Let x and y be two strings over some finite alphabet A. We would like to transform x into y allowing only operations given below:

Deletion: a letter in x is missing in y at a corresponding position.
Insertion: a letter in y is missing in x at a corresponding position.
Change: letters at corresponding positions are distinct

Certainly, we would like to minimize the number of all possible operations.

Illustration

```
A  G  T  A  A  G  T  *  A  G  G  C
|  |  |           |     |     |  |
A  G  T  *  C  *  T  G  A  C  G  C
```

Deletion: $*$ in the bottom line
Insertion: $*$ in the top line
Change: when the letters at the top and bottom are distinct

This tells us that to transform $x =$ `AGTCTGACGC` into $y =$ `AGTAAGTAGGC` we could be required to perform 5 operations (2 changes, 2 deletions and 1 insertion).

If we want to minimize the number operations, we should do it like

```
A  G  T  A  A  G  T  A  G  G  C
|  |  |        |     |     |  |
A  G  T  C  T  G  *  A  C  G  C
```

and 4 moves would be required (3 changes and 1 deletion).

In this problem we would always consider strings x and y to be fixed, such that the number of letters in x is m and the number of letters in y is n where $n \geq m$.

Assign 1 as the cost of an operation performed. Otherwise, assign 0 if there is no operation performed.

Write a program that would minimize the number of possible operations to transform any string x into a string y.

Input

Input contains several datasets. Each dataset consists of the strings x and y prefixed by their respective lengths, one in each line.

Output

For each dataset, an integer representing the minimum number of possible operations to transform any string x into a string y.

Sample Input

```
10 AGTCTGACGC
11 AGTAAGTAGGC
```

Sample Output

```
4
```