# 堆树

1.优先队列：大家还记得我们上节课讲的赫夫曼树，我使用了一个优先队列大大减轻了我们的开发任务，但是大家知道这个优先队列内部是如何实现的呢？

2.如何实现一个用户热门搜索排行榜功能（微博热搜）？给你一个包含1亿关键词的用户检索的日志，如何取出排行前10的关键词。

给你的处理机器：2CPU 2G内存 一台。年龄，0~200

Map 分治 + 堆树

A b c d a

1.统计出现的频率。Hash

2.维护一个大小为10的大顶堆

时间复杂度，空间复杂度。

放到硬盘

完全二叉树：满

2.如何实现一个用户热门搜索排行榜功能（微博热搜）？给你一个包含1亿关键词的用户检索的日志，如何取出排行前10的关键词。

给你的处理机器：2CPU 2G内存 一台。年龄，0~200

Map 分治 + 堆树

A b c d a，有可能内存爆炸，因为数据太多。

放到硬盘。分治，分成很多份。1亿个我分成 10个文件。分布式，分库分表。我要知道我的数据在哪张表，hash%分表数

Hash%10=当前这个词放在哪个文件。

分成了10个文件后：分别求top10，然后再把这个top10合起来。也就是有100个数，再求一次。

衍生出很多情况：求新闻点击率

MapReduce思想，hadoop。

完全二叉树：满

堆是什么？堆是一种特殊的树，他需要满足以下两点：

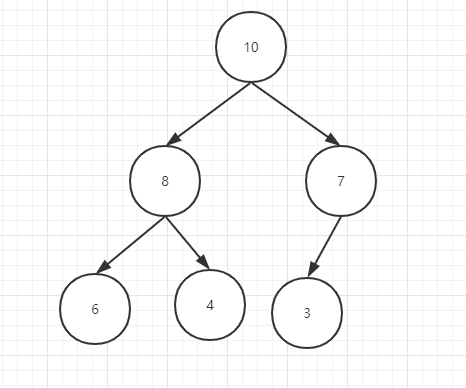
1.是一颗完全二叉树

2.其每一个节点的值都大于等于或者小于等于其左右子节点的值。

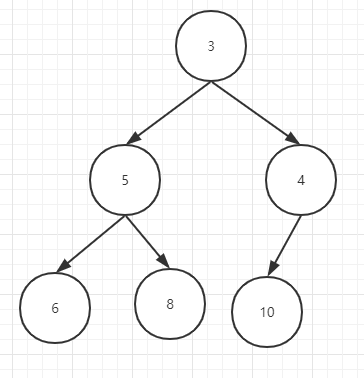
二叉搜索树：左节点小于根节点，右节点大于根节点

完全二叉树：除了最后一层，其他层每个节点都是满的且最后一层的节点都要靠左排列。

大顶堆：



小顶堆：



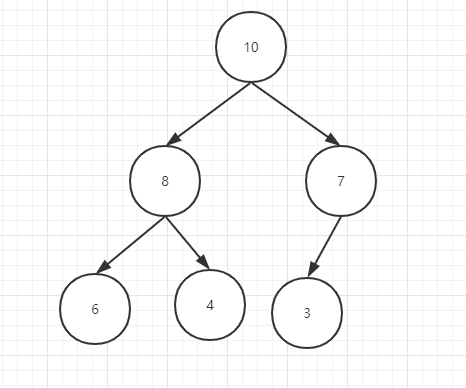
1.堆树如何来存储？

如果大家还记得我之前讲的，那肯定都很清楚完全二叉树的最佳存储结构就是数组。因为它有着特殊的属性，直接利用下标就可以表示左右节点

Node{

}

比如右边的结构：



1：10

2：8

3：7

4：6

5：4

6：3

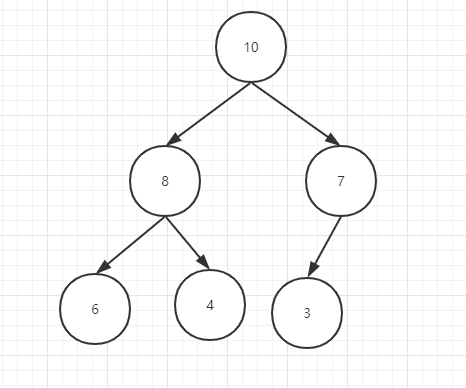
还记得我之前讲的那个左右子节点的公式吧：

左=2\*I i就是我们当前点所在数组的下标。

右=2\*i+1

如果下标从0开始开始 ，这两个公式：2\*i+1,2\*i+2

2.堆的插入操作:



堆的插入有两种实现方式：

（1）从下往上

（2）从上往下

其插入过程就叫做堆化。

思考下 删除任意一个节点怎么搞？

从下往上：加入我们在右图插入9，很显然此时插入9后我们是不满足堆树的性质的，那怎么办呢？

其实只要做下交换就可以了，直到依次往上做到不能交换为止，看下图。

完全二叉树。类似于满二叉树，32层就可以21亿+个点，n个数交换多少 次？Logn-1次

2.堆的插入操作:

堆的插入有两种实现方式：

（1）从下往上。

（2）从上往下

其插入过程就叫做堆化。

从上往下：其实就是把插入的点放到堆顶，然后依次往下比较即可。

3.堆顶的删除操作:

其实就是把堆顶拿掉，然后从下往上堆化就ok啦。

画图演示：

思考：为啥不能直接删除，然后用左右节点代替呢？ 会造成我们的树不是一颗完全二叉树，因为会少点。

堆操作的时间复杂度分析：

Logn

堆排序

大家还记得我之前讲排序的时候我们还有一个堆排序没讲，今天我们来看看，如果利用这个堆树进行排序呢？

假设给你一个序列：

8 4 20 7 3 1 25 14 17

利用堆树进行排序：

1.先按照序列顺序存储在完全二叉树中。

建堆：

2.从最后一个非叶子节点堆化。为什么是最后一个非叶子节点而不是最后一个叶子节点呢？

看例子：

一种：从无到有，就是从头往后建

二种：修正，从后往前。从数组的后面往前走

大家还记得我之前讲排序的时候我们还有一个堆排序没讲，今天我们来看看，如果利用这个堆树进行排序呢？

假设给你一个序列：

8 4 20 7 3 1 25 14 17

利用堆树进行排序：

1.先按照序列顺序存储在完全二叉树中。

2.从最后一个非叶子节点堆化。为什么是最后一个非叶子节点而不是最后一个叶子节点呢？

代码实现:

1.优先级队列

2.TOP K问题，比如给你一串1000万的数字 求前k大的数。

一种是静态的数据

一种是动态的的数据

3.定时任务