Common Prefix Length

Given a string, split the string into two substrings at every possible point. The rightmost substring is a *suffix.* The beginning of the string is the *prefix.* Determine the lengths of the common prefix between each suffix and the original string. Sum and return the lengths of the common prefixes. Return an array where each element *i* is the sum for string *i.*

**Example**

Consider the only string in the array *inputs = ['abcabcd']*. Each suffix is compared to the original string.

| **Remove to leave suffix** | **Suffix** | **Common Prefix** | **Length** |
| --- | --- | --- | --- |
| '' | 'abcabcd' | 'abcabcd' | 7 |
| 'a' | 'bcabcd' | '' | 0 |
| 'ab' | 'cabcd' | '' | 0 |
| 'abc' | 'abcd' | 'abc' | 3 |
| 'abca' | 'bcd' | '' | 0 |
| 'abcab' | 'cd' | '' | 0 |
| 'abcabc' | 'd' | '' | 0 |

The sum is *7 + 0 + 0 + 3 + 0 + 0 + 0 = 10.*

**Function Description**

Complete the function *commonPrefix* in the editor below.

commonPrefix has the following parameter(s):

*string inputs[n]:*  an array of strings

Returns:

*int[]:* the sums of the common prefix lengths for each test case

**Constraints**

*1 ≤ n ≤ 10*

*1 ≤ | inputs[i] | ≤ 105*

Each *inputs[i]* contains only letters in the range *ascii[a-z]*.

Input Format For Custom Testing

The first line contains the number of test cases *n*.

Each of the next *n* lines contains a string, *inputs[i],*one for each test case.

Sample Case 0

**Sample Input**

STDIN Function

----- -----

1 → number of test cases n = 1

ababaa → inputs = ['ababaa']

**Sample Output**

11

**Explanation**

*n = 1*

*inputs = ['ababaa']*  
The suffixes are *['ababaa', 'babaa', 'abaa', 'baa', 'aa', 'a'].* The common prefix lengths of each of these suffixes with the original string are *[6, 0, 3, 0, 1, 1]* respectively, and they sum to *11*.

Sample Case 1

**Sample Input**

STDIN Function

----- -----

1 → number of test cases n = 1

aa → inputs = ['aa']

**Sample Output**

3

**Explanation**

*n = 1*

*inputs = ['aa']*

The suffixes are *['aa', 'a'].* The common prefix lengths of each of these suffixes with the original string are *[2, 1]*  which sum to *3.*

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.function.\*;

import java.util.regex.\*;

import java.util.stream.\*;

import static java.util.stream.Collectors.joining;

import static java.util.stream.Collectors.toList;

class Result {

/\*

\* Complete the 'commonPrefix' function below.

\*

\* The function is expected to return an INTEGER\_ARRAY.

\* The function accepts STRING\_ARRAY inputs as parameter.

\*/

public static List<Integer> commonPrefix(List<String> inputs) {

// Write your code here

}

}

public class Solution {

public static void main(String[] args) throws IOException {

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

int inputsCount = Integer.parseInt(bufferedReader.readLine().trim());

List<String> inputs = IntStream.range(0, inputsCount).mapToObj(i -> {

try {

return bufferedReader.readLine();

} catch (IOException ex) {

throw new RuntimeException(ex);

}

})

.collect(toList());

List<Integer> result = Result.commonPrefix(inputs);

bufferedWriter.write(

result.stream()

.map(Object::toString)

.collect(joining("\n"))

+ "\n"

);

bufferedReader.close();

bufferedWriter.close();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*