Задача A. Плотники

Имя входного файла:	`carpenters.in`
Имя выходного файла:	`carpenters.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Поспорили, как-то раз, два плотника кто из них в своем ремесле искуснее. Чтобы выяснить все по-честному, было решено устроить соревнование по распилу квадратного куска фанеры на несколько прямоугольников. Побеждал тот, кто распилит как можно ровнее и с минимальным количеством опилок. Вас они пригласили в качестве судьи.

Как только плотники распилили выданные им квадраты, они принесли все вам на проверку. Велико же было ваше удивление, когда оказалось, что оба получили абсолютно ровные прямоугольники! Единственное, что осталось проверить — можно ли сложить данные прямоугольники в квадрат. Ведь может быть, что кто-то из участников сжульничал и не принес какую-то часть квадрата. Чтобы все было точно, вы решили написать программу, которая определяла бы можно ли из данных прямоугольников сложить квадрат или нет.

Формат входного файла

В первой строке входного файла задано целое число N — количество прямоугольников ($1 \le N \le 7$). Следующие N строк содержат по два целых числа W и H каждая. Это длина и ширина соответствующего прямоугольного куска фанеры. Числа в строке разделяются одним пробелом. Гарантируется, что ($1 \le W, H \le 10$).

Формат выходного файла

В выходной файл вывести слово "YES" (большими буквами и без кавычек) если из данных прямоугольников можно сложить целый квадрат так, чтобы никакие куски не накладывались бы друг не друга. Иначе, выведите "NO" (большими буквами и без кавычек).

Пример

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

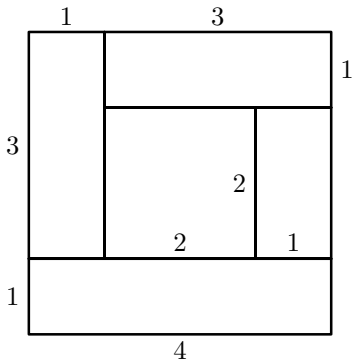

Иллюстрация к первому примеру.

Задача B. Необычный конь

Имя входного файла:	`knight.in`
Имя выходного файла:	`knight.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Как хорошо известно, обычный шахматный конь - это фигура, которая ходит сначала по горизонтали или вертикали на две клетки, а потом в перпендикулярном направлении еще на одну клетку. Таким образом, траектория его движения похожа на букву "Г".

Однако, в этой задаче мы будем рассматривать необычного шахматного коня. Он отличается тем, что у него есть направление, куда он смотрит. А именно он может смотреть в одном из четырех направлений: север, юг, запад, восток.

Мы будем рассматривать шахматную доску из N рядов и N столбцов. Нумерация ведется числами от 1 до N, ряды нумеруются сверху вниз, а столбцы — слева направо. Направление "север" соответствует увеличению номера ряда, "юг" — уменьшению номера ряда, "запад" — уменьшению номера столбца и, наконец, "восток" — увеличению номера столбца.

Необычный конь может совершать всего четыре хода: поворот на 90 градусов налево, поворот на 90 градусов направо, прыжок налево и прыжок направо. Под прыжком налево и направо понимается ход на две клетки вперед, т.е. туда, куда смотрит конь, а потом на одну клетку налево или направо (относительно направления взгляда коня) соответственно. Выходить за пределы доски запрещено. При прыжке, направление взгляда коня не меняется.

Вам даны координаты начальной клетки, где стоит конь, направление взгляда коня, а также координаты конечной клетки. Ваша задача — найти минимальное количество ходов, за которое необычный конь может дойти от начальной до конечной клетки.

Формат входного файла

В первой строке входного файла задано целое число N — размер доски ($2 \le N \le 20$). Во второй строке через пробел заданы два целых числа C и R — номер колонки и ряда начальной клетки. В третьей строке задана одна буква из набора: 'N', 'S', 'E', 'W' — направление взгляда коня (север, юг, восток или запад, соответственно). И, наконец, в четвертой строке через пробел идут два целых числа C_2 и R_2 — номер колонки и ряда конечной клетки. Гарантируется, что все данные корректны и начальная клетка отличается от конечной.

Формат выходного файла

В первой строке выходного файла выведите минимальное количество ходов, за которое конь может совершить требуемый путь. Если попасть из начальной клетки в конечну невозможно, то выведите -1.

Пример

knight.in	knight.out
5 2 3 S 5 5	5
3 1 1 E 2 2	-1

Задача C. Денежный перевод

Имя входного файла:	`money.in`
Имя выходного файла:	`money.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Ваш друг хочет сделать денежный перевод своим родителям в одну из стран ближнего зарубежья. Всего он планирует перевести N рублей. Однако условия перевода в банке оказались довольно странные. Ему сказали, что возможно выгоднее будет разбить сумму на несколько частей. В частности, банк готов существенно снизить комиссию, если каждая часть суммы будет представлена положительным числом больше единицы, у которого нет простых делителей, кроме 2 или 3. Более того, никакая часть не должна делиться нацело на другую часть.

Ваш друг не силен в математике, поэтому он попросил вас помочь ему. А вы, чтобы не ошибиться в расчетах, решили написать программу, которая по данной сумме N найдет нужное разбиение, или скажет что это невозможно.

Формат входного файла

В первой строке входного файла задано целое число N — общая сумма перевода ($1 \le N \le 10^{18}$).

Формат выходного файла

В первой строке выходного файла выведите количество слагаемых в найденном разбиении, или, если его не существует, то выведите 0. Во второй строке, через пробел, выведите все слагаемые в любом порядке. Если ответов несколько, то выведите любой из них.

Пример

money.in	money.out
10	2 6 4
1000	3 96 648 256
1	0

Задача D. Формула?

Имя входного файла:	`formula.in`
Имя выходного файла:	`formula.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

У короля Байтландии есть придворный математик. Но в последнее время формулы стали подводить мудреца. Чашу королевского терпения переполнил вчерашний случай: монарх приехал на пляж позагорать, а тут вдруг — солнечное затмение. А ведь математик утверждал, что точно вычислил по формулам, что затмение будет только через год.

Решил король проверить, не дурачит ли его математик. Для этого задал он ему следующую задачу (которую вычитал в каком-то древнегреческом журнале).

Пусть дано множество A, состоящее из n положительных чисел $a_1, a_2, ..., a_n$. Определим на нем функцию:

$$\sigma(A) = \frac{\sum\limits_{i=1}^{n} a_i}{\prod\limits_{i=1}^{n} a_i}$$

Другими словами, значение этой функции — это отношение суммы элементов множества A к их произведению.

Далее, определим другую функцию:

$$\lambda(A) = \sum_{S \subseteq A} \sigma(S)$$

Здесь, суммирование ведется по всем непустым подмножествам A.

Вопрос задачи: найти значение функции λ от множества всех натуральных чисел от 1 до N, т.е. $\lambda(\{1, 2, ..., N\})$.

Король задал эту задачу математику, но как-то не подумал, что сам-то он ее решить не может, а значит и проверить правильность решения ему не под силу. Вас он просит написать программу, которая находила бы значение функции λ в зависимости от N. Используя ее, король, по крайней мере, сможет проверить правильность ответа.

Формат входного файла

В первой строке входного файла задано целое число N ($1 \le N \le 1000$).

Формат выходного файла

В первой строке выходного файла выведите ответ на вопрос задачи. Абсолютная погрешность не должна превышать 10^{-6}.

Пример

formula.in	formula.out
1	1
2	3.5
5	21.3

Задача E. Сортировка

Имя входного файла:	`sorting.in`
Имя выходного файла:	`sorting.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Задача очень проста. Вам дано N целых чисел $a_1, a_2, ..., a_N$. Все, что нужно сделать — отсортировать их по не убыванию. Правда вам разрешается производить только одно действие: выбрать любой элемент и перевернуть последовательности чисел справа и слева от него. Например, если вам даны числа $(7, 1, 3, 9, 8)$, то при выборе первого элемента вы получите набор $(7, 8, 9, 3, 1)$, при выборе второго — $(7, 1, 8, 9, 3)$, третьего — $(1, 7, 3, 8, 9)$, четвертого — $(3, 1, 7, 9, 8)$, и, наконец, пятого — $(9, 3, 1, 7, 8)$.

Отсортируйте массив по не убыванию (слева направо), используя данное действие ноль или более раз.

Формат входного файла

В первой строке входного файла задано целое число N — количество чисел ($1 \le N \le 100$). Во второй строке, через пробел, идут сами числа. Гарантируется, что числа будут целыми и не больше 1000 по модулю.

Формат выходного файла

В первой строке выходного файла выведите количество действий (не более 11000), произведенных над набором чисел. Если отсортировать числа нельзя, или в лучшем случае придется произвести более 11000 действий, то выведите -1. Иначе, во второй строке, через пробел, выведите действия в том порядке, в котором они должны быть произведены, чтобы отсортировать данный набор. Каждое действие определяется позицией выбранного элемента. Элементы нумеруются, начиная с единицы, слева направо. Если ответов несколько, то выведите любой из них.

Пример

sorting.in	sorting.out
5 7 1 3 9 8	2 4 3
2 2 1	-1
3 1 2 2	0

Задача F. Колокола

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

Король Байтландии очень любит слушать по утрам колокольный перезвон. Эти звуки наполняют его силами на целый день. Однажды он решил узнать у звонаря, как тот играет на этих колоколах.

Оказалось, что у звонаря имеется N колоколов. Для того, чтобы сыграть какую-то мелодию, звонарь задает в своем хитром механизме для каждого колокола четыре целочисленных параметра: S, A, B и M. S — секунда относительно начала мелодии, в которую данный колокол ударит в первый раз. Все остальные удары колокола описываются остальными тремя параметрами, а именно, если колокол ударил в секунду X, то в следующий раз он ударит ровно через $((A \cdot X + B) \bmod M) + 1$ секунд.

Король мало что понял из этих объяснений звонаря и приказал ему составить список колоколов в том порядке, в котором они ударяют. Звонарь не знает, как это сделать, поэтому он попросил вас написать программу, которая быстро могла бы сказать, какой из колоколов ударяет K-ым по счету.

Помогите ему!

Формат входного файла

В первой строке входного файла задано целое число N — количество колоколов у звонаря ($1 \le N \le 10^4$). В следующих N строках идет описание колоколов. Каждая из этих строк содержит четыре целых числа, разделенных пробелом. Это параметры S, A, B и M соответствующего колокола ($0 \le S \le 10^9$; $0 \le A, B \le 10^6$; $2 \le M \le 10^6$). И, наконец, в последней строке содержится целое число K ($1 \le K \le 10^6$).

Формат выходного файла

В первой строке выходного файла выведите номер колокола, который ударит K-ым по счету. Нумерация колоколов ведется с единицы и соответствует порядку, в котором они заданы во входном файле. Если в какую-то секунду одновременно ударит несколько колоколов, то будем считать, что они ударяют в порядке их номеров во входном файле.

Пример

bells.in	bells.out
2 5 3 7 2 0 0 0 100 6	1
2 5 3 7 2 0 0 0 100 7	2

Задача G. Жюри олимпиады

Имя входного файла:	jury.in
Имя выходного файла:	jury.out
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

В этой задаче вы почувствуете себя на месте жюри олимпиады, т.к. вам предстоит написать программу проверки правильности ответа на задачу "Сортировка". Подобную программу пришлось писать жюри, чтобы автоматически проверять ваши решения.

Итак, сформулируем задачу. Вам дано N целых чисел $a_1, a_2, ..., a_N$. Все, что нужно сделать — отсортировать их по не убыванию. Правда вам разрешается производить только одно действие: выбрать любой элемент и перевернуть последовательности чисел справа и слева от него. Например, если вам даны числа $(7, 1, 3, 9, 8)$, то при выборе первого элемента вы получите набор $(7, 8, 9, 3, 1)$, при выборе второго — $(7, 1, 8, 9, 3)$, третьего — $(1, 7, 3, 8, 9)$, четвертого — $(3, 1, 7, 9, 8)$, и, наконец, пятого — $(9, 3, 1, 7, 8)$.

Кто-то дал вам последовательность действий в виде номеров элементов, которые следует выбирать. И этот кто-то утверждает, что, выполнив эти действия по порядку, вы получите отсортированный по не убыванию набор чисел.

Ваша задача проверить, прав этот кто-то или нет.

Формат входного файла

В первой строке входного файла задано целое число N — количество чисел ($1 \le N \le 1000$). Во второй строке, через пробел, идут сами числа. Гарантируется, что числа будут целыми и не больше 1000 по модулю. В третьей строке идет целое число K — количество действий ($0 \le K \le 10^4$). В четвертой строке, через пробел, заданы действия в том порядке, в котором их надо выполнять. Все действия корректны, т.е. их можно выполнить.

Формат выходного файла

В первой строке выходного файла выведите "YES" (большими буквами и без кавычек) если приведенная последовательность действий действительно сортирует данный набор чисел, иначе выведите "NO" (большими буквами и без кавычек).

Пример

jury.in	jury.out
5 7 1 3 9 8 2 4 3	YES
5 7 1 3 9 8 2 4 2	NO
1 10 10 1 1 1 1 1 1 1 1 1 1	YES
3 1 2 2 0	YES

Задача H. Двойные подстроки

Имя входного файла:	`string.in`
Имя выходного файла:	`string.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Один ваш друг захотел написать свой архиватор. Принцип архивирования, который он хочет использовать, заключается в кодировании двойных подстрок текста одной. То есть, если где-то в тексте встретится подстрока вида $\alpha\alpha$, то он заменит ее на что-то вроде $2(\alpha)$. Но, прежде чем начинать процесс архивирования, он хочет узнать, есть ли вообще в тексте такие двойные подстроки.

Он просит вас помочь ему написать эту часть кода, поскольку она оказалась неожиданно трудной для него.

Формат входного файла

В первой строке входного файла задана строка для кодирования. Гарантируется, что она будет содержать только латинские буквы (большие и маленькие) и ее длина не превзойдет 10^5. В этой задаче регистр символов имеет значение, т.е. 'A' не то же самое, что 'a'.

Формат выходного файла

В первой строке выходного файла выведите, через пробел, два числа: позицию начала двойной подстроки, а затем ее длину. Позиции в строке нумеруются с единицы. Если в строке нет ни одной двойной подстроки, то выведите два числа 0 0, разделенные пробелом. Если ответов несколько, то выведите любой из них.

Пример

string.in	string.out
abcdefg	0 0
blabla	1 6
Blabla	0 0
TheQueen	6 2

Задача I. Супермаркет

Имя входного файла:	`supermarket.in`
Имя выходного файла:	`supermarket.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Один супермаркет заказал вам написать программу, которая определяла бы, видит охранник посетителей или нет. В дальнейшем, эту программу планируется использовать для нахождения оптимального места для камеры наблюдения. Схематически представим себе вид сверху, тогда посетители супермаркета будут обозначены точками, а стенды с товарами отрезками. Будем считать, что два человека видят друг друга, тогда и только тогда, когда отрезок, соединяющий их не имеет ни одной общей точки, ни с одним из отрезков, обозначающих стенды.

В супермаркете есть один охранник и несколько посетителей. Для каждого посетителя вам надо определить, видит его охранник или нет.

Формат входного файла

В первой строке входного файла заданы два целых числа: N — количество посетителей в супермаркете и M — количество стендов ($1 \le N, M \le 100$). В следующих N строках заданы посетители своими координатами X и Y. Затем, идет M строк с координатами X_1, Y_1, X_2, Y_2, которые задают концы отрезков стендов. И, наконец, в последней строке заданы координаты охранника. Гарантируется, что все координаты заданы целыми числами и не превосходят по модулю 10000, а также все люди находятся в разных точках. Отрезки стендов могут пересекаться и накладываться друг на друга. Ни один человек не будет касаться стенда.

Формат выходного файла

В выходной файл выведите ровно N строк, каждая из которых описывает, видит охранник соответствующего посетителя или нет. Если он его видит, то выведите "SEES" , иначе "DOES NOT SEE". Все строки нужно выводить, используя большие буквы, и без кавычек.

Пример

supermarket.in	supermarket.out
3 2 -1 -1 -2 2 0 4 2 0 -2 4 -2 0 2 4 0 0	SEES DOES NOT SEE DOES NOT SEE