EVERNOTE

(/)

Take the Evernote Coding Challenge

Do you have what it takes to be an Evernote engineer? We hope so! We challenge you to complete the three questions below. Score an Evernote Smart Notebook by Moleskine (/moleskine/) and the opportunity for an interview with our team if your code catches our eye. Good luck!

Question #1

Implement a circular buffer of size N. Allow the caller to append, remove and list the contents of the buffer. Implement the buffer to achieve maximum performance for each of the operations.

The new items are appended to the end and the order is retained i.e elements are placed in increasing order of their insertion time. When the number of elements in the list elements exceeds the defined size, the older elements are overwritten.

There are four types of commands.

"A" n - Append the following n lines to the buffer. If the buffer is full they replace the older entries.

"R" n - Remove first n elements of the buffer. These n elements are the ones that were added earliest among the current elements.

"L" - List the elements of buffer in order of their inserting time.

"Q" - Quit.

Your task is to execute commands on circular buffer.

Input format

First line of input contains N , the size of the buffer.

A n - append the following n lines to the buffer.

R n - remove first n elements of the buffer.

L - list the buffer.

Q - Quit.

Output format

Whenever L command appears in the input, print the elements of buffer in order of their inserting time. Element that was added first should appear first.

Sample Input

10

A 3

Fee

Fi

Fo

A 1

Fum

R 2

L

Q

Sample Output

Fo

Fum

Constraint

0 <= N <= 10000

Number of removing elements will <= number of elements presents in circular buffer.

Total number of commands <= 50000.

Total number of characters in input <= 20000000.

Question #2

Frequency Counting of Words / Top N words in a document.

Given N terms, your task is to find the k most frequent terms from given N terms.

Input format

First line of input contains N, denoting the number of terms to add.

In each of the next N lines, each contains a term.

Next line contains k, most frequent terms.

Output format

Print the k most frequent terms in descending order of their frequency. If two terms have same frequency print them in lexicographical order.

Sample input

14

Fee

Fi

Fo

Fum

Fee

Fo

Fee

Fee

Fo

Fi

Fi

Fo

Fum

Fee

3

Sample output

Fee

Fo

Fi

Constraint

0 < N < 300000

0 < term length < 25.

Question #3

Given a list of integers, your task is to write a program to output an integer-valued list of equal length such that the output element at index 'i' is the product of all input elements except for the input element at 'i'.

In other words, let inputArray by an integer array of length 'n'. The solution,computed into outputArray, would be:

for each j from 1 to n-2:

outputArr[j] = inputArray[0] * inputArray[1] * inputArray[2] * ... * inputArray[j-1] * inputArray[j+1] * inputArray[j+2] *...* inputArray[n-1]

for j = 0

outputArray[0] = inputArray[1] * outputArray[2] * ... * outputArray[n-1]

for j = n-1

outputArray[n-1] = outputArray[0] * outputArray[1] * outputArray[2] * ... * outputArray[n-2]

As an example, if inputArray = { 1, 2, 3, 4 }, then

outputArray = { 2*3*4, 1*3*4, 1*2*4, 1*2*3 }.

Your program should run in O(n) time and should be space efficient.

Input format

First line of input contains N , number of elements in list.

Next N lines will each contain an element (a signed integer)

Output format

Print the output list of numbers.

Sample input

4

5

2

2

3

Sample output

12

30

30

20

Constraint

You may assume that:

- The input array size will always have at least two elements in it, that is, n >= 2.
- The product of any subset of the input will never exceed the value of a 64 bit integer.
- The maximum length of input array is 1000.