant a \leqslant b \leqslant c \leqslant 1000$) — цена в бурлях за однократную покупку первого, второго и третьего тарифа аренды соответственно.

Вторая строка содержит целое число n ($1 \leqslant n \leqslant 10^{15}$) — количество дней, в течение которых Роман планирует отдыхать во Флатландии и оставлять машину на охраняемой парковке.

Формат выходных данных

Выведите единственное целое число — минимальное количество бурлей, которое Роману придётся потратить на аренду парковочного места.

Примеры

стандартный ввод	стандартный вывод
4 7 20 10	14
2 9 38 36	47

Замечание

В первом примере Роману выгодно взять 2 абонемента на неделю, это будет стоить $2 \cdot 7 = 14$ бурлей и позволит оплатить парковку на 14 дней.

Во втором примере выгодно купить 5 абонементов на неделю и 1 на день. Количество оплаченных дней будет ровно 36, а цена составит $1 \cdot 2 + 5 \cdot 9 + 0 \cdot 38 = 47$ бурлей.

Система оценки

Тесты к этой задаче состоят из пяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп.

Группа	Тесты	Баллы	Дополнительные ограничения	Комментарий
			n	
0	1 – 2	0	–	Тесты из условия.
1	3 – 17	10	$n \leqslant 200$	
2	18 – 26	20	$n \leqslant 2000$	
3	27 – 35	30	$n \leqslant 200\,000$	
4	36 – 48	40	–	

Задача B. Деревни

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

В одной стране все населённые пункты являются деревнями и расположены вдоль длинной прямой дороги. Всего вдоль дороги имеется n остановок, расстояние между любыми двумя соседними остановками равно одному километру. Рядом с некоторыми остановками расположены деревни.

Министерство транспорта решило запустить между деревнями рейсовые автобусы, ровно по одному маршруту **из каждой деревни**. Планируется, что автобус будет выезжать из деревни и двигаться направо вдоль дороги (в сторону увеличения номеров остановок) до тех пор, пока не встретит k деревень, либо не доедет до последней. После этого автобус будет возвращаться в начальную деревню. Так, для самой последней деревни автобус проедет расстояние 0 (направо, ему, конечно, ехать некуда, и, казалось бы, он вообще никому не нужен, но этот вопрос остаётся за рамками данной задачи). Необходимо посчитать длину маршрута каждого автобуса.

Формат входных данных

В первой строке входных данных находится целое число k ($1 \leqslant k \leqslant 300\,000$) — количество деревень правее стартовой, которые должен посетить каждый автобус.

Во второй строке следует строка s, состоящая только из символов из '`0`' и '`1`', — описание дороги. Если i-й символ строки равен '`1`', то рядом с i-й остановкой есть деревня, а если '`0`', то нет. Гарантируется, что всегда есть хотя бы одна деревня, то есть s содержит не менее одного символа '`1`'. Длина строки s не превосходит 300 000 символов.

Формат выходных данных

Пусть всего во входных данных m деревень, то есть m символов строки s равны '`1`'. Пронумеруем деревни слева направо вдоль дороги, то есть от начала строки s к концу. Необходимо вывести ровно m целых чисел, i-е из которых должно быть равно длине маршрута автобуса, выезжающего из i-й деревни.

Примеры

стандартный ввод	стандартный вывод
2 101101	6 6 4 0
1 10010001000	6 8 0
10 111	4 2 0

Замечание

Рассмотрим первый пример.

1. Автобус из деревни, расположенной у остановки 1, будет ездить до деревни, расположенной у остановки 4, потому что это вторая деревня на его пути.

2. Автобус из деревни, расположенной у остановки 3, будет ездить до деревни, расположенной у остановки 6, потому что это вторая деревня на его пути.

3. Автобус из деревни, расположенной у остановки 4, будет ездить до деревни, расположенной у остановки 6, потому что это последняя деревня вдоль дороги.

4. Автобус из деревни, расположенной у остановки 6, будет ездить до деревни, расположенной у остановки 6, потому что это последняя деревня вдоль дороги.

Система оценки

Тесты к этой задаче состоят из трёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Тесты	Баллы за группу	Ограничения		Комментарии
			K	S	
0	1 – 3	0	—	—	Тесты из условия.
1	4 – 27	55	$k \leqslant 1000$	$\lvert s\rvert \leqslant 1000$	
2	28 – ∞	45	–	–	**Offline-проверка.**

Задача C. Дома в Берляндии

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

Как известно, в столице Берляндии есть n перекрёстков. На некоторых их этих перекрёстков находятся жилые дома. Пусть a_i — это количество человек в семье, проживающей в доме, который находится на перекрёстке i для всех $1 \leqslant i \leqslant n$. Если a_i равно нулю, будем считать, что на этом i-м перекрёстке нет жилого дома.

Некоторые перекрёстки соединены дорогами. Всего в Берляндии m дорог. По каждой дороге можно ходить в любом направлении. Каждая дорога соединяет два различных перекрёстка. Любые два перекрёстка соединены не более чем одной дорогой. От любого перекрёстка можно добраться по дорогам города до любого другого перекрёстка. Назовём расстоянием между перекрёстками i и j минимальное количество дорог, по которым нужно пройти, чтобы добраться от перекрёстка i до перекрёстка j.

Когда одна семья ходит в гости к другой семье, каждый человек находит себе ровно одного собеседника из другой семьи, и они вдвоем разговаривают между собой. Двум семьям общаться скучно, если какой-то человек останется без собеседника.

По какой-то непонятной причине правительство Берляндии попросило вас найти два ближайших жилых дома v и u, таких что если семья из дома v придёт в гости к семье из дома u, то им будет скучно общаться, то есть $a_v > 0$, $a_u > 0$, $a_v \neq a_u$ и расстояние от v до u минимально. Гарантируется, что существуют два жилых дома с различным количеством жильцов.

Формат входных данных

В первой строке входных данных заданы числа n и m ($2 \leqslant n \leqslant 1\,000\,000, 1 \leqslant m \leqslant 1\,000\,000$) — количество перекрёстков в столице Берляндии и количество двунаправленных дорог соответственно.

Во второй строке входных данных заданы n чисел ($0 \leqslant a_i \leqslant 10^9$), где a_i равно 0, если на i-м перекрёстке нет жилого дома, или a_i равно количеству человек в семье, проживающей в доме, который находится на i-м перекрёстке.

В следующих m строках заданы дороги. В i-й из этих строк заданы два числа v_i и u_i ($1 \leqslant v_i, u_i \leqslant n, v_i \neq u_i$) — номера перекрёстков, соединённых соответствующей дорогой.

Формат выходных данных

В единственной строке выведите одно целое число — минимальное расстояние между двумя жилыми домами с различным количеством жильцов.

Примеры

стандартный ввод	стандартный вывод
4 4 1 0 2 0 1 2 2 3 3 4 4 1	2
5 4 1 0 2 1 0 1 5 4 5 5 2 2 3	3

Замечание

В первом примере населены только два дома — у перекрёстков 1 и 3 и количество жильцов в них разное. Расстояние между домами равно 2, кратчайший путь использует дороги 1 и 2.

Система оценки

Тесты к этой задаче состоят из четырёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп.

Группа	Тесты	Баллы	Дополнительные ограничения	Комментарий
			n, m	
0	1 – 2	0	–	Тесты из условия.
1	3 – 53	30	$n, m \leqslant 1000$	
2	54 – 100	50	$n, m \leqslant 100\,000$	
3	101 – 147	20	–	

Задача D. НОД объединяет

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

На одну из кафедр филфака университета Байтландии поступили n студентов. Поскольку они совершенно не знакомы друг с другом, им необходимо найти способ быстро и надежно передавать друг другу информацию о предстоящих экзаменах и зачётах.

Для этого студенты создадут несколько диалогов в новом Байтландском мессенджере. В каждом диалоге будет участвовать ровно два студента. Информация будет распространяться следующим образом: после того, как один из студентов узнал о предстоящей контрольной работе, он рассылает это сообщение во все диалоги со своими одногруппниками. После этого все те люди, которым он разослал сообщения и которые его ещё не получали, рассылают сообщение во все диалоги, в которых они участвуют и так далее.

Староста загорелся идеей построить сеть диалогов таким образом, чтобы каждый получил сообщение хотя бы один раз. При этом он хочет, чтобы количество диалогов было минимально возможным. Поскольку таких вариантов всё ещё очень много, староста придумал свою странную оценку качества сети диалогов: каждому человеку из группы он сопоставил число a_i. Он считает, что оценка одного диалога, созданного между одногруппником u и одногруппником v, равна (a_u, a_v), где (x, y) — наибольший общий делитель чисел x и y, то есть максимальное целое положительное d, которое делит и x, и y. Староста хочет, чтобы суммарная оценка всех диалогов была как можно больше.

Помогите старосте найти максимальную суммарную оценку!

Формат входных данных

В первой строке входных данных дано единственное число n ($1 \leqslant n \leqslant 100\,000$) — количество студентов на кафедре филфака университета Байтландии.

Во второй строке даны n целых положительных чисел a_i ($1 \leqslant a_i \leqslant 1\,000\,000$) — числа, которые староста сопоставил студенту.

Формат выходных данных

Выведите единственное целое число — максимальную суммарную оценку сети, которую может построить староста, при условии, что количество диалогов в сети будет минимально.

Примеры

стандартный ввод	стандартный вывод
5 1 2 3 4 5	5
2 5 10	5

Замечание

В первом примере оптимальным ответом будет создать диалог между следующими парами студентов: $(1, 2)$, $(2, 3)$, $(2, 4)$, $(4, 5)$. Суммарная оценка такой сети диалогов равняется $1 + 1 + 2 + 1 = 5$.

Система оценки

Тесты к этой задаче состоят из четырёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Тесты	Баллы	Ограничения		Комментарий
			n	a_i	
0	1 – 2	0	–	–	Тесты из условия
1	3 – 32	30	$n \leqslant 3000$	$a_i \leqslant 3000$	
2	33 – 59	30	$n \leqslant 3000$	$a_i \leqslant 1\,000\,000$	
3	60 – 89	40	–	–	**Offline-проверка.**

Задача E. Вупсень и Пупсень

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3.5 секунд
Ограничение по памяти:	128 мегабайт

Вупсень очень любит давать задачи на поиск наибольшей общей подпоследовательности. Пупсень очень любит давать задачи на поиск наибольшей правильной скобочной подпоследовательности. Нет ничего удивительного в том, что они решили объединиться и подготовить очень сложную задачу на поиск наибольшей общей правильной скобочной подпоследовательности.

Подпоследовательностью строки a называется такая строка b, которую можно получить удалением из строки a символов на каких-либо (возможно, никаких) позициях.

Последовательность круглых скобок называется *правильной* в следующих случаях:

1. Если она пустая.
2. Если она состоит из правильной скобочной последовательности, заключённой в скобки.
3. Если она состоит из двух правильных скобочных последовательностей, записанных одна за другой.

Вам даны две строки s и t, состоящие из круглых открывающих и закрывающих скобок. Найдите правильную скобочную последовательность w максимальной длины, являющуюся подпоследовательностью строк s и t.

Формат входных данных

Две строки s и t из круглых скобок, длины которых не превосходят n ($1 \leqslant n \leqslant 700$), по одной в строке. Любая из строк (в том числе обе) может быть пустой.

Формат выходных данных

Выведите одну строку w — наибольшую общую правильную скобочную подпоследовательность исходных строк s и t. Если таких строк несколько, разрешается вывести любую из них.

Примеры

стандартный ввод	стандартный вывод
()(()()())(())(())	(())()
))(((())	

Система оценки

Тесты к этой задаче состоят из четырёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп.

Группа	Тесты	Баллы	Дополнительные ограничения	Комментарий
			n	
0	1 – 2	0	–	Тесты из условия.
1	3 – 18	20	$n \leqslant 10$	
2	19 – 46	30	$n \leqslant 50$	
3	47 – 74	30	$n \leqslant 300$	
4	75 – 88	20	–	

Задача F. Трудовые будни

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Бригадир Павел руководит командой рабочих, занимающихся возведением концертного зала по новейшему проекту иностранных архитекторов. Главной особенностью здания должна стать колоннада у главного входа, состоящая из n колонн. При этом, каждая из колонн, вопреки классическим архитектурным представлениям, будет иметь свою высоту, не совпадающую с высотой крыши над входом. По текущему плану высоты колонн составляют $a_1, a_2, \ldots, a_n$ метров относительно уровня крыши в порядке следования слева направо (например, высота в 10 метров означает, что колонна выдаётся на десять метров над крышей, а высота в -5 метров означает, что между верхом колонны и крышей остаётся зазор в пять метров).

За три дня до сдачи объекта и торжественного открытия зала архитекторы прибыли на место строительства и изменили проект, выдвинув новое требование: в соответствии с последними веяниями европейской моды разность высот любых двух соседних колонн должна быть одной и той же, то есть, для любых двух целых i и j от 1 до $n-1$ должно выполняться условие: $a_{i+1} - a_i = a_{j+1} - a_j$. Точное значение высоты каждой колонны при этом не имеет значения. По техническим причинам колонны могут только иметь высоту, выражающуюся целым числом метров относительно уровня крыши.

Изменение высоты колонны на x метров как в сторону увеличения, так и в сторону уменьшения, будет стоить x бурлей. Павел просит вас помочь ему выбрать новую высоту для каждой колонны так, чтобы выполнить поставленное требование и затратить при этом суммарно как можно меньше денег на изменение высот колонн. Помогите ему, или его больше никогда не будут приглашать возводить здания по иностранным проектам.

Формат входных данных

В первой строке входных даных находится целое число n ($2 \leqslant n \leqslant 3\,000\,000$) — количество колонн перед входом в здание.

Во второй строке следуют n целых чисел $a_1, a_2, \ldots, a_n$ ($-10^9 \leqslant a_i \leqslant 10^9$) — текущие высоты колонн.

Формат выходных данных

Выведите два целых числа: высоту первой колонны и разность высот между двумя соседними колоннами в оптимальном плане.

Абсолютная величина обоих выведенных чисел не должна превышать 10^{16}. Гарантируется, что существует оптимальный ответ, удовлетворяющий этому условию.

Примеры

стандартный ввод	стандартный вывод
2 3 -3	3 -6
5 3 8 10 13 20	3 4

Замечание

В первом тесте из условия менять высоту колонн не требуется, так как их всего две, и они автоматически удовлетворяют поставленному требованию.

Во втором тесте из условия оптимальным набором высот колонн будет $3, 7, 11, 15, 19$, а суммарная стоимость изменений есть $|3-3| + |7-8| + |11-10| + |15-13| + |19-20| = 5$.

Система оценки

В данной задаче 102 теста, два из которых являются тестами из условия, а каждый из оставшихся оценивается независимо и стоит 1 балл. Задача является полностью **Offline**, то есть результаты

проверки вашего решения станут доступны уже после окончания соревнования. Ваше решение будет проверяться на основном наборе тестов только в случае успешного прохождения всех тестов из условия.

Гарантируется, что решения, корректно работающие при дополнительных условиях $n \leqslant 50$ и $-50 \leqslant a_i \leqslant 50$, наберут не менее 20 баллов.

Гарантируется, что решения, корректно работающие при дополнительных условиях $n \leqslant 5000$ и $-5000 \leqslant a_i \leqslant 5000$, наберут не менее 40 баллов.

Гарантируется, что решения, корректно работающие при дополнительном условии $n \leqslant 100\,000$, наберут не менее 60 баллов.

Задача G. Включи свет, закрой двери!

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Это **интерактивная** задача. В секции «Протокол взаимодействия» вы найдёте информацию о том, как сбрасывать буфер вывода (делать операцию 'flush').

В новом квесте, набирающем популярность в городе N, игроку необходимо выбраться из лабиринта, состоящего из неизвестного числа комнат. В каждой комнате имеется некоторое количество дверей, одна лампочка и два переключателя. Первый переключатель в каждой из комнат отвечает за включение и выключение лампочки и может быть использован произвольное количество раз. Второй переключатель может быть активирован только один раз и он запирает все двери, ведущие из данной комнаты, сразу после того как игрок пройдет через одну из них, после чего комната становится недоступной.

Изначально все комнаты открыты, и в некоторых из них горит свет. Целью игры является включить свет во всех комнатах лабиринта, а также закрыть все комнаты, кроме одной (любой). Данная задача была бы простой, если бы у игрока была возможность оставлять в комнатах какие-то отметки, но это запрещено правилами квеста. Когда игрок заходит в какую-либо комнату, он видит только количество дверей в этой комнате, горит или не горит в ней свет, а также знает через какую дверь он только что вошёл. Заходя в комнату, игрок всегда видит двери в одном и том же порядке и нумерует из одинаково.

Игрок может выполнять следующие действия:

- Включить свет в комнате, где он сейчас находится.
- Выключить свет в комнате, где он сейчас находится.
- Попробовать пройти через какую-то из дверей, данное действие завершится успехом, если комната по ту сторону двери не заперта. Проходя через какую-либо из дверей можно также активировать закрытие покидаемой комнаты.
- Завершить игру, если игрок считает, что он уже запер все комнаты кроме текущей и включил везде свет.

Формат входных данных

Изначально программе на ввод подаётся информация о том включен ли свет в стартовой комнате, сколько в этой комнате дверей и число 0 (поскольку игрок не пришёл в эту комнату через какую-то из дверей).

Гарантируется, что число комнат не превосходит 50, а число дверей не превосходит 100 (каждая дверь соединяет две комнаты и считается один раз). Также гарантируется, что из любой комнаты можно дойти до любой другой, двигаясь только с использованием имеющихся дверей, и что никакая пара комнат не соединена более чем одной дверью.

В секции «Примеры» приведены некоторые варианты возможных конфигураций лабиринта. Первая строка содержит количество комнат и количество дверей в лабиринте, затем идёт описание начального состояния лампочки в комнате (1 если лампочка горит, и 0, если не горит), далее следуют пары комнат, соединённых дверьми, а в конце содержится одно число — номер стартовой комнаты. Обратите внимание, данное описание показывает структуру первых трёх тестов, но **не соответствует тому, что ваша программа получит на вход**. Пример возможного взаимодействия вашей и проверяющей программ приведён в разделе «Замечание».

Формат выходных данных

Число запросов, сделанных вашей программой, не должно превосходить 30 000. При нарушении этого ограничения вы получите вердикт `Wrong Answer` (неправильный ответ). Завершение игры также считается одним запросом.

После выполнения **каждого** запроса необходимо сбрасывать буфер вывода (делать операцию 'flush'), для этого можно использовать:

- `fflush(stdout)` в языке C++;
- `System.out.flush()` в Java;
- `stdout.flush()` в Python;
- `flush(output)` в Pascal;
- смотрите документацию для других языков.

В секции «Примеры» в графе выходные данные приведено число запросов, использованных некоторым из решений жюри. Данное число следует игнорировать.

Протокол взаимодействия

Для выполнения запросов используйте следующий протокол взаимодействия с проверяющей программой:

- «`turn on`» — включить свет в текущей комнате. Если свет уже включен, то ничего не происходит. Проверяющая программа никак не отвечает на данный запрос.
- «`turn off`» — выключить свет в текущей комнате. Если свет уже выключен, то ничего не происходит. Проверяющая программа никак не отвечает на данный запрос.
- «`go edgeId keep|lock`» — пойти в дверь номер *edgeId*, считая что двери нумеруются начиная с 1. При этом параметр `keep` означает, что комната остается открытой, а `lock` означает, что после выхода из комнаты она закроется и больше не будет доступна. Ответом проверяющей программы будет:
  - `locked`, если комната, в которую ведёт данная дверь, закрыта. В этом случае игрок остаётся в данной комнате а действие `lock` (если оно было указано) не выполняется.
  - строка `on|off edgeCount edgeId`, если игрок успешно перейдёт в другую комнату. Слово `on` означает, что свет в текущей комнате включен, а слово `off` — что выключен. *edgeCount* равняется количеству дверей в данной комнате, а *edgeId* — номеру двери, через которую игрок только что пришёл.
- «`done`» — завершить игру. Данный запрос выполняется в конце, когда игрок считает, что уже включил свет во всех комнатах и запер все комнаты, кроме той в которой сейчас находится. Обратите внимание, что игрок может закончить игру в любой комнате. Если в какой-то комнате свет будет выключен или какая-то комната кроме данной будет не заперта, то решение получит вердикт `Wrong Answer`. После выполнения данного запроса программа должна немедленно завершить работу.

Если Ваша программа выведет какую-то другую команду или попытается перейти по двери с номерм, которого не существует, то она получит вердит Presentation error.

Система оценки

Тесты к этой задаче состоят из четырёх групп. Баллы за каждую группу ставятся при прохождении всех тестов данной группы, но прохождение тестов каких-либо других групп (в том числе тестов из условия) не требуется.

Группа	Тесты	Баллы за группу	Комментарии
0	1 – 3	0	Тесты из условия
1	4 – 19	30	Граф является деревом
2	20 – 29	30	Граф является циклом
3	30 – 46	40	Дополнительные ограничения отсутствуют

Примеры

стандартный ввод	стандартный вывод
2 1 0 0 1 2 1	7
3 2 1 1 1 1 2 2 3 2	17
3 3 0 1 1 1 2 2 3 3 1 1	43

Замечание

Комментарий о том, что граф является деревом, означает, что в лабиринте не существует ни одного циклического маршрута длины больше двух, не посещающего никакую комнату дважды.

Комментарий о том, что граф является циклом, означает что все комнаты лежат на одном циклическом маршруте, не посещающем никакую комнату дважды.

Ниже приведён пример возможного протокола взаимодействия на первом тестовом примере. Левый столбец соответствует выводу проверяющей программы, а правый — возможному выводу программы участника.

```
off 1 0
                turn on
                go 1 keep
off 1 1
                turn off
                turn on
                turn on
                go 1 lock
on 1 1
                go 1 lock
locked
                turn off
                turn on
                done
```

Задача H. Бинарная игра

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Искандер и Оля любят придумывать ребусы. Но больше, чем придумывать ребусы, они любят придумывать какие-нибудь игры на строках. Вот и сейчас им в голову пришла забавная игра со следующими правилами:

- Выбирается какой-то набор *запрещённых* двоичных (состоящих из нулей и единиц) строк $f_1, f_2, \ldots, f_n$.
- Выбирается некоторая стартовая бинарная строка s, такая что ни одна из запрещённых строк не входит в неё как подстрока.
- Игроки по очереди дописывают в конец строки s по одному символу «`0`» или «`1`». Оля ходит первой.
- Проигрывает тот, после чьего хода хотя бы одна из запрещённых строк $f_1, f_2, \ldots f_n$ входит в s как подстрока.
- В случае если при оптимальной игре обоих игроков игра может продолжаться сколь угодно долго, то объявляется ничья.

Вы обожаете портить другим людям их любимые развлечения, поэтому решили написать программу, которая будет определять исход игры по заданному набору запрещённых строк и стартовой строке s.

Формат входных данных

В первой строке входных данных записаны два целых числа n и m ($0 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 1\,000\,000$) — количество запрещённых строк и изначальная длина строки s.

В каждой из последующих n строк содержится одна запрещённая строка. Гарантируется, что все эти строки непусты, состоят из символов «`0`» и «`1`» и никакая из них не является подстрокой строки s. Дополнительно гарантируется, что **суммарная длина** всех запрещённых строк не превосходит $1\,000\,000$.

В последней строке входных данных записана стартовая строка s длины m, состоящая только из символов «`0`» и «`1`». Обратите внимание, строка s может быть пустой, в этом случае соответствующая строка входных данных отсутствует (в том числе символ перевода строки). Длина s не превосходит $1\,000\,000$.

Формат выходных данных

В зависимости от результата игры при оптимальной игре обоих игроков выведите:

- «`Olya`» (без кавычек), если Оля может победить вне зависимости от того как будет играть Искандер. Напомним, что Оля ходит первой.
- «`Iskander`» (без кавычек), если Искандер может победить не зависимо от ходов Оли.
- «`Friendship`» (без кавычек), если при оптимальной игре обоих игроков игра будет продолжаться бесконечно долго.

Примеры

стандартный ввод	стандартный вывод
1 0 1	Friendship
3 1 000 001 011 0	Olya
2 3 1001 000 100	Iskander

Замечание

В первом примере строка s изначально пустая. Любой из игроков может не проиграть на любом ходу просто приписав к s символ «0».

Система оценки

Тесты к этой задаче состоят из пяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп, кроме тестов из условия. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

В таблице ниже через L обозначается суммарная длина всех строк, то есть $L = \sum_{i=1}^{n} |f_i|$.

Группа	Тесты	Баллы	Дополнительные ограничения			Комментарий
			n	m, $\lvert f_i\rvert$	L	
0	1 – 3	0	–	–	–	Тесты из условия.
1	4 – 33	25	$n \leqslant 2$	$\lvert f_i\rvert, m \leqslant 10$	–	
2	34 – 70	25	$n \leqslant 150$	$\lvert f_i\rvert, m \leqslant 10$	$L \leqslant 500$	
3	71 – 117	25	$n \leqslant 500$	$\lvert f_i\rvert, m \leqslant 500$	$L \leqslant 500$	
4	118 – ∞	25	–	–	–	**Offline-проверка**.