Задача A. Перемешанные сообщения

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Сегодня Никите пришло несколько сообщений. Одно из сообщений — это кодовое слово «`spbsu`». До и после этого сообщения могло быть любое количество других. Все другие сообщения — произвольные строки из маленьких букв английского алфавита.

Сообщения передавались одно за другим по секретному каналу. Так что строка, передаваемая по каналу, изначально была просто конкатенацией всех сообщений.

Однако, поскольку канал секретный, передаче мог мешать шум: различные сообщения могли перемешаться. Формально шум проявляется как последовательность *обменов*. При каждом обмене канал выбирает и меняет местами две соседние буквы в строке, **которые изначально были в разных сообщениях**. Буквы каждого сообщения при этом сохраняют свой относительный порядок.

После всех обменов итоговая строка оказалась у Никиты. По итоговой строке выясните минимально возможное количество обменов, произошедших в канале.

Формат входных данных

В первой строке задано целое число n — количество принятых символов ($5 \leqslant n \leqslant 10^5$).

В следующей строке задана строка s, состоящая из n маленьких букв английского алфавита — итоговая строка, оказавшаяся у Никиты. Гарантируется, что эта строка получена так, как описано выше.

Формат выходных данных

Выведите одно целое число — ответ на задачу.

Примеры

стандартный ввод	стандартный вывод
6 spbssu	1
15 spongebaseurban	11

Задача B. Я календарь переверну...

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Как-то, переворачивая календарь, Николай задумался: сколько рядов в календаре в конкретном году?

Календарь состоит из 12 листов, каждый лист соответствует месяцу — с января по декабрь. В каждом месяце перечислены все дни этого месяца. Дни на каждом листе собраны в ряды по неделям: дни одной недели в одном ряду, дни разных недель — в разных. Неделя в этом календаре начинается с понедельника.

Например, если в месяце 31 день, и первый день месяца — воскресенье (как в январе 2023 года), то на листе календаря для этого месяца получится шесть рядов:

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

Помните, что в високосном году в феврале 29 дней, а в невисокосном — 28. Год считается високосным, если его номер кратен 400, либо кратен 4 и не кратен 100. Например, 2000, 2004 и 2040 — високосные годы, а 1900, 1982 и 2039 — невисокосные.

Формат входных данных

В первой строке задано целое число y — номер года ($1970 \leqslant y \leqslant 2037$).

Формат выходных данных

Выведите искомое число: количество рядов в календаре в заданном году.

Пример

стандартный ввод	стандартный вывод
2023	63

Задача C. Много-много циклов

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Дан неориентированный граф G. Найдите такое наибольшее число d, что длина всех простых циклов делится на d. Если такого числа не существует, выведите 0.

Формат входных данных

В первой строке записаны два целых числа n и m: количество вершин и рёбер в графе соответственно ($1 \leqslant n \leqslant 5000$, $0 \leqslant m \leqslant 10\,000$). В каждой из следующих m строк даны три числа a, b и c, которые описывают неориентированное ребро между вершинами a и b длины c ($1 \leqslant a, b \leqslant n$, $1 \leqslant c \leqslant 10^9$). Гарантируется, что граф не содержит петель и кратных рёбер.

Формат выходных данных

Выведите одно целое число — ответ на задачу.

Примеры

стандартный ввод	стандартный вывод
4 4 1 2 1 2 3 1 3 4 1 4 1 1	4
4 5 1 2 1 1 3 2 1 4 1 2 3 1 3 4 1	4

Задача D. Слоны

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Шахматный слон бьёт все клетки доски, которые находятся с ним на одной диагонали.

Расставьте наибольшее возможное количество слонов на доске $n \times m$ таким образом, что никакая пара слонов не бьёт друг друга.

Формат входных данных

В первой строке записаны два целых числа n и m — размеры доски ($1 \leqslant n, m \leqslant 10^5 + 1$).

Формат выходных данных

В первой строке выведите число k: наибольшее возможное количество слонов на доске $n \times m$, которых можно расположить так, что они не бьют друг друга. В каждой из следующих k строк выведите по два целых числа — координаты слонов. Первая координата должна быть в промежутке $[1, n]$, а вторая — в промежутке $[1, m]$. Если есть несколько возможных ответов, выведите любой из них.

Примеры

стандартный ввод	стандартный вывод
2 5	6 2 5 1 5 2 3 1 1 1 3 2 1
5 5	8 1 1 1 2 5 4 1 3 5 3 1 4 5 2 1 5

Задача E. Угадайка для шахмат Фишера

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Это интерактивная задача.

В шахматах Фишера начальная позиция выбирается случайным образом из 960 позиций. Начальная позиция удовлетворяет следующим требованиям:

- Фигуры белых перемешаны (на первой горизонтали расположены король, ферзь, две ладьи, два слона и два коня).
- Фигуры чёрных расположены на восьмой горизонтали в том же порядке, что и фигуры белых.
- Король располагается между двумя ладьями (между королем и ладьями могут быть другие фигуры). Это нужно для правила рокировки.
- Слоны белых располагаются на полях различного цвета. Другими словами, один слон на чётной вертикали, а другой на нечётной.

Джил загадала одну из начальных позиций. Задача Джека состоит в том, чтобы её угадать. Джек расставляет на шахматной доске какую-нибудь легальную начальную позицию, а Джил ему сообщает, сколько фигур белых он расставил правильным образом. Ваша задача состоит в том, чтобы помочь Джеку угадать позицию, сделав не более шести расстановок.

Протокол взаимодействия

Взаимодействие состоит из не более чем 100 игр. В начале игры необходимо из ввода прочитать строку, содержащую слово «`GAME`», и порядковый номер игры n. Номер игры задаётся целым числом, начиная с 1.

При попытке угадать позицию выведите строку из восьми символов, задающую расположение фигур белых. Король обозначается символом «`K`», ферзь — «`Q`», ладья — «`R`», слон — «`B`», а конь — «`N`». В ответ из ввода необходимо прочитать строку, содержащую целое число от 0 до 8 — количество правильно расставленных белых фигур. Игра заканчивается, если прочитано число 8 (все фигуры расставлены правильно) или превышен лимит на количество угадываний (в этом случае решение получит вердикт «`Wrong Answer`», если оно немедленно завершит работу, или какой-то не-OK, если оно попытается читать из ввода дальше).

В конце взаимодействия из ввода необходимо прочитать строку, содержащую слово «`END`».

После вывода каждой позиции не забывайте выводить перевод строки и очищать буфер вывода, иначе решение получит вердикт «`Idleness Limit Exceeded`». Очистить буфер можно, например, вызовом `fflush (stdout)` в C или C++, `System.out.flush ()` в Java, `flush (output)` в Pascal или `sys.stdout.flush ()` в Python.

В каждой игре позиция загадана заранее и не меняется от действий решения.

Пример

стандартный ввод	стандартный вывод
GAME 1 4 3 5 8 END	RQKBBNRN RQKNBRNB RKRNBBQN RKRBBQNN

Задача F. Пешки, бьющие вперёд

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Во многих комбинаторных играх даже небольшое изменение правил может полностью поменять «ландшафт» игры. Шахматы — не исключение. В обычных шахматах пешка — единственная фигура, которая ходит и бьёт по-разному: она ходит вперёд, но бьёт по диагонали. Но что было бы, если бы этого исключения не было, и пешки бы били вперёд?

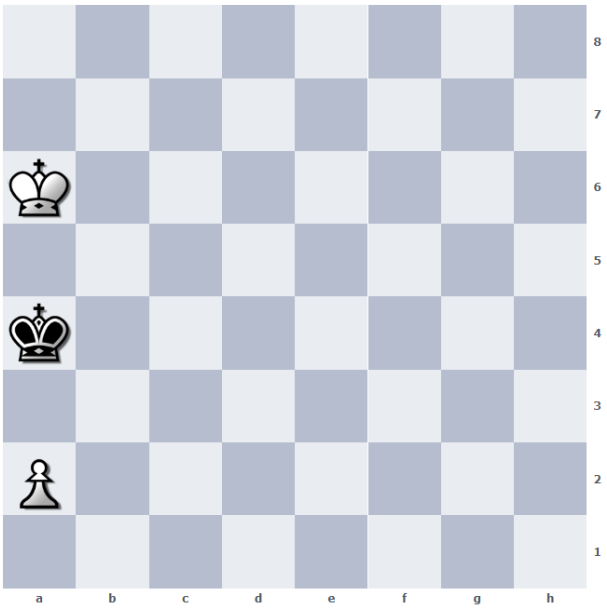
Определённо, получившаяся игра бы сильно отличалась от знакомых нам шахмат. Не было бы никаких закрытых пешечных структур, так как пешку в принципе не было бы возможно заблокировать. В наших «новых» шахматах есть только один способ остановить продвигающуюся вперёд пешку: взять её. Именно поэтому в «новых» шахматах не бывает «закрытых» (в обычном смысле) позиций. Более того, нет никакого способа сдвоить пешки.

Становится понятно, что подобное изменение полностью меняет то, как мы играем на каждом этапе шахматной партии: в дебюте, в миттельшпиле и в эндшпиле. Ваша задача состоит в том, чтобы понять, как это изменение влияет на самую простую часть теории окончаний: окончания «король и пешка против короля».

Все остальные правила не меняются. Вам достаточно рассматривать только *вменяемые* позиции. Позиция вменяема, если и только если все следующие свойства выполняются:

1. Все три фигуры (белый король, белая пешка, чёрный король) стоят на попарно различных полях.
2. Белая пешка не находится ни на 1-й, ни на 8-й горизонтали.
3. Если сейчас ход белых, то король чёрных не под шахом. Если же ход чёрных, то король белых не под шахом. В частности, короли не могут стоять в клетках, соседних по стороне или по углу.

Стоит заметить две важные вещи. Во-первых, **пешка, стоящая на 2-й горизонтали, бьёт два поля впереди себя**. Другими словами, пешка бьёт *в точности* так же, как она ходит. Например, пешка `c2` бьёт оба поля `c3` и `c4`. Рассмотрим следующую позицию:



Вменяема лишь при ходе чёрных. Чёрные под шахом.

Если в ней ход белых, то она не является вменяемой позицией, так как чёрный король под шахом. Если же ход чёрных, то она вменяема, и у чёрных есть всего два возможных хода: `Kb3` и `Kb4`, потому что поле `a3` под боем белой пешки, а поля `a5` и `b5` — под боем белого короля.

Второе важное замечание состоит в следующем. Хоть эта позиция и вменяема при ходе чёрных, но она не является корректной в обычном смысле этого слова (то есть её нельзя достичь из начальной никакой последовательностью возможных ходов). Если немного подумать, то становится понятно, что нет никакого предыдущего хода белых, который мог бы привести к этой позиции. Все корректные позиции вменяемы, но не все вменяемые позиции корректны. Неформально, вменяемые позиции — это в точности те, которые «выглядят корректными на первый взгляд». **Вас просят решить задачу именно для вменяемых позиций**. Таким образом, часть позиций во входных данных могут быть некорректными в обычном смысле этого слова.

Как уже упоминалось, все обычные правила шахмат не меняются. Если пешка стоит на 2-й горизонтали, она может сделать ход вперёд на два поля при условии, что оба эти поля свободны (но, разумеется, она **не может** «перепрыгнуть» через белого короля). Когда пешка достигает 8-й горизонтали, её можно превратить в ферзя, ладью, слона или коня. Пат и троекратное повторение позиции — всё ещё ничьи. Можете считать, что правила 50 ходов нет (можно доказать, что добавление или удаление правила 50 ходов не повлияет на ответ).

Формат входных данных

В первой строке дано одно число t: сколько позиций вам нужно рассмотреть ($t \geqslant 1$). Следующие t строк задают эти позиции.

Каждая позиция задана следующим образом: поля с белым королём, белой пешкой и чёрным королём (в таком порядке), чей сейчас ход («`w`» для белых или «`b`» для чёрных). Поля задаются стандартным образом («`a`»–«`h`» для вертикалей и «`1`»–«`8`» для горизонталей). Все эти четыре «токена» в строке разделены единичными пробелами. В конце строки нет лишних пробелов. Если что-то непонятно, то сверьтесь с примером.

Все данные на вход позиции вменяемы и попарно различны. Позиции, отличающиеся лишь очередью хода, считаются **различными**.

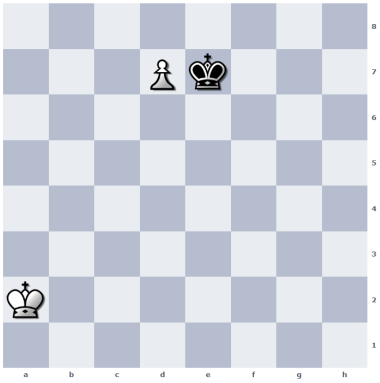
Формат выходных данных

Для каждой позиции выведите одну строку: «`Win`», если белые выигрывают при правильной игре, и «`Draw`» иначе.

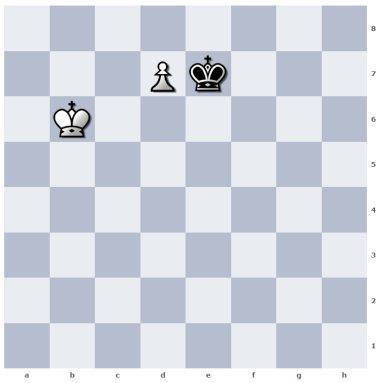
Пример

стандартный ввод	стандартный вывод
6 a2 d7 e7 w b6 d7 e7 b b6 d7 e7 w b5 a2 b2 w a6 a2 a4 b g6 g7 h8 b	Draw Draw Win Win Draw Draw

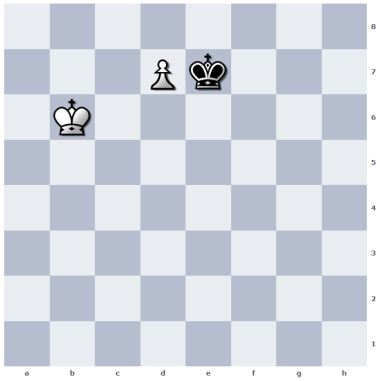
Замечание



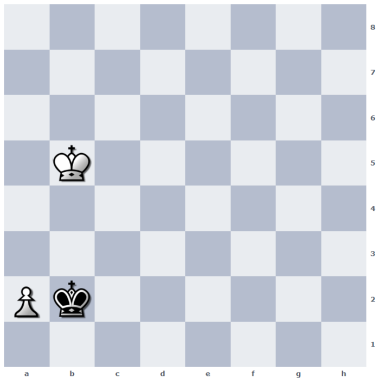
Ход белых, ничья. Белые могут провести пешку, но чёрные тогда немедленно возьмут получившуюся фигуру.



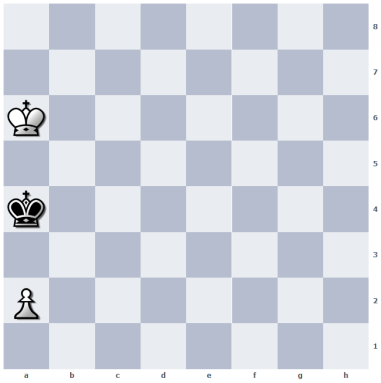
Ход чёрных, ничья. Чёрные немедленно бьют пешку.



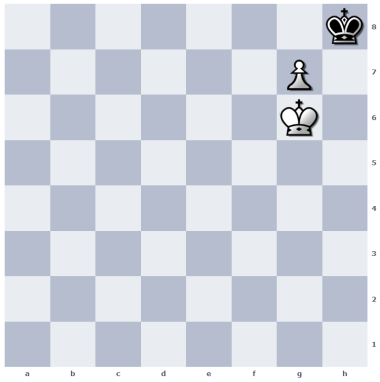
Ход белых, победа белых. Белые ходят `Kc7` и спокойно проводят пешку. Заметьте, что эта позиция отличается от предыдущей только очередью хода.



Ход белых, победа белых. Белые играют `a4`, используя право пешки, стоящей на 2-й горизонтали, пойти сразу на 2 поля вперёд.



Ход чёрных, ничья. Чёрные играют `Kb3` и забирают пешку следующим ходом.



Ход чёрных, ничья. У чёрных нет ходов, но их король не под шахом. Следовательно, это — пат, и позиция ничейна по определению.

Задача G. Разрезы графа

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	1024 мебибайта

Дан неориентированный граф без кратных рёбер и петель. Также имеется множество его вершин U, которое изначально пусто. Ваша задача — написать программу, которая будет отвечать на следующие виды запросов.

1. «`+` v». Добавить вершину v в U. Гарантируется, что $v \notin U$.
2. «`-` v». Удалить вершину v из U. Гарантируется, что $v \in U$.
3. «`?`». Найти ребро, у которого ровно один конец находится в U, и удалить его из графа, или определить, что таких рёбер нет. Если существует несколько рёбер, удовлетворяющих этому свойству, можно выбрать любое из них.

Формат входных данных

Первая строка содержит два целых числа n и m: количество вершин и рёбер в графе соответственно ($0 \leqslant n, m \leqslant 10^5$). Каждая из следующих m строк содержит два целых числа u и v: концы двунаправленного ребра ($1 \leqslant u, v \leqslant n$). Гарантируется, что в графе нет кратных рёбер и петель.

Следующая строка содержит одно целое число q — количество запросов ($0 \leqslant q \leqslant 10^5$). Следующие q строк содержат запросы в описанном выше формате (во всех запросах $1 \leqslant v \leqslant n$).

Формат выходных данных

Для каждого запроса третьего типа ваша программа должна вывести либо номер найденного ребра в порядке, в котором оно было представлено на входе, либо 0, если такого ребра не существует.

Пример

стандартный ввод	стандартный вывод
4 5 1 2 1 3 1 4 2 3 2 4 10 + 1 + 2 ? ? ? ? ? - 2 ? ?	5 4 3 2 0 1 0

Задача H. Очень разреженная таблица

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	10 секунд
Ограничение по памяти:	512 мебибайт

Это интерактивная задача.

Вам дан ориентированный граф G с вершинами, пронумерованными от 0 до n. Изначально G содержит ровно n рёбер вида $v \to v + 1$. Ваша задача состоит в том, чтобы добавить в этот граф несколько рёбер так, чтобы для любых двух вершин v, u $(v < u)$ существовал направленный путь из v в u, состоящий не более чем из трёх рёбер. Вы также должны выполнить два дополнительных требования:

1. Вы можете добавить ребро $a \to c$ тогда и только тогда, когда существует такое b, что рёбра $a \to b$ и $b \to c$ уже присутствуют в G.
2. Всего можно добавить не более $6 \cdot n$ рёбер.

Протокол взаимодействия

Взаимодействие начинается с того, что интерактор печатает единственное целое число n $(0 \leqslant n \leqslant 2^{16})$ на первой строке. После этого ваша программа должна вывести k $(0 \leqslant k \leqslant 6 \cdot n)$ — количество рёбер, которые вы хотите добавить в граф. Следующие k строк должны содержать описания добавленных рёбер в порядке их добавления. Каждое ребро должно быть описано тремя целыми числами a, b, c, что означает, что вы добавляете ребро $a \to c$, а рёбра $a \to b$ и $b \to c$ уже присутствуют в G.

Не забудьте после этого сбросить буфер вывода (команда `cout << flush;` в C++), иначе решение получит «`Idleness Limit Exceeded`».

Затем интерактор выводит количество запросов q $(0 \leqslant q \leqslant 2 \cdot 10^5)$ и q запросов в следующих q строках. Каждый запрос описывается двумя целыми числами ℓ, r $(0 \leqslant \ell < r \leqslant n)$, для которых ваша программа должна вывести строку, содержащую путь в G длины не более трёх, который начинается в вершине ℓ и заканчивается в вершине r. Путь должен быть напечатан как последовательность всех вершин, которые он посещает (включая обе конечные точки), разделённых пробелами.

Чтобы получить следующий запрос, **нужно** вывести ответ на предыдущий, завершить строку переводом строки и сбросить буфер вывода. Запросы **зависят** от выведенных рёбер.

В приведенном ниже примере **нет** фактических пустых строк на входе и выходе: они добавлены только для визуального выравнивания входных и выходных данных.

Пример

стандартный ввод	стандартный вывод
9 3 1 8 2 4 0 5	7 1 2 3 0 1 3 6 7 8 6 8 9 3 4 5 4 5 6 3 5 6 1 3 6 8 2 3 4 0 1 3 5

Задача I. Пароль

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

После очередной утечки персональных данных администратор Почты.ком решил ужесточить правила для паролей сотрудников. Теперь пароль каждого сотрудника должен состоять ровно из n символов, а небуквенные символы должны встречаться среди каждых трёх идущих подряд символов. Дополнительное ограничение: небуквенным должен быть символ в середине пароля — один, если n нечётно, или оба ближайших к середине символа, если n чётно.

Например, для $n = 9$ следующие пароли подходят: «`p4ss#or0s`», «`1a2b34CD5`». Пароль «`1234a56bc`» не подходит, потому что пятый символ должен быть небуквенным. Пароль «`9ASE#orkd`» не подходит, потому что он содержит три буквы подряд.

Для $n = 6$ подходят пароли «`ab23bc`» и «`5a428E`». Не подходят пароли «`111e11`» и «`4sy1um`».

Сотрудники задумались: какое наименьшее и наибольшее количество небуквенных символов может встречаться в пароле заданной длины? Помогите им это выяснить.

Формат входных данных

В первой строке записано целое число n — длина пароля ($1 \leqslant n \leqslant 1\,000\,000$).

Формат выходных данных

Выведите через пробел два целых числа: минимальное и максимальное количество небуквенных символов в пароле.

Примеры

стандартный ввод	стандартный вывод
1	1 1
2	2 2
3	1 3

Задача J. Транспортные плюсики

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Камбит живёт на плоскости. Он хочет попасть из дома, расположенного в точке (x_h, y_h), на выставку, находящуюся в точке (x_e, y_e). По плоскости можно перемещаться двумя способами, каждый из которых требует затрат энергии.

Во-первых, можно просто двигаться по отрезку между двумя точками. Энергия, нужная для такого перемещения, равна длине отрезка: при перемещении из точки (x_a, y_a) в точку (x_b, y_b) тратится $\sqrt{|x_b - x_a|^2 + |y_b - y_a|^2}$ единиц энергии.

Во-вторых, на плоскости расположены n *транспортных плюсиков*, пронумерованных целыми числами от 1 до n. Плюсик i имеет центр в точке (x_i, y_i) и соединяет все точки, у которых $x = x_i$ или $y = y_i$: из любой такой точки можно мгновенно попасть в любую другую, достаточно ввести координаты в мобильном приложении. Каждое использование любого плюсика требует t единиц энергии.

Камбит может использовать любые доступные способы перемещения в любом порядке. Помогите ему найти путь, требующий минимальных затрат энергии.

Формат входных данных

В первой строке заданы два целых числа n и t — количество транспортных плюсиков и затраты энергии на однократное использование плюсика ($0 \leqslant n, t \leqslant 100$). Во второй строке заданы два целых числа x_h и y_h — координаты дома Камбита ($0 \leqslant x_h, y_h \leqslant 100$). В третьей строке заданы два целых числа x_e и y_e — координаты выставки ($0 \leqslant x_e, y_e \leqslant 100$). В каждой из следующих n строк заданы два целых числа x_i и y_i — координаты центра i-го транспортного плюсика ($0 \leqslant x_i, y_i \leqslant 100$).

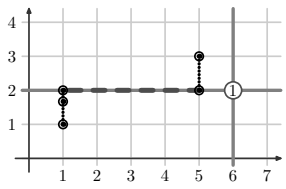
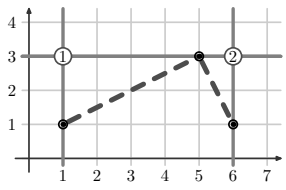
Все числа во входных данных целые. Однако это не мешает Камбиту перемещаться в точки, координаты которых — любые вещественные числа.

Формат выходных данных

В первой строке выведите вещественное число — суммарные затраты энергии. Во второй строке выведите целое число k — количество перемещений в пути ($0 \leqslant k \leqslant 10\,000$). В каждой из следующих k строк выведите три числа p_j, x_j и y_j — номер используемого транспортного плюсика и координаты очередной цели. Значение p_j может быть нулём — это означает перемещение по отрезку — или целым числом от 1 до n — это означает использование плюсика с таким номером. Координаты могут быть любыми **вещественными** числами от 0 до 100. При использовании плюсика этот плюсик должен соединять нынешнее положение и очередную цель. Путь должен заканчиваться в (x_e, y_e).

Можно выводить любой путь, требующий минимальных затрат энергии. Ответ будет считаться верным, если затраты энергии отличаются от минимальных не более чем на 10^{-3}.

Примеры

стандартный ввод	стандартный вывод	*иллюстрация*
1 2 1 1 5 3 6 2	4.000000 4 0 1 1.67 0 1 2 1 5 2 0 5 3	
2 1 1 1 6 1 1 3 6 3	2.0 2 1 5 3 2 6 1	

Задача K. Несчастные студенты

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мебибайт

Близится конец семестра — самая жаркая пора для студента. У группы было k курсов, в этой группе n студентов, и каждый студент должен сдать ровно один экзамен.

Если студент i будет сдавать экзамен j, его уровень расстройства будет равен $c_{i,j}$. Суммарное расстройство студентов — это сумма расстройств всех студентов.

Преподаватели поставили условие: каждый экзамен j может быть выбран не более чем a_j студентами. Найдите минимально возможное суммарное расстройство студентов при таком условии.

Формат входных данных

В первой строке записаны два целых числа n и k: количество студентов в группе и количество экзаменов ($1 \leqslant n \leqslant 50\,000$, $1 \leqslant k \leqslant 10$).

Далее следует n строк. В i-й из них записаны k целых чисел $c_{i,1}, c_{i,2}, \ldots, c_{i,k}$: расстройство студента i при выборе экзамена $1, 2, \ldots, k$ ($1 \leqslant c_{i,j} \leqslant 10^9$).

В последней строке записаны k целых чисел $a_1, a_2, \ldots, a_k$: максимальное количество студентов, которые могут выбрать экзамены $1, 2, \ldots, k$ ($0 \leqslant a_j \leqslant n$). Гарантируется, что $\sum a_j \geqslant n$.

Формат выходных данных

Выведите одно целое число: минимально возможное суммарное расстройство студентов.

Примеры

стандартный ввод	стандартный вывод
6 2 1 2 1 3 1 4 1 5 1 6 1 7 3 4	12
3 3 1 2 3 2 4 6 6 5 4 1 1 1	8

Задача L. Подсказка в игре «Мемо»

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

«Мемо», или «Мемори» — игра с карточками. Для игры в этой задаче используется 25 картинок, каждая из которых изображена ровно на двух прямоугольных карточках — всего получается 50 карточек. Изначально все карточки случайно перемешаны и перевёрнуты рубашкой вверх — так, что картинки не видны. Карточки лежат на позициях, пронумерованных целыми числами от 1 до 50.

Василиса тренирует память, играя в «Мемо». Она делает ходы в игре. Каждый ход состоит из двух действий. Первым действием Василиса выбирает позицию, где карточка лежит рубашкой вверх, и переворачивает эту карточку картинкой вверх. Вторым действием она выбирает другую такую позицию и также переворачивает карточку в ней. Если картинки на двух карточках одинаковые, эти карточки остаются лежать картинками вверх. Иначе обе карточки опять переворачиваются рубашкой вверх, а Василисе засчитывается *промах*.

Василиса побеждает, когда все карточки оказываются перевёрнуты картинками вверх. Её задача — победить, допустив как можно меньше промахов.

Василиса натренировала память и придумала себе хорошую стратегию, и теперь в среднем допускает за игру примерно 14.83 промахов. Чтобы было интереснее, она позвала сестру Сашу играть вместе.

Рисунок на рубашке всех 50 карточек один и тот же, но он выглядит иначе, если *повернуть* карточку на 180 градусов, не переворачивая. Теперь Саша с Василисой играют так. Сначала Саша перемешивает карточки и раскладывает их картинками вверх. После этого Саша переворачивает карточки рубашкой вверх, но каждую карточку может либо повернуть на 180 градусов, либо не поворачивать. Наконец, Василиса исходно видит не просто 50 карточек с одинаковой рубашкой, но и повёрнута ли каждая из них на 180 градусов.

Придумайте, как Саше и Василисе договориться о поворачивании карточек, чтобы Василиса могла допускать за игру в среднем не более 13.5 промахов.

Протокол взаимодействия

Это интерактивная задача. Кроме того, в этой задаче ваше решение будет запущено на каждом тесте два раза. В конце каждой строки входных данных следует символ перевода строки.

Первый запуск

При первом запуске решение действует за Сашу. В первой строке записано слово «`prepare`». Вторая строка содержит целое число t — количество тестовых случаев ($1 \leqslant t \leqslant 100$). Каждая из следующих t строк описывает начальное положение карточек — 50 заглавных английских букв «`A`»–«`Y`» без пробелов. Одинаковым картинкам соответствуют одинаковые буквы. Каждая буква встречается в строке ровно два раза. Гарантируется, что все строки, кроме примеров, в каждом тесте выбраны заранее, случайно и равновероятно.

В ответ на каждый тестовый случай выведите строку из 50 двоичных цифр — в каждой позиции единицу, если карточку следует повернуть на 180 градусов, и ноль, если не следует.

Технически этот запуск интерактивный, но все входные данные даются сразу.

Второй запуск

При втором запуске решение действует за Василису. В первой строке записано слово «`play`». Вторая строка содержит целое число t — количество тестовых случаев, такое же, как при первом запуске.

Программа жюри считает суммарное количество промахов за все тестовые случаи. Это количество должно быть не больше 1350: например, если тестовых случаев ровно 100 (а не меньше), это означает, что в среднем должно получиться не более 13.5 промахов в каждом.

Каждый тестовый случай начинается строкой из 50 двоичных цифр — той, которая была выведена при первом запуске. Далее следует интерактивно выводить действия, пока все карточки не

окажутся перевёрнуты картинками вверх. Чтобы перевернуть карточку на позиции p (целое число от 1 до 50), следует вывести строку, содержащую число p. В ответ решение получает строку из двух символов. Эта строка равна «##» (две решётки), если ход некорректен: например, позиция p некорректна, или карточка в ней уже перевёрнута, или этим ходом превышено максимальное количество промахов. В этом случае решение должно завершить работу. Иначе первый символ — заглавная английская буква «A»–«Y» на позиции p, соответствующая картинке. Второй символ может быть таким:

- «.» (точка), если это первое действие хода;
- «-» (минус), если это второе действие хода, и картинки разные;
- «+» (плюс), если это второе действие хода, картинки одинаковые, но Василиса ещё не победила;
- «!» (восклицательный знак), если это второе действие хода, картинки одинаковые, и это действие приводит к победе.

После ответа с восклицательным знаком со следующей строки начинается следующий тестовый случай — или взаимодействие завершается, если они закончились.

Не забывайте выводить перевод строки и очищать буфер вывода сразу после каждого выведенного действия — иначе вы, вероятно, получите вердикт «Idleness Limit Exceeded».

При получении строки «##» решение может немедленно завершить работу — тогда оно точно получит вердикт «Wrong Answer», а не какой-нибудь другой.

Примеры

На каждом тесте входные данные при втором запуске зависят от того, что вывело решение при первом запуске.

Далее показаны два запуска какого-то решения на первом тесте. Пустые строки добавлены лишь для удобства чтения — при реальном запуске их не будет. Для экономии места почти все ходы заменены на «(...)».

стандартный ввод	стандартный вывод
prepare 2 ABCDEFGHIJKLMNOPQRSTUVWXYABCDEFGHIJKLMNOPQRSTUVWXY AABBCCDDEEFFGGHHIIJJKKLLMMNNOOPPQQRRSSTTUUVVWWXXYY	 00000000000000000000000001111111111111111111111111 11010011001111111111111111111111111111111111111010

стандартный ввод	стандартный вывод
play 2 00000000000000000000000001111111111111111111111111 A. E- A. A+ (...) Y. Y! 11010011001111111111111111111111111111111111111010 E. E+ (...) A. A!	 1 5 26 1 (...) 25 50 10 9 (...) 1 2

Задача M. Жёсткий подсчёт строк

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	8 секунд
Ограничение по памяти:	512 мебибайт

Вам дана непустая строка s из строчных латинских букв. Назовём строку w из строчных латинских букв *хорошей*, если все собственные префиксы w не содержат s как подстроку, а сама w — содержит.

Посчитайте количество хороших строк длины m. Это количество может быть очень большим, поэтому выведите его по модулю простого числа $998\,244\,353 = 2^{23} \cdot 119 + 1$.

Формат входных данных

В первой строке дано два числа через пробел: n, длина строки s, и m, длина тех строк, количество которых вам нужно посчитать ($1 \leqslant n \leqslant 10^5$, $n \leqslant m \leqslant 10^9$). Во второй строке дана сама s, состоящая из n строчных букв латинского алфавита.

Формат выходных данных

Выведите одно неотрицательное целое число: количество хороших строк длины m по модулю $998\,244\,353$.

Примеры

стандартный ввод	стандартный вывод
6 7 aaaaaa	25
3 5 aba	675