

Задача А. Создание архива

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

Системный администратор вспомнил, что давно не делал архива пользовательских файлов. Однако, объем диска, куда он может поместить архив, может быть меньше чем суммарный объем архивируемых файлов.

Известно, какой объем занимают файлы каждого пользователя.

Напишите программу, которая по заданной информации о пользователях и свободному объему на архивном диске определит максимальное число пользователей, чьи данные можно поместить в архив, при этом используя свободное место как можно более полно.

Формат входных данных

Программа получает на вход в одной строке число S — размер свободного места на диске ($0 \leq S \leq 10^7$) и число N — количество пользователей ($1 \leq N \leq 10^5$), после этого идет N натуральных чисел, не превосходящих 300 — объем данных каждого пользователя, записанных каждое в отдельной строке.

Формат выходных данных

Выведите наибольшее количество пользователей, чьи данные могут быть помещены в архив.

Примеры

<code>archive.in</code>	<code>archive.out</code>
100 2 200 50	1
100 3 50 30 50	2

Задача В. Коньки

Имя входного файла: `skates.in`
Имя выходного файла: `skates.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

В ЛКШ-Зима школьники любят ходить на каток. В прокате коньков есть много коньков самых разных размеров. Школьник может надеть коньки любого размера, который не меньше размера его ноги. Известны размеры всех коньков и размеры ног школьников. Определите, какое наибольшее число школьников сможет одновременно пойти покататься.

Формат входных данных

Первая строка входных данных содержит число N — количество коньков в прокате ($1 \leq N \leq 10^5$). Во второй строке записано N чисел — размеры коньков. В третьей строке содержится число M — количество школьников в ЛКШ ($1 \leq M \leq 10^5$), четвертая строка содержит размеры их ног. Размеры коньков и ног — натуральные числа, не превосходящие 100.

Формат выходных данных

Выведите единственное число — наибольшее количество школьников, которое сможет пойти на каток.

Примеры

<code>skates.in</code>	<code>skates.out</code>
4 41 40 39 42 3 42 41 42	2

Задача С. Выбор заявок

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

Вы прекрасно знаете, что в ЛКШ.Зима 2017 лекции читают лучшие преподаватели мира. К сожалению, лекционных аудиторий у нас не так уж и много, поэтому каждый преподаватель составил список лекций, которые он хочет прочитать ЛКШатам. Чтобы ЛКШата, утром идя на завтрак, увидели расписание лекций, необходимо его составить прямо сейчас. И без вас нам здесь не справиться.

У нас есть список заявок от преподавателей на лекции для одной из аудиторий. Каждая заявка представлена в виде временного интервала $[s_i, f_i)$ — время начала и конца лекции. Лекция считается открытым интервалом, то есть какая-то лекция может начаться в момент окончания другой, без перерыва. Необходимо выбрать из этих заявок такое подмножество, чтобы суммарно выполнить максимальное количество заявок. Учтите, что одновременно в лекционной аудитории, конечно же, может читаться лишь одна лекция.

Формат входных данных

В первой строке вводится натуральное число N , не более 1000 — общее количество заявок на лекции. Затем вводится N строк с описаниями заявок — по два числа в каждом s_i и f_i . Гарантируется, что $s_i < f_i$. Время начала и окончания лекции — натуральные числа, не превышают 1440 (в минутах с начала суток).

Формат выходных данных

Выведите одно число — максимальное количество заявок, которые можно выполнить.

Примеры

<code>request.in</code>	<code>request.out</code>
1 5 10	1
3 1 5 2 3 3 4	2

Задача D. Планирование заданий

Имя входного файла: `schedule.in`
Имя выходного файла: `schedule.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Имеется некоторое множество заданий и один исполнитель. На выполнение одного задания уходит единица времени. Задания можно выполнять начиная с момента времени 0. У каждого задания есть две характеристики: d_i и w_i . Если задание не было выполнено к моменту времени d_i , взимается штраф в размере w_i . Требуется минимизировать суммарный штраф.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество заданий ($1 \leq n \leq 1000$). Следующие n строк содержат по два натуральных числа, разделенных пробелом — d_i и w_i ($0 \leq d_i, w_i \leq 1000$).

Формат выходных данных

Выведите одно число — минимальный суммарный штраф.

Примеры

<code>schedule.in</code>	<code>schedule.out</code>
2 1 1 1 2	1
1 0 5	5

Задача Е. Код Хаффмана

Имя входного файла: `huffman.in`
Имя выходного файла: `huffman.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Заданы числа p_1, p_2, \dots, p_n .

Предположив, что имеется текст, содержащий p_1 символов c_1 , p_2 символов c_2 , и т. д., постройте код Хаффмана и найдите суммарное число битов, необходимое для кодирования такого текста.

Формат входных данных

Первая строка входного файла содержит число n ($2 \leq n \leq 1000$). Вторая строка содержит n целых чисел p_1, p_2, \dots, p_n ($1 \leq p_i \leq 10^9$).

Формат выходных данных

Выведите одно число — число битов, необходимое для кодирования текста с заданным во входном файле количеством вхождений каждого символа.

Пример

<code>huffman.in</code>	<code>huffman.out</code>
10 1 2 3 4 5 6 7 8 9 10	173

Задача F. Белоснежка и n гномов

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

«Ну не гномы, а наказание какое-то!», — подумала Белоснежка, в очередной раз пытаясь уложить гномов спать. Одного уложишь — другой уже проснулся! И так всю ночь.

У Белоснежки n гномов, и все они очень разные. Она знает, что для того, чтобы уложить спать i -го гнома нужно a_i минут, и после этого он будет спать ровно b_i минут. Помогите Белоснежке узнать, может ли она получить хотя бы минутку отдыха, когда все гномы будут спать, и если да, то в каком порядке для этого нужно укладывать гномов спать.

Например, пусть есть всего два гнома, $a_1 = 1, b_1 = 10, a_2 = 10, b_2 = 20$. Если Белоснежка сначала начнет укладывать первого гнома, то потом ей потребуется целых 10 минут, чтобы уложить второго, а за это время проснется первый. Если же она начнет со второго гнома, то затем она успеет уложить первого и получит целых 9 минут отдыха.

Формат входных данных

Первая строка входного файла содержит число n ($1 \leq n \leq 10^5$), вторая строка содержит числа a_1, a_2, \dots, a_n , третья — числа b_1, b_2, \dots, b_n ($1 \leq a_i, b_i \leq 10^9$).

Формат выходных данных

Выведите в выходной файл n чисел — порядок, в котором нужно укладывать гномов спать. Если Белоснежке отдохнуть не удастся, выведите число -1 . Если решений несколько, выведите любое.

Примеры

стандартный ввод	стандартный вывод
2 1 10 10 20	2 1
2 10 10 10 10	-1
3 1 4 1 5 3 4	2 1 3

Задача G. Алиса и яблоки

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

Алисе в стране чудес попались n волшебных яблок. Про каждое яблоко известно, что после того, как его съешь, твой рост сначала уменьшится на a_i сантиметров, а потом увеличится на b_i сантиметров. Алиса очень голодная и хочет съесть все n яблок, но боится, что в какой-то момент ее рост станет равным нулю или еще меньше, и она пропадет совсем. Помогите ей узнать, можно ли съесть яблоки в таком порядке, чтобы в любой момент времени рост Алисы был больше нуля.

Формат входных данных

В первой строке вводятся натуральные числа n и s ($1 \leq n \leq 1000$, $1 \leq s \leq 100\,000$) — число яблок и начальный рост Алисы. В следующих n строках вводятся пары натуральных чисел a_i , b_i , не больших 1000.

Формат выходных данных

Если яблоки съесть нельзя, выведите число -1 . Иначе выведите n чисел — номера яблок, в том порядке, в котором их нужно есть.

Примеры

<code>apples.in</code>	<code>apples.out</code>
3 5 2 3 10 5 5 10	1 3 2
3 5 2 3 10 5 5 6	-1