4213 DNA Subsequences

Thomas, a computer scientist that works with DNA sequences, needs to compute longest common subsequences of given pairs of strings. Consider an alphabet Σ of letters and a word $w = a_1 a_2 \ldots a_r$, where $a_i \in \Sigma$, for $i = 1, 2, \ldots, r$. A *subsequence* of w is a word $x = a_{i_1} a_{i_2} \ldots a_{i_s}$ such that $1 \leq i_1 < i_2 < \ldots < i_s \leq r$. Subsequence x is a *segment* of w if $i_{j+1} = i_j + 1$, for $j = 1, 2, \ldots s - 1$. For example the word "`ove`" is a segment of the word "`lovely`", whereas the word "`loly`" is a subsequence of "`lovely`", but not a segment.

A word is a *common subsequence* of two words w_1 and w_2 if it is a subsequence of each of the two words. A *longest common subsequence* of w_1 and w_2 is a common subsequence of w_1 and w_2 having the largest possible length. For example, consider the words w_1=`lovxxelyxxxxx` and w_2=`xxxxxxxlovely`. The words w_3=`lovely` and w_4=`xxxxxxx`, the latter of length 7, are both common subsequences of w_1 and w_2. In fact, w_4 is their longest common subsequence. Notice that the empty word, of length zero, is always a common subsequence, although not necessarily the longest.

In the case of Thomas, there is an extra requirement: the subsequence must be formed from common segments having length K or more. For example, if Thomas decides that $K = 3$, then he considers "`lovely`" to be an acceptable common subsequence of "`lovxxelyxxxxx`" and "`xxxxxxxlovely`", whereas "`xxxxxxx`", which has length 7 and is also a common subsequence, is not acceptable. Can you help Thomas?

Input

The input contains several test cases. The first line of a test case contains an integer K representing the minimum length of common segments, where $1 \leq K \leq 100$. The next two lines contain each a string on lowercase letters from the regular alphabet of 26 letters. The length l of each string satisfies the inequality $1 \leq l \leq 10^3$. There are no spaces on any line in the input. The end of the input is indicated by a line containing a zero.

Output

For each test case in the input, your program must print a single line, containing the length of the longest subsequence formed by consecutive segments of length at least K from both strings. If no such common subsequence of length greater than zero exists, then `0` must be printed.

Sample Input

```
3
lovxxelyxxxxx
xxxxxxxlovely
1
lovxxelyxxxxx
xxxxxxxlovely
3
lovxxxelxyxxxx
xxxlovelyxxxxxxx
4
lovxxxelyxxx
xxxxxxlovely
```

```
0
```

Sample Output

```
6
7
10
0
```