Задача A. Подстановочный шифр

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

Существует два основных типа шифрования: с секретным ключом и с открытым ключом. При шифровании с секретным ключом требуется, чтобы все стороны, имеющие право на прочтение информации, имели один и тот же ключ. Это позволяет свести общую проблему безопасности информации к проблеме обеспечения защиты ключа. Шифрование на секретном ключе также называется симметричным шифрованием, так как для шифрования и дешифрования данных используется один и тот же ключ.

Одним из типов шифров с секретным ключом является подстановочный шифр. Подстановочные шифры существуют уже около 2500 лет. Самым ранним примером является шифр Атбаш. Он возник примерно в 600 году до н.э. и заключался в использовании еврейского алфавита в обратном порядке.

Юлий Цезарь использовал подстановочный шифр, который так и назывался — шифр Цезаря. Этот шифр заключался в замещении каждой буквы другой буквой, расположенной в алфавите на три буквы дальше от шифруемой. Таким образом, буква A преобразовывалась в D, B преобразовывалась в E, a Z преобразовывалась в C. Из этого примера видно, что подстановочный шифр обрабатывает за один раз одну букву открытого текста. Сообщение может быть прочитано обоими абонентами при использовании одной и той же схемы подстановки. Ключом в шифре подстановки является либо число букв сдвига, либо полностью переупорядоченный алфавит.

Таким образом, подстановочный шифр состоит в замещении каждой буквы алфавита некоторой буквой, причем разные буквы замещаются разными. Более формально, пусть в исходном и в зашифрованном сообщениях используются первые k букв латинского алфавита и пусть определена функция $f(c)$, получающая на вход символ и выдающая символ, на который его надо заменить при шифровании. При этом выполняется условие: если $c_1 \neq c_2$, то $f(c_1) \neq f(c_2)$.

Задано исходное сообщение и оно же в зашифрованном виде. Ваша задача состоит в том, чтобы восстановить по ним ключ шифрования (функцию f) или определить, что шифрование выполнялось не шифром подстановочного типа.

Формат входного файла

Первая строка входного файла содержит целое число k ($1 \le k \le 26$). Вторая строка входного файла содержит исходное сообщение s, состоящее только из строчных букв латинского алфавита. Третья строка входного файла содержит зашифрованное сообщение t, состоящее только из строчных букв латинского алфавита. В s и в t используются только первые k букв латинского алфавита, s и t имеют одинаковую положительную длину, не превосходящую 100.

Формат выходного файла

Если существует подстановочный шифр, удовлетворяющий условию задачи, то в первой строке выходного файла выведите слово `POSSIBLE`, иначе — выведите `IMPOSSIBLE`. В случае положительного ответа в последующих k строках выведите описание ключа шифра. Каждая из этих строк должна иметь формат `c -> f(c)`, где c — символ алфавита, а $f(c)$ — символ, на который он заменяется при шифровании. Эти строки должны быть перечислены в алфавитном порядке заменяемых символов.

Если вариантов ключа шифра несколько, выведите любой.

Примеры

cipher.in	cipher.out
4 abcd dcba	POSSIBLE a -> d b -> c c -> b d -> a
4 abcda dcbaa	IMPOSSIBLE
4 ab ba	POSSIBLE a -> b b -> a c -> c d -> d

Задача B. Выход

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

В одном из университетов на выходе стоит вертушка. Она одинаково легко крутиться в обе стороны. Студенты, которые в этом заведении учатся, делятся на 4 категории:

- Обычные — они всегда проходят в направлении вращения пропускного устройства;
- Нонконформисты — они всегда меняют направление вращения вертушки;
- «Правые» — они всегда проходят так, чтобы вертушка крутилась по часовой стрелке (если необходимо, то они её разворачивают);
- «Левые» — такие же, как и правые, но проходят против часовой;

Охранник, который сидит у выхода знает, что в университете учатся a_1 обычных студентов, a_2 нонконформистов, a_3 «правых» и a_4 «левых». Ваша задача — помочь охраннику узнать, за какое наименьшее число смен вращения вертушки все студенты могут выйти из здания. Изначально вертушка крутится по часовой стрелке. Можно считать, что студенты выходят так быстро, что вертушка не успевает остановиться.

Формат входного файла

Четыре неотрицательных целых числа a_1, a_2, a_3 и a_4, каждое из которых не превосходит 10^9.

Формат выходного файла

В выходной файл выведите одно число — ответ на задачу.

Примеры

exit.in	exit.out
1 1 1 1	1
1 0 1 0	0

Задача C. Развлечение с числами

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

Дано два пятизначных числа a и b. Необходимо найти сумму чисел, составленных из трех средних цифр этих чисел.

Например, если $a = 10021$, $b = 12345$, то искомая сумма равна $002 + 234 = 236$.

Формат входного файла

Первая строка входного файла содержит два целых числа a и b ($10000 \le a, b \le 99999$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

fun.in	fun.out
10021 12345	236

Задача D. Система неравенств

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

Задана система неравенств, в которой каждое неравенство имеет вид `x > y`, `x > a`, `x < a` где x и y — некоторые переменные, а a — некоторое целое число от 0 до 100. В этой системе неравенств используются только переменные x, y и z.

Ваша задача — найти число решений этой системы, в которых x, y и z являются целыми числами от 0 до 100.

Формат входного файла

Первая строка входного файла содержит целое число n ($0 \le n \le 25$) — число неравенств в системе. Каждая из последующих строк содержит одно из неравенств.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

ineq.in	ineq.out
6 x > 2 x < 5 y > 2 y < 5 z > 2 z < 5	8
1 x > y	510050

Задача E. k-ый минимум

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

Задано n попарно различных целых чисел $a_1, a_2, \ldots, a_n$. Назовем k-ым минимумом из них такое число $x = a_j$ (для некоторого j), что среди $a_1, \ldots, a_n$ существует ровно $k - 1$ число, меньшее x.

Напишите программу, которая по заданному набору чисел находит k-ый минимум в нем.

Формат входного файла

Первая строка входного файла содержит два целых числа n и k ($1 \le n \le 10^5$, $1 \le k \le min(n, 10)$). Вторая строка входного файла содержит n попарно различных целых чисел $a_1, \ldots, a_n$,находящихся в диапазоне от -10^9 до 10^9.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

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

Задача F. Точки и окружности

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

На плоскости задано n геометрических объектов, каждый их которых является либо точкой, либо окружностью. Необходимо найти расстояние между двумя ближайшими объектами.

Формат входного файла

Первая строка входного файла содержит целое число n ($2 \le n \le 1000$). Каждая из последующих строк описывает один из объектов и содержит три целых числа x, y и r ($|x|, |y| \le 10^4$, $0 \le r \le 10^4$). Если $r = 0$, то этот объект является точкой с координатами (x, y), иначе — окружностью с центром в точке (x, y) и радиусом r.

Формат выходного файла

В выходной файл выведите расстояние между ближайшими объектами с точностью не хуже 10^{-6}.

Примеры

pac.in	pac.out
3 1 0 0 1 1 0 0 0 0	1.00000000
4 0 0 4 35 67 3 0 48 2 10 13 6	6.40121947

Задача G. Снукер

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

На турнире по снукеру (одна из разновидностей бильярда) очень любят подсчитывать различную статистику. Наиболее популярным является подсчет количества сотенных серий — подходов к столу, когда игрок набирает более ста очков.

У организаторов накопилось порядочное количество данных о таких сериях и теперь они просят вас составить таблицу результатов по этим данным. Для этого был разработан метод сравнения двух игроков.

Для начала сравниваются максимальные по количеству очков серии этих игроков, и тот, у кого очков было получено больше, считается лучше другого игрока. Если же максимальные серии совпадают, то лучше считается игрок, вторая по величине серия которого больше. Если же и они совпадают, то сравнеие происходит по третьим сериям и т.д. Если же статистики игроков полностью совпадают, то лучше считается игрок, сделавший сотенную серию первым.

Формат входного файла

В первой строке одно целое число n — количество зафиксированных на турнире сотенных серий ($1 \le n \le 1000$). Далее следуют n строк вида “a_i s_i”, где a_i — количество очков набранных в i-ой сотенной серии ($100 \le a_i \le 155$), а s_i — имя игрока совершившего серию, состоящие из не более чем 30 заглавных и строчных латинских букв, а также пробелов. Имя игрока не может начинаться с пробела или им заканчиваться. Строчные и заглавные буквы считаются различными.

Формат выходного файла

В выходной файл следует вывести таблицу результатов. В первой строке должна быть статистика по лучшему игроку, во второй — по второму и т.д. Статистика по каждому игроку должна содержать имя игрока и далее все его сотенные серии, перечисленные в порядке невозрастания.

Примеры

snooker.in	snooker.out
5 140 Allister Carter 147 Andrew Higginson 147 Allister Carter 140 Andrew Higginson 144 Neil Robertson	Allister Carter 147 140 Andrew Higginson 147 140 Neil Robertson 144

Задача H. Два шланга

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

Иван Петрович — заядлый садовод. Он настолько любит свои посевы, что для их полива использует два шланга, а не один, как это делает его сосед Петр Иванович. Естественным образом эти шланги часто запутываются. Вот и сейчас они, естественно, переплелись.

Чтобы распутать получившуюся конфигурацию шлангов, Иван Петрович вытягивает их вдоль дорожки таким образом, что, идя вдоль дорожки, всё время можно наблюдать два параллельно идущих шланга, но в некоторых местах можно наблюдать, как шланги "меняются местами", то есть один шланг проходит над другим. Иван Петрович классифицировал два вида таких наложений: 1 — дальний шланг проходит сверху, 2 — ближний проходит сверху.

Теперь Ивана Петровича интересует вопрос, если он попросит Петра Ивановича подержать шланги с одного конца, а сам потянет за два других конца, то распутаются ли шланги.

Формат входного файла

В первой строке целое число n — количество наложений шлангов ($1 \le n \le 1000$). Во второй строке n чисел — виды наложений в порядке прохода по дорожке от Петра Ивановича в Ивану Петровичу.

Формат выходного файла

В выходной файл выведите «`YES`», если данным образом удастся распутать шланги, и «`NO`» иначе.

Примеры

twohoses.in	twohoses.out
3 1 1 1	NO
2 1 2	YES