# Задачи от Яндекса

# Часть 1

# A. Покраска деревьев

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вася и Маша участвуют в субботнике и красят стволы деревьев в белый цвет. Деревья растут вдоль улицы через равные промежутки в 1 метр. Одно из деревьев обозначено числом ноль, деревья по одну сторону занумерованы положительными числами 1,2 и т.д., а в другую — отрицательными −1,−2 и т.д.

Ведро с краской для Васи установили возле дерева P, а для Маши — возле дерева Q. Ведра с краской очень тяжелые и Вася с Машей не могут их переставить, поэтому они окунают кисть в ведро и уже с этой кистью идут красить дерево. Краска на кисти из ведра Васи засыхает, когда он удаляется от ведра более чем на V метров, а из ведра Маши — на M метров. Определите, сколько деревьев может быть покрашено.

## Формат ввода

В первой строке содержится два целых числа P и V — номер дерева, у которого стоит ведро Васи и на сколько деревьев он может от него удаляться.

В второй строке содержится два целых числа Q и M — аналогичные данные для Маши.

Все числа целые и по модулю не превосходят 108.

## Формат вывода

Выведите одно число — количество деревьев, которые могут быть покрашены.

## Пример

| **Ввод** Скопировать ввод | **Вывод**Скопировать вывод |
| --- | --- |
| 0 7  12 5 | 25 |

# B. Футбольный комментатор

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Раунд плей-офф между двумя командами состоит из двух матчей. Каждая команда проводит по одному матчу «дома» и «в гостях». Выигрывает команда, забившая большее число мячей. Если же число забитых мячей совпадает, выигрывает команда, забившая больше мячей «в гостях». Если и это число мячей совпадает, матч переходит в дополнительный тайм или серию пенальти.

Вам дан счёт первого матча, а также счёт текущей игры (которая ещё не завершилась). Помогите комментатору сообщить, сколько голов необходимо забить первой команде, чтобы победить, не переводя игру в дополнительное время.

## Формат ввода

В первой строке записан счёт первого мачта в формате *G1:G2*, где *G1* — число мячей, забитых первой командой, а *G2* — число мячей, забитых второй командой.

Во второй строке записан счёт второго (текущего) матча в аналогичном формате. Все числа в записи счёта не превышают *5*.

В третьей строке записано число *1*, если первую игру первая команда провела «дома», или *2*, если «в гостях».

## Формат вывода

Выведите единственное целое число "— необходимое количество мячей.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 0:0  0:0  1 | 1 |

### Пример 2

| **Ввод** Скопировать ввод | **Вывод** Скопировать вывод |
| --- | --- |
| 0:2  0:3  1 | 5 |

### Пример 3

| **Ввод** Скопировать ввод | **Вывод** Скопировать вывод |
| --- | --- |
| 0:2  0:3  2 |  |

C. Форматирование файла

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Петя - начинающий программист. Сегодня он написал код из n строк.

К сожалению оказалось, что этот код трудно читать. Петя решил исправить это, добавив в различные места пробелы. А точнее, для i-й строки ему нужно добавить **ровно**ai пробелов.

Для добавления пробелов Петя выделяет строку и нажимает на одну из трёх клавиш: Space, Tab, и Backspace. При нажатии на Space в строку добавляется один пробел. При нажатии на Tab в строку добавляются четыре пробела. При нажатии на Backspace в строке удаляется один пробел.

Ему хочется узнать, какое наименьшее количество клавиш придётся нажать, чтобы добавить необходимое количество пробелов в каждую строку. Помогите ему!

Формат ввода

Первая строка входных данных содержит одно целое положительное число n(1≤n≤105) – количество строк в файле.

Каждая из следующих n строк содержит одно целое неотрицательное число ai(0≤ai≤109) – количество пробелов, которые нужно добавить в i-ю строку файла.

Формат вывода

Выведите одно число – минимальное количество нажатий, чтобы добавить в каждой строке необходимое количество пробелов.

Пример

| **Ввод** Скопировать ввод | **Вывод** Скопировать вывод |
| --- | --- |
| 5  1  4  12  9  0 | 8 |

Примечания

В примере можно:

* 1 раз нажать на Space в первой строке.
* 1 раз нажать на Tab на второй строке.
* 3 раза нажать на Tab в третьей строке.
* 2 раза нажать на Tab и один раз нажать на Space в четвёртой строке.
* Ничего не нажимать в пятой строке.

В итоге получается 1+1+3+3=8 нажатий. Можно доказать, что нельзя добавить необходимое количество пробелов за 7 нажатий или меньше.

# D. Слоны и ладьи

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

На шахматной доске стоят слоны и ладьи, необходимо посчитать, сколько клеток не бьется ни одной из фигур.

Шахматная доска имеет размеры 8 на 8. Ладья бьет все клетки горизонтали и вертикали, проходящих через клетку, где она стоит, до первой встретившейся фигуры. Слон бьет все клетки обеих диагоналей, проходящих через клетку, где он стоит, до первой встретившейся фигуры.

## Формат ввода

В первых восьми строках ввода описывается шахматная доска. Первые восемь символов каждой из этих строк описывают состояние соответствующей горизонтали: символ B (заглавная латинская буква) означает, что в клетке стоит слон, символ R — ладья, символ \* — что клетка пуста. После описания горизонтали в строке могут идти пробелы, однако длина каждой строки не превышает 250 символов. После описания доски в файле могут быть пустые строки.

## Формат вывода

Выведите количество пустых клеток, которые не бьются ни одной из фигур.

### Пример 1

| **Ввод** Скопировать ввод | **Вывод** Скопировать вывод |
| --- | --- |
| \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*R\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\* | 49 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*B\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\* | 54 |

### Пример 3

| **Ввод** Скопировать ввод | **Вывод** Скопировать вывод |
| --- | --- |
| \*\*\*\*\*\*\*\*  \*R\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*B\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\* | 40 |

# E. Прибыльный стартап

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

k друзей организовали стартап по производству укулеле для кошек. На сегодняшний день прибыль составила n рублей. Вы, как главный бухгалтер компании, хотите в каждый из ближайших d дней приписывать по одной цифре в конец числа, выражающего прибыль. При этом в каждый из дней прибыль должна делиться на k.

## Формат ввода

В единственной строке входных данных через пробел записаны три числа: n,k,d — изначальная прибыль, количество учредителей компании и количество дней, которое вы собираетесь следить за прибылью (1≤n,k≤109,1≤d≤105). **НЕ** гарантируется, что n делится на k.

## Формат вывода

Выведите одно целое число x — прибыль компании через d дней. Первые цифры числа x должны совпадать с числом n. Все префиксы числа x, которые длиннее числа n на 1,2,…,d цифр, должны делиться на k. Если возможных ответов несколько, выведите любой из них. Если ответа не существует, выведите −1.

### Пример 1

| **Ввод** Скопировать ввод | **Вывод** Скопировать вывод |
| --- | --- |
| 21 108 1 | 216 |

### Пример 2

| **Ввод** Скопировать ввод | **Вывод** Скопировать вывод |
| --- | --- |
| 5 12 4 | -1 |

## Примечания

В первом тестовом примере всего один день нужно следить за прибылью. Можно дописать цифру 6 в конец числа 21 и получить прибыль, делящуюся на 108.

Во втором тестовом примере мы в первый же день не можем получить прибыль, делящуюся на k, какая бы цифра не была дописана у числу n, поэтому ответа не существует.

# F. Миша и математика

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Миша сидел на занятиях математики в Высшей школе экономики и решал следующую задачу: дано n целых чисел и нужно расставить между ними знаки + и × так, чтобы результат полученного арифметического выражения был нечётным (например, между числами 5, 7, 2, можно расставить арифметические знаки следующим образом: 5×7+2=37). Так как примеры становились все больше и больше, а Миша срочно убегает в гости, от вас требуется написать программу решающую данную задачу.

## Формат ввода

В первой строке содержится единственное число n (2≤n≤105). Во второй строке содержится n целых чисел ai, разделённых пробелами (−109≤ai≤109). Гарантируется, что решение существует.

## Формат вывода

В одной строке выведите n−1 символ + или ×, в результате применения которых получается нечётный результат. (Для вывода используйте соответственно знаки «+» (ASCII код—43) и «x» (ASCII код—120), без кавычек).

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  5 7 2 | x+ |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2  4 -5 | + |

# G. Разрушить казарму

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вы играете в интересную стратегию. У вашего соперника остались всего одна казарма — здание, в котором постоянно появляются новые солдаты. Перед атакой у вас есть *x* солдат. За один раунд каждый солдат может убить одного из солдат противника или нанести 1 очко урона казарме (вычесть единицу здоровья у казармы). Изначально у вашего оппонента нет солдат. Тем не менее, его казарма имеет *y* единиц здоровья и производит *p* солдат за раунд.

Ход одного раунда:

1. Каждый солдат из вашей армии либо убивает одного из солдат вашего противника, либо наносит 1 очко урона казарме. Каждый солдат может выбрать своё действие. Когда казарма теряет все свои единицы здоровья, она разрушается.
2. Ваш противник атакует. Он убьет *k* ваших солдат, где *k* — количество оставшихся у противника солдат.
3. Если казармы еще не разрушены, ваш противник производит *p* новых солдат.

Ваша задача — разрушить казарму и убить всех солдат противника. Если это возможно, посчитайте минимальное количество раундов, которое вам нужно для этого. В противном случае выведите *-1*.

## Формат ввода

На вход подаётся три целых числа *x*, *y*, *p* (*1 ≤ x, y, p ≤ 5000*) — количество ваших солдат на старте игры, количество очков здоровья казармы и количество производимых за раунд казармой солдат, соответственно. Каждое число расположено в новой строке.

## Формат вывода

Если возможно убить всех вражеских солдат и разрушить казарму, выведите минимальное количество раундов, необходимых для этого. В противном случае выведите *-1*.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 10  11  15 | 4 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1  2  1 | -1 |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1  1  1 | 1 |

### Пример 4

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 25  200  10 | 13 |

## Примечания

В первом примере в первом раунде сначала все ваши солдату атакуют казарму, после этого не происходит ничего, потому что у врага нет солдат, затем у врага появляется *15* солдат. Во втором раунде один ваш солдат добивает казарму, остальные *9* солдат убивают *9* солдат врага. Оставшиеся *6* солдат врага убивают *6* ваших солдат, но армия врага не пополняется, поскольку казарма разрушена. В третьем раунде сначала вы убиваете четверых солдат врага, затем враг двоих ваших солдат. В последнем, четвертом, раунде вы добиваете двух оставшихся солдат врага.

# H. Забег по стадиону

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Стадион представляет собой окружность **длиной**L метров, на которой отмечена точка старта. По стадиону бегают Кирилл и Антон. У каждого мальчика есть своя точка старта (она представляет собой расстояние в метрах от старта, отсчитанное по часовой стрелке) и своя скорость в метрах в секунду (положительная скорость означает, что мальчик бежит по часовой стрелке, отрицательная — что бежит против часовой, а нулевая — что он стоит на месте).

Вам нужно сказать, через какое минимальное время мальчики окажутся на одинаковом расстоянии от точки старта. Обратите внимание, что в этот момент они могли находиться в разных точках. Расстоянием от точки A до точки B называется минимальное из расстояний, которое нужно пробежать из точки A по или против часовой стрелки, чтобы оказаться в B.

## Формат ввода

В единственной строке вводится 5 целых чисел L,x1,v1,x2,v2 (1≤L≤109, 0≤x1,x2<L, ∣∣v1∣∣,∣∣v2∣∣≤109) — длины стадиона в метрах, начальная точка Кирилла, скорость Кирилла, начальная точка Антона, скорость Антона.

## Формат вывода

В первой строке выведите слово «YES», если случится момент, когда мальчики будут на одинаковом расстоянии от старта, или «NO», если такого момента не произойдёт.

Если ответ «YES», то во второй строке выведите одно вещественное число — через какое минимальное количество времени мальчики окажутся на одинаковом расстоянии от старта.

Вы можете выводить каждую букву в любом регистре (строчную или заглавную). Например, строки «yEs», «yes», «Yes» и «YES» будут приняты как положительный ответ.

Ваш ответ будет считаться правильным, если его абсолютная или относительная ошибка не превосходит 10−9.

Формально, пусть ваш ответ равен a, а ответ жюри равен b. Ваш ответ будет зачтен, если и только если |a−b|max(1,|b|)≤10−9.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 6 3 1 1 1 | YES  1.0000000000 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 12 8 10 5 20 | YES  0.3000000000 |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5 0 0 1 2 | YES  2.0000000000 |

### Пример 4

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 10 7 -3 1 4 | YES  0.8571428571 |

## Примечания

В первом тесте Кирилл изначально находится в точке 3 и бежит по часовой стрелке со скоростью 1. Антон находится в точке 1 и также бежит по часовой стрелке со скоростью 1. Через 1 секунду мальчики окажутся в точках 4 и 2 соответственно. Обе эти точки расположены на расстоянии 2 метра от старта (точки 0, совпадающей с точкой 6). Можно показать, что до этого они всегда находились на разном расстоянии от старта. Значит, ответ — 1.

Во втором тесте оба мальчика окажутся в точке 11 через 0.3 секунды.

В третьем Антон прибежит к Кириллу в точку 0 за 2 секунды.

# I. Расписание

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Как же Илье надоело учиться! Сначала школа, потом университет... Вот, наконец, наступил тот долгожданный день, когда Илье утром не надо ехать на учебу. Но, к несчастью для Ильи, оказалось, что после окончания университета начинается самое трудное — надо устраиваться на работу.

Во всемирно известной фирме «Goondex», в которую устроился Илья, принято очень много работать, в частности, для сотрудников установлена шестидневная рабочая неделя. Но, в качестве бонуса, «Goondex» каждый год предлагает своим сотрудникам выбрать любой день недели в качестве выходного. В свою очередь, оставшиеся шесть дней недели будут рабочими.

Илья сообразил, что с учётом государственных праздников (которые всегда являются выходными) с помощью правильного выбора выходного дня недели можно варьировать количество рабочих дней в году. Теперь он хочет знать, какой день недели ему следует выбрать в качестве выходного, чтобы отдыхать как можно больше дней в году, или, наоборот, демонстрировать чудеса трудолюбия, работая по максимуму.

## Формат ввода

В первой строке входных данных находится одно целое число *N* (*0 ≤ N ≤ 366*) — количество государственных праздников.

Во второй строке содержится одно целое число *year* (*1800 ≤ year ≤ 2100*) — год, в который необходимо помочь Илье.

В каждой из последующих *N* строк расположено по паре чисел *day* *month* (*day* — целое число, *month* — слово, между *day* и *month* ровно один пробел), обозначающих, что день *day* месяца *month* является государственным праздником.

В последней строке расположено слово https://contest.yandex.ru/testsys/tex/render/ZGF5XF9vZlxfd2Vlaw==.png — день недели первого января в год *year*.

Гарантируется, что все даты указаны корректно (в том числе указанный день недели первого января действительно является днём недели первого января соответствующего года *year*) и все дни государственных праздников различны.

## Формат вывода

Выведите через пробел два дня недели — лучший и худший варианты дней недели для выходного (то есть дни недели, для которых достигается соответственно максимальное и минимальное количество выходных дней в году). Если возможных вариантов ответа несколько, выведите любой из них.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2  2015  1 January  8 January  Thursday | Monday Thursday |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  2013  1 January  8 January  15 January  Tuesday | Monday Tuesday |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  2013  6 February  13 February  20 February  Tuesday | Tuesday Wednesday |

## Примечания

Рассмотрим подробно **третий пример**.

2013 год начинается и заканчивается во вторник (Tuesday), при этом на вторник приходится *53* дня года, а на все остальные дни недели –— по *52* дня. Все три государственных праздника выпадают на среду (Wednesday). Если Илья выберет в качестве выходного дня вторник, то в году у него будет *53 + 3 = 56* выходных дней (*53* вторника и *3* государственных праздника). Если Илья выберет в качестве выходного дня среду, то у него будет только *52* выходных дня. Если же Илья выберет в качестве выходного дня любой другой день недели, то у него будет *52 + 3 = 55* выходных дней.

Таким образом, лучший вариант для выходного дня — вторник, худший — среда, и **единственным** правильным ответом в данном примере является Tuesday Wednesday.

Соответствие названий месяцев и дней недели в английском и русском языках:

# J. Форматирование документа

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вася пишет новую версию своего офисного пакета "Closed Office". Недавно он начал работу над редактором "Dword", входящим в состав пакета.

Последняя проблема, с которой столкнулся Вася — размещение рисунков в документе. Он никак не может добиться стабильного отображения рисунков в тех местах, в которые он их помещает. Окончательно отчаявшись написать соответствующий модуль самостоятельно, Вася решил обратиться за помощью к вам. Напишите программу, которая будет осуществлять размещение документа на странице.

Документ в формате редактора "Dword" представляет собой последовательность абзацев. Каждый абзац представляет собой последовательность элементов – слов и рисунков. Элементы одного абзаца разделены пробелами и/или переводом строки. Абзацы разделены пустой строкой. Строка, состоящая только из пробелов, считается пустой.

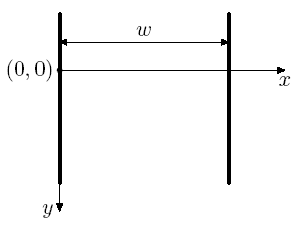
Слово — это последовательность символов, состоящая из букв латинского алфавита, цифр, и знаков препинания: ".", ",", ":", ";", "!", "?", "-", "'".

Рисунок описывается следующим образом: "(**image** image parameters)". Каждый параметр рисунка имеет вид "имя=значение". Параметры рисунка разделены пробелами и/или переводом строки. У каждого рисунка обязательно есть следующие параметры:

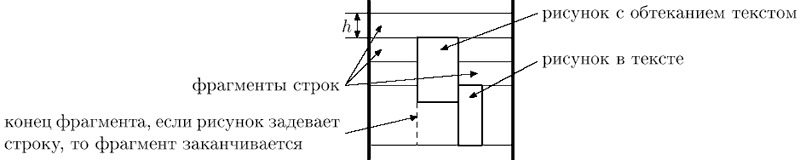
width — целое положительное число, ширина рисунка в пикселях height — целое положительное число, высота рисунка в пикселях layout — одно из следующих значений: embedded (в тексте), surrounded (обтекание текстом), floating (свободное), описывает расположение рисунка относительно текста

Документ размещается на бесконечной вверх и вниз странице шириной *w* пикселей (разбиение на конечные по высоте страницы планируется в следующей версии редактора). Одна из точек на левой границе страницы условно считается точкой с ординатой равной нулю. Ордината увеличивается вниз.

Размещение документа происходит следующим образом. Абзацы размещаются по очереди. Первый абзац размещается так, что его верхняя граница имеет ординату 0.



Абзац размещается следующим образом. Элементы располагаются по строкам. Каждая строка исходно имеет высоту *h* пикселей. В процессе размещения рисунков высота строк может увеличиваться, и строки могут разбиваться рисунками на фрагменты.



Слова размещаются следующим образом. Считается, что каждый символ имеет ширину c пикселей. Перед каждым словом, кроме первого во фрагменте, ставится пробел шириной также в *c* пикселей. Если слово помещается в текущем фрагменте, то оно размещается на нем. Если слово не помещается в текущем фрагменте, то оно размещается в первом фрагменте текущей строки, расположенном правее текущего, в котором оно помещается. Если такого фрагмента нет, то начинается новая строка, и поиск подходящего фрагмента продолжается в ней. Слово всегда "прижимается" к верхней границе строки.

Размещение рисунка зависит от его расположения относительно текста.

Если расположение рисунка относительно текста установлено в "embedded", то он располагается так же, как слово, за тем исключением, что его ширина равна ширине, указанной в параметрах рисунка. Кроме того, если высота рисунка больше текущей высоты строки, то она увеличивается до высоты рисунка (при этом верхняя граница строки не перемещается, а смещается вниз нижняя граница). Если рисунок типа "embedded" не первый элемент во фрагменте, то перед ним ставится пробел шириной *c* пикселей. Рисунки типа "embedded" также прижимаются к верхней границе строки.

Если расположение рисунка относительно текста установлено в "surrounded", то рисунок размещается следующим образом. Сначала аналогично находится первый фрагмент, в котором рисунок помещается по ширине. При этом перед рисунком этого типа не ставится пробел, даже если это не первый элемент во фрагменте.

После этого рисунок размещается следующим образом: верхний край рисунка совпадает с верхней границей строки, в которой находится найденный фрагмент, а сам рисунок продолжается вниз. При этом строки, через которые он проходит, разбиваются им на фрагменты.

Если расположение рисунка относительно текста установлено в "floating", то рисунок размещается поверх текста и других рисунков и никак с ними не взаимодействует. В этом случае у рисунка есть два дополнительных параметра: "dx" и "dy" — целые числа, задающие смещение в пикселях верхнего левого угла рисунка вправо и вниз, соответственно, относительно позиции, где находится верхний правый угол предыдущего слова или рисунка (или самой левой верхней точки первой строки абзаца, если рисунок — первый элемент абзаца).

Если при размещении рисунка таким образом он выходит за левую границу страницы, то он смещается вправо, так, чтобы его левый край совпадал с левой границей страницы. Аналогично, если рисунок выходит за правую границу страницы, то он смещается влево, чтобы его правый край совпадал с правой границей страницы.

Верхняя граница следующего абзаца совпадает с более низкой точкой из нижней границы последней строки и самой нижней границы рисунков типа "surrounded" предыдущего абзаца.

По заданным *w*, *h*, *c* и документу найдите координаты верхних левых углов всех рисунков в документе.

## Формат ввода

Первая строка входного файла содержит три целых числа: *w*, *h* и *c* (*1 ≤ w ≤ 1000*, *1 ≤ h ≤ 50*, *1 ≤ c ≤ w*).

Далее следует документ. Размер входного файла не превышает 1000 байт. Гарантируется, что ширина любого слова и любого рисунка не превышает *w*. Высота всех рисунков не превышает *1000*. Относительное смещение всех рисунков типа «floating» не превышает *1000* по абсолютной величине.

## Формат вывода

Выведите в выходной файл по два числа для каждого рисунка — координаты его верхнего левого угла. Выводите координаты рисунков в том порядке, в котором они встречаются во входном файле.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 120 10 8  start (image layout=embedded width=12 height=5)  (image layout=surrounded width=25 height=58)  and word is  (image layout=floating dx=18 dy=-15 width=25 height=20)  here new  (image layout=embedded width=20 height=22)  another  (image layout=embedded width=40 height=19)  longword  new paragraph  (image layout=surrounded width=5 height=30)  (image layout=floating width=20 height=35 dx=50 dy=-16) | 48 0  60 0  74 -5  32 20  0 52  104 81  100 65 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1000 2 3 |  |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 100 2 3  (image dx=10 dy=11 height=100 width=20 layout=floating) | 10 11 |

# Часть 2

# A. Минимальный прямоугольник

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

На клетчатой плоскости закрашено K клеток. Требуется найти минимальный по площади прямоугольник, со сторонами, параллельными линиям сетки, покрывающий все закрашенные клетки.

## Формат ввода

Во входном файле, на первой строке, находится число *K* (*1 ≤ K ≤ 100*). На следующих *K* строках находятся пары чисел *Xi* и *Yi* — координаты закрашенных клеток (*|Xi|*, *|Yi| ≤ 109*).

## Формат вывода

Выведите в выходной файл координаты левого нижнего и правого верхнего углов прямоугольника.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 4  1 3  3 1  3 5  6 3 | 1 1 6 5 |

# B. Продавец рыбы

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вася решил заняться торговлей рыбой. С помощью методов машинного обучения он предсказал цены на рыбу на *N* дней вперёд. Он решил, что в один день он купит рыбу, а в один из следующих дней — продаст (то есть совершит или ровно одну покупку и продажу или вообще не совершит покупок и продаж, если это не принесёт ему прибыли). К сожалению, рыба — товар скоропортящийся и разница между номером дня продажи и номером дня покупки не должна превышать *K*.

Определите, какую максимальную прибыль получит Вася.

## Формат ввода

В первой строке входных данных задаются числа *N* и *K* (*1 ≤ N ≤ 10000*, *1 ≤ K ≤ 100*).

Во второй строке задаются цены на рыбу в каждый из *N* дней. Цена — целое число, которое может находится в пределах от 1 до *109*.

## Формат вывода

Выведите одно число — максимальную прибыль, которую получит Вася.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5 2  1 2 3 4 5 | 2 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5 2 5 C. Петя, Маша и верёвочки  |  |  | | --- | --- | | Ограничение времени | 1 секунда | | Ограничение памяти | 512Mb | | Ввод | стандартный ввод или input.txt | | Вывод | стандартный вывод или output.txt |   На столе лежали две одинаковые верёвочки целой положительной длины.  Петя разрезал одну из верёвочек на *N* частей, каждая из которых имеет целую положительную длину, так что на столе стало *N+1* верёвочек. Затем в комнату зашла Маша и взяла одну из лежащих на столе верёвочек. По длинам оставшихся на столе *N* верёвочек определите, какую **наименьшую** длину может иметь верёвочка, взятая Машей. Формат ввода Первая строка входных данных содержит одно целое число *N* — количество верёвочек, оставшихся на столе (*2 ≤ N ≤ 1000*). Во второй строке содержится *N* целых чисел *li* — длины верёвочек (*1 ≤ li ≤ 1000*). Формат вывода Выведите одно целое число — наименьшую длину, которую может иметь верёвочка, взятая Машей. Пример 1  | **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод | | --- | --- | | 4  1 5 2 1 | 1 |  Пример 2  | **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод | | --- | --- | | 4  5 12 4 3 | 24 |  D. Шахматная доска  |  |  | | --- | --- | | Ограничение времени | 1 секунда | | Ограничение памяти | 64Mb | | Ввод | стандартный ввод или input.txt | | Вывод | стандартный вывод или output.txt |   Из шахматной доски по границам клеток выпилили связную (не распадающуюся на части) фигуру без дыр. Требуется определить ее периметр. Формат ввода Сначала вводится число N (1 ≤ N ≤ 64) – количество выпиленных клеток. В следующих N строках вводятся координаты выпиленных клеток, разделенные пробелом (номер строки и столбца – числа от 1 до 8). Каждая выпиленная клетка указывается один раз. Формат вывода Выведите одно число – периметр выпиленной фигуры (сторона клетки равна единице). Пример 1  | **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод | | --- | --- | | 3  1 1  1 2  2 1 | 8 |  Пример 2  | **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод | | --- | --- | | 1  8 8 | 4 |  Примечания 1) Вырезан уголок из трех клеток. Сумма длин его сторон равна 8.  2) Вырезана одна клетка. Ее периметр равен 4. | 0 |

# E. Амбициозная улитка

|  |  |
| --- | --- |
| Ограничение времени | 5 секунд |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Домашний питомец мальчика Васи — улитка Петя. Петя обитает на бесконечном в обе стороны вертикальном столбе, который для удобства можно представить как числовую прямую. Изначально Петя находится в точке 0.

Вася кормит Петю ягодами. У него есть n ягод, каждая в единственном экземпляре. Вася знает, что если утром он даст Пете ягоду с номером i, то поев и набравшись сил, за остаток дня Петя поднимется на ai единиц вверх по столбу, но при этом за ночь, потяжелев, съедет на bi единиц вниз. Параметры различных ягод могут совпадать.

Пете стало интересно, а как оно там, наверху, и Вася взялся ему в этом помочь. Ближайшие n дней он будет кормить Петю ягодами из своего запаса таким образом, чтобы максимальная высота, на которой побывал Петя за эти n дней была максимальной. К сожалению, Вася не умеет программировать, поэтому он попросил вас о помощи. Найдите, максимальную высоту, на которой Петя сможет побывать за эти n дней и в каком порядке Вася должен давать Пете ягоды, чтобы Петя смог её достичь!

## Формат ввода

В первой строке входных данных дано число n (1≤n≤5⋅105) — количество ягод у Васи. В последующих n строках описываются параметры каждой ягоды. В i+1 строке дано два числа ai и bi (0≤ai,bi≤109) — то, насколько поднимется улитка за день после того, как съест i ягоду и насколько опуститься за ночь.

## Формат вывода

В первой строке выходных данных выведите единственное число — максимальную высоту, которую сможет достичь Петя, если Вася будет его кормить оптимальным образом. В следующей строке выведите n различных целых чисел от 1 до n — порядок, в котором Вася должен кормить Петю (i число в строке соответствует номеру ягоды, которую Вася должен дать Пете в i день чтобы Петя смог достичь максимальной высоты).

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  1 5  8 2  4 4 | 10  2 3 1 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2  7 6  7 4 | 10  2 1 |

## Примечания

Во втором примере изначально улитка находится на высоте 0. Пусть сначала Петя накормит её второй ягодой, а затем первой. После того как она съест вторую ягоду, за день она поднимется на 7 (и окажется на высоте 7), а за ночь опустится на 4 (и окажется на высоте 3). После того как она съест первую ягоду, за день она поднимется на 7 (и окажется на высоте 10), а за ночь опустится на 6 (и окажется на высоте 4).

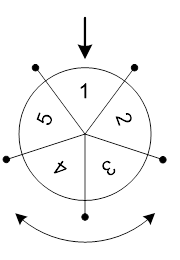
Таким образом, максимальная высота, на которой побывает улитка при данном порядке кормления, равна 10. Нетрудно видеть, что если Петя накормит улитку сначала первой ягодой, а затем второй, то максимальная высота, на которой побывает улитка, будет меньше

# F. Колесо Фортуны

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Развлекательный телеканал транслирует шоу «Колесо Фортуны». В процессе игры участники шоу крутят большое колесо, разделенное на сектора. В каждом секторе этого колеса записано число. После того как колесо останавливается, специальная стрелка указывает на один из секторов. Число в этом секторе определяет выигрыш игрока.

Юный участник шоу заметил, что колесо в процессе вращения замедляется из-за того, что стрелка задевает за выступы на колесе, находящиеся между секторами. Если колесо вращается с угловой скоростью *v* градусов в секунду, и стрелка, переходя из сектора *X* к следующему сектору, задевает за очередной выступ, то текущая угловая скорость движения колеса уменьшается на *k* градусов в секунду. При этом если *v ≤ k*, то колесо не может преодолеть препятствие и останавливается. Стрелка в этом случае будет указывать на сектор *X*.



Юный участник шоу собирается вращать колесо. Зная порядок секторов на колесе, он хочет заставить колесо вращаться с такой начальной скоростью, чтобы после остановки колеса стрелка указала на как можно большее число. Колесо можно вращать в любом направлении и придавать ему начальную угловую скорость от *a* до *b* градусов в секунду.

Требуется написать программу, которая по заданному расположению чисел в секторах, минимальной и максимальной начальной угловой скорости вращения колеса и величине замедления колеса при переходе через границу секторов вычисляет максимальный выигрыш.

## Формат ввода

Первая строка входного файла содержит целое число *n* — количество секторов колеса (*3 ≤ n ≤ 100*).

Вторая строка входного файла содержит *n* положительных целых чисел, каждое из которых не превышает 1000 — числа, записанные в секторах колеса. Числа приведены в порядке следования секторов по часовой стрелке. Изначально стрелка указывает на первое число.

Третья строка содержит три целых числа: *a*, *b* и *k* (*1 ≤ a ≤ b ≤ 109*, *1 ≤ k ≤ 109*).

## Формат вывода

В выходном файле должно содержаться одно целое число — максимальный выигрыш.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  1 2 3 4 5  3 5 2 | 5 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  1 2 3 4 5  15 15 2 | 4 |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  5 4 3 2 1  2 5 2 | 5 |

## Примечания

В первом примере возможны следующие варианты: можно придать начальную скорость колесу равную 3 или 4, что приведет к тому, что стрелка преодолеет одну границу между секторами, или придать начальную скорость равную 5, что позволит стрелке преодолеть 2 границы между секторами. В первом варианте, если закрутить колесо в одну сторону, то выигрыш получится равным 2, а если закрутить его в противоположную сторону, то — 5. Во втором варианте, если закрутить колесо в одну сторону, то выигрыш будет равным 3, а если в другую сторону, то — 4.

Во втором примере возможна только одна начальная скорость вращения колеса — 15 градусов в секунду. В этом случае при вращении колеса стрелка преодолеет семь границ между секторами. Тогда если его закрутить в одном направлении, то выигрыш составит 4, а если в противоположном направлении, то — 3.

Наконец, в третьем примере оптимальная начальная скорость вращения колеса равна 2 градусам в секунду. В этом случае стрелка вообще не сможет преодолеть границу между секторами, и выигрыш будет равен 5.

# G. Ни больше ни меньше

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Дан массив целых положительных чисел *a* длины *n*. Разбейте его на **минимально возможное** количество отрезков, чтобы каждое число было не меньше длины отрезка которому оно принадлежит. Длиной отрезка считается количество чисел в нем.

Разбиение массива на отрезки считается корректным, если каждый элемент принадлежит ровно одному отрезку.

## Формат ввода

Первая строка содержит одно целое число *t* *(1 ≤ t ≤ 1 000)* — количество наборов тестовых данных. Затем следуют *t* наборов тестовых данных.

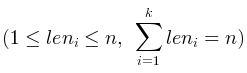
Первая строка набора тестовых данных содержит одно целое число *n* *(1 ≤ n ≤ 105)* — длину массива.

Следующая строка содержит *n* целых чисел *a1, a2, …, an* *(1 ≤ ai ≤ n)* — массив *a*.

Гарантируется, что сумма *n* по всем наборам тестовых данных не превосходит *2 ⋅ 105*.

## Формат вывода

Для каждого набора тестовых данных в первой строке выведите число *k* — количество отрезков в вашем разбиении.

Затем в следующей строке выведите *k* чисел *len1, len2, …, lenk*  — длины отрезков в порядке слева направо.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  5  1 3 3 3 2  16  1 9 8 7 6 7 8 9 9 9 9 9 9 9 9 9  7  7 2 3 4 3 2 7 | 3  1 2 2  3  1 6 9  3  2 3 2 |

## Примечания

Ответы в примере соответствуют разбиениям:

*{[1], [3, 3], [3, 2]}*

*{[1], [9, 8, 7, 6, 7, 8], [9, 9, 9, 9, 9, 9, 9, 9, 9]}*

*{[7, 2], [3, 4, 3], [2, 7]}*

В первом наборе тестовых данных набор длин *{1, 3, 1}*, соответствующий разбиению *{[1], [3, 3, 3], [2]}*, также был бы корректным.

# H. Наилучший запрет

|  |  |
| --- | --- |
| Ограничение времени | 3 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Константин и Михаил играют в настольную игру «Ярость Эльфов». В игре есть *n* рас и *m* классов персонажей. Каждый персонаж характеризуется своими расой и классом. Для каждой расы и каждого класса существует ровно один персонаж такой расы и такого класса. Сила персонажа *i*-й расы и *j*-го класса равна *ai j*, и обоим игрокам это прекрасно известно.

Сейчас Константин будет выбирать себе персонажа. Перед этим Михаил может запретить одну расу **и** один класс, чтобы Константин не мог выбирать персонажей, у которых такая раса **или** такой класс. Конечно же, Михаил старается, чтобы Константину достался как можно более слабый персонаж, а Константин, напротив, выбирает персонажа посильнее. Какие расу и класс следует запретить Михаилу?

## Формат ввода

Первая строка содержит два целых числа *n* и *m* (*2 ≤ n,m ≤ 1000*) через пробел — количество рас и классов в игре «Ярость Эльфов», соответственно.

В следующих *n* строках содержится по *m* целых чисел через пробел. *j*-е число *i*-й из этих строк — это *ai j* (*1 ≤ ai j ≤ 109*).

## Формат вывода

В единственной строке выведите два целых числа через пробел — номер расы и номер класса, которые следует запретить Михаилу. Расы и классы нумеруются с единицы. Если есть несколько возможных ответов, выведите любой из них.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2 2  1 2  3 4 | 2 2 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3 4  1 3 5 7  9 11 2 4  6 8 10 12 | 3 2 |

I. Пираты Баренцева моря

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вася играет в настольную игру «Пираты Баренцева моря», которая посвящена морским битвам. Игровое поле представляет собой квадрат из N×N клеток, на котором расположено N кораблей (каждый корабль занимает одну клетку).

Вася решил воспользоваться линейной тактикой, для этого ему необходимо выстроить все N кораблей в одном столбце. За один ход можно передвинуть один корабль в одну из четырёх соседних по стороне клеток. Номер столбца, в котором будут выстроены корабли, не важен. Определите минимальное количество ходов, необходимых для построения кораблей в одном столбце. В начале и процессе игры никакие два корабля не могут находиться в одной клетке.

Формат ввода

В первой строке входных данных задаётся число N (1≤N≤100).

В каждой из следующих N строк задаются координаты корабля: сначала номер строки, затем номер столбца (нумерация начинается с единицы).

Формат вывода

Выведите одно число — минимальное количество ходов, необходимое для построения.

Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  1 2  3 3  1 1 | 3 |

Примечания

В примере необходимо выстроить корабли в столбце номер 2. Для этого необходимо переставить корабль из клетки 3 3 в клетку 3 2 за один ход, а корабль из клетки 1 1 в клетку 2 2 за два хода. Существуют и другие варианты перестановки кораблей, однако ни в одном из них нет меньше трёх ходов.

# J. Два прямоугольника

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Недавно один известный художник-абстракционист произвел на свет новый шедевр — картину «Два черных непересекающихся прямоугольника». Картина представляет собой прямоугольник *m× n*, разбитый на квадраты *1× 1*, некоторые из которых закрашены любимым цветом автора — черным. Федя — не любитель абстрактных картин, однако ему стало интересно, действительно ли на картине изображены два непересекающихся прямоугольника. Помогите ему это узнать. Прямоугольники не пересекаются в том смысле, что они не имеют общих клеток.

## Формат ввода

Первая строка входного файла содержит числа *m* и *n* (*1 ≤ m, n ≤ 200*). Следующие *m* строк содержат описание рисунка. Каждая строка содержит ровно *n* символов. Символ «.» обозначает пустой квадрат, а символ «#» — закрашенный.

## Формат вывода

Если рисунок можно представить как два непересекающихся прямоугольника, выведите в первой строке «YES», а в следующих *m* строках выведите рисунок в том же виде, в каком он задан во входном файле, заменив квадраты, соответствующие первому прямоугольнику на символ «a», а второму — на символ «b». Если решений несколько, выведите любое.

Если же этого сделать нельзя, выведите в выходной файл «NO».

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2 1  #  . | NO |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2 2  ..  ## | YES  ..  ab |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 3  ### | YES  abb |

### Пример 4

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 5  ####. | YES  abbb. |

# Часть 3

# A. Плейлисты

|  |  |
| --- | --- |
| Ограничение времени | 1.5 секунд |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Костя успешно прошел собеседование и попал на стажировку в отдел разработки сервиса «Музыка».

Конкретно ему поручили такое задание — научиться подбирать плейлист для группы друзей, родственников или коллег. При этом нужно подобрать такой плейлист, в который входят исключительно нравящиеся всем членам группы песни.

Костя очень хотел выполнить это задание быстро и качественно, но у него не получается. Помогите ему написать программу, которая составляет плейлист для группы людей.

## Формат ввода

В первой строке расположено одно натуральное число n(1≤n≤2⋅105), где n – количество человек в группе.

В следующих 2⋅n строках идет описание любимых плейлистов членов группы. По 2 строки на каждого участника.

В первой из этих 2-х строк расположено число ki — количество любимых треков i-го члена группы. В следующей строке расположено ki строк через пробел — названия любимых треков i-го участника группы.

Каждый трек в плейлисте задан в виде строки, все строки уникальны, сумма длин строк не превосходит 2⋅106. Строки содержат большие и маленькие латинские буквы и цифры.

## Формат вывода

Выведите количество, а затем сам список песен через пробел — список треков, которые нравятся каждому участнику группы. Ответ необходимо **отсортировать**в лексикографическом порядке!

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1  2  GoGetIt Life | 2  GoGetIt Life |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2  2  Love Life  2  Life GoodDay | 1  Life |

# B. Анаграмма?

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Задано две строки, нужно проверить, является ли одна анаграммой другой. Анаграммой называется строка, полученная из другой перестановкой букв.

## Формат ввода

Строки состоят из строчных латинских букв, их длина не превосходит 100000. Каждая записана в отдельной строке.

## Формат вывода

Выведите "YES" если одна из строк является анаграммой другой и "NO" в противном случае.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| dusty  study | YES |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| Rat  Bat | NO |

# C. Удаление чисел

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Дан массив a из n чисел. Найдите минимальное количество чисел, после удаления которых попарная разность оставшихся чисел по модулю не будет превышать 1, то есть после удаления ни одно число не должно отличаться от какого-либо другого более чем на 1.

## Формат ввода

Первая строка содержит одно целое число n (1≤n≤2⋅105) — количество элементов массива a.

Вторая строка содержит n целых чисел a1,a2,…,an (0≤ai≤105) — элементы массива a.

## Формат вывода

Выведите одно число — ответ на задачу.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  1 2 3 4 5 | 3 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 10  1 1 2 3 5 5 2 2 1 5 | 4 |

# D. Повторяющееся число

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вам дана последовательность измерений некоторой величины. Требуется определить, повторялась ли какое-либо число, причём расстояние между повторами не превосходило *k*.

## Формат ввода

В первой строке задаются два числа *n* и *k* (*1 ≤ n, k ≤ 105*).

Во второй строке задаются *n* чисел, по модулю не превосходящих *109*.

## Формат вывода

Выведите YES, если найдется повторяющееся число и расстояние между повторами не превосходит *k* и NO в противном случае.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 4 2  1 2 3 1 | NO |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 4 1  1 0 1 1 | YES |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 6 2  1 2 3 1 2 3 | NO |

# E. Два из трех

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вам даны три списка чисел. Найдите все числа, которые встречаются хотя бы в двух из трёх списков.

## Формат ввода

Во входных данных описывается три списка чисел. Первая строка каждого описания списка состоит из длины списка *n* (*1 ≤ n ≤ 1000*). Вторая строка описания содержит список натуральных чисел, записанных через пробел. Числа не превосходят *109*.

## Формат вывода

Выведите все числа, которые содержатся хотя бы в двух списках из трёх, в порядке возрастания. Обратите внимание, что каждое число необходимо выводить только один раз.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2  3 1  2  1 3  2  1 2 | 1 3 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  1 2 2  3  3 4 3  1  5 |  |

# F. Замена слов

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

С целью экономии чернил в картридже принтера было принято решение укоротить некоторые слова в тексте. Для этого был составлен словарь слов, до которых можно сокращать более длинные слова. Слово из текста можно сократить, если в словаре найдется слово, являющееся началом слова из текста. Например, если в списке есть слово "лом", то слова из текста "ломбард", "ломоносов" и другие слова, начинающиеся на "лом", можно сократить до "лом".

Если слово из текста можно сократить до нескольких слов из словаря, то следует сокращать его до самого короткого слова.

## Формат ввода

В первой строке через пробел вводятся слова из словаря, слова состоят из маленьких латинских букв. Гарантируется, что словарь не пуст и количество слов в словаре не превышет 1000, а длина слов — 100 символов.

Во второй строке через пробел вводятся слова текста (они также состоят только из маленьких латинских букв). Количество слов в тексте не превосходит *105*, а суммарное количество букв в них — *106*.

## Формат вывода

Выведите текст, в котором осуществлены замены.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| a b  abdafb basrt casds dsasa a | a b casds dsasa a |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| aa bc aaa  a aa aaa bcd abcd | a aa aa bc abcd |

# G. Построить квадрат

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Задано множество, состоящее из *N* различных точек на плоскости. Координаты всех точек — целые числа. Определите, какое минимальное количество точек нужно добавить во множество, чтобы нашлось четыре точки, лежащие в вершинах квадрата.

## Формат ввода

В первой строке вводится число *N* (*1 ≤ N ≤ 2000*) — количество точек.

В следующих *N* строках вводится по два числа *xi*, *yi* (*-108 ≤ xi, yi ≤ 108*) — координаты точек.

## Формат вывода

В первой строке выведите число *K* — минимальное количество точек, которые нужно добавить во множество.

В следующих *K* строках выведите координаты добавленных точек *xi*, *yi* через пробел. Координаты должны быть целыми и не превышать *109* по модулю.

Если решений несколько — выведите любое из них.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2  0 1  1 0 | 2  0 0  1 1 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  0 2  2 0  2 2 | 1  0 0 |

### Пример 3

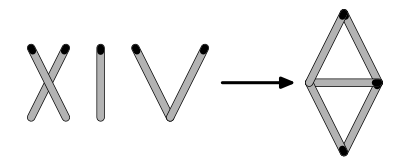
| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 4  -1 1  1 1  -1 -1  1 -1 | 0 |

# H. Спички детям не игрушка!

|  |  |
| --- | --- |
| Ограничение времени | 3 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Вася любит решать головоломки со спичками. Чаще всего они формулируется следующим образом: дано изображение *A*, составленное из спичек; переложите в нем минимальное количество спичек так, чтобы получилось изображение *B*.

Например, из номера текущего командного чемпионата школьников Санкт-Петербурга по программированию, можно получить ромб с диагональю, переложив всего три спички.



Головоломки, которые решает Вася, всегда имеют решение. Это значит, что набор спичек, используемый в изображении *A*, совпадает с набором спичек, используемым в изображении *B*. Кроме того, в одном изображении никогда не встречаются две спички, у которых есть общий участок ненулевой длины (то есть спички могут пересекаться, но не могут накладываться друг на друга).

Вася устал решать головоломки вручную, и теперь он просит вас написать, программу, которая будет решать головоломки за него. Программа будет получать описания изображений *A* и *B* и должна найти минимальное количество спичек, которые надо переложить в изображении *A*, чтобы полученная картинка получалась из *B* параллельным переносом.

## Формат ввода

В первой строке входного файла содержится целое число *n* — количество спичек в каждом из изображений (*1 ≤ n ≤ 1000*).

В следующих *n* строках записаны координаты концов спичек на изображении *A*. Спичка номер *i* описывается целыми числами *x1i*, *y1i*, *x2i*, *y2i* — координатами ее концов. Следующие *n* строк содержат описание изображения *B* в таком же формате. Набор длин этих спичек совпадает с набором длин спичек с изображения *A*.

Все координаты по абсолютной величине не превосходят *104*. Все спички имеют ненулевую длину, то есть *x1i ≠ x2i* или *y1i ≠ y2i*.

## Формат вывода

Выведите в выходной файл минимальное количество спичек, которые следует переложить, чтобы изображение *A* совпало с изображением *B*, с точностью до параллельного переноса.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  0 0 1 2  1 0 0 2  2 0 2 2  4 0 3 2  4 0 5 2  9 -1 10 1  10 1 9 3  8 1 10 1  8 1 9 -1  8 1 9 3 | 3 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1  3 4 7 9  -1 3 3 8 | 0 |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1  -4 5 2 -3  -12 4 -2 4 | 1 |

# I. Играйте в футбол!

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Ася Вуткина — известный футбольный комментатор. Будучи профессионалом своего дела, Ася тщательно следит за всеми матчами всех европейских чемпионатов.

Благодаря накопленной информации, Ася может во время трансляции матча сообщить какую-нибудь интересную статистику, например: «Индзаги третий матч подряд забивает гол на 9-й минуте» или «Матерацци никогда не открывает счет в матче».

Но мозг Аси не безграничен, а помнить всю историю футбола просто невозможно. Поэтому Ася попросила вас написать программу, которая собирает статистику матчей и умеет отвечать на некоторые запросы, касающиеся истории футбола.

Информация о матче сообщается программе в следующей форме:

"<Название 1-й команды>" - "<Название 2-й команды>" <Счет 1-й команды>:<Счет 2-й команды>

<Автор 1-го забитого мяча 1-й команды> <Минута, на которой был забит мяч>'

<Автор 2-го забитого мяча 1-й команды> <Минута, на которой был забит мяч>'

...

<Автор последнего забитого мяча 1-й команды> <Минута, на которой был забит мяч>'

<Автор 1-го забитого мяча 2-й команды> <Минута, на которой был забит мяч>'

...

<Автор последнего забитого мяча 2-й команды> <Минута, на которой был забит мяч>'

Запросы к программе бывают следующих видов:  
  
Total goals for <Название команды>

— количество голов, забитое данной командой за все матчи.

Mean goals per game for <Название команды>

— среднее количество голов, забиваемое данной командой за один матч. Гарантирутся, что к моменту подачи такого запроса команда уже сыграла хотя бы один матч.

Total goals by <Имя игрока>

— количество голов, забитое данным игроком за все матчи.

Mean goals per game by <Имя игрока>

— среднее количество голов, забиваемое данным игроком за один матч его команды.

Гарантирутся, что к моменту подачи такого запроса игрок уже забил хотя бы один гол.

Goals on minute <Минута> by <Имя игрока>

— количество голов, забитых данным игроком ровно на указанной минуте матча.

Goals on first <T> minutes by <Имя игрока>

— количество голов, забитых данным игроком на минутах с первой по *T*-ю включительно.

Goals on last <T> minutes by <Имя игрока>

— количество голов, забитых данным игроком на минутах с *(91 - T)*-й по 90-ю включительно.

Score opens by <Название команды>

— сколько раз данная команда открывала счет в матче.

Score opens by <Имя игрока>

— сколько раз данный игрок открывал счет в матче.

## Формат ввода

Входной файл содержит информацию о матчах и запросы в том порядке, в котором они поступают в программу Аси Вуткиной.

Во входном файле содержится информация не более чем о 100 матчах, в каждом из которых забито не более 10 голов. Всего в чемпионате участвует не более 20 команд, в каждой команде не более 10 игроков забивают голы.

Все названия команд и имена игроков состоят только из прописных и строчных латинских букв и пробелов, а их длина не превышает 30. Прописные и строчные буквы считаются различными. Имена и названия не начинаются и не оканчиваются пробелами и не содержат двух пробелов подряд. Каждое имя и название содержит хотя бы одну букву.

Минута, на которой забит гол — целое число от 1 до 90 (про голы, забитые в дополнительное время, принято говорить, что они забиты на 90-й минуте).

Для простоты будем считать, что голов в собственные ворота в европейских чемпионатах не забивают, и на одной минуте матча может быть забито не более одного гола (в том числе на 90-й). Во время чемпионата игроки не переходят из одного клуба в другой.

Количество запросов во входном файле не превышает 500.

## Формат вывода

Для каждого запроса во входном файле выведите ответ на этот запрос в отдельной строке. Ответы на запросы, подразумевающие нецелочисленный ответ, должны быть верны с точностью до трех знаков после запятой.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| "Juventus" - "Milan" 3:1  Inzaghi 45'  Del Piero 67'  Del Piero 90'  Shevchenko 34'  Total goals for "Juventus"  Total goals by Pagliuca  Mean goals per game by Inzaghi  "Juventus" - "Lazio" 0:0  Mean goals per game by Inzaghi  Mean goals per game by Shevchenko  Score opens by Inzaghi | 3  0  1.0  0.5  1.0  0 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| Total goals by Arshavin | 0 |

# J. P2P обновление

|  |  |
| --- | --- |
| Ограничение времени | 15 секунд |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

В системе умного дома под управлением голосового помощника Лариса *n* устройств, соединяющихся между собой по сети LoRaWAN. Устройство номер 1 подключено к интернету и на него было скачано обновление, которое необходимо передать на все устройства.

Сеть LoRaWAN очень медленная, поэтому для распространения протокола был придуман peer-to-peer (P2P) протокол. Файл обновления разбивается на *k* одинаковых по размеру частей, занумерованных от 1 до *k*.

Передача части обновления происходит во время таймслотов. Каждый таймслот занимает одну минуту. За один таймслот каждое устройство может получить и передать ровно одну часть обновления. То есть устройство во время таймслота может получать новую часть обновления и передавать уже имеющуюуся у него к началу таймслота часть обновления, или совершать только одно из этих действий, или вообще не осуществлять прием или передачу. После приема части обновления устройство может передавать эту часть обновления другим устройствам в следующих таймслотах.

Перед каждым таймслотом для каждой части обновления определяется, на скольких устройствах сети скачана эта часть. Каждое устройство выбирает отсутствующую на нем часть обновления, которая встречается в сети реже всего. Если таких частей несколько, то выбирается отсутствующая на устройстве часть обновления с наименьшим номером.

После этого устройство делает запрос выбранной части обновления у одного из устройств, на котором такая часть обновления уже скачана. Если таких устройств несколько — выбирается устройство, на котором скачано наименьшее количество частей обновления. Если и таких устройств оказалось несколько — выбирается устройство с минимальным номером.

После того, как все запросы отправлены, каждое устройство выбирает, чей запрос удовлетворить. Устройство *A* удовлетворяет тот запрос, который поступил от наиболее ценного для *A* устройства. Ценность устройства *B* для устройства *A* определяется как количество частей обновления, ранее полученных устройством *A* от устройства *B*. Если на устройство *A* пришло несколько запросов от одинаково ценных устройств, то удовлетворяется запрос того устройства, на котором меньше всего скачанных частей обновления. Если и таких запросов несколько, то среди них выбирается устройство с наименьшим номером.

Далее начинается новый таймслот. Устройства, чьи запросы удовлетворены, скачивают запрошенную часть обновления, а остальные не скачивают ничего.

Для каждого устройства определите, сколько таймслотов понадобится для скачивания всех частей обновления.

## Формат ввода

Вводится два числа *n* и *k* (*2 ≤ n ≤ 100*, *1 ≤ k ≤ 200*).

## Формат вывода

Выведите *n-1* число — количество таймслотов, необходимых для скачивания обновления на устройства с номерами от 2 до *n*.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3 2 | 3 3 |

## Примечания

Для удобства будем пользоваться обозначениями устройств буквами A, B, C (соответствует устройствам с номерами 1, 2 и 3). На устройстве A есть обе части обновления, а на устройствах B и C — ни одной.

Перед первым таймслотом для каждой части определяется количество устройств, на которых скачана каждая часть обновления: и 1 и 2 часть обновления присутствуют только на одном устройстве.

Устройства B и C выбирают самую редкую отсутствующую у них часть обновления с минимальным номером: самая редкая часть с минимальным номером — это часть 1. Она отсутствует и на устройстве B, и на устройстве С. Они запрашивают ее у устройства A. Ценность устройств B и C для устройства A равна нулю. Количество имеющихся у устройств B и C частей обновления одинакова и равно нулю. Поэтому устройство A выбирает устройство с минимальным номером (B). Во время первого таймслота выполняется передача части 1 с устройства A на устройство B. Ценность устройства A для устройства B становится равной 1.

Перед вторым таймслотом для каждой части определяется количество устройств, на которых скачана каждая часть обновления: самой редкой оказывается часть 2 (присутствует только на устройстве A), следующая по редкости часть 1 (присутствует на устройствах A и B).

Устройства B и C выбирают среди отсутствующих у них частей обновления самую редкую: для обоих устройств выбирается часть 2. Каждое из них делает запрос части 2 у единственного обладателя этой части — устройства A. Ценность устройств B и C для устройства A одинакова и равна нулю. Количество имеющихся у устройства C частей (0) меньше, чем у устройства B (1), поэтому выбирается устройство C. Во время второго таймслота выполняется передача части 2 с устройства A на устройство C. Ценность устройства A для устройства C становится равной 1.

Перед третьим таймслотом для каждой части определяется количество устройств, на которых скачана каждая часть обновления: обе части 1 и 2 присутствуют на двух устройствах (часть 1 на устройствах A и B, часть 2 — на устройствах A и C)

Устройство B может сделать запрос недостающей части 2 у обладающей ей устройств A и C, но выбирает устройство C, т.к. на устройстве C скачано меньше частей (1), чем у устройства A (2).

Устройство C может сделать запрос недостающей части 1 у обладающей ей устройств A и B, но выбирает устройство B, т.к. на устройстве B скачано меньше частей (1), чем у устройства A (2).

Во время третьего таймслота оба запроса оказываются единственными запросами у устройств B и C и удовлетворяются. Часть 2 передается с устройства C на устройство B. Часть 1 передается с устройства B на устройство C. Ценность устройства B для устройства C становится равной 1. Ценность устройства C для устройства B становится равной 1.

Все части обновления оказываются на всех устройствах и на этом обновление заканчивается.

# Часть 4

# A. Быстрый поиск в массиве

|  |  |
| --- | --- |
| Ограничение времени | 3 секунды |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Дан массив из N целых чисел. Все числа от −109 до 109.

Нужно уметь отвечать на запросы вида “Cколько чисел имеют значения отL доR?”.

## Формат ввода

Число N (1≤N≤105). Далее N целых чисел.

Затем число запросов K (1≤K≤105).

Далее K пар чисел L,R (−109≤L≤R≤109) — собственно запросы.

## Формат вывода

Выведите K чисел — ответы на запросы.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  10 1 10 3 4  4  1 10  2 9  3 4  2 2 | 5 2 2 0 |

# B. Одномерный морской бой

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Поле в игре в одномерный морской бой имеет размеры 1×n. Ваша задача — найти такое максимальное k, что на поле можно расставить один корабль размера 1×k, два корабля размера 1×(k−1), …, k кораблей размера 1×1, причем корабли, как и в обычном морском бое, не должны касаться друг друга и пересекаться.

## Формат ввода

В единственной строке входных данных дано число n — количество клеток поля (0≤n≤1018).

## Формат вывода

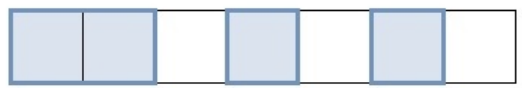
Выведите единственное число — такое максимальное k, что можно расставить корабли, как описано в условии.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 7 | 2 |

## Примечания

Пояснение к примеру: для поля 1×7 ответ равен 2. Расставить один корабль размера 1×2 и два корабля размера 1×1 можно следующим образом:



# C. Саруман

|  |  |
| --- | --- |
| Ограничение времени | 4 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Как известно, Саруман Радужный очень любит порядок. Поэтому все полки его войска стоят друг за другом, причем каждый следующий полк содержит количество орков не меньше, чем предыдущий.

Перед тем как напасть на Хельмову Падь, Саруман решил провести несколько вылазок для разведки. Чтобы его отряды никто не заметил, он решил каждый раз отправлять несколько подряд идущих полков так, чтобы суммарное количество орков в них было равно определенному числу. Так как это всего лишь разведка, каждый полк после вылазки возвращается на свое место. Задачу выбрать нужные полки он поручил Гриме Змеиному Языку. А Грима не поскупится на вознаграждение, если вы ему поможете.

## Формат ввода

В первой строке входного файла находится два целых числа: *n* (*1 ≤ n ≤ 2⋅105*) — количество полков и *m* (*1 ≤ m ≤ 2⋅105*) – количество предстоящих вылазок.

В следующей строке записано *n* чисел *ai*, где *ai* — число орков в *i*-ом полке (*1 ≤ ai ≤ 109, ai ≤ ai+1*).

Далее в *m* строках записаны запросы вида: количество полков *l* (*1 ≤ l ≤ n*), которые должны будут отправиться в эту вылазку, и суммарное количество орков в этих полках *s* (*1 ≤ s ≤ 2⋅1016*)

## Формат вывода

Для каждого запроса выведите номер полка, с которого начнутся те *l*, которые необходимо отправить на вылазку. Если таких полков несколько, выведите любой. Если же так выбрать полки нельзя, выведите *-1*.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5 2  1 3 5 7 9  2 4  1 3 | 1  2 |

# D. Рапорт

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Верс нужно подготовить рапорт о последнем боевом вылете. Она уже сочинила в голове текст, осталось лишь его записать. Рапорт будет состоять из двух частей: первая будет содержать n слов, i-е из которых состоит из ai букв, вторая — m слов, j-е из которых состоит из bj букв. Язык Крии не содержит никаких знаков препинания. Верс должна записать рапорт на клетчатом рулоне бумаги, шириной w клеток. Так как рапорт состоит из двух частей, она разделит вертикальной чертой рулон на две части целой ширины, после чего в левой части напишет первую часть, а в правой — вторую.

Обе части рапорта записываются аналогично, каждая на своей части рулона. Одна буква слова занимает ровно одну клетку. Первое слово записывается в первой строке рулона, начиная с самой левой клетки этой части рулона. Каждое следующее слово, если это возможно, должно быть записано в той же строке, что и предыдущее, и быть отделено от него ровно одной пустой клеткой. Иначе, оно пишется в следующей строке, начиная с самой левой клетки. Если ширина части рулона меньше, чем длина какого-то слова, которое должно быть написано в этой части, написать эту часть рапорта на части рулона такой ширины невозможно.

Гарантируется, что можно провести вертикальную черту так, что обе части рапорта возможно написать. Верс хочет провести вертикальную черту так, чтобы длина рулона, которой хватит, чтобы написать рапорт, была минимальна. Помогите ей найти эту минимальную длину.

## Формат ввода

В первой строке даны три целых числа w, n и m — ширина рулона, количество слов в первой и второй части рапорта (1≤w≤109; 1≤n,m≤100000).

В следующей строке дано n целых чисел ai — длина i-го слова первой части рапорта 1≤ai≤109.

В следующей строке дано m целых чисел bj — длина j-го слова второй части рапорта 1≤bj≤109.

Гарантируется, что возможно провести черту так, что обе части рапорта возможно написать.

## Формат вывода

В единственной строке выведите одно целое число — минимальную длину рулона, которой достаточно, чтобы написать рапорт.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 15 6 6  2 2 2 3 2 2  3 3 5 2 4 3 | 3 |

## Примечания

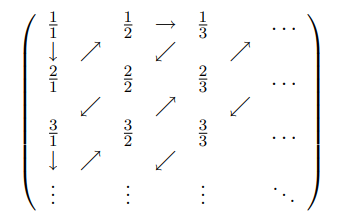
В тесте из примера рулон можно разделить на две части, проведя черту между 7 и 8 столбцом клеток, а затем записать по два слова в каждой строке в обеих частях рапорта.

# E. Нумерация дробей

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Георг Кантор доказал, что множество всех рациональных чисел счетно (т.е. все рациональные числа можно пронумеровать).

Чтобы упорядочить дроби необходимо их положить в таблицу, как показано на рисунке. В строку с номером *i* этой матрицы по порядку записаны дроби с числителем *i*, а в столбец с номером *j* дроби с знаменателем *j*.



Дальше необходимо выписать все дроби в том порядке, как показано на рисунке стрелками. Получится такая последовательность: https://contest.yandex.ru/testsys/tex/render/XGZyYWN7MX17MX0=.png, https://contest.yandex.ru/testsys/tex/render/XGZyYWN7Mn17MX0=.png, https://contest.yandex.ru/testsys/tex/render/XGZyYWN7MX17Mn0=.png, https://contest.yandex.ru/testsys/tex/render/XGZyYWN7MX17M30=.png, https://contest.yandex.ru/testsys/tex/render/XGZyYWN7Mn17Mn0=.png, https://contest.yandex.ru/testsys/tex/render/XGZyYWN7M317MX0=.png, *…*

Вам требуется по числу *n* найти числитель и знаменатель *n*-ой дроби.

## Формат ввода

Во входном файле дано число *n* (*1 ≤ n ≤ 1018*) — порядковый номер дроби в последовательности.

## Формат вывода

В выходной файл требуется вывести через символ / два числа: числитель и знаменатель соответствующей дроби.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 | 1/1 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 6 | 3/1 |

### Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2 | 2/1 |

# F. Велодорожки

|  |  |
| --- | --- |
| Ограничение времени | 4 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Мэр одного города очень любит следить за тенденциями и воспроизводить их в своём городе. До него дошла новость о популярности велодорожек. Теперь он хочет проложить велодорожки в своём городе и сделать это лучше, чем в других городах! Поэтому он решил сделать велодорожки даже на главной площади города.

Главная площадь представляет собой прямоугольник шириной w и высотой h, замощённый квадратными плитками со стороной 1. Мэр хочет, чтобы было проложено две велодорожки **одинаковой ширины**: одна горизонтальная и одна вертикальная. К сожалению, ремонт на площади проводился достаточно давно и на некоторых плитках уже появились трещины. Мэр хочет проложить велодорожки так, чтобы после этого на площади остались только целые плитки. При строительстве велодорожек плитки на их месте убираются. Можно только убирать плитки с площади и нельзя менять местами или добавлять новые. Чтобы потратить меньше денег, мэр хочет сделать велодорожки наименьшей возможной ширины, при этом ширина дорожек должна быть целым числом. Определите, какой должна быть ширина велодорожек.

## Формат ввода

В первой строке входных данных содержатся три целых числа w,h,n (1≤w,h≤109, 1≤n≤min(w×h,3⋅105))  — ширина и высота площади и количество потрескавшихся плиток соответственно.

В следующих n строках содержится по 2 целых числа xi,yi (1≤xi≤w, 1≤yi≤h)  — координаты потрескавшихся плиток. (xi,yi)≠(xj,yj) при i≠j.

## Формат вывода

Выведите единственное число c (1≤c≤min(w,h))  — наименьшую возможную ширину велодорожек.

### Пример 1

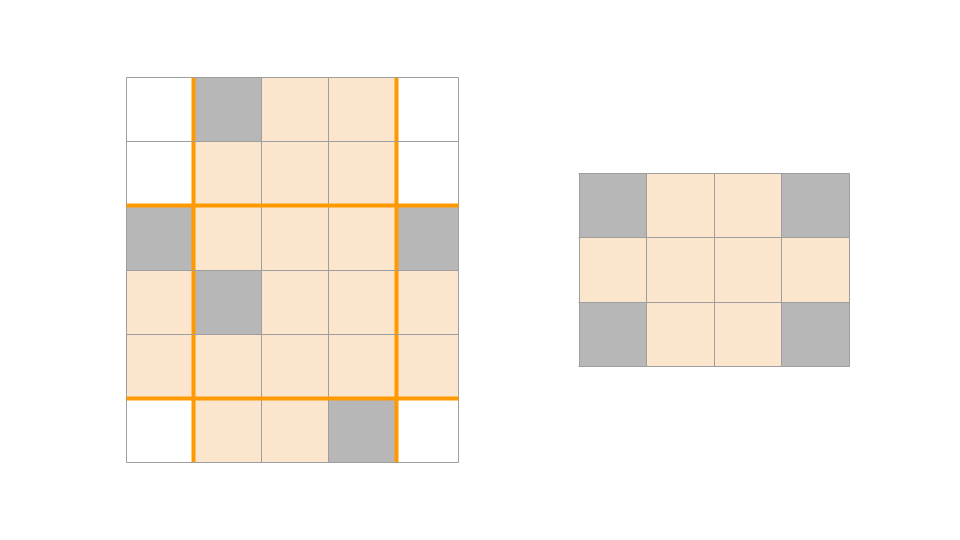
| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5 6 5  5 4  2 6  4 1  2 3  1 4 | 3 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 4 3 4  1 1  4 3  4 1  1 3 | 3 |

## Примечания

Ниже приведены картинки к примерам из условия. Серым отмечены потрескавшиеся плитки. Во втором примере ширина дорожек равна меньшей из сторон прямоугольника.



# G. Новый офис плюса

|  |  |
| --- | --- |
| Ограничение времени | 5 секунд |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Сервис Тындекс.Плюс так быстро растет, что для сотрудников и серверов потребовалось потребовалось построить новый офис.

Участок под застройку представляет из себя клетчатое поле n×m, часть клеток которого пригодна для строительства, а часть нет.

Новый офис должен выглядеть как знак "плюс"какого-то целого положительного размера k. Знак "плюс"размера k  — это такая клетчатая фигура, состоящая из пяти квадратов k×k клеток, при этом есть один центральный квадрат, а остальные четыре являются его соседями по стороне.

Новый офис должен быть как можно больше, поэтому необходимо найти максимальное k, такое что офис удастся разместить на участке под застройку.

Определите максимальное k. Гарантируется, что он можно построить офис хотя бы с k=1.

## Формат ввода

В первой строке задано два целых числа n и m (1≤n,m≤2000) — длина и ширина участка под застройку.

В каждой из последующих n строк задана строка, состоящая из m символов, j-й символ в i-й строке равен #, если клетка с координатами (i,j) пригодна для строительства и . иначе.

## Формат вывода

Выведите одно целое положительное число — максимально возможное k.

### Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 9 12  ...##.###...  ...##.###...  .########...  .###########  ...#########  ...#########  ......###...  ......###...  ......###... | 3 |

### Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 6 6  .##...  .##...  ######  ######  .##...  .##... | 1 |

## Примечания

В первом тесте из примера можно выбрать плюс с k=3. Этот плюс выглядит следующим образом:

...###...

...###...

...###...

#########

#########

#########

...###...

...###...

...###...

# H. Выборы

|  |  |
| --- | --- |
| Ограничение времени | 3 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

В одной демократической стране приближаются парламентские выборы. Выборы проходят по следующей схеме: каждый житель страны, достигший восемнадцатилетнего возраста, отдает свой голос за одну из политических партий. После этого партия, которая набрала максимальное количество голосов, считается победившей на выборах и формирует правительство. Если несколько партий набрали одинаковое максимальное количество голосов, то они должны сформировать коалиционное правительство, что обычно приводит к длительным переговорам.

Один бизнесмен решил выгодно вложить свои средства и собрался поддержать на выборах некоторые партии. В результате поддержки он планирует добиться победы одной из этих партий, которая затем сформирует правительство, которое будет действовать в его интересах. При этом возможность формирования коалиционного правительства его не устраивает, поэтому он планирует добиться строгой победы одной из партий.

Чтобы повлиять на исход выборов, бизнесмен собирается выделить деньги на агитационную работу среди жителей страны. Исследование рынка показало, что для того, чтобы один житель сменил свои политические воззрения, требуется потратить одну условную единицу. Кроме того, чтобы i-я партия в случае победы сформировала правительство, которое будет действовать в интересах бизнесмена, необходимо дать лидеру этой партии взятку в размере pi условных единиц. При этом некоторые партии оказались идеологически устойчивыми и не согласны на сотрудничество с бизнесменом ни за какие деньги.

По результатам последних опросов известно, сколько граждан планируют проголосовать за каждую партию перед началом агитационной компании. Помогите бизнесмену выбрать, какую партию следует подкупить, и какое количество граждан придется убедить сменить свои политические воззрения, чтобы выбранная партия победила, учитывая, что бизнесмен хочет потратить на всю операцию минимальное количество денег.

## Формат ввода

В первой строке вводится целое число n – количество партий (*1 ≤ n ≤ 105*). Следующие n строк описывают партии. Каждая из этих строк содержит по два целых числа: *vi* – количество жителей, которые собираются проголосовать за эту партию перед началом агитационной компании, и *pi* – взятка, которую необходимо дать лидеру партии для того, чтобы сформированное ей в случае победы правительство действовало в интересах бизнесмена (*1 ≤ vi ≤ 106*, *1 ≤ pi ≤ 106* или *pi* = -1). Если партия является идеологически устойчивой, то *pi* равно -1. Гарантируется, что хотя бы одно *pi* не равно -1.

## Формат вывода

В первой строке выведите минимальную сумму, которую придется потратить бизнесмену. Во второй строке выведите номер партии, лидеру которой следует дать взятку. В третьей строке выведите n целых чисел – количество голосов, которые будут отданы за каждую из партий после осуществления операции. Если оптимальных решений несколько, выведите любое.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  7 -1  2 8  1 2 | 6  3  3 2 5 |

# I. Лапта

|  |  |
| --- | --- |
| Ограничение времени | 3 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

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 ввода

В первой строке записаны два числа: *D* — максимальное расстояние удара и *N* — количество соперников на поле (*D* и *N* натуральные числа, *D* ≤ 1000, *N* ≤ 200). В следующих *N* строках записаны по три числа – начальные координаты *xi* и *yi* и максимальная скорость *vi* соответствующего игрока (скорости и координаты — целые числа, *–1000 ≤ xi ≤ 1000*, *0 ≤ yi ≤ 1000*, *0 < vi ≤ 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 ≥ 0*).

## Формат вывода

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

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 10 2  1 1 1  -1 1 1 | 9.05539  0.00000 10.00000 |

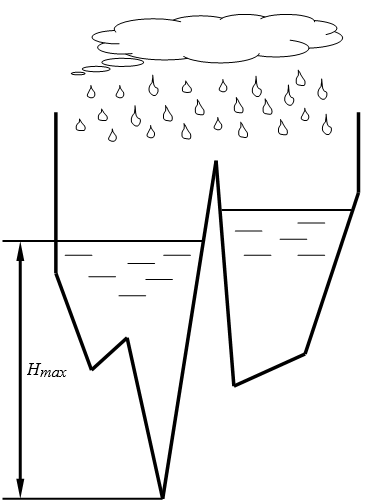
# J. Дождик

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

В НИИ метеорологии решили изучить процесс образования водоемов на различных рельефах местности во время дождя. Ввиду сложности реальной задачи была создана двумерная модель, в которой местность имеет только два измерения — высоту и длину. В этой модели рельеф местности можно представить как N-звенную ломаную c вершинами (*x0*, *y0*), ..., (*xN*, *yN*), где *x0* < *x1* < ... < *xN* и *yi ≠ yj*, для любых *i ≠ j*. Слева в точке *x0* и справа в точке *xN* рельеф ограничен вертикальными горами огромной высоты.

Если бы рельеф был горизонтальным, то после дождя вся местность покрылась бы слоем воды глубины *H*. Но поскольку рельеф — это ломаная, то вода стекает и скапливается в углублениях, образуя водоемы.

Требуется найти максимальную глубину в образовавшихся после дождя водоемах.



## Формат ввода

В первой строке расположены натуральное число *N* (*1 ≤ N ≤ 100*) и *H* — действительное число, заданное с тремя цифрами после десятичной точки (*0 ≤ H ≤ 109*). В последующих *N + 1* строках — по два целых числа *xi*, *yi* (*-10000 ≤ xi, yi ≤ 10000*).

Числа в строках разделены пробелами.

## Формат вывода

Выведите единственное число — искомую глубину с точностью *10-4*.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 7 7.000  -5 10  -3 4  -1 6  1 –4  4 17  5 3  9 5  12 15 | 15.8446 |