PERMUTATION RECOVERY

Ghiţă is a guy really keen on competitive programming. His favourite activities are playing with permutations and spending time with his wife, Ana. At their 10^{th} anniversary, Ana gave him a very beautiful permutation as she knew this is the best present Ghiţă could receive. Let P_j be the j^{th} element of the permutation for every j with $1 <= j <= N$.

Ghiţă was so excited by his present that he began computing the value Q_i for each i with $1 <= i <= N$. Q_i is defined as the number of increasing subsequences that he could find in the prefix of length i of his permutation.

More formally, for each i with $1 <= i <= N$, Q_i is the number of integer arrays j_1, j_2, .., j_k so that $1 <= j_1 < j_2 < \ldots < j_{k-1} < j_k <= i$ and $P_{j_1} < P_{j_2} < \ldots < P_{j_k}$.

He thought that Q, even though it wasn't a permutation, was pretty nice too. That's why he saved it near the permutation P.

Everything was ok until Lică Sămădăul came. He wanted to use Ghiţă's surveillance system for immoral purposes and Ghiţa, being a fair man, didn't help him. Enraged by Ghiţă's answer, Lică Sămădăul hired Buză Spartă to help him steal Ghiţă's most valuable asset: his permutation and wife. And so he did.

The next day Ghiţă found out that P was missing and now, the only solution for Ghiţă to recover the permutation is by using the array Q that he still has. You can guess that your job is to help Ghiţă recover array P being provided with array Q.

INPUT

On the first line of the input there is N, the length of the permutation. On the second line, separated by spaces, there are Q_1, Q_2, …, Q_n.

OUTPUT

On the first and only line of the output, you should print the array P representing the stolen permutation.

SPECIFICATIONS AND CONSTRAINTS

- It is guaranteed that there is **exactly** one possible answer (only one P has the given Q).
- $N <= 70.000$
- The input size is less than 111 MB.

- We advise you to independently check the running time and the usage of memory of the reading part of your program in order to make sure that the eventual inefficiency of your program is not due to this part, as the reading of the input is not intended to be an impediment in solving this task.

SCORING

Subtask	Restriction	T=Number of digits of Q(N)	Maximum input size	Score
Subtask 1	N <= 9	-	-	10 points
Subtask 2	N <= 400	T <= 18	-	15 points
Subtask 3	N <= 700	-	-	18 points
Subtask 4	N <= 40.000	T <= 171	4.5 MB	17 points
Subtask 5	N <= 70.000	T <= 258	10 MB	11 points
Subtask 6	N <= 70.000	T <= 314	16 MB	7 points
Subtask 7	N <= 70.000	-	85 MB	16 points
Subtask 8	N <= 70.000	-	111 MB	6 points

EXAMPLE

Input	Output
4 1 2 5 6	3 2 4 1
6 1 3 5 9 11 21	1 6 3 4 2 5

In the first example, N = 4 and P = {3, 2, 4, 1}

Q_1 = 1 because {3} is the only increasing subsequence of {3}

Q_2 = 2 because {3} and {2} are the only increasing subsequences of {3, 2}

Q_3 = 5 because {3}, {3, 4}, {2}, {2, 4} and {4} are the only increasing subsequences of {3, 2, 4}

Q_4 = 6 because {3}, {3, 4}, {2}, {2, 4}, {4} and {1} are the only increasing subsequences of {3, 2, 4, 1}.