Задача A. Катание на автобусах

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

В городе n автобусных остановок, через которые проходят k кольцевых автобусных маршрутов. Каждый маршрут задается списком номеров остановок, через которые он проходит, i-ый маршрут проходит по остановкам $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$ (в этом порядке). По маршруту ходит ровно один автобус. В момент времени 0 этот автобус находится на остановке $a_{i,1}$. На то, чтобы доехать до следующей на своем маршруте остановки, автобус тратит ровно одну минуту. Временем стоянки автобуса на остановке можно пренебречь. Все маршруты кольцевые, то есть через минуту после остановки a_{i,l_i} автобус оказывается на остановке $a_{i,1}$ и едет по маршруту еще раз.

Несколько человек в этом городе решили покататься на автобусах. При этом каждый из них составил план своего катания. План j-го человека состоит из остановки b_j, на которой человек начнет свое катание и последовательности чисел $c_{j,1}, c_{j,2}, \ldots, c_{j,m_j}$. Эти числа означают следующее: в момент времени 0 человек придет на остановку b_j и дождется ближайшего автобуса (если в этот момент какой-то автобус находится на остановке b_j, человек сядет в него). На этом автобусе он проедет $c_{j,1}$ остановок, после чего выйдет и дождется следующего автобуса на той остановке, где он окажется. На нем он проедет $c_{j,2}$ остановок, снова выйдет и снова дождется следующего автобуса. И так далее. Если в какой-то момент к остановке подъедет сразу несколько автобусов, то человек сядет в автобус с минимальным номером маршрута. Когда человек выходит из автобуса на какой-то остановке, он может уехать с этой остановки не раньше, чем через минуту.

Для каждого человека определите, через сколько минут после начального момента и на какой остановке закончится его катание.

Формат входных данных

Во входном файле записано сначала число n, затем число k. Далее записано k строк, задающих автобусные маршруты. Каждая строка начинается с числа l_i, задающего длину маршрута, затем идет список остановок, через которые проходит маршрут: $a_{i,1}, a_{i,2}, \ldots a_{i,l_i}$. Маршрут может несколько раз проходить через одну и ту же остановку.

Далее идет число p — количество людей, и затем p строк, задающих планы людей. Каждая строка содержит сначала числа b_j — номер начальной остановки и m_j — количество чисел в последовательности. Затем идут числа $c_{j,1}, c_{j,2}, \ldots, c_{j,m_j}$.

Все числа во входном файле натуральные и не превышают 50.

Формат выходных данных

В выходной файл для каждого человека выведите два числа: время в минутах, когда закончится его катание, и номер остановки, на которой это произойдет. Если же человек не сможет реализовать свой план до конца (на какой-либо остановке он не дождется автобуса), выведите для него два нуля.

Пример

buses.in	buses.out
6 4 4 1 2 3 5 2 3 4 5 5 2 1 3 2 2 4 3 3 1 4 1 2 3 4 2 1 1 6 3 1 2 3	20 1 2 3 0 0

Задача B. Старая крепость

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

В одной далекой стране ученые обнаружили странное скопление камней. Изучив его, ученые пришли к выводу, что это части старой крепостной стены, имевшей форму окружности. К сожалению, время и вандалы разрушили некоторые части стены.

Чтобы защитить оставшиеся фрагменты стены и продолжить их изучение в спокойной обстановке, ученые хотят обнести фрагменты стены забором из колючей проволоки. Если сделать отдельный забор для каждого фрагмента, будет неудобно переходить от одного фрагмента к другому, поэтому ученые хотят сделать один общий забор, окружающий все фрагменты.

Помогите ученым посчитать минимальную возможную длину забора, чтобы они знали, сколько просить колючей проволоки.

Формат входных данных

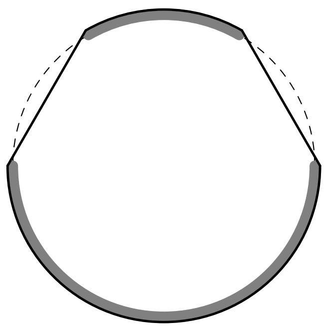
Во входном файле задано два натуральных числа: число фрагментов n ($1 \le n \le 180$) и радиус крепости r ($1 \le r \le 100$). Далее следует n пар целых чисел, описывающих сохранившиеся фрагменты стены: a_i, b_i — углы в градусах, соответствующие началу и концу фрагмента. Углы отмеряются от направления на север из центра крепости, против часовой стрелки ($0 \le a_i, b_i < 360$, $a_i \ne b_i$). Каждый фрагмент от начального угла к конечному также проходится против часовой стрелки. Фрагменты не имеют общих точек.

Формат выходных данных

Выведите минимальную возможную длину забора. Ответ должен отличаться от правильного не более, чем на 10^{-3}.

Пример

castle.in	castle.out
2 10 330 30 90 270	61.8879020479



Задача C. Гистограмма

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

Вовочка ломает систему безопасности Пентагона. Для этого ему понадобилось узнать, какие символы в секретных зашифрованных посланиях употребляются чаще других. Для удобства изучения Вовочка хочет получить графическое представление встречаемости символов. Поэтому он хочет построить гистограмму количества символов в сообщении. Гистограмма — это график, в котором каждому символу, встречающемуся в сообщении хотя бы один раз, соответствует столбик, высота которого пропорциональна количеству этих символов в сообщении.

Формат входных данных

Входной файл содержит зашифрованный текст сообщения. Он содержит строчные и прописные латинские буквы, цифры, знаки препинания («`.`», «`!`», «`?`», «`:`», «`-`», «`,`», «`;`», «`(`», «`)`»), пробелы и переводы строк. Размер входного файла не превышает 10^4 байт. Текст содержит хотя бы один непробельный символ. Все строки входного файла не длиннее 200 символов.

Формат выходных данных

Для каждого символа c кроме пробелов и переводов строк выведите столбик из символов «`#`», количество которых должно быть равно количеству символов c в данном тексте. Под каждым столбиком напишите символ, соответствующий ему. Отформатируйте гистограмму так, чтобы нижние концы столбиков были на одной строке, первая строка и первый столбец были непустыми. Не отделяйте столбики друг от друга. Отсортируйте столбики в порядке увеличения кодов символов.

Пример

<table>
<tr><th>chart.in</th><th>chart.out</th></tr>
<tr><td><pre>Hello, world!</pre></td><td><pre>   #
     ##
#########
!,Hdelorw</pre></td></tr>
<tr><td><pre>Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.</pre></td><td><pre>   #
         #
         #
         #
         #
         #   #
         # #   #
      # # ### ####
      ## ###### ####
      ##############
      ############## ##
###
########################
,.;ADTabdeghilmnorstuvwy</pre></td></tr>
</table>

Задача D. Два коня

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

Петя учится играть в шахматы. Недавно он заметил, что несмотря на то, что кони умеют прыгать через фигуры, они могут мешать друг другу дойти до нужных клеток. Петя поставил на шахматную доску размером 8×8 двух коней: черного и белого, и для каждого их них выбрал клетку, на которой он хочет его видеть. Теперь ему интересно, какое минимальное число ходов потребуется коням, чтобы дойти до нужных клеток.

Кони ходят по шахматным правилам (на одну клетку по горизонтали и две по вертикали или на одну клетку по вертикали и на две по горизонтали). Порядок ходов черного и белого коня может быть произвольным. Коням не разрешается одновременно вставать на одну и ту же клетку.

Формат входных данных

Во входном файле записаны четыре клетки шахматной доски в следующем порядке: начальное положение белого коня, начальное положение черного коня, конечное положение белого коня, конечное положение черного коня. Клетка шахматной доски задается горизонталью (буква от «`a`» до «`h`») и вертикалью (цифра от 1 до 8), не разделенными пробелами. Описания клеток отделяются друг от друга одним пробелом.

Гарантируется, что исходно кони находятся на различных клетках, и в конце кони также должны оказаться на различных клетках.

Формат выходных данных

Выведите в первой строке выходного файла количество необходимых ходов. Далее выведите последовательность ходов. Ход описывается следующим образом: буква, соответствующая цвету коня («`W`» для белого или «`B`» для черного) и клетка, на которую нужно пойти. Клетку выведите в таком же формате, как во входном файле.

Если искомой последовательности ходов не существует, выведите на первой строке выходного файла число -1.

Пример

knights.in	knights.out
a1 a2 a2 a1	6 W b3 W c1 B b4 W a2 B c2 B a1

Задача E. Сапер

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

Мальчику Васе очень нравится известная игра «Сапер». В нее играет один человек. Игра идет на клетчатом поле размером $m \times n$ (m строк, n столбцов). В некоторых клетках поля стоят мины. В каждой из остальных клеток записано либо число от 1 до 8 — количество мин в соседних с ней клетках, либо ничего не написано — это означает, что в соседних клетках мин нет. Клетки являются соседними, если они имеют хотя бы одну общую вершину. В одной клетке не может стоять более одной мины. Будем называть поле с расположенными на нем минами и числами *картой*.

Изначально все клетки поля закрыты. Игрок за один ход может открыть какую-нибудь клетку. После этого игроку показывается содержимое этой клетки, и если в открытой им клетке оказывается мина, он проигрывает. В противном случае игра продолжается. Цель игры — открыть все клетки, в которых нет мин.

У Васи на компьютере есть эта игра, но ему кажется, что все карты, которые в ней есть, некрасивые и неинтересные. Поэтому он решил нарисовать свои. При этом он хочет, чтобы карты, которые он нарисует, после того, как они будут открыты, выглядели красиво.

У Васи есть рисунки, нарисованные на клетчатой бумаге следующим образом: некоторые клетки закрашены в черный цвет, а некоторые оставлены белыми. Вася хочет по каждому такому рисунку сделать соответствующее ему поле для игры в «Сапера» по следующему правилу: если на рисунке клетка покрашена в черный цвет, то на этом месте должна быть либо мина, либо число от 1 до 8, если же клетка оставлена белой, то на игровом поле она должна быть пустой.

Напишите программу, которая сделает это за Васю.

Формат входных данных

В первой строке входного файла содержатся числа m и n ($1 \le m, n \le 100$) — количество строк и столбцов соответственно. Далее идет таблица из m строк, по n чисел в каждой строке, задающая Васин рисунок. Каждое число в таблице равно 0 или 1, число 0 означает, что соответствующая клетка на рисунке белая, 1 — черная. Числа в строках разделяются пробелами.

Формат выходных данных

Выходной файл должен содержать m строк по n символов — карту игрового поля, j-ый символ i-ой строки должен содержать символ «`*`» (звездочка) если в клетке (i, j) стоит мина, цифру от 1 до 8, если в этой клетке стоит соответствующее число, либо «`.`» (точка), если клетка (i, j) пустая. Символы пробелами не разделяйте. Если решений несколько, выведите любое из них.

Если построить поле, соответствующее рисунку, невозможно, выходной файл должен содержать одну строку с сообщением «`No solution`».

Пример

<table>
<tr><th>mines.in</th><th>mines.out</th></tr>
<tr><td><pre>3 4
0 1 1 1
1 1 1 1
1 1 0 0</pre></td><td><pre>.1*1
1211
*1..</pre></td></tr>
<tr><td><pre>3 3
0 1 0
1 0 1
0 1 0</pre></td><td><pre>No solution</pre></td></tr>
</table>

Задача F. Интересное число

Имя входного файла:	`number.in`
Имя выходного файла:	`number.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	64 мегабайта

Для заданного числа n найдите наименьшее положительное целое число с суммой цифр n, которое делится на n.

Формат входных данных

Во входном файле содержатся целое число n $(1 \le n \le 1000)$.

Формат выходных данных

Выходной файл должен содержать искомое число. Ведущие нули выводить не разрешается.

Пример

number.in	number.out
1	1
10	190

Задача G. Треугольник Паскаля

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

Треугольник Паскаля — это бесконечный треугольник из чисел, который имеет следующий вид:

$$\begin{array}{ccccccccccccccccc}
&&&&&&&&1&&&&&&&& \\
&&&&&&&1&&1&&&&&&& \\
&&&&&&1&&2&&1&&&&&& \\
&&&&&1&&3&&3&&1&&&&& \\
&&&&1&&4&&6&&4&&1&&&& \\
&&&1&&5&&10&&10&&5&&1&&& \\
&&1&&6&&15&&20&&15&&6&&1&& \\
&1&&7&&21&&35&&35&&21&&7&&1& \\
\ldots&&\ldots&&\ldots&&\ldots&&\ldots&&\ldots&&\ldots&&\ldots&&\ldots
\end{array}$$

Строки треугольника Паскаля нумеруются с нуля, числа в каждой строке также нумеруются с нуля. Нулевая строка содержит единственное число — единицу, а каждая следующая содержит на одно число больше, чем предыдущая. Нулевое и последнее число в каждой строке равны единице, а каждое из остальных равно сумме двух чисел предыдущей строки, расположенных над ним.

Таким образом, i-ая строка содержит $i+1$ число. Если обозначить j-ый элемент i-ой строки как $a_{i,j}$, то выполняется равенство $a_{i,j} = a_{i-1,j-1} + a_{i-1,j}$. Заметим, что это равенство выполняется и для крайних элементов, если положить отсутствующие элементы предыдущей строки (элементы с номерами -1 и i) равными нулю.

Коля хочет узнать, сколько нечетных чисел в n-ой строке треугольника Паскаля. Он начал рисовать треугольник, но очень скоро тот перестал помещаться на листочек. Тогда Коля решил сделать это с помощью компьютера. Помогите ему.

Формат входных данных

Во входном файле содержится число n ($0 \le n \le 2 \cdot 10^9$).

Формат выходных данных

Выходной файл должен содержать одно число — количество нечетных чисел в n-ой строке треугольника Паскаля.

Пример

pascal.in	pascal.out
0	1
5	4
7	8

Задача H. Два прямоугольника

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

Недавно один известный художник-абстракционист произвел на свет новый шедевр — картину «Два черных непересекающихся прямоугольника». Картина представляет собой прямоугольник $m \times n$, разбитый на квадраты 1×1, некоторые из которых закрашены любимым цветом автора — черным. Федя — не любитель абстрактных картин, однако ему стало интересно, действительно ли на картине изображены два непересекающихся прямоугольника. Помогите ему это узнать. Прямоугольники не пересекаются в том смысле, что они не имеют общих клеток.

Формат входных данных

Первая строка входного файла содержит числа m и n ($1 \le m, n \le 200$). Следующие m строк содержат описание рисунка. Каждая строка содержит ровно n символов. Символ «`.`» обозначает пустой квадрат, а символ «`#`» — закрашенный.

Формат выходных данных

Если рисунок можно представить как два непересекающихся прямоугольника, выведите в первой строке «`YES`», а в следующих m строках выведите рисунок в том же виде, в каком он задан во входном файле, заменив квадраты, соответствующие первому прямоугольнику на символ «`a`», а второму — на символ «`b`». Если решений несколько, выведите любое.

Если же этого сделать нельзя, выведите в выходной файл «`NO`».

Пример

rect.in	rect.out
4 6 .###.. .###.. ...##.	YES .aaa.. .aaa.. ...bb. ......
3 6 .###.. .####. ..###.	NO
4 5 .#### .#### .####	YES .abbb .abbb .abbb .....
3 3 ... .#. ...	NO
3 3 ... #.# ..#	YES ... b.a ..a

Задача I. Цветные нули

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

Толик только что узнал, что на свете существует двоичная система счисления. Обрадованный этим, он записал в столбик двоичные формы чисел 1, 2, ..., n. Получились числа 1, 10, 11, 100, 101, 110, 111, ...

После этого он стер все написанные единицы и стал изучать расположение нулей. Он выбрал число k и в каждой строке, идя слева направо, выделил красным цветом каждый k-ый ноль, начиная с первого. Таким образом, оказались выделенными нули с номерами $1, k+1, 2k+1, \ldots$ Например если $k = 2$, $n = 56$ то получились бы такие строки:

1	1 **0** 0 **0**	1 1 1 1	1 **0** 1 1 0	1 1 1 **0** 1	1 **0** 0 1 **0** 0	1 **0** 1 0 1 1	1 1 **0** 0 1 **0**
1 **0**	1 **0** 0 1	1 **0** 0 **0** 0	1 **0** 1 1 1	1 1 1 1 **0**	1 **0** 0 1 **0** 1	1 **0** 1 1 0 **0**	1 1 **0** 0 1 1
1 1	1 **0** 1 0	1 **0** 0 **0** 1	1 1 **0** 0 **0**	1 1 1 1 1	1 **0** 0 1 1 **0**	1 **0** 1 1 0 1	1 1 **0** 1 0 **0**
1 **0** 0	1 **0** 1 1	1 **0** 0 1 **0**	1 1 **0** 0 1	1 **0** 0 **0** 0 **0**	1 **0** 0 1 1 1	1 **0** 1 1 1 0	1 1 **0** 1 0 1
1 **0** 1	1 1 **0** 0	1 **0** 0 1 1	1 1 **0** 1 0	1 **0** 0 **0** 0 1	1 **0** 1 0 **0** 0	1 **0** 1 1 1 1	1 1 **0** 1 1 0
1 1 **0**	1 1 **0** 1	1 **0** 1 0 **0**	1 1 **0** 1 1	1 **0** 0 **0** 1 0	1 **0** 1 0 **0** 1	1 1 **0** 0 **0** 0	1 1 **0** 1 1 1
1 1 1	1 1 1 **0**	1 **0** 1 0 1	1 1 1 **0** 0	1 **0** 0 **0** 1 1	1 **0** 1 0 1 **0**	1 1 **0** 0 **0** 1	1 1 1 **0** 0 **0**

(красные нули выделены жирным шрифтом и подчеркнуты)

Теперь Толику интересно, сколько же ноликов он выделил. Помогите ему их посчитать.

Формат входных данных

Во входном файле содержатся числа n и k ($1 \le n < 2^{31}$, $1 \le k \le 30$).

Формат выходных данных

Выходной файл должен содержать одно число — количество красных нулей.

Пример

zeroes.in	zeroes.out
4 1	3
56 2	74