Задача A. Сборка компьютеров

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

В рамках национальной программы «Электронная Россия» одно государственное учреждение заказало несколько системных блоков и столько же мониторов. При составлении заказа, однако, никто не учел, что существует два типа интерфейсов для соединения системных блоков и мониторов: VGA и DVI. При этом существуют системные блоки и мониторы, которые поддерживают как только один из этих интерфейсов, так и оба.

Поставщик техники оказался не очень добросовестным — в поставке оказалось a_1 системных блоков, которые поддерживают только VGA, a_2 системных блоков, которые поддерживают только DVI, и a_3 системных блоков, которые поддерживают оба интерфейса. С мониторами ситуация аналогична: b_1 мониторов поддерживают только VGA, b_2 — только DVI, b_3 — поддерживают оба интерфейса.

Необходимо выяснить, сколько комплектов из монитора и системного блока можно собрать. При этом соединить монитор и системный блок можно только если у них есть общий интерфейс.

Формат входного файла

Первая строка входного файла содержит три числа a_1, a_2 и a_3 ($0 \le a_1, a_2, a_3 \le 100$). Вторая строка входного файла содержит три числа b_1, b_2 и b_3 ($0 \le b_1, b_2, b_3 \le 100$). При этом выполняется равенство $a_1 + a_2 + a_3 = b_1 + b_2 + b_3$.

Формат выходного файла

В выходной файл выведите максимальное число комплектов из монитора и системного блока, которые можно собрать.

Примеры

computers.in	computers.out
3 4 6 2 3 8	13
3 4 6 2 11 0	12

Задача B. Домашнее задание

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

Сережа очень не любит делать домашние задания, но на последнем уроке информатики учитель задал классу n разных домашних заданий. Причем некоторые домашние задания можно делать только после того, как сделаны некоторые другие.

Для каждого задания Сережа оценил, сколько минут потребуется для его выполнения. После этого Сережа понял, что сделать все задания он точно не успеет. Тогда он решил сделать все задания кроме одного — за одно несделанное задание учитель, наверное, не будет слишком сильно ругаться. Теперь Сереже надо выбрать, какое задание не сделать.

Помогите Сереже выбрать задание, которое можно не делать, чтобы закончить все остальные задания как можно быстрее.

Формат входного файла

Первая строка входного файла содержит целые числа n и m — количество заданий и количество зависимостей между заданиями ($1 \le n \le 100$, $0 \le m \le 1000$). Вторая строка содержит n целых чисел: $t_1, t_2, \ldots, t_n$. Число t_i означает количество минут, необходимое Сереже для выполнения i-го задания ($1 \le t_i \le 1000$).

Затем следует m строк, каждая строка содержит два целых числа. Числа a и b означают, что задание a должно быть выполнено до задания b. Гарантируется, что все задания можно выполнить.

Формат выходного файла

Выведите в выходной файл одно число — минимальное количество минут, необходимое Сереже для выполнения всех заданий кроме одного.

Примеры

homework.in	homework.out
5 5 1 2 3 4 5 1 2 5 3 1 3 3 4 2 4	11

В приведенном примере Сережа может не выполнять четвертое задание. Остальные задания суммарно требуют 11 минут.

Задача C. Цирковое шоу

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

В цирке планируется грандиозное театрализованное шоу с участием львов и тигров. Чтобы уменьшить агрессию хищников, дрессировщики хотят составить программу таким образом, чтобы львы и тигры никогда не встречались на сцене.

Шоу состоит из n небольших представлений, в каждом из которых могут участвовать или львы, или тигры (также может случиться, что в представлении не участвуют ни те, ни другие). Представление i начинается через s_i минут от начала шоу и продолжается t_i минут. При этом в некоторые моменты времени на сцене могут идти одновременно несколько представлений (в этом случае в них не могут участвовать разные виды хищников).

Публика любит и представления со львами, и представления с тиграми. Дрессировщики просят вас помочь им распределить представления между львами и тиграми так, чтобы минимум из числа представлений с львами и числа представлений с тиграми был как можно больше.

Формат входного файла

Первая строка входного файла содержит число n ($1 \leqslant n \leqslant 200$). Следующие n строк содержат пары чисел s_i, t_i. ($0 \leqslant s_i \leqslant 10^9$, $1 \leqslant t_i \leqslant 10^9$)

Формат выходного файла

Выведите в выходной файл n чисел. Число номер i должно быть равно 1, если в i-ом представлении участвуют львы, или 2, если участвуют тигры, или 0, если не участвуют ни те ни другие.

Примеры

show.in	show.out
5 8 3 0 7 4 5 1 2 11 3	1 0 1 2 2

Задача D. Красивая таблица результатов

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

Олег — известный поклонник соревнований по программированию. Он знает всех участников всех соревнований за последние десять лет и может про любого участника сказать, сколько задач решила команда с его участием на любом соревновании. И еще Олег очень любит теорию чисел.

В таблице результатов соревнования по программированию команды упорядочены по убыванию количества решенных задач. Олег называет таблицу результатов *красивой*, если для всех команд количество решенных ими задач равно нулю или является делителем количества задач на соревновании. Когда какая-нибудь команда сдает задачу, количество сданных задач у нее увеличивается на один. Никакая команда не может сдать две или более задач одновременно, также две команды не могут одновременно сдать задачу.

Глядя на красивую таблицу результатов, Олег заинтересовался: а сколько еще задач смогут суммарно сдать команды так, чтобы после каждой сданной задачи таблица результатов оставалась красивой? Помогите ему выяснить это.

Формат входного файла

Первая строка входного файла содержит два целых числа: n и m — количество команд и количество задач на соревновании, соответственно ($1 \le n \le 100$, $1 \le m \le 10^9$). Вторая строка содержит n целых чисел, упорядоченных по невозрастанию: для каждой команды задано, сколько задач она решила. Гарантируется, что все отличные от нуля числа являются делителями числа m.

Формат выходного файла

Выведите в выходной файл одно число: максимальное количество задач, которое суммарно могут еще сдать команды так, чтобы после каждой сданной задачи таблица результатов оставалась красивой.

Примеры

standing.in	standing.out
7 12 12 6 4 3 3 1 0	9

В приведенном примере команды на 4 и 5 месте могут сдать по одной задаче, команда на 6 месте — три, а команда на 7 месте — 4. Суммарно таким образом команды смогут сдать 9 задач.

Задача E. Следующее разбиение на слагаемые

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

Разбиения числа n на слагаемые — это набор целых положительных чисел, сумма которых равна n. При этом разбиения, отличающиеся лишь порядком слагаемых, считаются одинаковыми, поэтому можно считать, что слагаемые в разбиении упорядочены по неубыванию.

Например, существует 7 разбиений числа 5 на слагаемые:

$$5 = 1 + 1 + 1 + 1 + 1$$
$$5 = 1 + 1 + 1 + 2$$
$$5 = 1 + 1 + 3$$
$$5 = 1 + 2 + 2$$
$$5 = 1 + 4$$
$$5 = 2 + 3$$
$$5 = 5$$

В приведенном примере разбиения упорядочены *лексикографически* — сначала по первому слагаемому в разбиении, затем по второму, и так далее. В этой задаче вам потребуется по заданному разбиению на слагаемые найти следующее в лексикографическом порядке разбиение.

Формат входного файла

Входной файл содержит одну строку — разбиение числа n на слагаемые ($1 \le n \le 100\,000$). Слагаемые в разбиении следуют в неубывающем порядке.

Формат выходного файла

Выведите в выходной файл одну строку — разбиение числа n на слагаемые, следующее в лексикографическом порядке после приведенного во входном файле. Если во входном файле приведено последнее разбиение числа n на слагаемые, выведите «`No solution`».

Примеры

next.in	next.out
5=1+1+3	5=1+2+2
5=5	No solution

Задача F. Выбор дорог

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

Коровы фермера Джона желают свободно ходить по N ($1 \le N \le 200$) последовательно пронумерованным от 1 до N полям фермы, даже если поля разделены лесом. Коровы выбирают систему дорог между парами полей таким образом, чтобы можно было пройти от любого поля к любому другому полю, двигаясь по дорогам. Коровы могут ходить по дорогам в любом направлении.

Для дорог коровы могут использовать только тропы диких животных. Каждую неделю они могут выбрать для использования все или некоторые тропы диких животных, о которых им известно.

Интересно, что коровы всегда обнаруживают ровно одну новую тропу диких животных в начале каждой недели. Они должны принять решение о множестве троп, которые будут использоваться в качестве системы дорог в течение текущей недели.

Коровы могут выбирать любое подмножество троп диких животных независимо от того, какие тропы выбирались на предыдущей неделе.

Коровы всегда хотят минимизировать суммарную длину дорог, которые они собираются выбрать.

Тропы диких животных не являются прямыми. Две тропы, которые соединяют одни и те же два поля, могут иметь разную длину. Более того, в случае пересечения двух троп (что возможно только вне поля), коровы не могут переходить с одной тропы на другую в точке их пересечения.

В начале каждой недели имеется информация о тропе диких животных, которую обнаружили коровы. На основании этого ваша программа должна выводить минимальную общую длину дорог, выбранных коровами на этой неделе, либо указывать, что искомая система дорог не существует.

Формат входного файла

- Первая строка входного файла содержит два целых числа, разделенных пробелом, N и W. W – количество недель, которые должна обработать программа ($1 \le W \le 6000$).
- Для каждой недели считывайте одну строку, содержащую информацию о вновь обнаруженной тропе диких животных . Эта строка содержит три целых числа, разделенных пробелами: концевые точки тропы (номера полей) и целочисленную длину тропы ($1 \ldots 10000$). Не бывает тропы, у которой концевые точки находятся на одном поле.

Формат выходного файла

Сразу после того, как ваша программа узнает о вновь обнаруженной тропе диких животных, она должна вывести одну строку, содержащую минимальную общую длину дорог, которые следует выбрать, чтобы связать все поля. Если невозможно связать все поля системой дорог, используя только тропы диких животных, то следует вывести "-1".

Ваша программа должна заканчивать работу после вывода ответа для последней недели.

Пример

maintain.in	maintain.out
4 6 1 2 10 1 3 8 3 2 3 1 4 3 1 3 6 2 1 2	-1 -1 -1 14 12 8

Задача G. Фермер

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

У фермера есть несколько полей, каждое из которых окружено кипарисами. Также у него есть несколько участков земли, на каждом из которых растёт ряд кипарисов. И на полях, и на участках между любыми двумя последовательными кипарисами растёт ровно одна олива. Каждый кипарис фермера либо растёт на границе какого-то поля, либо образуют ряд кипарисов, а каждая олива находится между какими-то двумя последовательными кипарисами.

Однажды фермер сильно заболел и почувствовал, что он скоро умрёт. За несколько дней до того, как он скончался, он позвал своего старшего сына и сказал ему: «Я оставляю тебе любые Q кипарисов, которые ты выберешь, и все те оливы, для которых ты выбрал оба соседних с ней кипариса.» На каждом поле и на каждом участке сын может выбрать любой набор кипарисов. Так как старший сын любит оливы, он хочет выбрать Q кипарисов так, чтобы унаследовать как можно больше олив.

Напишите программу, которая по данной информации о полях и участках, а также по количеству кипарисов, которые может выбрать сын, посчитает максимальное возможное количество олив, которые он может унаследовать.

Формат входного файла

Первая строка ввода содержит три целых числа Q, M и K — количество кипарисов, которые сын должен выбрать, количество полей и количество участков, соответственно ($0 \le Q \le 150\,000$, $0 \le M \le 2000$, $0 \le K \le 2000$). Вторая строка содержит M целых чисел N_i — количество кипарисов на соответствующих полях ($1 \le i \le M$, $3 \le N_i \le 150$). Третья строка содержит K целых чисел R_i — количество кипарисов на соответствующих участках ($1 \le i \le K$, $3 \le R_i \le 150$). Суммарное количество кипарисов на всех полях и участках хотя бы Q.

Формат выходного файла

Выходной файл должен содержать единственную строку, в которой находится единственное целое число — максимальное возможное количество олив, которые может унаследовать сын.

Пример

farmer.in	farmer.out
17 3 3 13 4 8 4 8 6	17

Задача H. Фидий

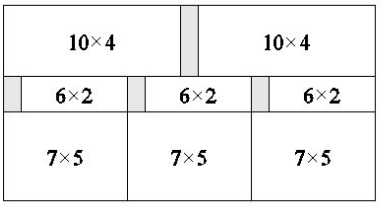
Имя входного файла:	`phidias.in`
Имя выходного файла:	`phidias.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Известный античный скульптор Фидий готовится создать ещё один удивительный монумент.

Для этого ему нужны прямоугольные мраморные плитки размеров $W_1 \times H_1$, $W_2 \times H_2$, ..., $W_N \times H_N$. Недавно Фидий получил огромную прямоугольную мраморную плиту. Он хочет разрезать эту плиту, чтобы получить плитки желаемых размеров. Любой кусок мрамора (изначальная плита или любая плитка, вырезанная из неё) можно разрезать горизонтальной или вертикальной прямой на две прямоугольные плитки с целыми шириной и высотой.

Разрезать плитки по-другому нельзя, объединять плитки тоже нельзя. Поскольку на мраморе есть узор, плитки нельзя поворачивать: если у Фидия есть плитка размера $A \times B$, он не может использовать её как плитку размера $B \times A$, если $A \neq B$.

Фидий может сделать ноль или больше плиток каждого из желаемых размеров. После того, как сделаны все разрезы, кусок мрамора выбрасывают, если его размеры не являются какими-то из желаемых. Фидию интересно, как разрезать изначальную плиту так, чтобы пришлось выбрасывать как можно меньше мрамора.



Напишите программу, которая по размерам изначальной плиты и желаемым размерам плиток посчитает минимальную суммарную площадь мрамора, которую придётся выбросить.

Формат входного файла

Первая строка ввода содержит два целых числа W и H — ширину и высоту изначальной плиты, соответственно ($1 \le W, H \le 600$).

Вторая строка содержит единственное целое число N — количество желаемых размеров плиток ($0 < N \le 200$).

Следующие N строк содержат желаемые размеры плиток. В каждой из этих строк содержится по два целых числа W_i и H_i — ширина и высота желаемого размера соответствующей плитки ($1 \le i \le N$, $1 \le W_i \le W$, $1 \le H_i \le H$).

Формат выходного файла

Вывод должен содержать единственную строку, в которой находится единственное число — минимальная возможная площадь мрамора, который придётся выбросить.

Пример

phidias.in	phidias.out
21 11 4 10 4 6 2 7 5 15 10	10

Задача I. Подарок

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

Начинающий программист Поликарп очень любит дарить подарки, особенно в коробках. Он давно заметил, что если коробка красиво оформлена, то радость от подарка возрастает многократно. Любой оберточной бумаге он предпочитает клетчатую. В самом деле, после распаковки подарка на ней можно играть в крестики-нолики, морской бой, точки, а также решать задачи и писать программы.

Поликарп очень аккуратен. Он упаковывает подарок в коробку, имеющую форму прямоугольного параллелепипеда, и оклеивает всю ее поверхность клетчатой бумагой. При этом каждая грань коробки представляет собой прямоугольник, состоящий из целых клеток.

В настоящий момент Поликарп собирается поздравить свою подругу, недавно вернувшуюся с очередной олимпиады. Он хочет подарить ей подарок в большой и красивой коробке.

У Поликарпа в наличии есть лист клетчатой бумаги, состоящий из n клеток. Каким будет максимальный объем коробки, которую можно оклеить с использованием этого листа бумаги описанным выше способом? Поликарп может разрезать лист клетчатой бумаги по границам клеток произвольным образом и оклеивать коробку получившимися фигурами, поэтому форма листа не важна, а имеет значение только количество клеток на нем. Поликарп может использовать для оклеивания коробки не все клетки.

Напишите программу, которая по заданному количеству клеток n находит размеры коробки максимального возможного объема.

Формат входного файла

Входной файл содержит одно целое число n ($6 \le n \le 10^{13}$) — количество клеток на листе клетчатой бумаги.

Формат выходного файла

Выведите в первую строку выходного файла максимальный объем коробки, которую может подарить Поликарп. Объем следует выводить в «кубических клетках», то есть единицей измерения является куб со стороной равной длине стороны клетки.

Во вторую строку выведите ширину, длину и высоту искомой коробки. Единица измерения — размер клетки. Числа разделяйте пробелами. Если решений несколько, то выведите любое из них.

Пример

gift.in	gift.out
6	1 1 1 1
37	12 3 2 2

Задача J. Вирусы и антивирусы

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

Антивирусная IT-компания имеет официальную иерархическую структуру управления. В ней есть босс — единственный сотрудник, над которым нет начальника. Каждый из остальных сотрудников подчинён ровно одному сотруднику — своему начальнику. Начальник может иметь нескольких подчинённых и отдавать или передавать приказы любому из них. Приказы могут передаваться от одного сотрудника другому только по цепочке, каждый раз от начальника к его подчинённому.

Сотрудник А главнее сотрудника Б в этой иерархии, если А может отдать или передать приказ сотруднику Б непосредственно, или через цепочку подчинённых. Босс главнее любого сотрудника.

Оказалось, что все сотрудники объединены ещё в одну организованную подобным образом тайную иерархическую структуру, производящую компьютерные вирусы. В тайной структуре может быть другой босс, а у сотрудников — другие начальники.

Будем называть пару сотрудников А и Б устойчивой, если А главнее Б и в основной, и в тайной иерархических структурах.

Требуется написать программу, определяющую количество устойчивых пар в компании.

Формат входного файла

В первой строке задано число N — количество сотрудников компании ($1 \le N \le 100\,000$).

Во второй строке — N целых чисел a_i, где $a_i = 0$, если в официальной иерархии сотрудник с номером i является боссом, в противном случае a_i равно номеру непосредственного начальника сотрудника номер i.

В третьей строке — N целых чисел b_i, где $b_i = 0$, если в тайной иерархии сотрудник с номером i является боссом, в противном случае b_i равно номеру непосредственного начальника сотрудника номер i. Нумерация сотрудников ведется с единицы в том порядке, в каком они упомянуты во входном файле.

Формат выходного файла

Выходной файл должен содержать единственное число – количество устойчивых пар.

Пример

virus.in	virus.out
3 0 3 1 0 1 1	2
5 2 0 1 3 4 3 1 0 2 4	7

Задача K. Разглядывание забора

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

Фермер Дон осматривает забор, которым окружено его квадратное плоское поле размером N на N метров ($2 \le N \le 500000$). Один угол забора расположен в точке $(0, 0)$, противоположный ему угол находится в точке (N, N). Стороны забора параллельны осям X и Y.

Столбы, к которым крепится забор, стоят не только по углам, но и через метр вдоль каждой стороны поля, всего в заборе $4 \cdot N$ столбов. Столбы стоят вертикально и не имеют толщины (радиус равен нулю). Фермер Дон хочет определить, сколько столбов он увидит, если встанет в определенной точке поля.

Проблема заключается в том, что на поле фермера Дона расположено R огромных камней ($1 \le R \le 30000$), за которыми не видны некоторые столбы, поскольку Дон недостаточно высок, чтобы смотреть поверх этих камней. Основание каждого камня представляет собой выпуклый многоугольник ненулевой площади, вершины которого имеют целочисленные координаты. Камни стоят на поле вертикально. Камни не имеют общих точек между собой, а также с забором. Точка, где стоит фермер Дон, лежит внутри, но не на границе поля, а также не лежит внутри и на границе камней.

По заданным размеру поля фермера Дона, положению и форме камней на поле, месту, где стоит фермер Дон, вычислите количество столбов, которые может видеть фермер Дон. Если вершина основания камня находится на одной линии с местом расположения Дона и некоторым столбом, то Дон не видит этот столб.

Формат входного файла

Первая строка входного файла содержит два целых числа N и R, разделенных пробелом. Вторая строка входного файла содержит два целых числа — координаты X и Y места, где стоит фермер Дон на поле. Последующие строки входного файла описывают положение R камней:

- Описание i-го камня начинается со строки, которая содержит целое число p_i ($3 \le p_i \le 20$), определяющее количество вершин в основании камня.
- Каждая из последующих p_i строк содержит пару разделенных пробелом целых чисел X и Y, которые являются координатами вершины. Вершины основания камня различны между собой и перечислены в направлении против часовой стрелки

Формат выходного файла

Выходной файл должен содержать одну строку с одним целым числом, обозначающим количество столбов, которые будет видеть фермер Дон.

Пример

boundary.in	boundary.out
100 1 60 50 5 70 40 75 40 80 40 80 50 70 60	319

Обратите внимание на то, что основание камня имеет три вершины на одной прямой: $(70, 40)$, $(75, 40)$ и $(80, 40)$.