Задача A. Азбучный путь

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

Вам дан прямоугольный лабиринт из символов, в котором каждая буква от «A» до «Z» встречается ровно один раз, а в оставшихся местах стоят точки («.»). Азбучным путём называется последовательность из 26-ти клеток лабиринта, в которой:

- В первой клетке последовательности стоит буква «A».
- Первая и вторая клетки — соседние (по горизонтали либо по вертикали).
- Во второй клетке стоит буква «B».
- Вторая и третья клетки — соседние. ...
- 25-я и 26-я клетки — соседние.
- В 26-й клетке стоит буква «Z».

По заданному лабиринты вам требуется определить, есть ли в нём такой путь.

Формат входного файла

В первой строчке входного файла вам даны два числа (n и m), это высота и ширина лабиринта, соответственно. Каждая из следующих n строчек содержит ровно m, эти n строчек задают сам лабиринт.

Ограничения:

- n и m — натуральные числа, не превосходящие 50.
- В лабиринте встречаются только буквы (от «A» до «Z») и точки («.»).
- Каждая буква встречается в лабиринте ровно один раз.

Формат выходного файла

В выходной файл выведите одну строку: «YES», если азбучный путь существует и «NO» — если нет.

Примеры

path.in	path.out
1 26 ABCDEFGHIJKLMNOPQRSTUVWXYZ	YES

path.in	path.out
4 8 ADEHI..Z BCFGJK.Y .PONML.X .QRSTUVW	YES
1 26 ACBDEFGHIJKLMNOPQRSTUVWXYZ	NO
4 10 ABC....... ...DEFGHIJ TSRQPONMLK UVWXYZ....	NO
11 14 ....DEFGHIJK... ...C......L... ...B......M... ...A......N... ....O... ..ZY..TSRQP... ...XWVU.......	YES

Задача B. Подсчёт последовательностей

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

Вам даны четыре больших числа: a, b, c и d. Числа a и b задают арифметическую прогрессию, которая состоит из всех чисел вида $a + b * x$, где x — произвольное целое неотрицательное число. Аналогично, c и d задают геометрическую прогрессию, состоящую из всех чисел вида $c * d^y$, где y — тоже произвольное целое неотрицательное число.

Ещё вам дано большое число $upperBound$. Подсчитайте, сколько чисел в промежутке от 1 до $upperBound$ (включительно) принадлежат хотя бы одной из указанных последовательностей.

Формат входного файла

В первой и единственной строке входного файла содержится пять чисел: a, b, c, d, $upperBound$, именно в таком порядке. Ограничения:

- a, b, c, $upperBound$ — целые числа, не большие $1\,000\,000\,000\,000 (10^{12})$, но не меньшие 1.
- d — целое число от 1 до $100\,000 (10^5)$ включительно.

Формат выходного файла

В выходной файл выведите одно число: ответ на задачу.

Примеры

series.in	series.out
1 1 1 2 1000	1000
3 3 1 2 1000	343
40 77 40 100000 40	1
452 24 4 5 600	10
234 24 377 1 10000	408

Задача C. Скинь монетку

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

Дан прямоугольник, разделённый на клетки 1×1. Каждая клетка либо пуста, либо в ней лежит монетка. Можно применять последовательно следущие операции: выбрать одно из направлений (верх, низ, лево, право) и сдвинуть все монетки на 1 по этому направлению. Если при этом какие-то монетки окажутся за пределами прямоугольника, то они упадут и исчезнут.

Требуется применить такие операции, чтобы осталось ровно k монеток.

Формат входного файла

В первой строке входного файла содержатся три числа: n, m и k ($1 \leqslant n, m \leqslant 30, 1 \leqslant k \leqslant 900$). Это высота прямоугольника, его ширина, и требуемое количество оставшихся монеток. В следующих n строках содержится описание прямоугольника: в j-м символе $(i+1)$-й строки стоит символ «o», если в j-м столбце i-й строки прямоугольника лежит монетка, иначе там стоит символ «.».

Формат выходного файла

Выведите одно число: минимальное количество операций, которое нужно применить, чтобы осталось ровно k монеток. Если это невозможно, выведите -1.

Примеры

coins.in	coins.out
3 4 3 .o.. oooo ..o.	2
6 6 12 .....o ..... oooooo oooooo ..... o.....	3
4 4 3 .... .oo. .oo.	-1
5 7 12 ....ooo.. ooooooo .oo.oo. oo...oo	4

coins.in	coins.out
12 17 58 .... .ooooooo...oooo.. .ooooooo..oooooo. .oo.......oo..oo. .oo.......oo..oo. .ooooo.....oooo.. .ooooooo...oooo.. .....ooo..oo..oo. ......oo..oo..oo. .ooooooo..oooooo. .oooooo....oooo..	6

Задача D. Дорога после дождя

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

Фокс Киэль собирается навестить своих друзей. Для этого ей нужно пойти по дороге, вымощенной квадратными плитками размера 1×1. Дорога имеет длину в n плиток и ширину в две плитки. Если представить, что дорога идёт слева направо, то она будет представлять из себя таблицу из двух строчек и n столбцов. Строчки пронумерованы от 0 до 1 сверху вниз, а столбцы — от 0 до $n-1$ слева направо. Фокс начинает в нулевом столбце нулевой строки, а попасть ей нужно в $(n-1)$-й столбец той же нулевой строки.

Фокс может шагать с плитки на соседнюю плитку. Две плитки называются соседними, если у них есть хотя бы одна общая точка (сторона или угол).

Вчера шёл дождь, так же на некоторых плитках лужи. Фокс не хочет промочить ноги, поэтому не может вставать на такие плитки. Вам дано описание дороги, требуется узнать, может ли Фокс дойти до друзей, не наступив в лужу.

Формат входного файла

В первой строке входного файла содержится число n ($2 \leqslant n \leqslant 50$). В следующих двух строках содержится описание дороги: j-й символ i-й строки соответствует j-му столбцу i-й строки, и содержит «`W`», если на соответствующей плитке лужа, и «`.`» — если лужи нет. Гарантируется, что лужи нет ни на плитке, с которой Фокс начинает, ни на той, куда ей нужно попасть.

Формат выходного файла

В выходной файл выведите строку «`YES`», если Фокс сможет добраться до друзей, иначе выведите «`NO`».

Примеры

road.in	road.out
4 .W..	YES
4 .W.. ..W.	YES
6 .W..W.. ...WWW.	NO
2 .. WW	YES
6 .WWWW. WWWWWW	NO
7 .W.W.W. W.W.W.W	YES

road.in
47 .....W. .....W.
road.out
NO

Задача E. Магический квадрат

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

Вам требуется поместить девять строк в квадрат 3×3. Строки квадрата пронумерованы от 0 до 2 сверху вниз, а столбцы — от 0 до 2 слева направо. Пусть s_{ij} — это строка, которая будет в j-м столбце i-й строки квадрата. Строки s_{ij} не обязательно должны быть различными. Также можно использовать пустые строки.

Вам дано два набора строк, по три строки в каждом: $rowStrings_i$ и $columnStrings_j$. Для любого i конкатенация строк в i-й строке квадрата должна давать $rowStrings_i$. Аналогичное условие должно выполняться для столбцов квадрата и строк $columnStrings_j$. Формально это означает следующее:

- Для $0 \leqslant i \leqslant 2$ $s_{i0} + s_{i1} + s_{i2} = rowStrings_i$
- Для $0 \leqslant j \leqslant 2$ $s_{0j} + s_{1j} + s_{2j} = columnStrings_j$

где «+» означает конкатенацию строк.

Подсчитайте количество способов заполнить квадрат строками так, чтобы выполнялись такие требования.

Формат входного файла

В шести строках входного файла содержится шесть строк $rowStrings_0$, $rowStrings_1$, $rowStrings_2$, $columnStrings_0$, $columnStrings_1$, $columnStrings_2$, именно в таком порядке. Длины строк не превосходят 50. Обратите внимание, что строки могут быть пустыми.

Формат выходного файла

В выходной файл выведите искомое число способов.

Примеры

square.in	square.out
f o x f o x	1
x x x x xx	3
cd cd cd dvd dvd dvd	0
abab ab abab abab ab abab	11
qwer asdf zxcv qaz wsx erdfcv	1

Задача F. Просто Floyd

Имя входного файла:	`floyd_joke.in`
Имя выходного файла:	`floyd_joke.out`
Ограничение по времени:	0.25 секунды
Ограничение по памяти:	256 мегабайт

Вам дан неориентируемый полный граф, состоящий из n вершин. Расстояние между вершинами a и b определяется как $dist(a, b) = b \cdot a^2 + b^2 \cdot a$ для $a \neq b$ и $dist(a, a) = 0$.
Напомним, что алгоритм Флойда на C++ пишется так:

```
for(int k = 0; k < n; ++k)
    for(int i = 0; i < n; ++i)
        for(int j = 0; j < n; ++j)
            d[i][j] = min(d[i][j], d[i][k] + d[k][j]);
```

Где массив d изначально заполнен значениями $d[i][j] = dist(i, j)$ для всех i и j.
Вершины нумеруются с единицы.

Формат входного файла

Во входном файле задано единственное число n ($2 \leqslant n \leqslant 1000$).

Формат выходного файла

В выходной файл выведите одно число сумму $dist(i, j)$ для всех i и j по простому модулю $1000000007 (10^9 + 7)$.

Примеры

floyd_joke.in	floyd_joke.out
2	12
13	21792