Problem A. Triangle Binary Search Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given N integers in order of their insertion to Binary Search Tree. You draw a set of horizontal lines that goes through nodes with the same height. After that you can see triangles with nodes instead vertices and edges instead sides. Your task is to calculate the number of the smallest triangles.

Input

The first line consists of an integer N - number of nodes in Binary Search Tree ($1 \leqslant N \leqslant 10000$).

The second line contains N integers a_i - value of each node in Binary Search Tree in order of their insertion ($1 \leqslant a_i \leqslant N$).

It is guaranteed that there are no duplicates.

Output

Print the number of mini-triangles in resulting Binary Search Tree.

Examples

standard input	standard output
3 3 5 1	1
3 1 3 5	0
16 13 9 3 7 6 16 1 11 12 10 4 2 14 5 8 15	5

Problem B. Balanced Binary Search Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You have an array with 2^N - 1 elements in it. You want to build the Binary Search Tree on this array, adding elements in order of their appearance in array (from left to right). But there is a probability of imbalance of such tree. That's why you decided to shuffle your array to obtain perfectly balanced Binary Search Tree after adding elements (from left to right, again). Your task is to print your array after appropriate shuffle. If there are several possible shuffles, print the array after applying any of them.

Note, that you are not asked for building Binary Search Tree, but only for shuffling array.

Input

The first line of input consists of single integer N that describes the length of the array ($1 \leqslant N \leqslant 15$).

The next line contains 2^N - 1 integers a_i - elements of the array ($0 \leqslant a_i \leqslant 2 \cdot 10^9$).

It is guaranteed that there is no duplicates in the array.

Output

Print 2^N - 1 integers - elements in your array after applying required shuffle.

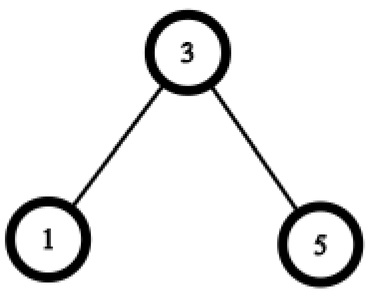
Examples

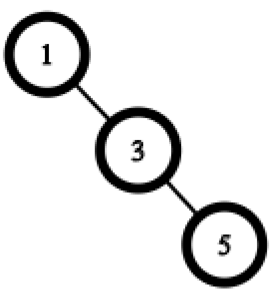
standard input	standard output
2 3 5 1	3 5 1
2 1 3 5	3 1 5

Note

In the first sample given array can be used for building balanced BST (left picture).

In the second sample given array gives such chain tree (right picture), so it must be shuffled.





Note, that for both samples [3, 5, 1] and [3, 1, 5] are correct answers.

Hint: Use divide and conquer method (recall advanced sorting algorithms) and implement recursive function to solve this problem

Problem C. Experiment with Mixtures

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Mark is making an experiment with the mixtures of different densities. For his experiment he wants all of the mixtures to have a density $\geq Min$. He uses the following formula to obtain a combined mixture: `new density = least density + 2*2nd least density`. Mark repeats the mixing until he gets all the mixtures with the density $\geq Min$. You are given the densities of mixtures. How many times Mark should mix his mixtures to get the densities of all mixtures $\geq Min$? Print -1 if this isn't possible.

Input

The first line consists of integers $N (1 \leq N \leq 10^6)$, the number of mixtures and $Min (0 \leq Min \leq 10^9)$, the minimum required density, separated by a space. The next line contains N space-separated integers $(0 \leq densities[i] \leq 10^6)$ describing the densities of mixtures.

Output

Print the number of operations that are needed to make all the densities $\geq Min$. Print -1 if it is impossible.

Examples

standard input	standard output
3 10 1 1 1	-1
6 7 1 2 3 9 10 12	2

Problem D. Don't be scared of magic square

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Do you know about magic squares? I think yes. Nevertheless I remind you. Magic square is a square in which sums of elements of each row, column or diagonal are equal. But now you deal with unusual magic square. This square is called magic if the greatest common divisor of elements in row and column is the same among all rows and columns. You are given a square with side N. Your task is to determine whether this square is magic or not.

Input

The first line of input contains single integer N - length of square's side ($2 \leqslant N \leqslant 100$).

Each of the next N lines contains N integers a_{ij} - elements of the square ($1 \leqslant a_{ij} \leqslant 2 \cdot 10^9$).

Output

Print «Yes» if square is magic, otherwise print «No» (without quotes).

Examples

standard input	standard output
2 4 10 6 14	Yes
2 4 8 12 16	No

Problem E. Simple Merge

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Given a set of strings print the set sorted according to their size. If the size of the strings are equal, must maintain the original order of the set.

Input

The first line of input has an integer T that indicates the number of sets of strings, each set may contain between 1 and 50 inclusive elements, and each of the strings of the set may contain between 1 and 50 inclusive characters('a' to 'z').

Output

The output should contain the set of input strings ordered by the length of strings.A blank space must be printed between two words.

Examples

standard input	standard output

`3`
`ab cd e j asd ljffg df`
`a a b b c c`
`xy yx zxy zx xzy xxx`
	`e j ab cd df asd ljffg`
	`a a b b c c`
	`xy yx zx zxy xzy xxx`

`4`
`aabc ddfd fbbe f baaad`
`fddf badcf aedbe cdb ae bbd`
`ada bc eaced dadab b aabc fbee`
`bedaf edde fcaa adbf beec`
	`f aabc ddfd fbbe baaad`
	`ae cdb bbd fddf badcf aedbe`
	`b bc ada aabc fbee eaced dadab`
	`edde fcaa adbf beec bedaf`

Note

You can solve this problem using mergesort or quicksort algorithm

Problem F. Wareta Ringo

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given an array a of size n and q queries. In each query you are given two integers l, r $(1 \leq l \leq r \leq n)$, you need to partition subarray $a_l, a_{l+1}, \ldots, a_r$ into minimal number of increasing subarrays. Output this number for each query.

Input

First line contains two integers n, q - size of array and number of queries. Second line contains n integers - array a $(1 \leq a_i \leq 10^5)$. Next q lines contain two integers l, r.

Output

Output q lines - answer to the queries.

Example

standard input	standard output
4 3 3 1 4 2 1 4 1 3 4 4	3 2 1

Note

Subarray $a_l, a_{l+1}, \ldots, a_r$ is called increasing if for each $l \leq i \leq r-1$ condition $a_i < a_{i+1}$ is satisfied.

Answer to the queries of the first testcase:
[3, 1, 4, 2] - [3], [1, 4], [2]
[3, 1, 4] - [3], [1, 4]
[4] - [4]

Problem G. Another one easy BST problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given implementation of Binary Search Tree. You need to introduce new feature to the Binary Search Tree. It must contain not only elements, but the number of copies of them too. Moreover, Binary Search Tree should not contain duplicate nodes.

Remember, when you remove some element, you delete it from Binary Search Tree if and only if it has a single appearance there. Otherwise you delete only one copy of element.

So, you are given several queries, each of them is of type '*insert X*', '*delete X*' or '*cnt X*'. You must answer on each query of the last type. The answer is the number of copies of X in Binary Search Tree.

To complete the task you need to download solution code from piazza.com and make some extra changes in it. Remaining code was written for you.

Input

The first line contains single integer Q - number of queries ($1 \leqslant Q \leqslant 10^3$). Each of the next Q lines contains one query.

Output

Print answer on each query of type '*cnt X*'.

Examples

standard input	standard output
4 insert 1 cnt 1 insert 1 cnt 1	1 2
8 insert 1 cnt 1 insert 1 cnt 1 delete 1 cnt 1 delete 1 cnt 1	1 2 1 0

Problem H. Mike and Pillars

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Mike is a short heighted person. He is standing facing N pillars of different heights with i-th pillar having height h_i. He tries to see all the possible pillars. He wants to know that how many buildings will he be able to see in the range $[L, R]$ both inclusive.

Input

The first line contains an integer N denoting the number of pillars. Next line contains N integers denoting height of i-th pillar. Next line contains a single integer Q. Next Q lines contain pairs L and R respectively.

Constraints

$1 \le N, Q \le 10^5$

$0 \le L \le R \le N - 1$

$1 \le h \le 10^9$

Output

For every Q queries print the number of buildings visible in the range $[L, R]$.

Example

standard input	standard output
7 5 2 3 7 9 8 11 4 0 6 1 5 2 6 3 4	4 4 4 2

Note

In query 1 5, 7, 9, 11 pillars are visible so answer is 4.

In query 2 2, 3, 7, 9 are visible so answer is 4.

Problem I. Robin Hood stealing the Gold

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Robin Hood wants to steal the golden bars from the bank of High Sheriff aiming to distribute them to poor local people. There are N bags of golden bars, the i-th bag has $bags[i]$ bars. Sheriff has gone and will return in H hours.

Robin can steal K bars per hour. Each hour, he chooses a single bag of golden bars, and steals K bars from that bag. If there are less than K bars in the bag, he steals them all, and won't steal any more during this hour.

Robin Hood wants to steal all of the golden bars before the Sheriff comes back.

Return the minimum number K such that Robin can steal ALL of the golden bars within H hours.

Input

The first line of the input contains two space-separated integers $N(1 \leq N \leq 10^4), H(N \leq H \leq 10^9)$, the number of bags of golden bars and the number of hours for which Sheriff has gone. The next line contains N space-separated integers $(1 \leq bags[i] \leq 10^9)$ denoting the number of golden bars in each bag.

Output

Print the minimum number K such that Robin Hood can steal all of the N golden bars within the limit of H hours.

Examples

standard input	standard output
4 8 3 6 7 11	4
5 5 30 11 23 4 20	30
5 6 30 11 23 4 20	23

Note

K is Robin's speed of stealing the bars such that $\sum_{i=1}^{N} \frac{bags[i]}{K} = H$.

If Robin can finish stealing all the bars (within H hours) with speed of K, he can finish with a larger speed too.

If we let $possible(K)$ be true if and only if Robin can finish with a speed of K, then there is some X such that $possible(K) = true$ if and only if $K \geq X$.

For the first test case there is some $X = 4$ so that $possible(1) = possible(2) = possible(3) = false$, and $possible(4) = possible(5) = \cdots = true$. $K = 4$ is the minimum K such that $\frac{3}{4}+\frac{6}{4}+\frac{7}{4}+\frac{11}{4} = 1+2+2+3 = 8$. $K = 5$ is also a right answer but it is not a minimum K.

Problem J. Another one easy BST problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given implementation of Binary Search Tree with *insert* and *find* functions. Your task is to calculate the size of the subtree of the node X.

Remember, that the subtree of node X is the set of all nodes whose ancestor is node X, including it. The size of the subtree is the size of such set.

To complete the task you need to download solution code from piazza.com and finish *getSubtreeSize()* function. Remaining code was written for you.

Input

The first line of the input contains an integer N - number of nodes in Binary Search Tree ($1 \leqslant N \leqslant 10^3$).

The second line contains N integers a_i - values of nodes in order of insertion to the Binary Search Tree.

The third line contains single integer - value of the node which subtree's size you must calculate.

Output

Print the size of the subtree of the given node.

Example

standard input	standard output
7 4 2 6 1 3 5 7 4	7

Problem K. Mutual Sort

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.25 seconds
Memory limit:	256 megabytes

At elementary school teacher gave children a task to sort one list of numbers based on the second list order. Precisely, students are given two lists of integer numbers $list1$ and $list2$, where the elements of $list2$ are distinct, and all elements in $list2$ are also in $list1$.

Children need to sort the elements of $list1$ such that the relative ordering of items in $list1$ are the same as in $list2$. Numbers that do not appear in $list2$ should be placed at the end of $list1$ in ascending order.

For example, for the lists $list1 = \{26, 21, 11, 20, 26, 50, 34, 1, 18, 26\}$ and $list2 = \{21, 11, 26, 20\}$ the sorted $list2 = \{21, 11, 26, 26, 26, 20, 1, 18, 34, 50\}$.

Help poor children to solve this problem.

Input

The first line contains two integers `N1 and N2` ($0 \le N1, N2 \le 1000$) denoting the sizes of the first and the second lists respectively. The next two lines represent $N1$ and $N2$ space-separated integers defining the $list1$ and $list2$ in respective order ($0 \le list1[i], list2[i] \le 1000$). Each $list2[i]$ is distinct and each $list2[i]$ is in $list1$.

Output

Print the $list1$ sorted by the relative ordering of items as in $list2$. If numbers do not appear in $list2$ they should be placed at the end in ascending order.

Example

standard input	standard output
11 6 2 9 2 19 3 1 3 2 4 6 7 2 1 4 3 9 6	2 2 2 1 4 3 3 9 6 7 19

Problem L. More One Night

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given a permutation of size n. Create an empty BST, and insert values $p_1, p_2, ..., p_n$ in this order. Find out number of leafs in BST.

Input

In the first line there is a single integer $1 \le n \le 5000$ size of permutation. Second line contains n distinct numbers from 1 to n - the permutation.

Output

Output one integer - answer to this task.

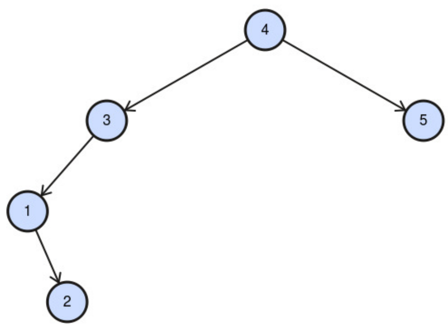
Examples

standard input	standard output
1 1	1
5 4 3 5 1 2	2

Note

Vertex is called a leaf if it doesn't have any sons.

In second testcase, BST looks like this.



Answer is 2(vertices 2 and 5).