Programming Part

Min-Heap Construction by Multiple Insertions (15 pts)

Problem Description

Given an array A that contains N distinct elements, we can call Min-Heap-Insert$(A, A[n])$ for $n = 1, 2, \ldots, N$ to convert A into a min heap. Although constructing the heap by those insertions may not be the fastest approach (check the textbook for faster ways!), it is somewhat simpler for you to understand and implement. :-) The Min-Heap-Insert algorithm, as modified from Max-Heap-Insert in the textbook, is

```
Min-Heap-Insert(A, key)
A.heapsize = A.heapsize + 1
A[A.heapsize] = ∞
Heap-Increase-Key(A, A.heapsize, key)
```

The Min-Heap-Insert algorithm calls Heap-Increase-Key to float the new *key* up. We strongly believe that you should know how to implement Parent(i) from our lecture. :-)

```
Heap-Increase-Key(A, i, key)
A[i] = key
while i > 1 and A[Parent(i)] > A[i]
      Swap(A[Parent(i)], A[i])
      i = Parent(i)
```

Please output the resulting min heap after the N insertions.

Input

The first line contains the number N. The second line contains N positive integers separated by spaces, representing the initial $A[1], A[2], \ldots, A[N]$.

Output

Output the array-represented min-heap after N insertions within a line, separating each number by a space.

Constraint

- $1 \leq N \leq 10^6$
- $1 \leq A[n] \leq 10^9$
- All $A[n]$ are distinct.

Sample Testcases

Sample Input 1

```
2
3 1
```

Sample Output 1

```
1 3
```

Sample Input 2

```
4
8 9 6 4
```

Sample Output 2

```
4 6 8 9
```

Sample Input 3

```
4
9 8 4 6
```

Sample Output 3

```
4 6 8 9
```

Sample Input 4

```
4
4 6 8 9
```

Sample Output 4

```
4 6 8 9
```

Hint

- By design, you can pass this homework by simulating the algorithms properly. There is no need for other arithmetic calculations or cuts.