ASK/VIEW DOUBT | SOLUTION | HINT | Your Code | Main Code | Code Pair | C++ (g++ 5)

Problem | Result

Code : In-place heap sort

Send Feedback

Given an integer array of size n. Sort this array (in decreasing order) using heap sort.

Space complexity should be O(1).

Input Format :

Line 1 : Integer n, Array size
Line 2 : Array elements, separated by space

Output Format :

Array elements after sorting

Constraints :
1 <= n <= 10^6

Sample Input:

6

PREVIOUS | NEXT

```
#include<bits/stdc++.h>

void inplaceHeapSort(int pq[], int n){
    /* Don't write main().
     * Don't read input, it is passed as function argument.
     * Change in the given input itself.
     * Taking input and printing output is handled automatically.
     */

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

        while(childIndex > 0) {
            int parentIndex = (childIndex - 1) / 2;

            if(pq[childIndex] < pq[parentIndex]) {
                int temp = pq[childIndex];
                pq[childIndex] = pq[parentIndex];
                pq[parentIndex] = temp;
            }
            else {
                break;
            }
            childIndex = parentIndex;
        }

    }


    for(int i=n-1;i>0;i--){

        int temp = pq[0];
        pq[0] = pq[i];
        pq[i] = temp;
        int childRight = 2*parent+2;
        int childLeft = 2*parent+1;


        while(childLeft < i) {
            int minIndex = parent;
            if(pq[minIndex] > pq[childLeft]) {
                minIndex = childLeft;
            }
            if(childRight < i && pq[childRight] < pq[minIndex]) {
                minIndex = childRight;
            }
            if(minIndex == parent) {
                break;
            }
            int temp = pq[minIndex];
            pq[minIndex] = pq[parent];
            pq[parent] = temp;

            parent = minIndex;
            childLeft = 2 * parent + 1;
            childRight = 2 * parent + 2;
        }

    }


}
```

CUSTOM INPUT | SUBMIT SOLUTION