```java
/*
 * Written by JJ Shepherd
 */
public class MinHeap <T extends Comparable<T>>{
    private T[] heap;
    public static final int DEF_SIZE = 128;
    private int lastIndex;//First null element
    public MinHeap()
    {
        init(DEF_SIZE);
    }
    public MinHeap(int size)
    {
        init(size);
    }
    private void init(int size)
    {
        if(size > 0)
        {
            heap = (T[])(new Comparable[size]);
            lastIndex = 0;
        }
        else
            init(DEF_SIZE);
    }
    public void add(T aData)
    {
        if(lastIndex >= heap.length)
            return;
        heap[lastIndex] = aData;
        bubbleUp();
        lastIndex++;
    }
    private void bubbleUp()
    {
        int index = lastIndex;
        while(index > 0 && heap[(index-1)/2].compareTo(heap[index])>0)
        {
            T temp = heap[(index-1)/2];
            heap[(index-1)/2] = heap[index];
            heap[index] = temp;
            index = (index-1)/2;
        }
    }
    public T remove()
    {
        if(lastIndex == 0)
            return null;
        T ret = heap[0];
        heap[0] = heap[lastIndex-1];
        heap[lastIndex-1] = null;
        lastIndex--;
        bubbleDown();
        return ret;
    }
    private void bubbleDown()
    {
        int index = 0;
        while(index*2+1 < lastIndex)
        {
            int smallIndex = index*2+1;
            if(index*2+2 < lastIndex && heap[index*2+1].compareTo(heap[index*2+2])>0)
                smallIndex = index*2+2;
            if(heap[index].compareTo(heap[smallIndex])>0)
            {
                T temp = heap[index];
                heap[index] = heap[smallIndex];
                heap[smallIndex] = temp;
            }
            else
                break;
            index = smallIndex;
        }
    }
    public void print()
    {
        for(T h : heap)
        {
            if(h == null)
                break;
            System.out.println(h);
        }
    }
    public void heapSort(T[] array)
    {
        if(array == null)
            return;
        MinHeap<T> mHeap = new MinHeap<T>(array.length);
        for(int i=0;i<array.length;i++)
            mHeap.add(array[i]);
        for(int i=0;i<array.length;i++)
            array[i] = mHeap.remove();

    }
}

```