Задача A. Расстановка книг

Имя входного файла:	`books.in`
Имя выходного файла:	`books.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В начале учебного года в одном Великом Университете библиотека выдавала книги своим студентам, чтобы они использовали их для получения познаний в течении года. Студентка Олечка, отстояв огромную очередь и получив книги в библиотеке, с великим трудом донесла их до своего дома. Теперь Олечка хочет расставить эти научные труды на свою книжную полку.

К сожалению, с восприятием содержимого книжной полки своей хозяйкой не так все просто — Олечку приводят в уныние книги, стоящие на полке по соседству и имеющие в общей сложности количество страниц, превышающее некоторое значение.

Говоря более точно, Олечка сильно расстраивается, если существует пара книг, стоящих на местах i и $(i+1)$ полки, и общее число станиц в этих двух книгах превышает a_i.

Помогите бедной Олечке понять, сколько способов существует расставить книги на полке так, чтобы она не расстраивалась.

Формат входного файла

В первой строке входного файла записано число n $(1 \le n \le 9)$ — число Олечкиных книг. В второй строке находится n чисел p_i $(1 \le p_i \le 10^9)$ — количество страниц в каждой из книг. В третьей строке находится $n-1$ число a_i $(1 \le a_i \le 2 \cdot 10^9)$ — допустимое общее количество страниц в книгах, стоящих на местах i и $i+1$ полки.

Формат выходного файла

В выходной файл выведите количество способов расставить книги на полке.

Примеры

books.in	books.out
3 1 2 1 2 3	2
3 1 1 2 1 2	0

Задача B. Кодовый замок

Имя входного файла:	`code.in`
Имя выходного файла:	`code.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Вася очень любит использовать кодовые замки. Как известно, кодовый замок состоит из барабана с несколькими кольцами. Причем, у каждого кольца есть несколько положений, каждое из которых соответствует некоторой цифре в k-ичной системе счисления. Таким образом, любое состояние всего кодового замка с n кольцами можно представить некоторым n-значным числом в k-ичной системе счисления. При этом замок открывается только в одном фиксированном состоянии.

Так как код замка очень легко забыть, для его запоминания Вася пользуется хитрым способом. После закрытия замка Вася поворачивает замок таким образом, чтобы число, кодирующее его состояние, было предыдущим (в порядке возрастания) числом, имеющим ровно такую же сумму цифр, что и то число, при котором замок открывается.

Тогда для открытия замка Васе достаточно найти следующее (в порядке возрастания) число с той же суммой цифр в k-ичной системе счисления, что и текущее, и это число откроет замок.

Но, после месяца использования такого замка, Васе надоело каждый раз решать вручную такую непростую задачу, и он попросил Вас помочь ему.

Требуется написать программу, которая будет по заданному числу m в k-ичной системе счисления находить следующее число (в порядке возрастания) в этой же системе счисления с такой же суммой цифр.

Формат входного файла

В первой строке входного файла задано два натуральных числа k и n ($2 \leq k \leq 36$). Во второй строке задано число m в k-ичной системе счисления, при этом число m состоит из n цифр и не содержит ведущих нулей. Цифрам от 10 до 35 соответствуют соответственно заглавные латинские буквы от A до Z. Число m не превосходит 100000.

Формат выходного файла

В выходной файл выведите ответ на задачу — искомое число в k-ичной системе счисления из n цифр без ведущих нулей. Если Вася где-то ошибся, и искомого числа не существует, то выведите в выходной файл единственное слово «Impossible».

Примеры

code.in	code.out
10 2 23	32
16 2 FF	Impossible
26 2 HP	IO

Задача C. Даты

Имя входного файла:	`dates.in`
Имя выходного файла:	`dates.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Вася конструирует свой собственный автоматический электронный календарь, который будет отображать текущую дату в формате «ДД.ММ.ГГГГ». Для этого ему нужно уметь отображать различные цифры. На данный момент его календарь умеет отображать k различных цифр. Поскольку добавление поддержки каждой новой цифры — процесс очень трудоемкий, то Вася решил, что если его календарь будет отображать довольно большое количество дат, то он будет доволен. Поэтому на данный момент Васю крайне интересует следующий вопрос: сколько дней из данного промежутка его календарь будет отображать дату.

Например, если Васин календарь может отображать только цифры 0, 2 и 9, то он сможет отобразить дату «второе февраля 2009 года» и не сможет отобразить дату «третье февраля 2009 года».

Стоит отметить, что программировать Вася умеет неплохо, поэтому он не забывает, что существует такое понятие, как високосный год, в котором в феврале 29 дней. Напомним, что год является високосным, если его номер кратен 4 и при этом не кратен 100, либо кратен 400.

Формат входного файла

Первая строка входного файла состоит из одного целого числа k ($0 \le k \le 10$). Во второй строке входного файла перечислены через пробел k различных цифр, которые Васин календарь умеет отображать. Следующие две строки содержат описание первой и последней даты интересующего его промежутка в формате «ДД.ММ.ГГГГ».

Формат выходного файла

В выходной файл выведите одно целое число — количество дней в промежутке включая концы, в которые календарь может показать дату.

Примеры

<table>
<tr><th>dates.in</th><th>dates.out</th></tr>
<tr><td>3
0 2 9
02.02.2009
03.02.2009</td><td>1</td></tr>
<tr><td>3
0 2 9
01.02.2009
28.02.2009</td><td>4</td></tr>
</table>

Задача D. Оптическое распознавание символов

Имя входного файла:	`ocr.in`
Имя выходного файла:	`ocr.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Всемирно известная компания *Maple* готовится к выпуску новейшего мобильного телефона *myPhone*. Среди прочих возможностей в myPhone будет функция сканирования текста с помощь встроенной фотокамеры.

Телефон должен распознавать n различных символов. Образец символа представляется в виде прямоугольной таблицы размера $w \times h$ ячеек. Каждая ячейка содержит 1, если в этом месте написания символа должны быть чернила, и 0 в обратном случае.

Сканирующая программа с помощью сложных алгоритмов разбивает сфотографированное изображение на прямоугольники размера $w \times h$ пикселей (при этом каждый пиксель считается либо черным, и тогда там стоит 1, либо белым, чему соответствует 0) и сравнивает их с образцами символа.

Для оценки сравнения специальный отдел компании Maple по *Критериям Оценки Шрифтов Конечными Автоматами* разработал величину, называемую *похожестью* двух таблиц. Похожесть — число ячеек, таких что в образце и изображении в соответствующей ячейке наблюдаются одинаковые значения.

Вам необходимо написать программу, которая по набору символов и изображению находила бы символ, наиболее похожий на изображение по критерию *КОШКА*, то есть символ с наибольшей похожестью.

Формат входного файла

В первой строке входного файла три натуральных числа n, w и h ($n, w, h \le 100$). Далее следует n блоков, описывающих образцы символов. Каждый блок состоит из h строк из нулей и единиц по w символов в каждой. Далее следует изображение в аналогичном формате.

Формат выходного файла

В вывходном файле должно быть одно число — номер наиболее похожего символа. Символы нумеруются с единицы в порядке появления во входном файле. Если ответов несколько, выведите любой из них.

Примеры

ocr.in	ocr.out

`2 2 2`
`11`
`11`
`00`
`00`
`00`
`01`
	`2`

`2 2 2`
`11`
`11`
`00`
`00`
`01`
`10`
	`1`

Задача E. Последовательность

Имя входного файла:	`seq.in`
Имя выходного файла:	`seq.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Задана последовательность из n чисел a_1, a_2, $\ldots$, a_n. *Подпоследовательностью* длины k этой последовательности называется набор индексов i_1, i_2, $\ldots$, i_k, удовлетворяющий неравенствам $1 \le i_1 < i_2 < \ldots < i_k \le n$. Подпоследовательность называется *возрастающей*, если выполняются неравенства $a_{i_1} < a_{i_2} < \ldots < a_{i_k}$.

Широко известна задача поиска максимальной возрастающей подпоследовательности, однако, Вам предлагается решить другую задачу: найти максимальную последовательность данной последовательности, которая не является возрастающей.

Формат входного файла

В первой строке входного файла находится число n $(1 \le n \le 100)$ — число элементов последовательности. В второй строке находится n чисел a_i $(0 \le a_i \le 1000)$ — элементы последовательности.

Формат выходного файла

В первой строке выходного файла выведите k длину максимальной не являющейся возрастающей последовательности или 0, если такой не существует. В случае, если искомая подпоследовательность существует, во второй строке выведите k чисел i_j — набор индексов подпоследовательности.

Примеры

seq.in	seq.out
3 3 2 1	3 1 2 3
2 1 2	0

Задача F. Налогообложение

Имя входного файла:	`taxes.in`
Имя выходного файла:	`taxes.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Одна малоизвестная, но амбициозная компания разрабатывает масштабный проект, цель которого упростить подсчет бюджета рядового гражданина страны Флатландии. Вы участвуете в этом проекте и вам поручили написать программу, которая вычисляет подоходный налог, то есть налог, взимаемый с получаемого дохода.

Правила вычисления подоходного налога довольно сложны и состоят из $k+1$ части:

- Если доход (X) менее U_1, то налог составляет p_1 процентов, то есть $X \cdot \frac{p_1}{100}$.
- Если X не менее U_1 и меньше U_2, то налог с U_1 составляет p_1, а с оставшейся суммы — p_2. Таким образом суммарный налог составляет $U_1 \cdot \frac{p_1}{100} + (X - U_1) \cdot \frac{p_2}{100}$.
- ...
- Если X не менее U_k, то налог с предыдущих частей остается прежним, а с остатка составляет p_{k+1} процентов, то есть суммарный налог составляет $U_1 \cdot \frac{p_1}{100} + (U_2 - U_1) \cdot \frac{p_2}{100} + \ldots + (X - U_k) \cdot \frac{p_{k+1}}{100}$.

Ваша задача — по данному доходу X вычислить подоходный налог.

Формат входного файла

Первая строка входного файла содержит два целых числа X и k ($0 \le X \le 10^9$, $1 \le k \le 100$). Вторая строка входного файла содержит k целых чисел $U_1, U_2, \ldots, U_k$ ($1 \le U_1 < U_2 < \ldots < U_k \le 10^9$). Третья строка входного файла содержит $k+1$ целое число $p_1, p_2, \ldots, p_k, p_{k+1}$ ($0 \le p_1 \le p_2 \le \ldots \le p_{k+1} \le 100$).

Формат выходного файла

В единственной строке выходного файла выведите подоходный налог, который должен заплатить честный гражданин Флатландии, заработавший X единиц флатландской валюты. Ответ следует выводить ровно с двумя знаками после запятой.

Примеры

taxes.in	taxes.out
1500 2 1000 2000 10 20 30	200.00

Задача G. Трамваи

Имя входного файла:	`trams.in`
Имя выходного файла:	`trams.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Как известно, еще несколько лет назад Санкт-Петербург был самым трамваизированным городом в мире. Но, к сожалению, в Санкт-Петербурге количество трамвайных линий постепенно уменьшается и уже несколько лет он уступает Мельбурну.

В славной стране Байтландии, а именно в ее столице БитСити, власти города поняли преимущества данного вида транспорта и решили серьезно расширить трамвайную сеть в городе. Но, для того, чтобы потратить на расширение как можно меньше средств из городского бюджета, для начала они решили провести небольшое статистическое исследование для каждой остановки.

Исследование заключается в следующем — для остановки записывается время прихода трамваев на нее и время прихода пассажиров. Причем, в Байтландии у пассажиров на остановке принято становится в очередь и заходить в трамваи по мере их прихода в порядке этой самой очереди. К сожалению, у трамваев ограниченная вместимость, и потому, возможно, некоторые пассажиры не смогут уехать на первом после их прихода трамвае.

Необходимо для каждого пассажира выяснить, на каком трамвае он уехал, чтобы потом можно было легко понять, сколько он ждал трамвая.

Для упрощения модели будем считать, что и трамваи и люди приходят в целые моменты времени, записанные с точностью до минуты. Причем, если пассажир подходит к остановке в ту же минуту, в которую туда приезжает трамвай, то пассажир может успеть на этот трамвай (конечно же, если там еще будут места). Трамвай стоит на остановке ровно минуту и сразу же после этого отправляется. Никакие два трамвая не приходят в одну минуту.

Также считается, что в никакую минуту не приходит более одного пассажира и в конце дня все оставшиеся пассажиры уходят с остановки, так и не дождавшись трамвая.

Формат входного файла

В первой строке входного файла заданы три целых числа n, m и k ($1 \le n, m, k \le 1000$) — соответственно число записей о приходе трамваев на остановку, число записей о приходе пассажиров на остановку и максимальная вместимость трамвая.

В следующих n строках задано время прихода трамваев в формате «`hh:mm`». Время изменяется от `00:00` до `23:59` и все события прихода трамваев даются в порядке возрастания времени.

В следующих m строках в аналогичном формате заданы времена прихода пассажиров в порядке возрастания времени.

Формат выходного файла

В m строках выходного файла выведите ответ на задачу.

В i-ой строке выведите одно число — номер трамвая, на котором уедет i-й в порядке появления во входном файле пассажир. Если же пассажир так и не уедет с остановки, выведите -1.

Трамваи нумеруются с 1 в порядке появления во входном файле.

Примеры

trams.in	trams.out
2 2 2 08:00 11:00 09:00 10:00	2 2
2 2 1 10:30 10:31 09:00 10:00	1 2

Задача H. Обобщенные числа-близнецы

Имя входного файла:	`twins.in`
Имя выходного файла:	`twins.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В теории чисел *простыми числами-близнецами* называют пару таких простых чисел (p, q), что $q-p=2$. Например, пары $(3, 5)$ и $(11, 13)$ являются парами простых чисел-близнецов. Назовем *обобщенными числами-близнецами* пару простых чисел (p, q), где $q-p=k$, k — некоторое натуральное число. Например, для $k=4$ пара $(3, 7)$ является парой обобщенных чисел-близнецов.

Существует предположение, что пар простых чисел-близнецов бесконечно много, однако это не доказано. Безусловно, выяснить по заданному k, сколько пар обобщенных близнецов содержит множество всех натуральных чисел, не менее сложная задача, чем аналогичная о простых близнецах.

Ваша же задача несколько проще — выяснить по заданному k, сколько пар обобщенных близнецов содержит множество натуральных чисел от 1 до n.

Формат входного файла

Во первой строке входного файла через пробел заданы два натуральных числа n и k $(1 \le n, k \le 10^4)$.

Формат выходного файла

В выходной файл выведите число пар простых чисел (p, q), таких, что $1 \le p < q \le n$ и $q-p=k$.

Примеры

twins.in	twins.out
17 2	3
10000 1	1
20 7	0