import java.util.\*;

public class HeapSort {

private int[] heap;

private int count;

public void downAdjust(int i) {

int j, temp, n;

n = heap[0]; // Number of elements in heap

while (2 \* i <= n) { // While there is a left child

j = 2 \* i; // Left child

if (j + 1 <= n && heap[j + 1] > heap[j]) {

j = j + 1; // Right child is larger

}

if (heap[i] < heap[j]) { // Swap parent and larger child

temp = heap[i];

heap[i] = heap[j];

heap[j] = temp;

i = j; // Move down

} else {

break;

}

}

}

public void upAdjust(int i) {

int temp;

while (i > 1 && heap[i] > heap[i / 2]) {

temp = heap[i];

heap[i] = heap[i / 2];

heap[i / 2] = temp;

i = i / 2; // Move up

}

}

public void insert(int x) {

heap[0]++; // Increase heap size

heap[heap[0]] = x;

upAdjust(heap[0]); // Maintain heap property

}

public void create() {

Scanner reader = new Scanner(System.in);

heap = new int[30]; // Max heap size

heap[0] = 0; // Initial count

System.out.println("\nEnter No. of elements: ");

int n = reader.nextInt();

count = n;

System.out.println("\nEnter heap data: ");

for (int i = 0; i < n; i++) {

int x = reader.nextInt();

insert(x);

}

reader.close();

}

public void sort() {

int last, temp;

while (heap[0] > 1) {

last = heap[0];

temp = heap[1];

heap[1] = heap[last];

heap[last] = temp;

heap[0]--; // Reduce heap size

downAdjust(1);

}

}

public void print() {

int n = count;

System.out.println("\nSorted data: ");

for (int i = 1; i <= n; i++) {

System.out.print(heap[i] + " ");

}

System.out.println();

}

public static void main(String[] args) {

HeapSort myObject = new HeapSort();

myObject.create();

myObject.sort();

myObject.print();

}

}