Problem A. 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 $N(1 \le N \le 10^4), H(N \le H \le 10^9)$, 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 $(1 \le bags[i] \le 10^9)$ 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 $K \ge X$.

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 B. 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 height 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 height of the subtree is the number of nodes lying on the path from the root of the subtree to its deepest leaf.

To complete the task you need to download solution code from `piazza.com` and finish *getSubtreeHeight()* 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 height you must calculate.

Output

Print the height of the subtree of the given node.

Examples

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

Problem C. 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.

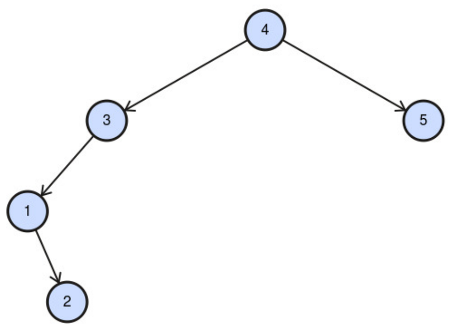
Examples

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).

Problem D. 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` ($0 \leq N1, N2 \leq 1000$) 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 ($0 \leq list1[i], list2[i] \leq 1000$). 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`