# Wildcard Patterns

## PROBLEM

Wildcard patterns are frequently used to specify sets of names. For example, we can specify all file names which start with `h` and end with `bak` by the pattern `h*bak`.

A *wildcard pattern* is a string that may contain * characters as wildcards. A string $W$ matches a pattern $P$ if $W$ can be obtained from $P$ by substituting some (possibly empty) string for each * character in $P$. (Different strings may be substituted for different occurrences of *.) For a pair of patterns $P_1$ and $P_2$, $Q$ is a *common pattern* of $P_1$ and $P_2$ if any string that matches $Q$ also matches both $P_1$ and $P_2$.

A set $Q_1$, $Q_2$, …, $Q_L$ of common patterns is called *complete* if any string that matches both $P_1$ and $P_2$ matches at least one $Q_i$ ($1 \le i \le L$).

You are to write a program that for a given pair of patterns $P_1$ and $P_2$ computes

A. at least one common pattern (but no wrong patterns); or

B. a complete set of common patterns with no more than 6666 entry; or

C. a complete set of common patterns which is minimal in the number of elements (i.e. there is no smaller set of common patterns, which is complete) Note that solving this task you will earn **100% plus an additional 50% of the scores**.

## INPUT

The first and the second lines of the file `pattern.in` contain the patterns $P_1$ and $P_2$. The patterns are composed of the lowercase letters from ‘a’ to ‘z’ and the ‘*’ character. Each pattern contains no more than 20 characters. The number of the * characters in each pattern is between 0 and 6.

## OUTPUT

The first line of the file `pattern.out` must contain the integer $K$, the number of common patterns computed as a solution. The next $K$ lines contain one common pattern in each line; these constitute the solution.

The order of the common patterns is irrelevant. Both case B and C can be solved within the specified limit of 6666 entries. If there is no common pattern of $P_1$ and $P_2$, then the first and only line must contain the number 0.

## EXAMPLE INPUT AND OUTPUT

| `pattern.in` | `pattern.out` for case C |
|---|---|
| `*ab*`<br>`ba*b` | `4`<br>`ba*ab*b`<br>`bab*b`<br>`ba*ab`<br>`bab` |

## GRADING

If your solution solves case A then you obtain 5 points. If your solution solves case B then you obtain 10 points. If your solution solves case C then you obtain 15 points.