

Задача А. Проверка

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Дано некоторое семейство множеств $S \subset 2^X$. Требуется проверить, может ли S быть семейством независимых множеств некоторого матроида.

Формат входных данных

Первая строка входного файла содержит два целых числа n и m — мощность множеств X и S соответственно ($1 \leq n \leq 10$, $0 \leq m \leq 2^n$). Каждая из следующих m строк содержит описание элемента множества S . Формат описания: количество элементов в подмножестве, затем через пробел номера этих элементов. Элементы множества X занумерованы, начиная с единицы.

Формат выходных данных

Выведите «YES», если S может быть семейством независимых множеств некоторого матроида, и «NO» иначе.

Примеры

стандартный ввод	стандартный вывод
2 4 0 1 1 1 2 2 1 2	YES
2 3 0 1 1 2 1 2	NO

Задача В. Планирование заданий

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
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 ($1 \leq n \leq 100000$). Следующие n строк содержат по два натуральных числа, разделенных пробелом — d_i и w_i ($0 \leq d_i, w_i \leq 10^9$).

Формат выходных данных

Выведите одно число — минимальный суммарный штраф.

Примеры

стандартный ввод	стандартный вывод
2 1 1 1 2	1

Задача С. Peep Blue

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

Суперкомпьютер Peep Blue умеет играть в игры. Однако Серёжа не побоялся принять его вызов.

Правила, предложенные компьютером, таковы. Есть n кучек камней. Игроки ходят по очереди, начинает Серёжа. На первом ходу каждому из игроков разрешается убрать любой набор кучек (в том числе пустой, но не все). На втором и следующих ходах каждый из игроков может взять любое ненулевое число камней из любой кучки. Проигрывает тот, кто не может сделать ход.

Ваша задача — определить, может ли Серёжа одержать победу, а также какой первый ход ему следует сделать. Среди всех первых ходов, приносящих победу, выберите такой, который оставит в игре как можно больше камней: это должно усложнить задачу Peep Blue.

Формат входных данных

В первой строке ввода записано целое число n — количество кучек камней ($1 \leq n \leq 100\,000$). В следующей строке записаны n натуральных чисел, a_i , не превышающих $2^{30} - 1$, — количества камней в кучках.

Формат выходных данных

Если Серёжа не может одержать победу, выведите единственное число -1 . Иначе выведите максимально возможное суммарное число камней, которое можно оставить.

Примеры

peep.in	peep.out
4 1 2 3 4	9

Задача D. Ремонт дорог

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

Славная страна Матроидландия состоит из N городов, соединенных M шоссе. Каждое шоссе соединяет два различных города, и по каждому шоссе можно перемещаться в обе стороны. Некоторые пары городов могут быть соединены более чем одной дорогой. Правительство Матроидландии решило произвести модернизацию дорожной системы в стране. В стране есть несколько компаний, занимающихся ремонтом дорог. Они уже договорились, кто что будет ремонтировать, таким образом, для каждого шоссе указана компания, которая будет его ремонтировать.

Чтобы поддержать малый бизнес в развивающейся стране, Правительство решило, что каждая компания может ремонтировать не более, чем одно шоссе. Так как жители Матроидландии не очень любят думать, между любыми двумя городами должен существовать максимум один путь по отремонтированным дорогам. Ваша задача состоит в том, чтобы определить максимальное число шоссе, которые можно отремонтировать, чтобы все были довольны.

Формат входных данных

Первая строка входного файла содержит два числа — N и M ($1 \leq N \leq 100$, $0 \leq M \leq 5000$). Следующие M строк содержат описания шоссе. Каждое описание состоит из двух номеров городов u и v , которые соединяет шоссе, и номера c компании, которая это шоссе собирается ремонтировать ($1 \leq v, u \leq N$, $1 \leq c \leq 200$).

Формат выходных данных

На первой строке выведите единственное число K — искомое максимальное количество шоссе. На следующей строке выведите номера этих шоссе в любом порядке. Если существует несколько решений, выведите любое.

Примеры

highways.in	highways.out
4 5 1 2 1 3 1 1 2 3 1 1 4 2 3 4 3	3 1 4 5

Задача Е. Два остовных дерева

Имя входного файла:	<code>twost.in</code>
Имя выходного файла:	<code>twost.out</code>
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Задан неориентированный граф, рёбра которого можно разбить на два непересекающихся остовных дерева. Вам необходимо найти одно из таких разбиений.

Формат входных данных

Первая строка входного файла содержит два натуральных числа N ($N \leq 600$) и M — количество вершин и рёбер в графе. Следующие M строк содержат описания рёбер графа. Каждое ребро задается номерами концов. Гарантируется, что в графе нет петель, но могут быть кратные рёбра. Вершины и рёбра графа нумеруются с единицы.

Формат выходных данных

Выходной файл должен содержать искомое разбиение рёбер графа. В первой строке выведите номера рёбер, которые войдут в первое остовное дерево, во второй — номера рёбер, которые войдут во второе остовное дерево. Каждое ребро графа должно появиться ровно в одной из этих двух строк.

Примеры

twost.in	twost.out
6 10 1 2 2 3 3 4 4 5 5 6 6 1 6 2 2 5 5 3 3 4	3 4 5 6 8 1 2 7 9 10
18 34 4 13 3 5 11 12 16 7 9 3 15 2 13 9 6 2 12 10 11 1 5 4 8 12 10 9 6 10 13 18 1 4 2 6 14 16 2 4 17 11 10 3 5 3 17 5 9 12 1 3 14 8 18 5 4 12 8 13 16 14 18 3 7 15 15 17 7 11	1 2 3 4 5 6 7 8 9 12 13 14 16 20 27 30 32 10 11 15 17 18 19 21 22 23 24 25 26 28 29 31 33 34

Задача F. Рюкзак-матроид

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

Одна из версий задачи о рюкзаке формулируется следующим образом: дано n предметов, i -й из них имеет вес w_i и стоимость v_i , требуется выбрать множество предметов с суммарным весом, не превышающим c (вместимости рюкзака) и максимальной возможной суммарной стоимостью. Известно, что задача о рюкзаке является NP-полной. Для нее существуют решения с помощью динамического программирования, но они обычно имеют время работы, линейное относительно суммы весов предметов и вместимости рюкзака, которое, таким образом, не является полиномиальным относительно размера входных данных.

Однако бывают случаи, когда задачу о рюкзаке можно так или иначе решить жадно. Один важный пример такого класса задач — если набор весов таков, что множество решений задачи образует матроид.

Матроидом называется пара $\langle X, \mathcal{I} \rangle$, где X — конечное множество, а $\mathcal{I}$ семейство подмножеств X , которые называют независимыми. При этом $\mathcal{I}$ должно удовлетворять следующим трем свойствам:

1. $\mathcal{I} \neq \emptyset$;
2. Если $A \in \mathcal{I}$ и $B \subset A$, то $B \in \mathcal{I}$;
3. Если $A, B \in \mathcal{I}$ и $|A| > |B|$, то найдется такой $x \in A \setminus B$, что $B \cup \{x\} \in \mathcal{I}$.

Например, ребра неориентированного графа и семейство их ациклических подмножеств образуют матроид.

Рассмотрим предметы с весами $w_1, w_2, \dots, w_n$. Пусть X представляет собой множество целых чисел от 1 до n . Будем называть подмножество $\{i_1, i_2, \dots, i_k\}$ независимым, если $w_{i_1} + w_{i_2} + \dots + w_{i_k} \leq c$. Проверьте, образует ли получившаяся пара матроид.

Формат входных данных

Первая строка входного файла содержит число n ($1 \leq n \leq 50$). Вторая строка содержит n целых чисел $w_1, w_2, \dots, w_n$ ($1 \leq w_i \leq 100$). Третья строка содержит число c ($\min w_i \leq c \leq \sum w_i$).

Формат выходных данных

Выведите «YES», если множество решений задачи о рюкзаке образует матроид. В противном случае выведите «NO».

Во втором случае выведите на второй строке число 2 или 3 — номер свойства, которое нарушается. Следующие две строки должны содержать контрпример к указанному свойству. Первая из строк должна описывать множество A , а вторая — B . Описание множества должно состоять из числа элементов в множестве и затем списка входящих в него предметов.

Примеры

knapsack.in	knapsack.out
3 1 2 3 4	YES
3 3 4 5 7	NO 3 2 1 2 1 3

Задача G. Рюкзак-матроид

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

Николай готовится к большому велопоходу. Это включает в себя много разных задач, в частности, ему нужно собрать огромный рюкзак. В этот рюкзак можно положить набор предметов, если сумма их весов не превосходит c . Размышляя над упаковкой рюкзака, Николай задался вопросом: а правда ли, что множества вещей, которые вмещаются в рюкзак, образуют матроид?

Николай быстро смекнул, что первые две аксиомы матроида точно выполняются. Помогите ему проверить третью!

Рассмотрим предметы с весами $w_1, w_2, \dots, w_n$. Пусть X представляет собой множество целых чисел от 1 до n . Будем называть подмножество $\{i_1, i_2, \dots, i_k\}$ независимым, если $w_{i_1} + w_{i_2} + \dots + w_{i_k} \leq c$. Проверьте, образует ли получившийся пара матроид.

Формат входных данных

Первая строка входного файла содержит число n ($1 \leq n \leq 200000$). Вторая строка содержит n целых чисел $w_1, w_2, \dots, w_n$ ($1 \leq w_i \leq 10^9$). Третья строка содержит число c ($1 \leq c \leq 10^{18}$).

Формат выходных данных

Выведите «YES», если множество решений задачи о рюкзаке образует матроид. В противном случае выведите «NO».

Выведите на второй строке число 3. Следующие две строки должны содержать контрпример к третьему свойству. Первая из строк должна описывать множество A , а вторая — B . Описание множества должно состоять из числа элементов в множестве и затем списка входящих в него предметов.

Примеры

стандартный ввод	стандартный вывод
3 1 2 3 4	YES
3 3 4 5 7	NO 3 2 2 1 1 3