Competitive Programming

Saarland University — Summer Semester 2022

Markus Bläser, Karl Bringmann, Martin Bromberger, Christoph Weidenbach
Julian Baldus, Yasmine Briefs, Simon Schwarz, Dominic Zimmer

Assignments Week 9

Deadline: **July 7, 2022 at 16:00 sharp**

Please submit solutions to the problems in our judge system, available at
`https://compro.mpi-inf.mpg.de/`.
You can find your credentials on your personal status page in our CMS.

Problem	`addressbook`	`concatcount`	`disco`	`lcs`
Points	3	3	3	3
Difficulty	🌶️🌶️	🌶️🌶️🌶️🌶️	🌶️🌶️🌶️	🌶️🌶️
Time Limit	2s	2s	1s	1s
Memory Limit	2 GB	2 GB	2 GB	2 GB

Please note:

- Your solution will be judged immediately after submitting. This may take some time, depending on the current server load.
- You can submit as many times as you want. However, don't abuse the server or try to extract the secret test cases.
- If your solution is **accepted**, you will receive the points specified in the table above.
- If you get **another verdict**, you will receive 0 points.

Addressbook

Problem ID: addressbook
Time limit: 2 seconds

We're going to make our own Address Book application! The application must perform two types of operations:

- `add name` where name is a string denoting a contact name. This operation stores `name` as a new contact in the application.
- `find query`, where `query` is a string denoting a partial name to search for. This operation counts the number of contacts starting with `query` and prints this number on a new line.

Given n sequential add and find operations, perform each operation in order.

Input

The first line contains a single integer, n ($1 \leq n \leq 10^5$), denoting the number of operations to perform.

Each line i of the n subsequent lines contains an operation in one of the two forms defined above.

All names and queries consist of english lowercase letters only. Furthermore, the length of the names and queries is at least 1 and at most 25. The names added to the address book are pairwise distinct.

Output

For each query, output the number of matching contacts.

Sample Input 1

```
14
add dieter
add dinic
find di
find f
add dijkstra
find di
add tarjan
add floyd
add ford
find d
find dij
find fl
find f
find dinic
```

Sample Output 1

```
2
0
3
3
1
1
2
1
```

Concatcount

Problem ID: concatcount
Time limit: 2 seconds

Counting the number of occurrences of a (contiguous) substring s in another string t is a well-known popular problem. For example, `knd` occurs twice in `kndlknd`. Overlapping occurences are also allowed and should be counted separately. Therefore, `fef` occurs two times in `covfefef`. We changed this classical problem a little bit.

Input

In the first line, you are given a string t ($1 \leq |t| \leq 2 \cdot 10^5$) consisting of english lowercase characters. The second line contains a positive integer n ($1 \leq n \leq 2 \cdot 10^5$). The following n lines contain each a single string s_i ($1 \leq |s_i| \leq 2 \cdot 10^5$). It is guaranteed that all s_i consist of only lowercase english characters and that $\sum_{i=1}^{n} |s_i| \leq 2 \cdot 10^5$

Output

For each pair i, j, we define the string s_{ij} to be the concatenation of s_i and s_j. For each string s, we call $o(s)$ the number of occurences of s in t. Print the sum of all $o(s_{ij})$, for all i, j.

Sample Inputs

Explanation of Sample Input 1

The strings s_1 and s_2 are `e` and `f`, respectively. $s_{11} =$ `ee` does not occur, $s_{12} =$ `ef` is found once, $s_{21} =$ `fe` twice and $s_{22} =$ `ff` not at all. The total is therefore three.

Explanation of Sample Input 2

The strings s_1 and s_2 are `a` and `aa`, respectively. We count $s_{11} =$ `aa` three times, $s_{12} =$ `aaa` once, $s_{21} =$ `aaa` once and $s_{22} =$ `aaaa` not at all. The total is therefore five.

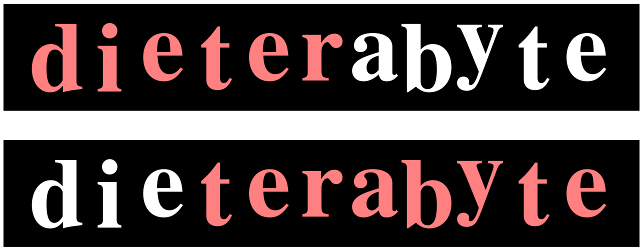
Sample Input 1	Sample Output 1
covfefe 2 e f	3

Sample Input 2	Sample Output 2
aaabacaa 2 a aa	5

Sample Input 3	Sample Output 3
aaabacaa 4 a a a b	33

Disco Lights

Problem ID: disco
Time limit: 1 second



After robbing a bank last week, there was no turning back for Dieter. This lifestyle is so much simpler and more exciting than his old life, which was all about impressing his boring wine friends. Of course, he now has new, cooler friends to impress. He already knows what he has to do to win them over. What he needs is an underground discothéque.

Dieter has already made up a good name: Terabyte. He can already imagine it: a foggy scene, large letters cut out of a piece of solid wood, illuminated from behind by red light, spelling the letters **TERABYTE**. So much for his dreams on public relations. Dieter wants to establish his reputation and make every gangster in the city know his name.

Whenever his lookouts signal him that the coast is clear and there is no police nearby, he will change the backlighting so that his letters read **DIETER**. However, cutting huge letters into blocks of wood is expensive. For example, it is not wise to cut **DIETERTERABYTE** but rather have the two words overlap. In order to estimate the costs, Dieter wants you to write a program that computes the number of letters he needs to order.

For artistic reasons, it is important

- that all letters of the first and second word need to be arranged as one contiguous string and
- that the first word is displayed further to the left than the right word.

Input

The input consists of two words s ($1 \leq |s| \leq 10^6$) and t ($1 \leq |t| \leq 10^6$), separated by a single space. Neither s nor t contain any spaces.

Output

Print the minimum number of letters Dieter needs to get manufactured so he can chose to display either of both words.

Sample Input 1	Sample Output 1
`dieter terabyte`	`11`

Sample Input 2	Sample Output 2
`teamilkricekiller killer`	`17`

Sample Input 3	Sample Output 3
`freshmilk freshblood`	`19`

LCS

Problem ID: lcs
Time limit: 1 second

Your local athletics club 1. Leichtathletik Club Sulzbach (LCS) needs your help. The summer season is in full swing and the club gained plenty of new members.

Sadly, the club is serverely underfunded and has little money to spend for its members' jerseys. Therefore, the administration has decided to recycle former members' jerseys. Not only will they recycle the fabric, but they also intend to reuse letters from the names printed onto the jerseys.

Given a shirt on which the name of the previous owner is printed you need to decide which letters you need to change. Fortunately, the names are printed on a piece of rubber that is located on the front of the jersey. This allows you to stretch the space between letters on the jersey and insert new letters. In order to achieve the new name, you may

- remove letters from the old name
- stretch the rubber and insert a new letter at any position

Note that it is impossible to switch the order of any two existing characters.

In order to keep the amount of new letters the club needs to buy low, you try to keep as many characters as possible. Your task is to determine how many characters of the old print you can keep.

Input

The input consists of two strings s and t ($1 \leq |t|, |s| \leq 1000$) consisting of english lowercase characters.

Output

Output the maximum number of characters of s you can keep in order to have the resulting jersey spell t.

Sample Input 1	Sample Output 1
`dieterschlau schlauerdieter`	`6`

Sample Input 2	Sample Output 2
`mark marcus`	`3`

Sample Input 3	Sample Output 3
`saarland saarbrooklyn`	`6`