排序 - 堆排序

作者：fengwang
原文链接：https://ld246.com/article/1529658736349
来源网站：链滴

本篇记录二叉堆的排序，以最大堆的排序为例。

几个概念

1. 完全二叉树：二叉堆是一个完全二叉树，所以可以用数组来定位任一节点的left和right。
2. left的定位：node=tree[i],node.left=tree[2*i+1]
3. right的定位：node=tree[i],node.right=tree[2*i+2].
4. 最大堆：父节点总是大于左右子节点。 node>max(node.left,node.right)
5. 堆排序和最大堆的关系：最大堆不保证整个堆是顺序的。只满足上述第3点的性质。堆排序反复每个点构建最大堆，最终实现了排序。
6. 最后一个非叶子：tree[tree.length / 2 - 1]

堆排序的步骤

1. 构建一个最大堆
2. 除去最大值，将余下的元素构建一个最大堆

详细步骤如下

1. 构建一个最大堆
2. 从最后一个非叶子节点到root，每个节点调整node，node.left,node.right，保证node是最大值。
3. 调整后的left，要保证left >(left.left,left.right).right也是如此

```
    public static void adjustHeap(int[] num, int start, int end)

    {
        int temp = num[start];
            //i是start的left节点，2*i+1是i的left节点，一直向上检查
        for(int i=2*start+1; i<=end; i = 2*i+1)
        {
            //找左右儿子中的最小值
            if(i<end&&num[i+1]<num[i])
                i++;
            if(temp<=num[i])
                break;
            num[start] = num[i];
            start = i;
        }
        num[start] = temp;
    }
  //从最后一个叶子节点，开始调整。每个节点的调整范围从他自己到结束。
  public static void heapSort(int[] a) {
      int i;
      for (i = a.length / 2 - 1; i >= 0; i--) {// 构建一个大顶堆
          adjustHeap(a, i, a.length - 1);
      }

  }```
```

2. 堆排序
 1. 交换最大值和最后一个元素
 2. 调整前面n-1的元素，为最大。这时从下到小，是调整好的，主需要从root向下调整。

```
for (i = a.length - 1; i >= 0; i--) {// 将堆顶记录和当前未经排序子序列的最后一个记录交换

    int temp = a[0];
    a[0] = a[i];
    a[i] = temp;
    adjustHeap(a, 0, i - 1);// 将a中前i-1个记录重新调整为大顶堆
}```
```