Задача A. Агроном-любитель

Имя входного файла:	`agro.in`
Имя выходного файла:	`agro.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Городской школьник Лёша поехал на лето в деревню и занялся выращиванием цветов. Он посадил n цветков вдоль одной длинной прямой грядки, и они успешно выросли. Лёша посадил различные цветки, i-й от начала грядки цветок имеет вид a_i, где a_i — целое число, номер соответствующего вида в «Каталоге юного агронома».

Теперь Лёша хочет сделать фотографию выращенных им цветов и выложить ее в раздел «мои грядки» в социальной сети для агрономов «ВКомпосте». На фотографии будет виден отрезок из одного или нескольких высаженных подряд цветков.

Однако он заметил, что фотография смотрится не очень интересно, если на ней много одинаковых цветков подряд. Лёша решил, что если на фотографии будут видны три цветка одного вида, высаженные подряд, то его друзья — специалисты по эстетике цветочных фотографий — поставят мало лайков.

Помогите ему выбрать для фотографирования как можно более длинный участок своей грядки, на котором нет трех цветков одного вида подряд.

Формат входных данных

В первой строке содержится целое число n ($1 \le n \le 200\,000$) — количество цветков на грядке.

Во второй строке содержится n целых чисел a_i ($1 \le a_i \le 10^9$), обозначающих вид очередного цветка. Одинаковые цветки обозначаются одинаковыми числами, разные — разными.

Формат выходных данных

Выведите номер первого и последнего цветка на самом длинном искомом участке. Цветки нумеруются от 1 до n.

Если самых длинных участков несколько, выведите описание любого из них.

Пример

agro.in	agro.out
6 5 6 6 6 23 9	3 6

Задача B. Верёвочный парк

Имя входного файла:	`amusement.in`
Имя выходного файла:	`amusement.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В парке развлечений «Пуперленд» открылся огромный верёвочный парк. Особая гордость парка — трасса, состоящая из n платформ, соединённых $n-1$ верёвками: первая платформа соединена со второй, вторая — с третьей, ..., $n-1$-я — с n-й, верёвка, соединяющая i-ю платформу с $i+1$-й имеет длину l_i.

В парке серьёзно относятся к технике безопасности, так что для трассы были разработаны следующие правила эксплуатации:

- для всех i от 2 до $n-1$ на i-й платформе разрешается находиться не более, чем p_i людям одновременно; (первая и последняя платформы достаточно надёжны, и на них может находиться произвольное число людей);
- на верёвке, протянутой между i-й и $i+1$-й платформами, разрешается находиться не более, чем r_i людям одновременно;
- для каждой верёвки известно минимальное безопасное расстояние d_i метров, такое что двум людям, одновременно идущим по этой верёвке, нельзя приближаться друг к другу ближе, чем на это расстояние.

В день открытия в парк пришло m человек, каждый из которых хочет пройти по трассе. Все они выстроились в очередь на первой платформе и сразу после открытия трассы готовы начать свое приключение. Посетители должны двигаться вдоль трассы в том порядке, в котором они стоят в очереди на первой платформе, меняться местами с другим посетителем во время прохождения трассы не разрешается. Все посетители должны добраться до n-й платформы и покинуть трассу.

Все люди разные и будут проходить трассу с разной скоростью. Пронумеруем посетителей от 1 до m в том порядке, в котором они выстроились в очередь. Для j-го посетителя известна скорость $v_{i,j}$ м/с — максимальная скорость, с которой он может проходить по верёвке, соединяющей i-ю и $i+1$-ю платформы. Таким образом, j-й посетитель может перемещаться по i-й верёвке с любой скоростью от 0 до $v_{i,j}$, соблюдая при этом условия про минимальное расстояние до других посетителей, идущих по той же верёвке.

Администрация парка не ожидала такого наплыва посетителей и теперь опасается, что все посетители могут не успеть пройти трассу до закрытия парка. Помогите им посчитать минимальное время, которое потребуется всем посетителям, чтобы пройти трассу.

Формат входных данных

В первой строке заданы два целых числа n ($2 \le n \le 100$) и m ($1 \le m \le 100$) — число платформ на трассе и число посетителей.

Во второй строке заданы $n-2$ целых числа $p_2, \ldots, p_{n-1}$ ($1 \le p_i \le 100$) — ограничения на число людей на платформах. Обратите внимание, что если $n = 2$, то эта строка пуста.

В следующей строке задано $n-1$ целое число $r_1, r_2, \ldots, r_{n-1}$ ($1 \le r_i \le 100$) — ограничение на число людей на i-й верёвке.

В следующей строке задано $n-1$ целое число $l_1, l_2, \ldots, l_{n-1}$ ($1 \le l_i \le 100$) — длины верёвок в метрах.

В следующей строке задано $n-1$ целое число $d_1, d_2, \ldots, d_{n-1}$ — ограничение в метрах на расстояние между людьми на i-й верёвке. Гарантируется, что $1 \le d_i \le l_i$.

В оставшихся $n-1$ строке находится по m целых чисел:

$v_{1,1}, v_{1,2}, \ldots, v_{1,m}$
$v_{2,1}, v_{2,2}, \ldots, v_{2,m}$
...
$v_{n-1,1}, v_{n-1,2}, \ldots, v_{n-1,m}$,

где $v_{i,j}$ — скорость в м/с j-го посетителя на i-й верёвке ($1 \le v_{i,j} \le 100$).

Формат выходных данных

Выведите единственное число — время в секундах, которое необходимо, чтобы все посетители прошли трассу.

Ваш ответ должен иметь относительную или абсолютную погрешность не больше 10^{-6}. Таким образом, он будет засчитан, если $\frac{|a-p|}{\max(a,1)} \leq 10^{-6}$, где p — ваш ответ, а a — правильный ответ.

Пример

amusement.in	amusement.out
2 1 1 30 2 2	15
3 2 1 2 2 10 10 5 5 2 2 1 2	17.5

Задача C. Школьная демократия

Имя входного файла:	`demo.in`
Имя выходного файла:	`demo.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В школе номер 932 проходят выборы в школьный совет. Выборы проходят по следующей системе. Завуч рассматривает список всех классов школы и разбивает их на группы. Каждая группа составлена из одного или нескольких классов, которые идут подряд в списке, каждый класс в результате разбиения попадает ровно в одну группу.

Каждая группа классов выдвигает двух кандидатов в школьный совет — одного мальчика и одну девочку. Далее каждый школьник голосует за одного из двух кандидатов своей группы. При этом известно, что мальчики всегда голосуют за кандидата-мальчика, а девочки — за кандидата-девочку. В каждой группе результаты голосования подводятся независимо. Кандидат, набравший наибольшее количество голосов в своей группе, считается избранным в школьный совет. В случае равенства голосов оба кандидата, выдвинутых в этой группе, считаются избранными в школьный совет.

Пусть в результате выборов в школьный совет пройдет B мальчиков и G девочек. По опыту прошлых лет завуч думает, что совет будет работать тем эффективней, чем больше разность $B - G$ числа мальчиков и числа девочек в совете. Обратите внимание, что эта величина может оказаться и отрицательной, завуч хочет максимизировать именно само значение этой величины, а не ее модуль. Например, из вариантов $B = 2$, $G = 5$, при котором $B - G = -3$, и $B = 3$, $G = 4$, при котором $B - G = -1$, второй вариант предпочтительнее.

Всего в школе n классов, и завуч уже подготовил их список. Теперь ему предстоит разбить их на группы. Группа не может содержать меньше чем l классов, иначе совет будет очень большим. В то же время группа не может содержать больше чем r классов, иначе учащиеся не смогут договориться о выдвигаемых кандидатах. Напомним, что каждая группа должна быть составлена из классов, которые идут подряд в списке завуча.

Помогите завучу найти оптимальное по его мнению разбиение на группы.

Формат входных данных

В первой строке входного файла содержится два целых числа n, l и r ($1 \le n \le 100\,000$, $1 \le l \le r \le n$) — количество классов в школе, максимальное и минимальное допустимое количество классов в одной группе соответственно. В следующих n строках содержится по два целых числа b_i и g_i ($1 \le b_i, g_i \le 10\,000$) — количество мальчиков и девочек в i-м классе, соответственно.

Формат выходных данных

В первой строке выведите целое число x — количество групп в оптимальном по мнению завуча разбиении. В следующих x строках выведите по два числа s_i и f_i ($1 \le s_i \le f_i \le n$). Это означает, что в i-ю группу следует включить классы с s_i-го по f_i-й, включительно. Группы можно выводить в любом порядке. Каждый класс должен войти ровно в одну группу.

Гарантируется, что существует хотя бы одно разбиение, удовлетворяющее всем ограничениям. Если существует несколько оптимальных ответов, выведите любой.

Примеры

demo.in	demo.out
5 1 2 7 5 10 1 2 3 2 6 4 3	4 1 1 2 3 4 4 5 5

Задача D. Полёт мечты

Имя входного файла:	`dreamrun.in`
Имя выходного файла:	`dreamrun.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Поля участвует в международной олимпиаде по парапланеризму. Каждому участнику олимпиады необходимо выполнить следующее задание. Стартовав из заданной точки, необходимо пролететь d километров на юг, затем d километров на запад и затем d километров на север. И в результате этого полета участник должен вернуться в исходную точку!

При этом участнику запрещается подлетать ближе чем на километр к южному полюсу, так как там расположена вышка, на которой находится жюри олимпиады.

Значение d участник олимпиады выбирает самостоятельно, необходимо только, чтобы выполнялось условие $d \geq 1$. Поля быстро сообразила, что выбрать d не так то просто, и обратилась к вам за помощью.

Будем считать Землю шаром с центром в точке $(0, 0, 0)$ и радиусом 6371 километр. Северный и южный полюса имеют координаты $(0, 0, 6371)$ и $(0, 0, -6371)$, соответственно. Стартовая точка имеет трехмерные координаты (x, y, z), где x, y и z — целые числа, причем старт находится на поверхности Земли, то есть $x^2 + y^2 + z^2 = 6371^2$.

Помогите Поле выбрать такое вещественное число $d \geq 1$, что стартовав из заданной точки и пролетев d километров на юг, затем d километров на запад, и затем d километров на север, она вернется в стартовую точку, при этом не пролетая близко к южному полюсу.

Формат входных данных

В единственной строке входного файла содержится три целых числа x, y, z — координаты старта в описанной системе координат.

Гарантируется, что старт находится на поверхности Земли, и расстояние от стартовой точки до южного полюса не менее 10 километров.

Формат выходных данных

Выведите одно вещественное число — искомое d. Ответ будет считаться верным, если $d \geq 1$, а расстояние между стартовой точкой и концом пути не больше 10^{-6}.

Если возможных d несколько, выведите любое из них. Гарантируется, что существует хотя бы одно d, удовлетворяющее условиям задачи.

Пример

dreamrun.in	dreamrun.out
0 0 6371	239.0

Задача E. Занимательное дежурство

Имя входного файла:	`duty.in`
Имя выходного файла:	`duty.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Однажды на перемене, во время дежурства по классу, Дима написал на доске несколько строчных английских букв и позвал Гришу на них посмотреть. Грише очень понравилась композиция на доске, но к началу урока доска должна быть идеально чистой. Ребятам жалко просто стирать буквы, и чтобы сделать этот процесс интереснее, Гриша предложил занимательную игру.

Ребята делают ходы по очереди. В свой ход игрок стирает с доски две одинаковые буквы, а вместо них записывает на доску одну любую букву. Так, например, из набора букв {a, b, a} можно получить наборы {a, b}, {b, b}, {b, c}, ..., {b, z}. Проигрывает тот, кто не может сделать ход, поскольку все записанные на доске буквы различны. Проигравший моет доску. Гриша ходит первым.

За происходящим внимательно наблюдает строгая учительница Дарья Владимировна. Она хочет узнать, кто выиграет в придуманной ребятами игре, если оба игрока будут придерживаться оптимальной стратегии.

Ваша задача — помочь ей узнать ответ на этот вопрос.

Формат входных данных

В единственной строке входного файла находится набор строчных английских букв, который был исходно записан на доске (число букв в наборе от 1 до 100 000, буквы не разделены пробелами).

Формат выходных данных

В выходной файл выведите `Grisha`, если выиграет Гриша, и `Dima` в противном случае.

Примеры

duty.in	duty.out
abc	Dima
aba	Grisha

Задача F. Рукопожатия

Имя входного файла:	`handshakes.in`
Имя выходного файла:	`handshakes.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В одной крупной компании работает n сотрудников. Сотрудники компании пронумерованы последовательными целыми числами от 1 до n в порядке, в котором они ежедневно приходят на работу. Никакие два сотрудника не приходят на работу одновременно, таким образом, первым приходит сотрудник с номером 1, вторым приходит сотрудник с номером 2 и так далее.

Некоторые пары сотрудников являются друзьями, при этом отношение дружбы симметрично, то есть если сотрудник с номером i считает своим другом сотрудника с номером j, то и сотрудник с номером j считает своим другом сотрудника с номером i. Известно, что когда очередной сотрудник приходит на работу, он быстро обходит весь офис и жмёт руку всем своим друзьям, уже находящимся в офисе, то есть тем, кто пришёл раньше него. Неизвестно, какие именно пары сотрудников являются друзьями, но для каждого сотрудника известно количество рукопожатий, которое он ежедневно совершает сразу после своего прихода на работу.

Директор компании хотел бы побеседовать с кем-нибудь из сотрудников о текущем состоянии дел. Для этого он хочет выбрать наиболее социально активного человека, а именно сотрудника с максимальным количеством друзей. По имеющейся информации определите, какое максимальное количество друзей может быть у одного из сотрудников.

Формат входных данных

В первой строке входных данных записано единственное число n ($1 \le n \le 200\,000$) — количество сотрудников в компании.

Во второй строке входных данных записаны n целых чисел h_i ($0 \le h_i < i$) — i-е число соответствует количеству рукопожатий, которое совершил сотрудник с номером i сразу после своего прихода на работу, то есть до прихода сотрудника с номером $i + 1$.

Формат выходных данных

Выведите единственное число — максимально возможное количество друзей у одного из сотрудников компании.

Примеры

handshakes.in	handshakes.out
2 0 1	1
5 0 0 1 1 1	3

Замечание

В первом примере есть всего одна пара сотрудников, и они, как следует из того, что $h_2 = 1$, являются друзьями.

Во втором примере, если все сотрудники 3, 4 и 5 здоровались с одним и тем же сотрудником, то у этого сотрудника 3 друга.

Задача G. Фишки

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Это интерактивная задача. Ваша программа будет взаимодействовать с программой жюри, используя стандартный ввод и вывод.

Программа жюри решила сыграть с вашей программой в игру. На доске $n \times n$ в двух различных клетках находятся две фишки. Ваша программа должна определить положение фишек. Для этого она может пытаться двигать фишки, а программа жюри будет сообщать результаты передвижений.

За один ход можно выбрать фишку и попросить переместить её на одну клетку влево, вправо, вверх или вниз. Программа жюри сообщает результат перемещения — если клетка в выбранном направлении существует и свободна, то перемещение считается успешным и фишка перемещается в эту клетку. В противном случае перемещение считается неудачным и фишка остается на той же клетке.

Вы выигрываете, если после очередного хода можете назвать исходное положение фишек на доске. Ваша задача — выиграть не более чем за $6n$ ходов.

Введем на доске систему координат таким образом, что клетки имеют координаты $(1,1), (1,2), \ldots, (1,n), (2,1), \ldots, (n,n)$. Команды для перемещения фишки кодируются латинскими буквами следующим образом:

- «`U`» — переместиться с клетки $(x,\ y)$ на клетку $(x,\ y+1)$.
- «`D`» — переместиться с клетки $(x,\ y)$ на клетку $(x,\ y-1)$.
- «`R`» — переместиться с клетки $(x,\ y)$ на клетку $(x+1,\ y)$.
- «`L`» — переместиться с клетки $(x,\ y)$ на клетку $(x-1,\ y)$.

Протокол взаимодействия с программой жюри

В самом начале программа жюри сообщает вашей программе натуральное число n ($2 \le n \le 50$) — размер доски.

Далее ваша программа должна повторять следующие ходы, выводя в стандартный поток вывода соответствующее сообщение и переводя строку. Перемещение фишки кодируется строкой «0 *id c*», где *id* — номер фишки, которую ваша программа хочет переместить (1 или 2), а символ *c* — направление движения. После каждого перемещения программа жюри сообщает вашей программе результат попытки перемещения:

- «`1`», если передвижение успешно;
- «`0`», если нет.

Когда ваша программа считает, что определила начальное положение фишек, следует вывести 5 чисел: «1 x_1 y_1 x_2 y_2» ($1 \le x_1, y_1, x_2, y_2 \le n$) — начальное положение первой фишки (x_1, y_1) и второй фишки (x_2, y_2), соответственно. После вывода этой команды ваша программа должна завершиться. Вывод этой команды не считается ходом и не включается в ограничение $6n$ на число ходов.

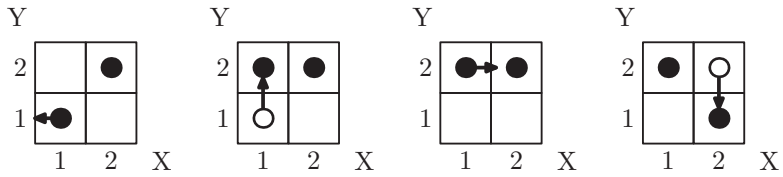
В каждом тесте исходное положение фишек зафиксировано, и программа жюри честно отвечает на запросы вашей программы.

Пример

stdin	stdout
2 0 1 0 1	0 1 L 0 1 U 0 1 R 0 2 D 1 1 1 2 2

Пояснение к примеру

В примере фишки перемещались следующим образом.



Замечание

После каждого действия вашей программы выводите символ перевода строки. Если вы используете «`writeln`» в Паскале, «`cout << ... << endl`» в C++, «`System.out.println`» в Java или «`print`» в Python, сброс потока вывода у вас происходит автоматически, дополнительно делать «`flush`» не обязательно. Если вы используете другой способ вывода, рекомендуется делать «`flush`», но все равно обязательно требуется выводить символ перевода строки.

Ниже приведены наиболее типичные причины получения тех или иных сообщений об ошибке.

Если ваша программа соблюдает протокол, но неверно определяет начальное положение фишек, либо выполняет слишком много ходов, вы получите результат «Wrong Answer».

Если ваша программа выводит некорректно отформатированные сообщения программе жюри, то вы получите результат «Presentation Error», либо «Wrong Answer».

Если ваша программа нарушила протокол и ждет ввода в то же время, когда его ждет и программа жюри, то вы получите результат «Idleness Limit Exceeded». Обратите внимание, что к такому же результату может привести и то, что вы не переводите строку после каждого выведенного сообщения, или выводите не тем способом, который описан в начале раздела, и не делаете «`flush`».

Задача H. Отчёт об ошибках

Имя входного файла:	`log.in`
Имя выходного файла:	`log.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Распечатки стека вызовов функций при ошибках — мощный инструмент отладки программ. Рассмотрим математическую модель взаимодействия между функциями в программе, которая называется *граф вызовов*.

Пусть в программе n функций, которые могут вызывать друг друга. Обозначим все функции в программе числами от 1 до n. Рассмотрим множество E пар чисел (f_i, g_i), где f_i и g_i — номера функций, такие, что f_i вызывает g_i. Размер этого множества будем называть *сложностью графа вызовов*.

Например, рассмотрим пример программы на примитивном языке программирования, в которой есть три функции.

```
function f(x)
    if x > 0 then
        return g(x)
    else
        return h(x)

function g(x)
    return x

function h(x)
    if x == 0 then
        return 1 / x
    else
        return h(x + 1) + 1
```

Пронумеруем функции, пусть f имеет номер 1, g имеет номер 2 и h имеет номер 3. Тогда множество E выглядит следующим образом: $E = \{(1,2),(1,3),(3,3)\}$, так как функция f вызывает функции g и h, функция g не вызывает других функций, а функция h вызывает себя. Сложность графа вызовов этой программы равна 3.

Распечатка стека вызовов при ошибке устроена следующим образом. Пусть в процессе выполнения программы произошла некоторая ошибка. Сначала в распечатку выводится номер функции i_1, в которой произошла ошибка, далее выводится номер функции i_2, из которой непосредственно была вызвана эта функция i_1, далее номер номер функции i_3, из которой была вызвана функция i_2, и так далее.

Пусть, например, в приведенном выше примере программы был выполнен вызов $f(-3)$. Тогда будет выполнен вызов $h(-3)$, который выполнит $h(-2)$, а далее в свою очередь $h(-1)$ и $h(0)$, в последнем вызове произойдет деление на 0. Тогда распечатка стека вызовов будет выглядеть так:

```
3
3
3
3
1
```

Юра попросил Лёшу найти ошибку в его программе, прислав ему распечатку стека вызовов после ошибок. К сожалению, файл, который Юра прислал Лёше, содержит несколько распечаток стеков вызовов при различных ошибках, причем эти распечатки выведены подряд и никак не отделены

друг от друга. При этом Юра утверждает, что ошибки могут происходить только в двух функциях, правда он не помнит, в каких.

Лёша понял, что ограничиться распечаткой стека вызовов не удастся, придется смотреть на программу. Но прежде чем это сделать, он хочет выяснить, какая минимальная сложность может быть у графа вызовов этой программы, если все утверждения Юры верны.

Помогите ему выяснить, какая может быть минимальная сложность графа вызовов программы, чтобы присланный ему файл мог содержать одну или несколько записанных подряд без разделителей распечаток стеков вызовов, а непосредственные ошибки происходили не более чем в двух различных функциях.

Формат входных данных

В первой строке находятся два целых числа n и m ($1 \le n, m \le 100\,000$) — количество функций в программе Юры и количество строк в файле с распечатками стека вызовов, которые Юра прислал Лёше. В следующих m строках находятся по одному целому числу f_i ($1 \le f_i \le n$) — номер функции в i-й строке файла.

Формат выходных данных

В первой строке выведите одно целое число k — минимальную возможную сложность графа вызовов программы Юры. В следующих k выведите по два целых числа a_i и b_i — такая пара означает, что функция с номером a_i может вызывать функцию с номером b_i. Если возможных вариантов графа вызовов несколько, выведите любой.

Пример

log.in	log.out
3 7 1 3 3 2 3 2 1	1 2 3

Пояснение к примеру

В примере возможна ситуация, что ошибки происходят только в функциях 1 и 3, а в приведенном файле содержится пять записанных подряд распечаток стеков вызовов. Ниже приведены те же распечатки, но разделенные пустой строкой.

```
1

3

3
2

3
2

1
```

В этом случае только функция 2 вызывает функцию 3, следовательно сложность графа вызовов равна 1.

Задача I. Обмен валюты

Имя входного файла:	`prices.in`
Имя выходного файла:	`prices.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Феоктист работает в обменном пункте на границе Флатландии и Байтландии. Каждый день он узнает по радио текущий курс обмена Флатландских флатов на Байтландские биты и вывешивает информацию об обменном курсе на дверях своего пункта.

В распоряжении Феоктиста есть n табличек, на которых записаны числа $c_1, c_2, \ldots, c_n$. Узнав сегодняшний курс обмена p, Феоктист выбирает две таблички с значениями c_i и c_j, такими, чтобы значение c_i/c_j было как можно ближе к p, и вывешивает их на двери, формируя таким образом объявление «меняю c_i флатов на c_j битов». Задача не из легких, и Феоктист решил её автоматизировать.

Помогите Феоктисту по заданному курсу p найти две соответствующие таблички.

Формат входных данных

В первой строке входного файла заданы два целых числа n и p ($2 \le n \le 100\,000$, $1 \le p \le 10^9$) — число табличек и текущий курс. Вторая строка содержит n целых чисел c_i ($1 \le c_i \le 10^9$) — числа, записанные на табличках.

Формат выходных данных

Выведите два целых числа i и j ($1 \le i, j \le n$, $i \ne j$) — номера двух табличек, таких что величина $|(c_i/c_j) - p|$ минимальна. Если таких пар несколько, то вы можно вывести любую из них.

Примеры

prices.in	prices.out
3 2 1 6 3	2 3
4 3 2 3 4 5	4 1

Задача J. Простая последовательность цифр

Имя входного файла:	`primeseq.in`
Имя выходного файла:	`primeseq.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

На перемене перед уроком математики Рома решил поупражняться в определении простоты числа. Напомним, что простым называется натуральное число, имеющее ровно два различных натуральных делителя — единицу и самого себя. Сначала он написал на доске первое простое число, после чего справа приписал к нему второе, затем третье и так далее. Всего Рома выписал на доску первые n простых чисел. В результате действий Ромы на доске появилось одно длинное число, которое начинается так: «23571113171923...».

Когда в кабинет вошла Елена Евгеньевна, учительница Ромы, она предложила классу решить следующую задачку: вычеркнуть из написанного на доске числа k цифр так, чтобы оставшееся на доске число было максимальным.

Помогите Роме и одноклассникам решить предложенную задачу, чтобы не получить двойку от строгой учительницы.

Формат входных данных

Входной файл к этой задаче содержит несколько наборов тестовых данных. В первой строке входного файла задано число T — количество наборов в файле.

В следующих T строках идут описания наборов, каждое из которых состоит из двух целых положительных чисел n и k. Гарантируется, что первые n простых чисел содержат в себе хотя бы $k + 1$ цифру суммарно.

Сумма всех n во входном файле не превосходит 400 000.

Формат выходных данных

Для каждого из тестовых наборов в отдельной строке выведите искомое максимальное число для соответствующих n и k.

Пример

primeseq.in	primeseq.out
2 4 2 5 3	57 711

Пояснение к примеру

В первом тесте Рома выписал число 2357. Максимальное число, которое может получиться после вычеркивания из него двух цифр: 57.

Во втором тесте Рома выписал число 235711. Максимальное число, которое может получиться после вычеркивания из него трех цифр: 711.

Задача K. Робот

Имя входного файла:	`robot.in`
Имя выходного файла:	`robot.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Компания «Филипп индастриз» разрабатывает программу для нового робота-марсохода. Участок Марса, на котором будет работать робот, представляет собой квадратное поле размером $n \times n$, разбитое на квадратные участки размером 1×1, некоторые из которых могут содержать скалу ($1 \le n \le 1000$, не более 300 участков содержат скалу).

Введем на поле систему координат таким образом, что участки имеют координаты $(1,1), (1,2), \ldots, (1,n), (2,1), \ldots, (n,n)$. Программа для робота представляет собой последовательность инструкций, каждая из которых кодируется одной латинской буквой:

- «`U`» — переместиться с участка (x, y) на участок $(x, y+1)$.
- «`D`» — переместиться с участка (x, y) на участок $(x, y-1)$.
- «`R`» — переместиться с участка (x, y) на участок $(x+1, y)$.
- «`L`» — переместиться с участка (x, y) на участок $(x-1, y)$.

Для экономии инженеры записывают в память последовательность инструкций s, пронумерованных от 1 до t. Затем можно заставить робота выполнить *подпрограмму* — одну или несколько следующих подряд инструкций. Каждая подпрограмма, таким образом, характеризуется двумя целыми числами (l, r) — номером первой и последней инструкции в подпрограмме.

В процессе лабораторного эксперимента робот был размещен на некотором участке тестового поля. Будем называть подпрограмму (l, r) корректной, если при последовательном выполнении инструкций $s[l], s[l+1], \ldots, s[r]$ робот не покидает поле и не перемещается на участок со скалой.

По описанию поля, программе для робота и его начальному положению определите, сколько у данной программы существует корректных подпрограмм.

Формат входных данных

В первой строке входного файла находятся два числа n и t ($1 \le n \le 1000$, $1 \le t \le 10^5$) — размер поля и количество инструкций в программе робота.

Во второй строке входного файла находится строка s длины t — программа робота. Гарантируется, что строка s состоит только из символов «`U`», «`D`», «`R`» и «`L`».

Следующие n строк содержат по n символов в каждой и задают поле. Символ «`.`» означает, что участок пустой и по нему может перемещаться робот. Символ «`#`» означает, что на участке находится скала. Символ «`@`» означает, что в этой клетке находится стартовая позиция робота. Ось X направлена слева направо, ось Y — снизу вверх. Гарантируется, что символ «`@`» встречается ровно один раз, а символ «`#`» встречается не более 300 раз.

Формат выходных данных

В единственной строке выходного файла выведите количество корректных подпрограмм.

Пример

robot.in	robot.out
4 4 ULUR ..#. .... .#@. #.#.	6

Пояснение к примеру

В примере следующие подпрограммы являются корректными: $(1,1)$=«`U`», $(1,2)$=«`UL`», $(1,3)$=«`ULU`», $(3,3)$=«`U`», $(3,4)$=«`UR`», $(4,4)$=«`R`».