Questions of ITP course at Shahid-Beheshti-University

Part 8

by Farbod Fooladi

Miniaturization

When receiving a string, remove its largest character.

Entrance

In the first line, n or the length of the desired string is written and in the next line the string itself is written.

$$2 \leq n \leq 200000$$

Output

Remove the largest character from the input string and print the remaining string on one line of standard output.

Note: If there are multiple deletion options, delete the character that is closest to the beginning of the string.

Homogenization

Write a program that, upon receiving two strings of equal length, determines the possibility of making two identical strings from them by performing the desired number of the following operations on each:

In each operation, you can select any two adjacent characters from one of the strings and set the value of one of those two characters equal to the other. Any number of these operations can be performed for each of the two strings.

For example, if the string is acbcacbccacbc, the following operations can be performed:

(S2=S1)aabc

(S2 = S1) aabc

(S2=S1)aabc

(S1=S2)ccbc

(S1 = S2) ccbc

(S1=S2)ccbc

(S3=S2orS3=S4)accc

(S3 = S2 or S3 = S4) accc

(S3=S2orS3=S4)accc

(S2=S3)abbc(S2 = S3) abbc

(S2=S3)abbc

(S4=S3)acbb(S4 = S3) acbb

(S4=S3)acbb

Hint: Pay attention to the text of the question and the operation above. Perhaps a very simple point can be obtained from them.

Entrance

n is written in the first line, which indicates the number of pairs of input strings.

1≤n≤501 \le n \le 501≤n≤50

In the n of the next line, in each line, two strings of English lowercase letters of length equal to LLL are written, separated by spaces.

$$1 \leq L \leq 15$$

Output

In the n line of the standard output, in the ith line, if it is possible to match the two strings corresponding to the ith input, print YES, otherwise, NO.

Summary

By receiving a set of words, for words whose length is more than ten, keep the first and last letters and put their number instead of the middle letters.

Entrance

In the first line n or the number of words and in the next line n the words with a maximum length of l are written.

$$1 \leq n, l \leq 1000$$

output

On line n of standard output, print the input words in the original order as requested.

Sorting nouns

When you get some irregular nouns, first change them so that the first letter of each word of each noun is uppercase. Then sort the nouns alphabetically.

Note: Be sure to use the insertion sort algorithm. If the first two words were equal, compare the next words.

Entrance

n or the number of names is written in the first line. In the n next line, n noun with a maximum length of l is written, with each word separated by a space.

$$1 \leq n \leq 100$$

$$1 \leq l \leq 50$$

output

On line n of standard output, print names as requested.

complex numbers

In this question, you are asked to write a program to work with complex numbers with the help of structures.

Define the following structure in your program and then by receiving the elements of two complex numbers and a desired operation, create two complex structures for them and after calculating the desired operation on those two, create a complex structure for the answer and that display.

```
struct complex
{
    int re;
    int img;
};
```

- The desired operation is addition, subtraction or multiplication. (It is assumed that you are familiar with them!!!)
- Input complex numbers are written in the pattern re+i∗imre+i*imre+i∗im even if the coefficients of rerere and imgimgimg are zero or one.
- The answer format is re+imgire+imgire+imgi. When printing it, pay attention that iii is written after the imgimgimg coefficient.

Entrance

The input consists of three lines. In the first two lines, complex numbers with the specified pattern are written, and in the third line, the desired action is written as +++, −-− or ∗*∗.

$$-100 < re, img < 100$$

Output

Print the result of calculations in one line of standard output with the mentioned format. Pay close attention to the way complex numbers are represented in the examples.

Telephone book

Array is not the only way to store a set of data together, but there are many other ways to do this. One of these methods is to use the linked list data structure, in which instead of keeping all the elements consecutively like an array, using structures, each element keeps a pointer (link) to the next element. In other words, one of the philosophies of using a linked list is to maintain a chain of data instead of a block of data. Considering these points, it can be assumed that to store a thousand data of one byte in the array, there must be at least one kilobyte of contiguous free memory, while using a linked list, that space does not need to be contiguous. The linked list has its own advantages and disadvantages, including the fact that to access the ith element of the array, it is enough to jump from the beginning of it, while in the linked list, each element knows only the element after it, and it must be done from the beginning on all The elements moved until it reached the ith element. At the same time, the dynamic management of the collection in the linked list is simpler, and to add a new element to the collection, it is enough to connect it to the last element of the list, that is, the last element no longer points to anywhere, but to the new element as the next element. slow

Note: The explanations provided are enough to solve this question. However, you will learn more about linked lists in future lessons. Some of its implementations are also not explicitly mentioned and are up to you, for example, what methods are there to access the last element. If you are interested in reading more about this data structure, it is recommended to take a look at this link.

You are going to write a phonebook for storing contact numbers, where the collection of contacts is stored using a linked list. Each contact has a first name (string), last name (string) and phone number (whole number). The code below shows the general shape of the structure used in your program. As you can see, the next pointer for each contact points to the contact after him. Also, the first global variable shows the first element of the linked list of contacts.

```
struct contact
{
    //Any Type of Data Set
    contact* next;
};
contact** first; //Or contact* first
```

Assuming that the maximum string length is 20, write a program that receives and executes the following commands until the exit command is entered:

- The Add command receives the information of the new contact as follows and adds him to the linked list of contacts.

  A [first_name] [last_name] [number]

- Edit command that receives the name and surname of a contact and sets his number equal to the received number. If there is no contact with the corresponding first and last name, the statement "Contact doesn't exist" should be printed. It is guaranteed that no two contacts have the same first and last name.

  E [first_name] [last_name] [number]

- The Number command receives the first and last name of a contact and prints his number in one line. If there is no contact with the corresponding first and last name, the statement "Contact doesn't exist" should be printed.

  N [first_name] [last_name]

- The Show command, as shown below, prints the information of each phone book addressee in order of entry in each line. If there is no contact in the phone book, the statement "Contact list is empty" should be printed.

  S [first_name] [last_name] [number]

- Exit command that ends the execution of the program.

  X