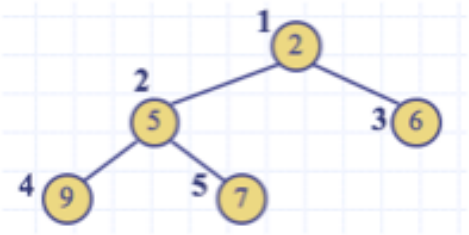
堆heap

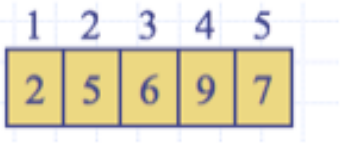
本质是个二叉树，顶小，是个平衡的，但是实现方式和平衡二叉树完全不同。

存储其实是在数组里，下图1，2，3，4，5为其id索引，里面放数据。

堆很容易知道—— 左儿子索引号2*id，右儿子2*id+1，父亲id。

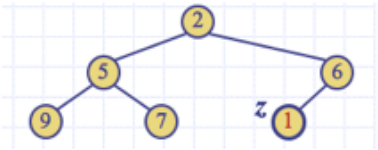


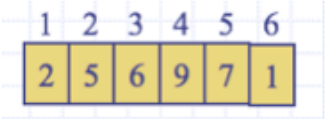
本质数组为：



heap.insert()　//heap.push()

如插入“1”，本质上数组末尾id索引为6的位置，加入了value1。





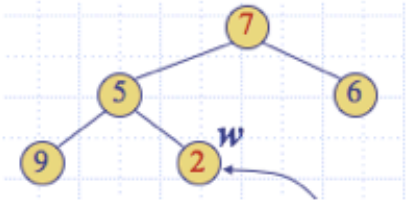
发现1不满足堆性质（小堆），1儿子＜6父，交换儿子父亲—> upheap算法。(swap数组)

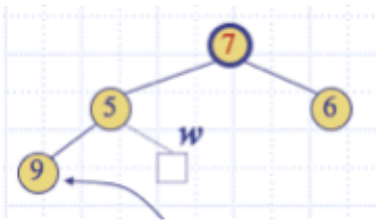
时间复杂度O(logn)

Heap.remove() // heap.pop()

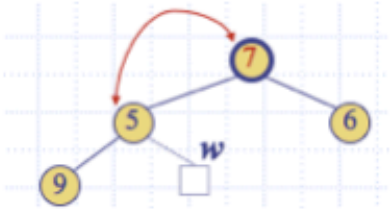
如删除2（根/1号节点）

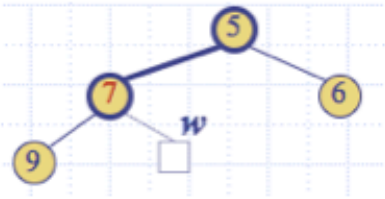
首先交换数组头尾的元素，直接删除数组末尾，则删了（根 / 1号节点了）





发现7不满足堆性质（小堆），7父 > 5,6儿子，父亲交换左右儿子里最小的5 —> downheap算法。





我们发现，叶子结点不用做downheap(一半不用做)，相当于倒叙处理数组。

第k层有2^(h-k)个节点，总共要做downhead的话，k从1到h, 2^(h-k)和次操作 = c*2^h = c*2^logn = c*n，(h = logn)

时间复杂度O(logn)，worse case是O(n)