Our Solution(s) Run Code

Solution 1

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

class Program {
  static class MinHeap {
    List<Integer> heap = new ArrayList<Integer>();

    public MinHeap(List<Integer> array) {
      heap = buildHeap(array);
    }

    // O(n) time | O(1) space
    public List<Integer> buildHeap(List<Integer> array) {
      int firstParentIdx = (array.size() - 2) / 2;
      for (int currentIdx = firstParentIdx; currentIdx >= 0; currentIdx--) {
        siftDown(currentIdx, array.size() - 1, array);
      }
      return array;
    }

    // O(log(n)) time | O(1) space
    public void siftDown(int currentIdx, int endIdx, List<Integer> heap) {
      int childOneIdx = currentIdx * 2 + 1;
      while (childOneIdx <= endIdx) {
        int childTwoIdx = currentIdx * 2 + 2 <= endIdx ? currentIdx * 2 + 2 : -1;
        int idxToSwap;
        if (childTwoIdx != -1 && heap.get(childTwoIdx) < heap.get(childOneIdx)) {
          idxToSwap = childTwoIdx;
        } else {
          idxToSwap = childOneIdx;
        }
        if (heap.get(idxToSwap) < heap.get(currentIdx)) {
          swap(currentIdx, idxToSwap, heap);
          currentIdx = idxToSwap;
          childOneIdx = currentIdx * 2 + 1;
        } else {
          return;
        }
      }
    }

    // O(log(n)) time | O(1) space
    public void siftUp(int currentIdx, List<Integer> heap) {
      int parentIdx = (currentIdx - 1) / 2;
      while (currentIdx > 0 && heap.get(currentIdx) < heap.get(parentIdx)) {
        swap(currentIdx, parentIdx, heap);
        currentIdx = parentIdx;
        parentIdx = (currentIdx - 1) / 2;
      }
    }

    public int peek() {
      return heap.get(0);
    }

    public int remove() {
      swap(0, heap.size() - 1, heap);
      int valueToRemove = heap.get(heap.size() - 1);
      heap.remove(heap.size() - 1);
      siftDown(0, heap.size() - 1, heap);
      return valueToRemove;
    }

    public void insert(int value) {
      heap.add(value);
      siftUp(heap.size() - 1, heap);
    }

    public void swap(int i, int j, List<Integer> heap) {
      Integer temp = heap.get(j);
      heap.set(j, heap.get(i));
      heap.set(i, temp);
    }
  }
}
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```java
import java.util.*;

// Do not edit the class below except for the buildHeap,
// siftDown, siftUp, peek, remove, and insert methods.
// Feel free to add new properties and methods to the class.
class Program {
  static class MinHeap {
    List<Integer> heap = new ArrayList<Integer>();

    public MinHeap(List<Integer> array) {
      heap = buildHeap(array);
    }

    public List<Integer> buildHeap(List<Integer> array) {
      // Write your code here.
      return null;
    }

    public void siftDown(int currentIdx, int endIdx, List<Integer> heap) {
      // Write your code here.
    }

    public void siftUp(int currentIdx, List<Integer> heap) {
      // Write your code here.
    }

    public int peek() {
      // Write your code here.
      return -1;
    }

    public int remove() {
      // Write your code here.
      return -1;
    }

    public void insert(int value) {
      // Write your code here.
    }
  }
}
```

Custom Output Raw Output Submit Code

Run or submit code when you're ready.