# D. Restore Permutation

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

An array of integers $p_1, p_2, \ldots, p_n$ is called a permutation if it contains each number from $1$ to $n$ exactly once. For example, the following arrays are permutations: $[3, 1, 2]$, $[1]$, $[1, 2, 3, 4, 5]$ and $[4, 3, 1, 2]$. The following arrays are not permutations: $[2]$, $[1, 1]$, $[2, 3, 4]$.

There is a hidden permutation of length $n$.

For each index $i$, you are given $s_i$, which equals to the sum of all $p_j$ such that $j < i$ and $p_j < p_i$. In other words, $s_i$ is the sum of elements before the $i$-th element that are smaller than the $i$-th element.

Your task is to restore the permutation.

## Input

The first line contains a single integer $n$ ($1 \le n \le 2 \cdot 10^5$) — the size of the permutation.

The second line contains $n$ integers $s_1, s_2, \ldots, s_n$ ($0 \le s_i \le \frac{n(n-1)}{2}$).

It is guaranteed that the array $s$ corresponds to a valid permutation of length $n$.

## Output

Print $n$ integers $p_1, p_2, \ldots, p_n$ — the elements of the restored permutation. We can show that the answer is always unique.

### Examples

**input** Copy

```
3
0 0 0
```

**output** Copy

```
3 2 1
```

**input** Copy

```
2
0 1
```

**output** Copy

```
1 2
```

**input** Copy

```
5
0 1 1 1 10
```

**output** Copy

```
1 4 3 2 5
```

## Note

In the first example for each $i$ there is no index $j$ satisfying both conditions, hence $s_i$ are always $0$.

In the second example for $i = 2$ it happens that $j = 1$ satisfies the conditions, so $s_2 = p_1$.

In the third example for $i = 2, 3, 4$ only $j = 1$ satisfies the conditions, so $s_2 = s_3 = s_4 = 1$. For $i = 5$ all $j = 1, 2, 3, 4$ are possible, so $s_5 = p_1 + p_2 + p_3 + p_4 = 10$.

- Announcement (en)
- Tutorial (en)