Информация о результатах оценивания решений во время тура для всех задач

В течение тура можно не более 10 раз по каждой задаче запросить информацию о результатах оценивания решения на тестах жюри.

Запрос по каждой задаче можно делать не чаще одного раза в 5 минут. Для каждой подзадачи сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте.

В каждой задаче можно задать, какое из прошедших предварительную проверку решений будет оцениваться. В этом случае баллы начисляются за лучшее решение из следующих:

- выбранного явно;
- последнего принятого на проверку решения.

Если выбор не сделан, то будет оцениваться лучшее решение из следующих:

- тех решений, по которым просмотрены баллы;
- последнего принятого на проверку решения.

Задача 1. «Пароль»

Имя входного файла:	`password.in`
Имя выходного файла:	`password.out`
Максимальное время работы на одном тесте:	1 *секунда*
Максимальный объем используемой памяти:	256 *мегабайт*
Максимальная оценка	100 *баллов*

Участник олимпиады разбирается с программой, которая шифрует пароль входа в систему. После работы эта программа выдает два натуральных числа, причем второе число получено из первого в результате замены некоторой непустой группы подряд идущих цифр первого числа на их сумму. Известно, что пароль – это группа цифр первого числа, замененная на их сумму во втором числе.

Требуется написать программу, которая по двум числам определяет номера позиций первой и последней цифры группы, являющейся искомым паролем.

Формат входных данных

Входной файл содержит две строки. В первой строке записано первое число, состоящее не более чем из 100 000 цифр, во второй строке – второе число. Гарантируется, что числа не начинаются с нуля.

Формат выходных данных

Выходной файл должен содержать два разделённых пробелом числа – номера позиций первой и последней цифры группы, которая была заменена в первом числе. Если решений несколько, можно вывести любое из них. Гарантируется, что решение существует.

Примеры входных и выходных данных

password.in	**password.out**
2148 213	2 4
8 8	1 1
1223 1223	4 4
10002 1002	3 4

Комментарий

В первом примере группа цифр 148 заменятся на число 13 = 1 + 4 + 8.

Подзадачи и система оценки

Данная задача содержит три подзадачи. Для оценки каждой подзадачи используется своя группа тестов. Баллы за подзадачу начисляются только в том случае, если все тесты из этой группы пройдены.

Подзадача 1 (оценивается в 30 баллов)

Первое число меньше 10^9.

Подзадача 2 (оценивается в 30 баллов)

Первое число меньше 10^{1000}.

Подзадача 3 (оценивается в 40 баллов)

Первое число меньше $10^{100\,000}$.

Задача 2. «Вирусы и антивирусы»

Имя входного файла:	`virus.in`
Имя выходного файла:	`virus.out`
Максимальное время работы на одном тесте:	2 *секунды*
Максимальный объем используемой памяти:	256 *мегабайт*
Максимальная оценка	100 *баллов*

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

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

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

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

Требуется написать программу, определяющую количество устойчивых пар в компании.

Формат входных данных

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

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

В третьей строке – N целых чисел b_i, где $b_i = 0$, если в тайной иерархии сотрудник с номером i является боссом, в противном случае b_i равно номеру непосредственного начальника сотрудника номер i.

Нумерация сотрудников ведется с единицы в том порядке, в каком они упомянуты во входном файле.

Формат выходных данных

Выходной файл должен содержать единственное число – количество устойчивых пар.

Примеры входных и выходных данных

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

Подзадачи и система оценки

Данная задача содержит три подзадачи. Для оценки каждой подзадачи используется своя группа тестов. Баллы за подзадачу начисляются только в том случае, если все тесты из этой группы пройдены.

Подзадача 1 (оценивается в 25 баллов)

Количество сотрудников N не превосходит 100.

Подзадача 2 (оценивается в 25 баллов)

Количество сотрудников N не превосходит 2000.

Подзадача 3 (оценивается в 50 баллов)

Количество сотрудников N не превосходит 100 000.

Задача 3. «Урюк»

Имя входного файла:	`apricot.in`
Имя выходного файла:	`apricot.out`
Максимальное время работы на одном тесте:	1 *секунда*
Максимальный объем используемой памяти:	256 *мегабайт*
Максимальная оценка	100 *баллов*

В давние времена Золотая Орда ежегодно собирала дань золотыми монетами. Известный крымский хан Гирей решил схитрить: выплачивая дань из N золотых монет, он подложил среди них одну фальшивую – более легкую монету. Об этом донесли казначею Золотой Орды. Для обнаружения подделки он решил использовать магические весы, работающие на урюке.

На чаши магических весов кладутся две кучи монет. Весы устанавливают, совпадает или различается вес этих куч. При этом, если кучи имеют разный вес, то весы указывают, какая из куч легче. При совпадении веса обеих куч весы требуют R плодов урюка, а при несовпадении – U плодов.

Казначей, сам любитель урюка, хочет и фальшивую монету обнаружить, и сэкономить на урюке.

Требуется написать программу, которая по заданному количеству монет N, при условии, что только одна из них легче других, укажет минимальное количество урюка, с помощью которого эта фальшивая монета гарантированно будет обнаружена.

Формат входных данных

Во входном файле в единственной строке находятся три целых числа N, R и U ($2 \le N \le 1\,000\,000$, $1 \le R, U \le 1\,000\,000$), где N – количество монет, R – количество плодов урюка, затрачиваемых в случае совпадения веса куч монет, U – количество плодов урюка, затрачиваемых в случае их различия. Все числа разделены пробелом.

Формат выходных данных

Выходной файл должен содержать одно число – минимальное количество урюка, с помощью которого гарантированно будет обнаружена фальшивая монета.

Пример входных и выходных данных

apricot.in	apricot.out
4 3 1	2
3 3 1	3
15 2 3	8
10 2 1	3

Подзадачи и система оценки

Данная задача содержит три подзадачи. Для оценки каждой подзадачи используется своя группа тестов. Баллы за подзадачу начисляются только в том случае, если все тесты из этой группы пройдены.

Подзадача 1 (оценивается в 40 баллов)

N, U, R не превосходят 200.

Подзадача 2 (оценивается в 30 баллов)

N, U, R не превосходят 2000.

Подзадача 3 (оценивается в 30 баллов)

N, U, R не превосходят 1 000 000.

Задача 4. «Древний календарь»

Имя входного файла:	`calendar.in`
Имя выходного файла:	`calendar.out`
Максимальное время работы на одном тесте:	1 *секунда*
Максимальный объем используемой памяти:	256 *мегабайт*
Максимальная оценка	100 *баллов*

Как известно, в 2012 году человечество с повышенным вниманием относится к древним календарям. Особый интерес представляют те из них, которые не заканчиваются 2012 годом. Потрясающее открытие в этом направлении сделано археологами Татарстана. В древних захоронениях они обнаружили прямоугольную табличку, которая после расшифровки сохранившихся знаков была записана в виде таблицы, состоящей из N строк по M десятичных цифр в каждой. Но полностью расшифровать табличку не удалось, так как некоторые цифры стерлись. Утраченные цифры в таблице были заменены символами «*».

По мнению археологов, найденная табличка представляет собой древний календарь, а записанные в ней M-значные числа являются номерами последовательных дней некоторого периода. Первое число является номером первого дня этого периода, а каждое следующее число на единицу больше предыдущего. По этому календарю конец света отсутствует, и после дня, обозначаемого с помощью M девяток, следует номер дня из M нулей.

Требуется написать программу, которая восстанавливает утраченные цифры так, чтобы число в каждой строке таблицы, начиная со второй, было на единицу больше предыдущего, и выводит номер первого дня в найденном календаре.

Формат входных данных

В первой строке входного файла записаны натуральные числа N и M – количество строк в таблице и длина каждой строки соответственно ($1 \le N \le 100\,000$, $1 \le M \le 100\,000$, $M \times N \le 100\,000$). Далее следуют N строк по M символов в каждой, состоящих только из десятичных цифр от 0 до 9 и символов «*».

Формат выходных данных

Выходной файл должен содержать одну строку, состоящую из M цифр – номер первого дня календаря. Если вариантов восстановления несколько, можно вывести любой из них. Гарантируется, что хотя бы один способ восстановления существует.

Примеры входных и выходных данных

calendar.in	calendar.out
1 2 23	23
3 3 1** *1* **1	109
2 3 9** 00*	999
3 4 **** *0** 01**	0098

Подзадачи и система оценки

Данная задача содержит три подзадачи.

Подзадача 1 (оценивается в 40 баллов)

$1 \le N \le 1000$, $1 \le M \le 100$. В этой подзадаче исходные данные таковы, что в каждом столбце таблицы есть, по крайней мере, одна сохранившаяся цифра. Баллы за подзадачу начисляются только в том случае, если все тесты из этой группы пройдены.

Подзадача 2 (оценивается из 30 баллов)

$1 \le N \le 1000$, $1 \le M \le 100$. В этой подзадаче хотя бы один столбец содержит только символы «*». Каждый тест в этой подзадаче оценивается отдельно.

Подзадача 3 (оценивается в 30 баллов)

$1 \le N \le 100\,000$, $1 \le M \le 100\,000$, $M \times N \le 100\,000$. Баллы за подзадачу начисляются только в том случае, если все тесты из этой группы пройдены.