

## Задача 1. Многоступенчатая ракета

|                         |              |
|-------------------------|--------------|
| Имя входного файла:     | input.txt    |
| Имя выходного файла:    | output.txt   |
| Ограничение по времени: | 1 секунда    |
| Ограничение по памяти:  | 256 мегабайт |

При запуске многоступенчатой ракеты необходимо обеспечить разъединение ступеней и запуск двигателей в правильном порядке. Для этой цели предназначено специальное программируемое устройство, командный секвенсор.

Ступени ракеты могут соединяться двумя способами, последовательным и пакетным. Пакетом называются две или более ступени, соединенные параллельно. Пакет не может включать в себя последовательно соединенные ступени.

При последовательном соединении двигатель ступени может быть включен только после разъединения с предыдущей ступенью. Двигатели ступеней, входящих в один пакет, могут включаться в любом порядке, но только после того, как отсоединены предыдущие ступени перед пакетом. Попытки включить несуществующие или отработанные двигатели приводят к ошибке. Никакая ступень не должна быть отсоединена раньше, чем отработают ее двигатели, если они есть в ступени. Ступени ракеты пронумерованы, и никакая ступень не должна быть отсоединена раньше меньшей по номеру. К концу работы программы, все ступени ракеты, кроме последней (полезной нагрузки) должны быть отсоединены.

Вам необходимо написать программу, которая проверяет корректность программы командного секвенсора.

### Формат входного файла

В первой строке входного файла содержится число  $N$ , количество описаний программ ( $1 < N < 1000$ ). В следующих  $2N$  строках содержатся описания программ. Каждая программа описывается двумя строками.

В первой строке задается конфигурация ракеты. Ракета состоит из ступеней, пронумерованных числами от 1 до  $K$  ( $K \leq 9$ ). Каждая ступень описывается одним, двумя или тремя символами:

- цифрой, обозначающей номер ступени.
- возможно, символом '#', обозначающим, что эта ступень не имеет двигателей.
- знаком '-' (последовательно) или знаком '|' (параллельно), в зависимости от способа присоединения к следующей ступени. У последней ступени (полезной нагрузки) этот знак отсутствует.

Номера ступеней в первой строке всегда заданы в порядке возрастания, каждая ступень описана ровно один раз. Все рядом стоящие ступени, соединенные параллельно, образуют один пакет.

Во второй строке задается программа, состоящая из подряд идущих команд. Каждая команда описывается двумя или тремя символами:

- одной или двумя цифрами, обозначающими номер (номера) ступеней.
- знаком '+', который обозначает включение двигателей, или знаком '-', обозначающим отделение ступени.

### Формат выходного файла

Для каждой описанной во входном файле ракеты, необходимо вывести строку 'CORRECT', если программа командного секвенсора корректна, или 'INCORRECT', если обнаружена некорректность в последовательности команд. Если у последней ступени есть двигатель, но он не включается, это не является ошибкой.

## Пример

| input.txt   | output.txt |
|-------------|------------|
| 4           | CORRECT    |
| 1 2-3#      | CORRECT    |
| 12+1-2-     | CORRECT    |
| 1 2-3-4     | INCORRECT  |
| 12+1-2-3+3- |            |
| 1 2# 3      |            |
| 13+1-2-     |            |
| 1 2-3#      |            |
| 1+1-2-      |            |

## Задача 2. Голосование

|                         |                                  |
|-------------------------|----------------------------------|
| Имя входного файла:     | <code>input.txt</code>           |
| Имя выходного файла:    | <code>output.txt</code>          |
| Ограничение по времени: | 4 секунды<br>8 секунд (для Java) |
| Ограничение по памяти:  | 256 мегабайт                     |

В Берляндии выборы президента происходят по сложной многоуровневой иерархической системе. Оппозиция уверена, что такая система выборов была изобретена с целью запутать население и скрыть несправедливость голосования. Требуется создать программу, которая будет определять исход выборов в постоянно меняющемся суровом мире.

Проще всего представить систему голосования в виде подвешенного дерева. Листья дерева — это отдельные избиратели. Каждый внутренний узел дерева — некоторая официальная социальная группа (семья, дом, район, округ и т.п.). Множество сыновей узла соответствует сущностям, которые непосредственно составляют эту группу — это отдельные избиратели и/или другие группы. Корень дерева — группа, включающая всю Берляндию.

Каждый из избирателей может либо отдать свой голос за одного конкретного кандидата, либо воздержаться, например, не прийдя на выборы. После этого в каждой группе определяются локальные результаты выборов. Для этого рассматриваются все голоса составляющих эту группу сущностей. В локальных выборах побеждает кандидат, набравший строго больше голосов, чем все остальные. Если такого кандидата нет, то локальные выборы признаются несостоявшимися. Если локальные выборы не состоялись, то группа считается “воздержавшейся”, и её мнение никак не учитывается. Далее результаты голосования в группе учитываются в локальном голосовании на уровень выше как один голос от всей группы.

Президентом Берляндии становится тот, кто победил в локальных выборах в самой верхней группе, то есть в корне дерева. Если выборы в этой группе не состоялись, то проводится повторное голосование.

В начальный момент времени для каждого отдельного избирателя известно, за кого он планирует голосовать. Затем отдельные избиратели по очереди изменяют своё мнение. Требуется после каждого изменения определить, какими бы были результаты выборов, если бы они состоялись в этот момент.

### Формат входного файла

В первой строке входного файла даны целые числа  $N$  — количество отдельных избирателей,  $M$  — количество социальных групп и  $K$  — количество кандидатов ( $2 \leq N \leq 100\,000$ ,  $1 \leq M \leq 100\,000$ ,  $2 \leq K \leq 10$ ). Избиратели, группы и кандидаты пронумерованы по отдельности, начиная с единицы. Во второй строке записано  $N$  целых чисел — исходные мнения избирателей. Мнение либо равно номеру кандидата, за которого избиратель планирует голосовать, либо равно нулю, если он воздерживается.

В следующих  $M$  строках описаны социальные группы. В  $t$ -ой из них описаны сущности, непосредственно составляющие  $t$ -ую социальную группу. Сначала записано целое положительное число  $S_t$  — количество этих сущностей, а затем через пробел  $S_t$  целых чисел, определяющих эти сущности ( $2 \leq S_t \leq N$ ). Если сущность является отдельным избирателем, то записан его номер, а если группой — то её номер со знаком минус. Корневая группа, соответствующая всей Берляндии, всегда имеет номер 1.

В следующей строке дано целое число  $Q$  — количество изменений в мнениях избирателей ( $1 \leq Q \leq 100\,000$ ). Каждая из следующих  $Q$  строк содержит два целых числа: номер избирателя и его новое мнение.

## Формат выходного файла

В выходной файл требуется вывести  $Q + 1$  число, по одному в строке. Первое из них — результаты выборов согласно исходным мнениям избирателей. Каждое из следующих  $Q$  чисел должно содержать результаты выборов после соответствующего изменения.

Результаты следует выдавать в том же формате, в котором задаётся мнение избирателей. Если есть победитель, нужно вывести его номер, а если нет — то ноль.

## Пример

| input.txt       | output.txt |
|-----------------|------------|
| 8 4 3           | 1          |
| 2 0 2 1 1 3 2 1 | 2          |
| 3 -2 -3 8       | 0          |
| 2 1 3           | 1          |
| 2 -4 2          | 0          |
| 4 4 7 5 6       | 1          |
| 7               | 0          |
| 8 2             | 3          |
| 8 0             |            |
| 3 1             |            |
| 7 3             |            |
| 1 1             |            |
| 8 3             |            |
| 5 3             |            |

## Задача 3. Фото для газеты

Имя входного файла: `input.txt`  
Имя выходного файла: `output.txt`  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 256 мегабайт

Фотограф получил задание от своей редакции. Ему нужно сфотографировать участников Всемирного форума программистов. Для общей фотографии участники выстроились в линию. Каждый надел футболку своей компании. Отметим, что цвета футболок различных компаний отличаются.

Для того, чтобы не обидеть ни одну компанию, нужно под фотографией перечислить названия всех компаний, представители которых изображены на снимке. Но место, которое отводится для подписи под фотографией, может содержать не более чем  $m$  названий компаний. Поэтому фотографу нужно сделать фотографию так, чтобы на ней были изображены представители не более чем  $m$  различных компаний. Фотограф стоит перед линией собравшихся участников форума. Он может фотографировать любое количество людей, стоящих рядом.

Необходимо узнать, сколько всего различных снимков, удовлетворяющих указанным ограничениям, может сделать фотограф. Две фотографии считаются разными, если на одной из них запечатлен хотя бы один человек, которого нет на другой фотографии.

### Формат входного файла

В первой строке входного файла содержится непустая строка, состоящая из букв латинского алфавита. Она задает расстановку людей при фотографировании. Каждая буква алфавита, независимо от регистра, обозначает цвет футболки. Длина строки не превосходит 50 000.

Во второй строке записано целое число  $m$  — ограничение на количество представителей различных компаний на одной фотографии ( $1 \leq m \leq 26$ ).

### Формат выходного файла

В выходной файл необходимо вывести целое число — общее количество возможных фотографий.

### Пример

| <code>input.txt</code> | <code>output.txt</code> |
|------------------------|-------------------------|
| abAc<br>2              | 8                       |

## Задача 4. В бочке

Имя входного файла: `input.txt`  
Имя выходного файла: `output.txt`  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 256 мегабайт

Александр Ивановичу Корейко приснился кошмарный сон — он заточен в бочку со своими миллионами и изо всех сил старается оттуда побыстрее выбраться.

Через Зосю все это дошло до знатного составителя задач и кроссвордов старика Синицкого, и тот тут же сочинил из всего этого задачу: найти математическое ожидание длины кратчайшего пути до поверхности для случайной точки, равномерно распределенной внутри цилиндра высоты  $H$  и радиуса  $R$ . В ответ на недоуменный взгляд Зоси он объяснил эту задачу так: найти среднее расстояние от точки внутри цилиндра до ближайшей точки на поверхности; точка внутри цилиндра может с равной вероятностью оказаться в любом его месте.

Сочинить-то задачу он сочинил, а вот решить сам никак не может. Так что все надежда на вас.

### Формат входного файла

В первой строке входного файла задано два целых числа  $R$  и  $H$  — радиус и высота цилиндра ( $1 \leq R, H \leq 10^9$ ).

### Формат выходного файла

В выходной файл необходимо вывести одно вещественное число — ответ на задачу с погрешностью (абсолютной или относительной) не более  $10^{-9}$ .

### Пример

| <code>input.txt</code> | <code>output.txt</code> |
|------------------------|-------------------------|
| 1 2                    | 0.250000000000          |

## Задача 5. Электронная книга

|                         |                                  |
|-------------------------|----------------------------------|
| Имя входного файла:     | <code>input.txt</code>           |
| Имя выходного файла:    | <code>output.txt</code>          |
| Ограничение по времени: | 2 секунды<br>4 секунд (для Java) |
| Ограничение по памяти:  | 256 мегабайт                     |

Высокие технологии проникают во все области жизни, вытесняя тёплые аналоговые эквиваленты. Электронные письма, электронные мыши, даже книги — и те уже электронные. Вот о них речь и пойдёт.

Известен факт, что электронные книги, дисплей которых основан на технологии e-ink, практически не тратят энергию на собственно отображение страниц. Единожды открытая страница может отображаться в течение года, не требуя заряда аккумулятора. Другое дело — перелистывание. В этом случае и тратится основная часть энергии. Она требуется для того, чтобы переключить часть пикселей на экране из зажжённого состояния в погашенное и наоборот. Поэтому-то каждый производитель электронных книг ищет методы, как бы сделать это эффективнее.

Компания “Новые Газеты и Учебники” придумала следующую идею для своей новой читалки “Knigle”. Немногие читатели смогут заметить, что текст на новой странице смещён на несколько пикселей по сравнению с предыдущей. То, что изменится размер полей — не так страшно. Конечно, некоторые особо дотошные индивидуумы могут возмутиться и этим, но значительно увеличенная продолжительность жизни устройства от одного заряда сможет “уговорить” кого угодно. Поэтому для каждой следующей страницы выбирается сдвиг относительно предыдущей с учётом следующих условий:

- размер каждой страницы равен размеру экрана;
- первая страница книги всегда выводится без сдвига;
- сдвиг относительно экрана по каждому из направлений не должен превышать по модулю заданной константы  $d$ , иначе сильно “убежавший” текст будет легко замечен;
- не разрешается построчный сдвиг, то есть вся страница должна сдвигаться целиком;
- все зажжённые пиксели новой страницы должны помещаться на экране, т.е. входить в прямоугольник, ограниченный размерами первой страницы;
- все пиксели экрана, находящиеся вне текущей страницы, должны быть погашены;
- включение пикселей для первого экрана не учитывается.

Соблюдая эти условия, можно при прочтении книги сохранить порядочно энергии. Но стоит понять, оправдано ли это. Для этого нужно взять хотя бы одну книгу и посчитать, какое минимальное количество переключений пикселей произойдёт при полном последовательном её просмотре. Такая задача и стоит перед вами.

### Формат входного файла

В первой строке входного файла содержатся целые числа  $w, h, n, d$ , где  $w$  и  $h$  — ширина и высота экрана в пикселях,  $n$  — общее количество страниц,  $d$  — максимально разрешённый сдвиг в каждом из направлений ( $1 \leq w, h \leq 50, 1 \leq n \leq 50, 0 \leq d \leq 10, d < w, h$ ).

Далее следуют  $n$  блоков по  $h + 1$  строк длины  $w$ , где первая строка содержит номер страницы, а следующие  $h$  строк — строки страницы, каждая из которых состоит из символов 0 и 1, где 0 соответствует погашенному пикселю, а 1 — зажжённому. Нумерация страниц начинается с единицы. Гарантируется, что страницы во входном файле идут последовательно.

### Формат выходного файла

В выходной файл нужно вывести одно целое число — минимально возможное количество

переключений пикселей, необходимое для просмотра этих страниц.

## Пример

| input.txt                                                                                                              | output.txt |
|------------------------------------------------------------------------------------------------------------------------|------------|
| 4 4 3 2<br>1<br>0101<br>0000<br>0101<br>0000<br>2<br>0000<br>1010<br>0000<br>1000<br>3<br>0001<br>0000<br>0100<br>0000 | 2          |

## Задача 6. Фигурки

|                         |                         |
|-------------------------|-------------------------|
| Имя входного файла:     | <code>input.txt</code>  |
| Имя выходного файла:    | <code>output.txt</code> |
| Ограничение по времени: | 5 секунд                |
| Ограничение по памяти:  | 768 мегабайт            |

Вася недавно узнал про технологию конструктивной блочной геометрии. Данная технология используется для моделирования твёрдых тел. Суть этого подхода состоит в следующем. Сначала создаются примитивы, такие как куб, тетраэдр, сфера, цилиндр, конус и т.д. Далее из них получаются новые тела посредством применения булевых, точнее, теоретико-множественных операций: объединения, пересечения, вычитания и пр. Эти операции выполняются над телами как над множествами точек.

Вася реализовал данную технологию программно. Поскольку стереометрию он ещё не проходил, то он строит фигуры на плоскости вместо тел в пространстве. В качестве примитивов было решено ограничиться следующими фигурами:

- треугольник,
- прямоугольник, стороны которого параллельны осям координат,
- эллипс, оси которого параллельны осям координат.

В пределах одного сеанса работы Вася создаёт  $m$  фигур. Они нумеруются, начиная с единицы, в порядке их создания. Из них первые  $n$  — это примитивы, а остальные  $(m - n)$  — производные фигуры. Каждая производная фигура получена при помощи применения булевой операции к двум уже созданным ранее фигурам. Разрешается использование следующих булевых операций:

- объединение (`uni`),
- пересечение (`int`),
- вычитание (`sub`).

Для проверки правильности работы программы Вася решил распечатывать получающиеся фигуры на принтере. На печать выдаётся только содержимое квадрата со стороной, равной 1, один из углов которого находится в начале координат, а противоположный — в точке с координатами  $(1, 1)$ .

Васю охватила вселенская скудость: очень не хочется тратить чернила. Поэтому он просит вас посчитать площадь каждой фигуры внутри распечатываемой области.

**Внимание:** для решения данной задачи важно прочитать комментарий.

### Формат входного файла

В первой строке входного файла записано два целых числа:  $n$  — количество созданных примитивов и  $m$  — общее количество созданных фигур ( $2 \leq n \leq 50$ ,  $n \leq m \leq 5000$ ). Каждая  $i$ -ая из следующих  $m$  строк содержит описание способа создания  $i$ -ой фигуры.

Первые  $n$  фигур являются примитивами и описываются в следующем формате:

- `R {x1} {y1} {x2} {y2}`  
В этом случае создаётся прямоугольник, в котором  $(x_1, y_1)$  и  $(x_2, y_2)$  — координаты двух его противоположных углов ( $0 \leq x_1, y_1, x_2, y_2 \leq 1$ ).
- `T {x1} {y1} {x2} {y2} {x3} {y3}`  
Создаётся треугольник по координатам трёх его вершин  $(x_1, y_1)$ ,  $(x_2, y_2)$  и  $(x_3, y_3)$ , где  $0 \leq x_1, y_1, x_2, y_2, x_3, y_3 \leq 1$ .
- `E {xc} {yc} {rx} {ry}`  
Создаётся эллипс, в котором  $(x_c, y_c)$  — координаты центра эллипса,  $r_x$  и  $r_y$  — длины полуосей эллипса, направленных вдоль осей  $X$  и  $Y$  соответственно ( $0 \leq x_c, y_c \leq 1$ ,

$$0.01 \leq r_x, r_y \leq 0.5).$$

Все вещественные числа указаны с не более чем 10-ю знаками после запятой. Прimitives могут быть вырожденными.

Последние  $(m - n)$  фигур являются производными.  $k$ -ая фигура описывается трёхбуквенным кодом булевой операции и номерами  $a$  и  $b$  фигур-аргументов ( $1 \leq a, b \leq k - 1$ ). Такое описание означает, что  $k$ -ая фигура получена при помощи применения булевой операции с заданным кодом к  $a$ -ой фигуре (первый аргумент) и  $b$ -ой фигуре (второй аргумент).

Гарантируется, что каждая производная фигура получена при помощи малого количества примитивов. Строго говоря: если производную фигуру выразить как развёрнутую булеву формулу от примитивов, то в этой формуле будет не более 4 булевых операций. Например, последняя фигура в первом тесте из условия выражается как "(P3 int (P1 uni P2)) sub P4", и в этой формуле 3 булевых операций.

## Формат выходного файла

В выходной файл необходимо вывести ровно  $M$  вещественных чисел, по одному на строке.  $k$ -ое число должно совпадать с площадью распечатываемой части  $k$ -ой фигуры с абсолютной погрешностью не более  $10^{-3}$ .

## Примеры

| input.txt                                                                                                            | output.txt                                                                             |
|----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|
| 4 7<br>E 0.5 1 0.5 0.5<br>T 0 0.5 1 0.5 0.5 1<br>R 0.5 1 1 0.25<br>E 0.5 1 0.5 0.25<br>uni 1 2<br>int 3 5<br>sub 6 4 | 0.3926990816<br>0.25<br>0.375<br>0.1963495408<br>0.4463495408<br>0.2231747704<br>0.125 |
| 2 6<br>R 0 0 1.0 0.5<br>R 0 0 0.9999999999 0.5000000001<br>uni 1 2<br>int 1 2<br>sub 3 4<br>sub 3 3                  | 0.5<br>0.5<br>0.5<br>0.5<br>0<br>0                                                     |

## Комментарий

В данной задаче тесты разделены на две категории: “ручные” (первые 20) и “случайные” (оставшиеся 10).

“Ручные” тесты имеют маленький размер. Дополнительно к ограничениям, указанным в формате входного файла, для “ручных” тестов верно  $n \leq 5$ ,  $m \leq 10$ .

“Случайные” тесты сгенерированы при помощи генератора псевдослучайных чисел следующим образом. Значения  $N$  и  $M$  устанавливаются максимально допустимыми. Каждое вещественное число во входных данных смоделировано независимой случайной величиной с равномерным распределением в пределах диапазона значений, указанного в формате входных данных. Все типы примитивов и все типы булевых операций равновероятны.

Аргументы каждой булевой операции  $a$  и  $b$  выбираются среди всех вариантов, при которых соблюдается ограничение на размер развёрнутой булевой формулы. Все такие пары  $(a, b)$  равновероятны.

## Задача 7. Скамейка

|                         |                         |
|-------------------------|-------------------------|
| Имя входного файла:     | <code>input.txt</code>  |
| Имя выходного файла:    | <code>output.txt</code> |
| Ограничение по времени: | 2 секунды               |
| Ограничение по памяти:  | 256 мегабайт            |

Люди любят уединение. Поэтому когда новый человек хочет сесть на скамейку, он старается выбрать себе место подальше от остальных людей, сидящих на ней.

Представим скамейку в виде отрезка  $[0, L]$  на координатной прямой. Будем считать, что каждый сидящий на скамейке человек занимает некоторый интервал длины  $W$ . В любой момент времени все интервалы, занимаемые людьми, не пересекаются и не выходят за пределы скамейки.

Допустим, на скамейке уже сидит некоторое количество человек, и хочет сесть ещё один. Если на скамейке нет подходящего свободного места, то он уходит. В противном случае он садится таким образом, чтобы расстояние от него до других людей и концов скамейки было наибольшим.

Формально говоря, место, на которое садится новый человек, определяется следующим образом. Пусть  $Y$  — множество, равное объединению всех интервалов, на которых уже сидят люди, и концевых точек скамейки  $\{0, L\}$ . Интервал, который занимает вновь пришедший, таков, что расстояние от его центра до множества  $Y$  максимально. Если на скамейке есть несколько таких точек, то в качестве центра из них выбирается точка с минимальной координатой.

Изначально нужно разместить на скамейке ровно  $N$  человек. Далее к скамейке по одному будут подходить люди и садиться на неё по вышеописанным правилам. Это будет продолжаться до тех пор, пока на скамейке не закончится свободное место.

В задаче требуется найти два варианта изначальной рассадки. В первом варианте окончательное количество сидящих на скамейке должно быть минимально возможным, в во втором — максимально возможным.

### Формат входного файла

В первой строке входного файла задано два целых числа  $N$  и  $W$  ( $1 \leq N \leq 100\,000$ ,  $1 \leq W \leq 10^9$ ). Во второй строке записано целое число  $L$  ( $1 \leq L \leq 10^{17}$ ).

Гарантируется, что можно разместить  $N$  человек на скамейке.

### Формат выходного файла

В первую строку выходного файла нужно вывести два целых числа — минимально возможное количество уместившихся на скамейке и максимально возможное количество. Далее необходимо вывести два варианта рассадки — рассадку, при которой достигается минимальное количество, а затем рассадку, при которой количество людей максимально.

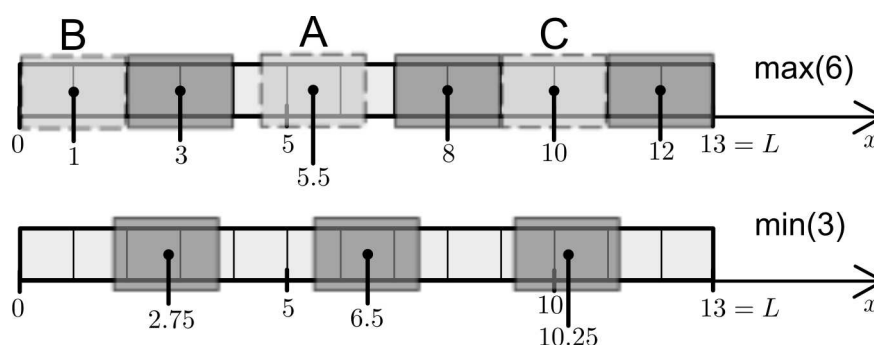
Каждая рассадка должна занимать  $N$  строк, по одному числу в каждой. Каждое число — это координата центра интервала, занимаемого человеком. Все числа в одной рассадке должны идти в порядке возрастания. Разрешается выводить вещественные числа, но не более чем с 9-ю десятичными знаками после запятой. Гарантируется, что существуют оптимальные рассадки, которые можно представить таким образом.

## Пример

| input.txt | output.txt                                       |
|-----------|--------------------------------------------------|
| 3 2<br>13 | 3 6<br>2.75<br>6.5<br>10.25<br>3<br>8.0<br>12.00 |

## Комментарий

На картинке изображены обе рассадки из примера.



В минимальной рассадке на скамейке есть четыре свободных отрезка, однако все они имеют длину 1.75, поэтому сесть больше никто не может.

В максимальной рассадке сначала сидят три человека. Потом подходят ещё трое, их места обозначены буквами **A**(5.5), **B**(1), **C**(10) в порядке их появления.

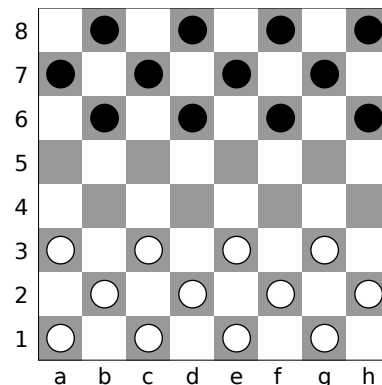
## Задача 8. Ничья?

Имя входного файла: `input.txt`  
Имя выходного файла: `output.txt`  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 256 мегабайт

По правилам русских шашек, участник может потребовать ничью, если какая-либо игровая позиция повторилась не менее трех раз. Хипстеры играют в русские шашки на планшете, поэтому возможны отмены хода. Дано описание партии, требуется по запросу определять, может ли игрок потребовать ничью.

Основные правила игры:

Игра ведется на стандартной шахматной доске ( $8 \times 8$ ), повернутой так, чтобы перед игроком, играющим белыми, в левом нижнем углу клетка была черного цвета. В игре используются только 32 поля черного цвета. В начале игры у каждого игрока по 12 шашек белого и черного цвета соответственно. Начальная игровая позиция представлена на рисунке. Шашки бывают двух типов: простые и дамки.



Простая шашка может ходить двумя способами.

1. Тихий ход – передвижение простой шашки вперед по диагонали на любое свободное соседнее поле. Назад тихий ход запрещается. Простая шашка, достигнувшая последней горизонтали, становится дамкой.
2. Бой – простая шашка может перепрыгнуть через простую шашку или дамку соперника, находящуюся на соседнем поле по диагонали, при условии, что следующее по той же диагонали поле свободно. В случае, если после боя бьющая шашка имеет возможность побить еще, то бой необходимо продолжать. Бой может осуществляться в любых направлениях. Простая шашка, в процессе боя достигнувшая последней горизонтали, становится дамкой и при возможности продолжает бой как дамка, по правилам боя для дамек.

Ход дамкой также может быть двух типов.

1. Тихий ход – дамка может двигаться в любом направлении по свободной диагонали, на любое количество полей в одну сторону за один ход.
2. Бой – дамка может перепрыгивать через любую простую шашку или дамку соперника, находящуюся в любом направлении от бьющей дамки по диагонали, если за ней (простой шашкой или дамкой соперника) по диагонали есть свободное поле. Если за бьющейся шашкой в направлении боя есть несколько свободных полей подряд, то бьющая дамка может занять любое из них. Если с одного или нескольких этих полей можно продолжить бой, либо в том же направлении, либо по любой из пересекающихся диагоналей, то дамка обязана продолжать взятие и этих шашек, сколько бы их ни находилось на ее пути. Если вариантов боя несколько, то необходимо бить по одному любому из направлений и так далее.

В том случае, когда при ходе есть возможность либо совершить бой, либо тихий ход, необходимо выбрать бой.

При взятии нескольких шашек противника снимать их с доски можно только после завершения всего хода. Запрещается при взятии перескакивать более одного раза бьющей шашкой или дамкой через одну и ту же шашку соперника, а на свободное поле наступать несколько раз разрешается.

## Формат входного файла

В первой строке входного файла находится единственное целое число  $N$  – количество запросов ( $1 \leq N \leq 5 \cdot 10^5$ ). В следующих  $N$  строках записаны запросы, по одному в строке.

Запросы могут быть трех видов:

1. “Move” – сделать ход, далее через пробел записывается ход.
2. “Undo” – отменить последний ход. Два последовательных запроса “Undo” отменяют последний и предпоследний ходы и так далее. Количество таких запросов ограничено количеством сделанных ходов с учетом других вызовов “Undo”.
3. “Draw” – проверить, было ли не менее трех повторений какой-либо игровой позиции.

В русских шашках горизонтальные ряды обозначаются цифрами от 1 до 8, вертикальные ряды – латинскими буквами от a до h. Тихий ход записывается в виде двух обозначений полей, разделённых знаком “-”, например, “f4-g5” описывает ход с поля f4 на поле g5. Ход со взятием записывается в виде обозначений полей, по которым проходит бьющая шашка, разделённых двоеточием. Например, “f4:h6” означает ход с поля f4 на h6 со взятием шашки, стоящей на поле g5, а “f4:h6:f8:c5:g1” – ход со взятием четырёх шашек. Поле a1 – левый угол доски соперника, играющего белыми шашками.

Гарантируется, что во входном файле записана корректная с точки зрения правил игра.

## Формат выходного файла

Для каждого запроса “Draw” на отдельной строке выходного файла необходимо вывести слово “Yes”, если имело место три и более повторений позиции, и “No” в противном случае.

## Примеры

| input.txt                                                                                                                 | output.txt |
|---------------------------------------------------------------------------------------------------------------------------|------------|
| 4<br>Move c3-b4<br>Draw<br>Undo<br>Draw                                                                                   | No<br>No   |
| 10<br>Move c3-b4<br>Move b6-a5<br>Move b2-c3<br>Move c7-b6<br>Move c3-d4<br>Move a5:c3:e5<br>Draw<br>Undo<br>Undo<br>Draw | No<br>No   |

## Задача 9. Мадам Грицацуева возвращается

|                         |              |
|-------------------------|--------------|
| Имя входного файла:     | input.txt    |
| Имя выходного файла:    | output.txt   |
| Ограничение по времени: | 1 секунда    |
| Ограничение по памяти:  | 256 мегабайт |

Мадам Грицацуева стремительным домкратом несется по узкому редакционному коридору, в конце которого стоит с широко разинутыми глазами обезумевший от внезапно свалившегося на него счастья Суслик. Остановить ее не может ничто, даже двери, которые, когда их открывают, наглухо перегораживают коридор. Встретившись с открытой дверью, влюбленная вдовушка проламывает ее и по-прежнему несется к своему счастью. Правда, скорость ее при этом несколько падает.

Дверь с номером  $i$  находится на расстоянии  $l_i$  от начала коридора и изначально закрыта. В момент времени  $t_i$  она открывается, и остается после этого открытой до окончания веков. Если в момент, когда мадам пробегает мимо этой двери, она закрыта, то ничего не происходит, и мадам как ни в чем не бывало бежит дальше. В противном же случае мадам ломает дверь, в результате чего ее скорость падает на величину  $v_i$ . А если скорость не превосходила  $v_i$ , то пылкая женщина, мечта поэта, застревает в двери.

Временной континуум в редакционном коридоре устроен так, что ломать двери можно только в целочисленные моменты времени. Так что если мадам подбегает к открытой двери в нецелый момент времени, то она вынуждена подождать до следующего целочисленного момента. Например, если мадам подбежит к открытой двери в моменты времени 26.2, или 26.6 или 27 секунд, то во всех этих случаях она пересечет дверь в момент  $t = 27$ .

Единственное, чего хочет вдовушка — крепко объять Великого Комбинатора, но при этом она никак не хочет его насмерть задушить. Для этого она заранее прикинула оптимальную скорость, с которой она должна достичь конца коридора. Увы, при этом она совершенно забыла рассчитать свою стартовую скорость, с которой она начинает свой забег от начала коридора. Мадам уверена — для успеха ее операции стартовая скорость должна быть целочисленной.

Помогите ей, пожалуйста, сосчитайте за нее ее стартовую скорость, такую, что при достижении конца коридора ее скорость как можно меньше будет отличаться от заданной оптимальной. Если таких решений несколько, то нужно найти минимальное из них.

### Формат входного файла

В первой строке входного файла записано три целых числа  $D$ ,  $n$  и  $v_{opt}$ , где  $D$  — длина коридора,  $n$  — количество дверей и  $v_{opt}$  — скорость, с которой мадам Грицацуева должна достичь конца коридора ( $1 \leq D \leq 10^{15}$ ,  $0 \leq n \leq 10^4$ ,  $1 \leq v_{opt} \leq 10^6$ ).

В следующих  $n$  строках записано по три целых числа  $l_i$ ,  $t_i$ ,  $v_i$ , где  $l_i$  — расстояние от начала коридора до  $i$ -ой двери,  $t_i$  — время, когда открывается эта дверь и  $v_i$  — на сколько падает скорость мадам Грицацовой при проламывании двери ( $1 \leq l_i < D$ ,  $0 \leq t_i \leq 10^9$ ,  $0 \leq v_i \leq 10^6$ ). Все двери расположены на разном расстоянии от начала коридора.

### Формат выходного файла

В выходной файл необходимо вывести одно целое число — минимальную возможную целочисленную стартовую скорость вдовы, при которой ее финишная скорость менее всего отличается от  $v_{opt}$ .

## Пример

| input.txt               | output.txt |
|-------------------------|------------|
| 4 2 3<br>1 1 1<br>3 1 2 | 4          |

## Комментарий

В примере из входного файла при начальной скорости 3 вдовушка прибегает ко второй двери в момент ее открытия — ее скорость падает на 2 и становится 1. При начальной скорости 4 мадам Грицацуева успевает пробежать все двери до открытия.

## Задача 10. Реформа на железной дороге

Имя входного файла: `input.txt`  
Имя выходного файла: `output.txt`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

В некоторой стране была объявлена реформа на железной дороге. В целях обеспечения всеобщего порядка и учета были введены новые правила формирования поездов. Номер каждого вагона в поезде, сформированном по новым правилам, должен быть ровно на единицу больше, чем номер предшествующего ему вагона. Номер первого вагона в поезде может быть любым.

Для выполнения этих требований на каждой станции организовали пункты переформирования поездов. В такой пункт загоняли состав с произвольной нумерацией вагонов. Затем какое-то количество вагонов из поезда можно было удалить, а какое-то количество вагонов с подходящими номерами вставить в любое место состава.

Эта реформа, как и любая другая, требует много денег. Для того, чтобы удалить один вагон из состава, требуется  $u$  рублей. Вставка вагона в состав на нужное место оценивается  $v$  рублями. В целях хоть какой-то экономии требуется так организовать переформирование состава в соответствии с новыми требованиями, чтобы суммарная стоимость вставки и удаления вагонов была бы минимальной.

### Формат входного файла

В первой строке входного файла заданы три целых числа  $n$ ,  $u$  и  $v$ , где  $n$  — количество вагонов в старом составе,  $u$  — стоимость удаления одного вагона, а  $v$  — стоимость вставки одного вагона в состав ( $1 \leq n \leq 100\,000$ ,  $1 \leq u, v \leq 100$ ).

Во второй строке через пробел перечислено  $n$  натуральных чисел, задающих исходную нумерацию вагонов. Номер вагона не может быть больше, чем 100 000.

### Формат выходного файла

В выходной файл необходимо вывести одно целое число — минимальную суммарную стоимость операций вставки и удаления вагонов, приводящих поезд к требуемому виду.

### Пример

| <code>input.txt</code> | <code>output.txt</code> |
|------------------------|-------------------------|
| 4 1 2<br>1 2 4 4       | 2                       |

### Комментарий

Оптимальное решение в тесте — удалить обе четверки (останется 1, 2).

Можно удалить одну четверку и добавить тройку (и получить 1, 2, 3, 4), но тогда суммарная стоимость будет равна 3.