Задача A. Псевдопростые числа

Имя входного файла:	numbers.in
Имя выходного файла:	numbers.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Пусть $a_1 = 2$, $a_2 = 3$, $a_n = a_1 \cdot a_2 \cdot \ldots \cdot a_{n-1} - 1$ при $n \geq 3$. Назовем числа a_i псевдопростыми. Для заданного натурального числа X нужно ответить на вопрос: можно ли X однозначно представить в виде произведения псевдопростых чисел (представления, отличающиеся только порядком множителей, считаются одинаковыми), и, если можно — выдать разложение.

Формат входных данных

Во входном файле записано одно натуральное число X, $1 < X \leq 10^9$.

Формат выходных данных

В выходной файл выдать псевдопростые числа, произведение которых равно X, в произвольном порядке. Если разложения не существует или оно не единственно, выдать 0.

Пример

numbers.in	numbers.out
6	3 2
5	5
7	0
4	2 2

Оценка задачи

1 балл будет набирать программа, верно работающая для $X \leq 100$.

Задача B. Сортировка «с конфеткой»

Имя входного файла:	sort.in
Имя выходного файла:	sort.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем используемой памяти:	8 мегабайт

Дан массив из N различных натуральных чисел от 1 до N. Сортировка массива по возрастанию "пузырьком" работает следующим образом. Сначала сравниваются первый и второй элемент, и, если первый больше второго, то они меняются местами. Затем та же процедура производится со вторым и третьим элементом, …, с предпоследним и последним. Затем эта процедура снова повторяется с первым и вторым, со вторым и третьим, …, с предпоследним и последним элементами. И так (N – 1) раз.

Сортировка «с конфеткой» выполняется по тем же правилам, но дополнительно задан список пар чисел, которые не меняются друг с другом ни при каких условиях (в таком случае сортирующий получает конфетку за то, что пропускает соответствующий обмен). Например, наличие в списке пары (4,1) обозначает, что если в какой-то момент рядом окажутся числа 4 и 1, и по алгоритму сортировки их нужно будет поменять местами, то обмена не произойдет, а сортирующий получит конфетку.

Требуется провести сортировку «с конфеткой» данного массива и выдать результат сортировки.

Формат входных данных

Во входном файле записаны сначала число N — количество чисел в массиве, затем N чисел — элементы массива. Далее записано число M — количество пар чисел, за которые дают конфетку, а затем M пар чисел. Если в списке есть пара (i,j), то и за пару (j,i) также дают конфетку.

$1 \le N \le 5000$, $0 \le M \le 10000$.

Формат выходных данных

Требуется вывести массив после сортировки.

Пример

sort.in	sort.out
4 1 4 2 3 2 4 3 1 2	1 2 4 3

Оценка задачи

1 балл будет набирать решение, верно работающее при $1 \le N \le 100$, $0 \le M \le 50$.

Задача C. Электричка

Имя входного файла:	train.in
Имя выходного файла:	train.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Вагоны в электричке пронумерованы натуральными числами, начиная с 1 (при этом иногда вагоны нумеруются от «головы» поезда, а иногда – с «хвоста»; это зависит от того, в какую сторону едет электричка). В каждом вагоне написан его номер.

Витя сел в i-й вагон от головы поезда и обнаружил, что его вагон имеет номер j. Он хочет определить, сколько всего вагонов в электричке. Напишите программу, которая будет это делать или сообщать, что без дополнительной информации это сделать невозможно.

Формат входных данных

Во входном файле записаны два числа i и j ($1 \le i \le 1000$, $1 \le j \le 1000$), разделенные пробелом.

Формат выходных данных

В выходной файл выведите одно число — количество вагонов в электричке. Если однозначно определить количество вагонов нельзя, выведите в выходной файл число 0.

Пример

train.in	train.out
3 4	6

Задача D. Обувной магазин

Имя входного файла:	shop.in
Имя выходного файла:	shop.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

В обувном магазине продается обувь разного размера. Известно, что одну пару обуви можно одеть на другую, если она хотя бы на три размера больше. В магазин пришел покупатель. Требуется определить, какое наибольшее количество пар обуви сможет предложить ему продавец так, чтобы он смог надеть их все одновременно.

Формат входных данных

Во входном файле сначала указан размер ноги покупателя (обувь меньшего размера он надеть не сможет), затем количество пар обуви в магазине и размеры каждой пары. Размер — натуральное число, не превосходящее 100, количество пар обуви в магазине не превосходит 1000.

Формат выходных данных

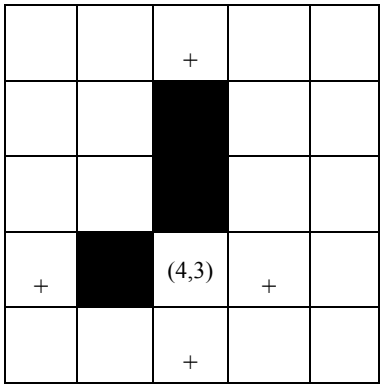
В выходной файл выведите единственно число — максимальное количество пар обуви.

Пример

shop.in	shop.out
60 2 60 63	2
35 5 30 40 35 42 35	2

Задача E. Перекраска клеток

Имя входного файла:	color.in
Имя выходного файла:	color.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем используемой памяти:	64 мегабайта



Дано клетчатое поле N x M, все клетки поля изначально белые. Автомат умеет:

- закрасить клетку (i,j) в черный цвет.
- для клетки (i,j) узнать её ближайших белых соседей по вертикали и горизонтали.

Дана последовательность команд для автомата. Требуется выполнить эти команды в указанной последовательности, и для каждой команды запроса ближайших белых соседей указать результат ее выполнения.

Формат входных данных

Во входном файле сначала записаны размеры поля N и M ($1 \le N \le 20$, $1 \le M \le 50000$), затем количество команд K ($1 \le K \le 10^5$), а затем сами команды. Команды записаны по одной в строке в следующем формате:

`Color` *i j* — окраска клетки (i,j) в черный цвет;

`Neighbors` *i j* — нахождение белых соседей для клетки (i,j).

Клетка (i,j) в описании любой команды может быть как белой, так и черной. $1 \le i \le N$, $1 \le j \le M$.

Формат выходных данных

На каждый запрос `Neighbors` требуется вывести сначала количество ближайших белых соседей (или 0, если ни с одной из сторон белых клеток не осталось), а затем их координаты (соседей можно перечислять в произвольном порядке). Если запросов `Neighbors` нет, оставьте выходной файл пустым.

Пример

color.in	color.out
5 5 6 Color 4 2 Neighbors 4 3 Color 2 3 Color 3 3 Neighbors 4 3 Neighbors 5 1	4 4 1 4 4 3 3 5 3 4 4 1 4 4 1 3 5 3 2 5 2 4 1

Оценка задачи

1 балл получат решения, верно работающие при $N \le 20$, $M \le 500$, $K \le 1000$.

Задача F. Быстрые шахматы

Имя входного файла:	chess.in
Имя выходного файла:	chess.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

На шахматной доске (размером 8 × 8 клеток) расставлены фигуры. За один ход разрешается взять одной фигурой другую (цвет фигур значения не имеет; ходы без взятия делать нельзя). Требуется найти последовательность ходов, после которой на доске останется одна фигура.

Ладья ходит по горизонтали или вертикали на любое число клеток. Слон ходит по диагонали на любое число клеток. Ферзь ходит по горизонтали, вертикали или диагонали на любое число клеток. Эти фигуры не могут перепрыгивать через другие фигуры. Конь ходит на две клетки по горизонтали или вертикали и на одну клетку в перпендикулярном направлении (например, на 2 клетки вверх и на одну клетку вправо и т.п.), при этом он может перепрыгивать через другие фигуры.

Формат входных данных

В восьми строках входного файла записаны по 8 символов, описывающих шахматную доску: R — ладья, B — слон, K — конь, Q — ферзь, точка обозначает пустую клетку. На доске изначально стоит не менее двух и не более десяти фигур.

Формат выходных данных

В выходной файл выведите возможную последовательность в следующем формате. Для каждого хода указывается сначала клетка, с которой делается ход, затем двоеточие, затем клетка, на которую делается ход. Клетка задается столбцом и строкой: столбцы нумеруются слева направо строчными латинскими буквами a, b, c, d, e, f, g, h; строки — снизу вверх цифрами 1, 2, 3, 4, 5, 6, 7, 8.

Если решений несколько, приведите любое из них. Если решений нет, выведите `NO SOLUTION`

Пример

chess.in	chess.out
.... .B...... ....K...	b6:e3
K......K .... .K...... ......Q. .... K......K	NO SOLUTION
..K..... ..K..... BR...... .B......	c7:a6 b6:a6 b5:a6 a6:c8

Оценка задачи

1 балл будут набирать решения, верно работающие когда на доске всего две фигуры.

Задача G. Красно-синий граф

Имя входного файла:	redblue.in
Имя выходного файла:	redblue.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем используемой памяти:	64 мегабайта

Даны N точек, занумерованных числами 1, 2, ..., N. От каждой точки с меньшим номером к каждой точке с большим номером ведет стрелка красного или синего цвета. Раскраска стрелок называется *однотонной*, если нет двух таких точек A и B, что от A до B можно добраться как только по красным стрелкам, так и только по синим.

Ваша задача — по заданной раскраске определить, является ли она однотонной.

Формат входных данных

В первой строке входного файла содержится единственное число N ($3 \le N \le 5000$).

В следующих N–1 строках идет описание раскраски. В (i+1)-й строке файла записано (N–i) символов `R` (красный) или `B` (синий), соответствующих цвету стрелок, выходящих из точки i и входящих в точки (i+1), (i+2), ..., N соответственно.

Формат выходных данных

В выходной файл выведите `YES`, если приведенная раскраска является однотонной, и `NO` в противном случае.

Пример

redblue.in	redblue.out
3 RB R	NO
3 RR R	YES

Оценка задачи

1 балл будут набирать решения, верно работающие для $N \le 50$.

Задача N. Лапта

Имя входного файла:	lapta.in
Имя выходного файла:	lapta.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем используемой памяти:	64 мегабайта

При игре в лапту одна команда ловит мяч и пытается осалить им бегущего. Игрок другой команды должен, перед тем как бежать, ударить мяч в поле. Известно, на какое максимальное расстояние он может ударить, а также скорости и начальные координаты игроков другой команды. Требуется выбрать направление и силу удара так, чтобы минимальное время, которое потребуется другой команде, чтобы поднять мяч с земли, было наибольшим. (Пока мяч летит, игроки стоят на местах.)

Формат входных данных

В первой строке входного файла записаны два числа: D — максимальное расстояние удара и N — количество соперников на поле (D и N натуральные числа, $D \le 1000$, $N \le 200$). В следующих N строках записаны по три числа – начальные координаты x_i и y_i и максимальная скорость v_i соответствующего игрока (скорости и координаты — целые числа, $-1000 \le x_i \le 1000$, $0 \le y_i \le 1000$, $0 < v_i \le 1000$), никакие два игрока не находятся изначально в одной точке. Игрок, бьющий мяч, находится в точке с координатами (0,0). Мяч выбивается в точку с неотрицательной ординатой ($y \ge 0$).

Формат выходных данных

В выходной файл выведите сначала время, которое потребуется игрокам, чтобы добежать до мяча, а затем координаты точки, в которую нужно выбить мяч. Если таких точек несколько, выведите координаты любой из них. Время и координаты нужно вывести с точностью 10^{-3}.

Пример

lapta.in	lapta.out
10 2 1 1 1 -1 1 1	9.055 0 10

Оценка задачи

1 балл получат программы, которые верно работают, когда в поле не более двух соперников.

Задача I. Машинки

Имя входного файла:	cars.in
Имя выходного файла:	cars.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

По двум однополосным дорогам едут машины. На перекрестке эти дороги сливаются в одну однополосную (машины едут как раз в ее сторону). На этом самом перекрестке стоит постовой милиционер, который регулирует движение, а именно, выбирает, с какой из двух дорог в данный момент машины будут заезжать.

В каждый момент времени либо одна из дорог является "активной", т.е. перекресток проезжают машины с этой дороги, либо регулировщик производит переключение потока на другую дорогу (в этот момент никакие машины через перекресток не едут). На то, чтобы "переключить" поток, тратится время T.

В некоторый момент образовалась пробка. На 1-й дороге скопилось N машин, а на 2-й — M. Для каждой машины известно время, которое ей потребуется для проезда развилки, если она первая в очереди и ее дорога "активна".

Требуется разработать план действий для постового по переключению потоков, при котором среднее время ожидания машины минимально (временем ожидания машины называется время с момента образования пробки до того момента, когда данная машина заканчивает проезжать перекресток; для вычисления среднего времени суммируются времена ожидания всех машин и полученная сумма делится на общее количество машин). В начальный момент активна первая дорога.

Формат входных данных

Во входном файле записаны сначала числа T, N и M, затем N чисел — времена проезда перекрестка машинами с первой дороги (в порядке удаления от перекрестка), а затем M чисел — времена проезда перекрестка машинами со второй дороги.

Все числа в файле целые неотрицательные, все времена не превосходят 100, $T \le 100$, $N, M \le 1000$, $N+M > 0$.

Формат выходных данных

В первой строке выходного файла выведите одно число — минимальное среднее время ожидания с точностью 10^{-3}. Далее выведите информацию о машинах в порядке их проезда перекрестка и о переключении потока в следующем формате. Для каждой машины выведите информацию в формате:

`Car k from road i`

где k — номер машины на своей дороге (ближайшая к перекрестку в момент образования пробки машина имеет номер 1, следующая на той же дороге — 2 и т.д.), i — номер дороги: 1 или 2.

Для каждого переключения потока выведите информацию:

`Switch road from 1 to 2`

для переключения потока с первой дороги на вторую или

`Switch road from 2 to 1`

для переключения потока со второй дороги на первую. Информация обо всех событиях (переключениях и проездах через перекресток) должна выводиться именно в том порядке, в котором они происходят.

Если решений несколько, выведите любое из них.

Пример

cars.in	cars.out
1 2 2 5 6 1 7	11.500 Switch road from 1 to 2 Car 1 from road 2 Switch road from 2 to 1 Car 1 from road 1 Car 2 from road 1 Switch road from 1 to 2 Car 2 from road 2

Оценка задачи

1 балл будет набирать решение, верно работающее при $N, M \le 10$.

Задача J. Контейнеры

Имя входного файла:	boxes.in
Имя выходного файла:	boxes.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем используемой памяти:	64 мегабайта

На складе хранятся контейнеры с товарами N различных видов. Все контейнеры составлены в N стопок. В каждой стопке могут находиться контейнеры с товарами любых видов (стопка может быть изначально пустой).

Автопогрузчик может взять верхний контейнер из любой стопки и поставить его сверху в любую стопку. Необходимо расставить все контейнеры с товаром первого вида в первую стопку, второго вида — во вторую стопку и т.д.

Программа должна вывести последовательность действий автопогрузчика или сообщение о том, что задача решения не имеет.

Формат входных данных

В первой строке входного файла записано одно натуральное число N, не превосходящее 500. В следующих N строках описаны стопки контейнеров: сначала записано число k_i — количество контейнеров в стопке, а затем k_i чисел — виды товара в контейнерах в данной стопке, снизу вверх. В каждой стопке вначале не более 500 контейнеров (в процессе переноса контейнеров это ограничение может быть нарушено).

Формат выходных данных

В выходной файл выдать описание действий автопогрузчика: для каждого действия напечатать два числа — из какой стопки брать контейнер и в какую стопку класть. (Обратите внимание, что минимизировать количество операций автопогрузчика не требуется.) Если задача не имеет решения, выдать одно число 0. Если контейнеры изначально правильно размещены по стопкам, то разрешается оставлять выходной файл пустым.

Пример

boxes.in	boxes.out
3 4 1 2 3 2 0 0	1 2 1 3 1 2

Изначально в первой стопке лежат четыре контейнера — снизу контейнер с товаром первого вида, над ним — с товаром второго вида, над ним — третьего, и сверху — еще один контейнер с товаром второго вида.

Оценка задачи

1 балл будут получать программы, верно работающие при следующих ограничениях: количество стопок не больше 10, в каждой стопке не более 10 контейнеров.