Задача A. Плавные числа

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

Назовём натуральное число плавным, если разность любых двух его соседних цифр не превосходит по модулю единицы. Вам необходимо определить количество N-значных плавных чисел.

Формат входных данных

В единственной строке входного файла одно число N ($1 \leqslant N \leqslant 20$).

Формат выходных данных

Вывести одно число — искомое количество плавных чисел.

Пример

numbers.in	numbers.out
2	26

Задача B. Наибольшая последовательнократная подпоследовательность

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

Для заданной числовой последовательности $a_1, a_2, \ldots, a_n$ требуется найти длину максимальной последовательнократной подпоследовательности.

Для последовательнократной подпоследовательности $a_{k_1}, a_{k_2}, \ldots, a_{k_t}$ ($k_1 < k_2 < \cdots < k_t$) верно, что $a_{k_i} | a_{k_j}$ при $1 \leqslant i < j \leqslant t$ (утверждение «$a|b$» эквивалентно «b кратно a»). Подпоследовательность из одного элемента полагается последовательнократной по определению.

Формат входных данных

В первой строке входного файла записаны N натуральных чисел ($1 \leqslant N \leqslant 1000$), не превосходящих $2 \cdot 10^9$ — последовательность.

Формат выходных данных

Вывести единственное число, равное длине максимальной последовательнократной подпоследовательности.

Пример

sequence.in	sequence.out
3 6 5 12	3

Задача C. Наибольшая общая подпоследовательность

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

Даны две последовательности. Найдите длину их наибольшей общей подпоследовательности (подпоследовательность — это то, что можно получить из данной последовательности вычеркиванием некоторых элементов).

Формат входных данных

В первой строке входного файла через пробел записаны N членов первой последовательности ($1 \leqslant N \leqslant 1000$) — целых чисел, не превосходящих 10 000 по модулю. Во второй строке через пробел записаны M членов второй последовательности ($1 \leqslant M \leqslant 1000$) — целые числа, не превосходящие 10 000 по модулю.

Формат выходных данных

В первую строку выходного файла требуется вывести единственное целое число: длину наибольшей общей подпоследовательности или число 0, если такой не существует. Во вторую строку выходного файла требуется вывести саму наибольшую общую подпоследовательность, через пробел (если подпоследовательностей несколько, выведите любую).

Пример

lcs.in	lcs.out
1 2 3 2 1 3 5	2 2 3

Задача D. Покупка билетов

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

За билетами на премьеру нового мюзикла выстроилась очередь из N человек, каждый из которых хочет купить 1 билет. На всю очередь работала только одна касса, поэтому продажа билетов шла очень медленно, приводя «постояльцев» очереди в отчаяние. Самые сообразительные быстро заметили, что, как правило, несколько билетов в одни руки кассир продаёт быстрее, чем когда эти же билеты продаются по одному. Поэтому они предложили нескольким подряд стоящим людям отдавать деньги первому из них, чтобы он купил билеты на всех.

Однако для борьбы со спекулянтами кассир продавала не более 3-х билетов в одни руки, поэтому договориться таким образом между собой могли лишь 2 или 3 подряд стоящих человека.

Известно, что на продажу i-му человеку из очереди одного билета кассир тратит A_i секунд, на продажу двух билетов — B_i секунд, трех билетов — C_i секунд. Напишите программу, которая подсчитает минимальное время, за которое могли быть обслужены все покупатели.

Обратите внимание, что билеты на группу объединившихся людей всегда покупает первый из них. Также никто в целях ускорения не покупает лишних билетов (то есть билетов, которые никому не нужны).

Формат входных данных

Во входном файле записано N троек натуральных чисел A_i, B_i, C_i ($1 \leqslant N \leqslant 5000$). Каждое из этих чисел не превышает 3600. Люди в очереди нумеруются начиная от кассы.

Формат выходных данных

В выходной файл выведите одно число — минимальное время в секундах, за которое могли быть обслужены все покупатели.

Пример

tickets.in	tickets.out
5 10 15 2 10 15 5 5 5 20 20 1 20 1 1	12

Задача E. Рюкзак

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

Найдите максимальный вес золота, который можно унести в рюкзаке вместительностью S, если есть N золотых слитков с заданными весами.

Формат входных данных

В первой строке входного файла запиано одно число — S ($1 \leqslant S \leqslant 10\,000$).

Далее следует N неотрицательных целых чисел ($1 \leqslant N \leqslant 300$), не превосходящих 100 000 — веса слитков.

Формат выходных данных

Выведите искомый максимальный вес.

Примеры

knapsack.in	knapsack.out
10 1 4 8	9
20 5 7 12 18	19

Задача F. Рюкзак с массами

Имя входного файла:	`knapsack2.in`
Имя выходного файла:	`knapsack2.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Дано N предметов массой $m_1, \ldots, m_N$ и стоимостью $c_1, \ldots, c_N$ соответственно.

Ими наполняют рюкзак, который выдерживает вес не более M. Определите набор предметов, который можно унести в рюкзаке, имеющий наибольшую стоимость.

Формат входных данных

В первой строке вводится натуральное число M, не превышающее 10 000.

Во второй строке вводятся N ($N \leqslant 100$) натуральных чисел m_i, не превышающих 100.

В третьей строке вводятся N натуральных чисел c_i, не превышающих 100.

Формат выходных данных

Выведите номера предметов (числа от 1 до N), которые войдут в рюкзак наибольшей стоимости.

Пример

knapsack2.in	knapsack2.out
6 2 4 1 2 7 2 5 1	1 3 4