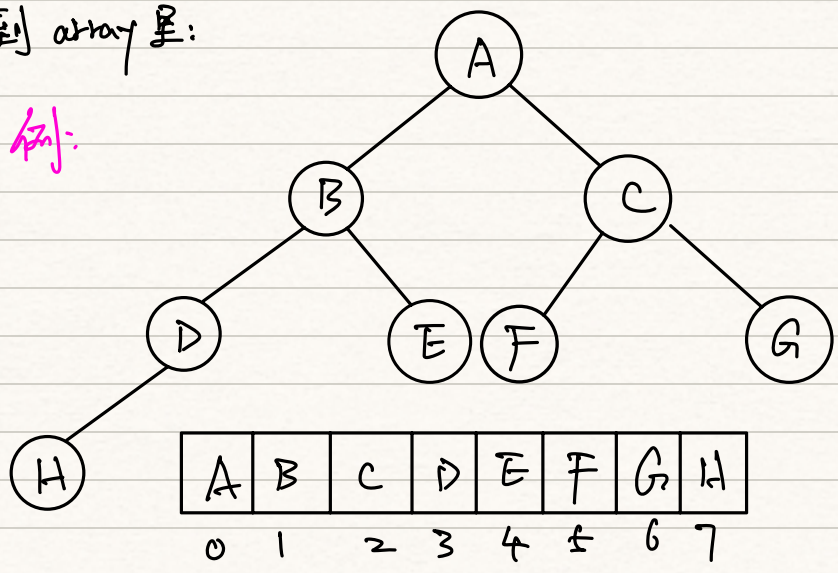
将某个 index 为 x 的 node 对应到 array 里：

left child: $(2x+1)$

right child: $(2x+2)$

Parent: $\left\lfloor \frac{x-1}{2} \right\rfloor$

例：



A	B	C	D	E	F	G	H
0	1	2	3	4	5	6	7

H 的 Parent: $\left\lfloor \frac{7-1}{2} \right\rfloor = 3 \rightarrow D$

E 的 left child: $2 \cdot 4 + 1 = 9 \rightarrow$ 无 child

C 的 right child: $2 \cdot 2 + 2 = 6 \rightarrow G$

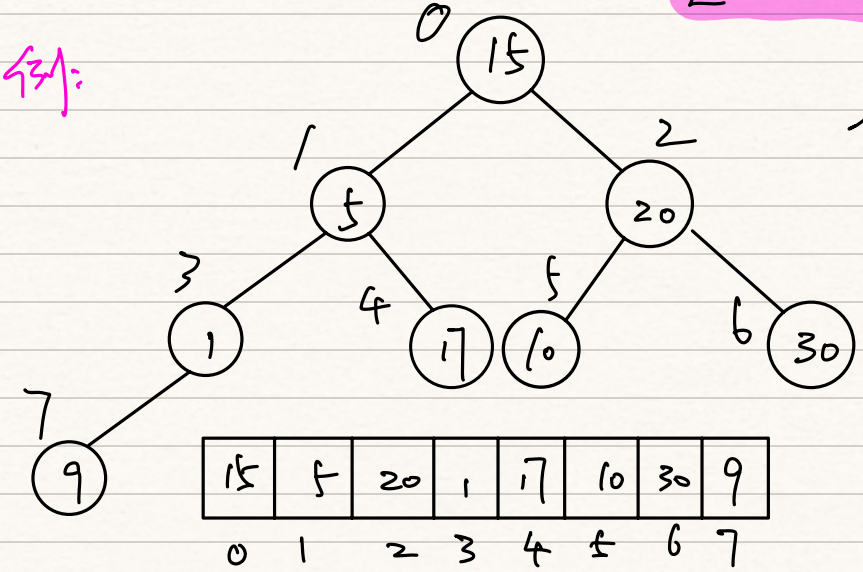
Heap

添加：从右往左!!!

用 bottom-up construction 创建一个 heap，可以在 $O(n)$ 时间内完成，每这个方法称为 heapify. Heapify 需要从第一个 index 最大的 non-leaf node 开始. (因为是 array 来存)

计算所有 leaf node 的 index 范围：$\left[\left\lfloor \frac{n}{2} \right\rfloor + 1, n\right]$ n 为 array heap 的长度.

例：



15	5	20	1	17	10	30	9
0	1	2	3	4	5	6	7

leaf node: 5-8

$\left\lfloor \frac{8}{2} \right\rfloor + 1 = 5$

$\therefore [5, 8]$

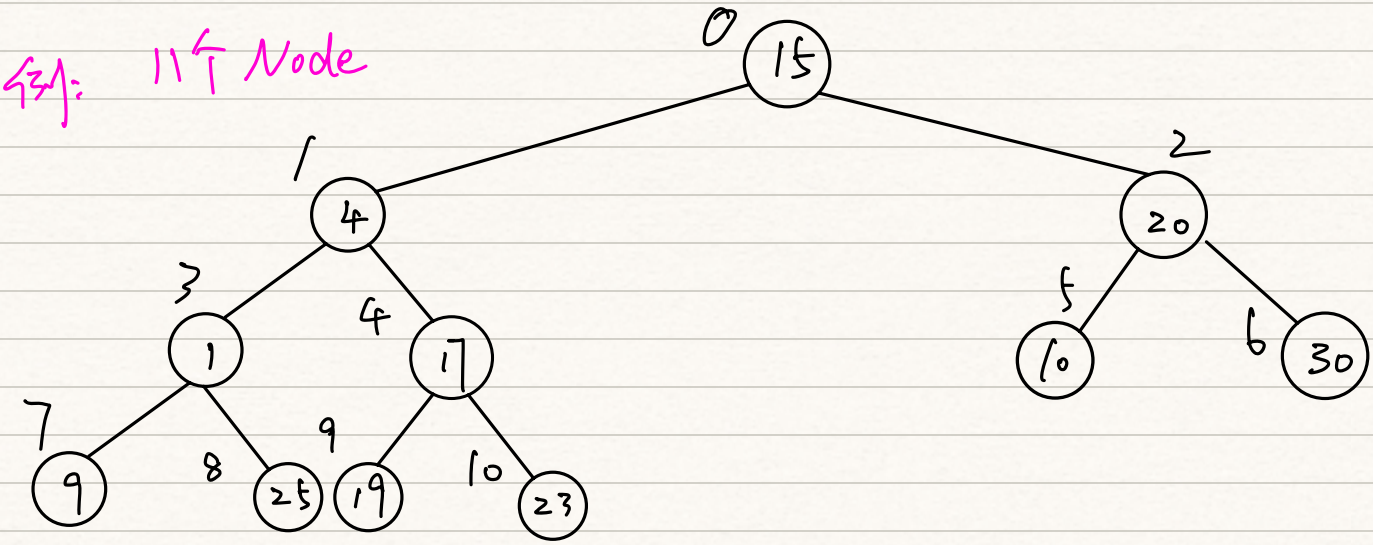
注，跟据实际实现 0 或 1 based，调整这个 $\lfloor \frac{n}{2} \rfloor$ 公式.

所以，heapify 从 $\left\lfloor \frac{n}{2} \right\rfloor - 1$ 开始!!!

删除：在heap中删除，永远都是删除root.（也叫 down heap）

为了实现heap sort，in place，我们将root与最后一个leaf交换位置，即可实现。

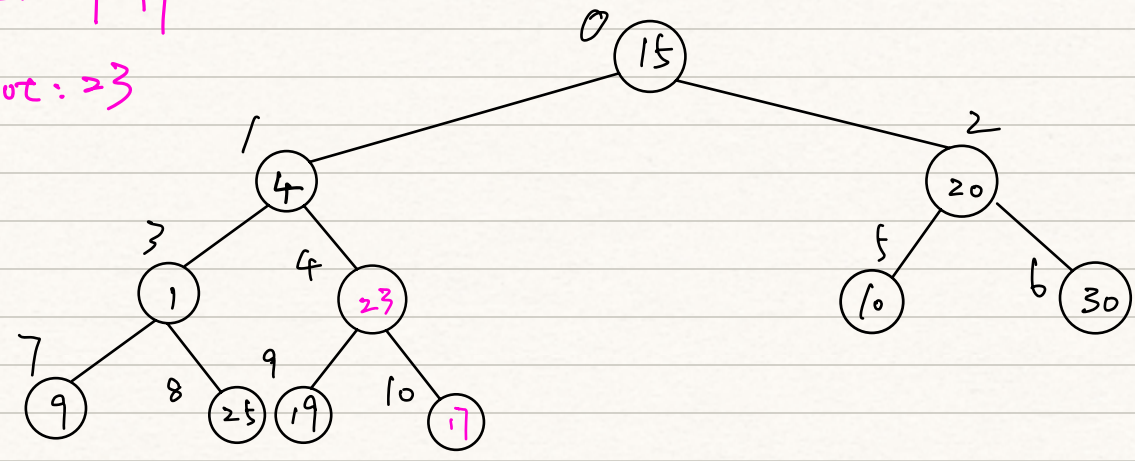
例：11个Node



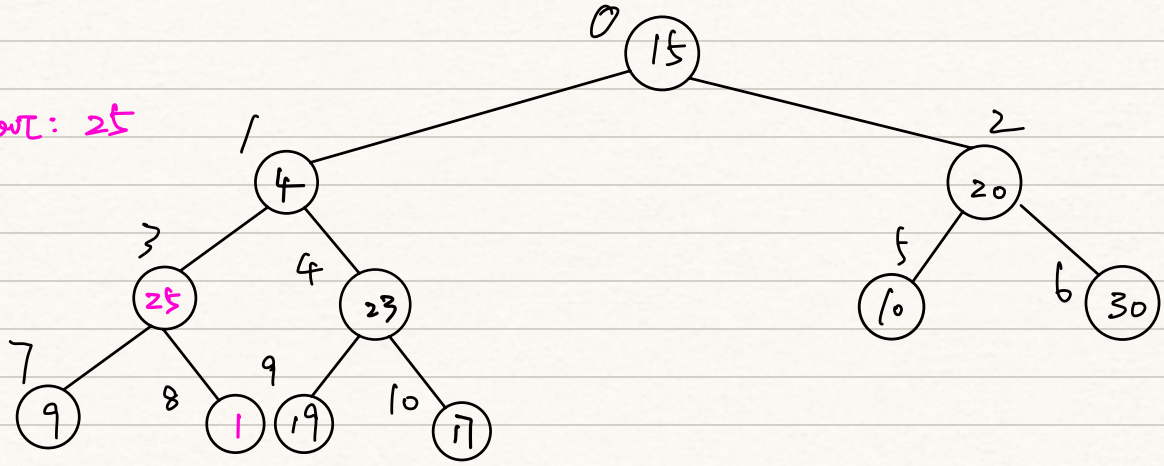
15	4	20	1	17	10	30	9	25	19	23
0	1	2	3	4	5	6	7	8	9	10

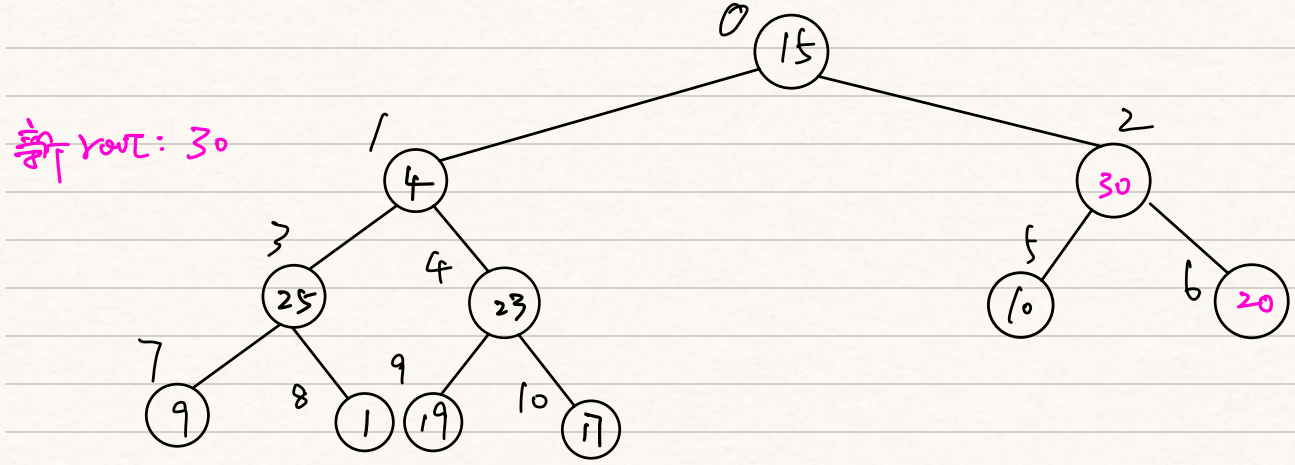
开始heapify：

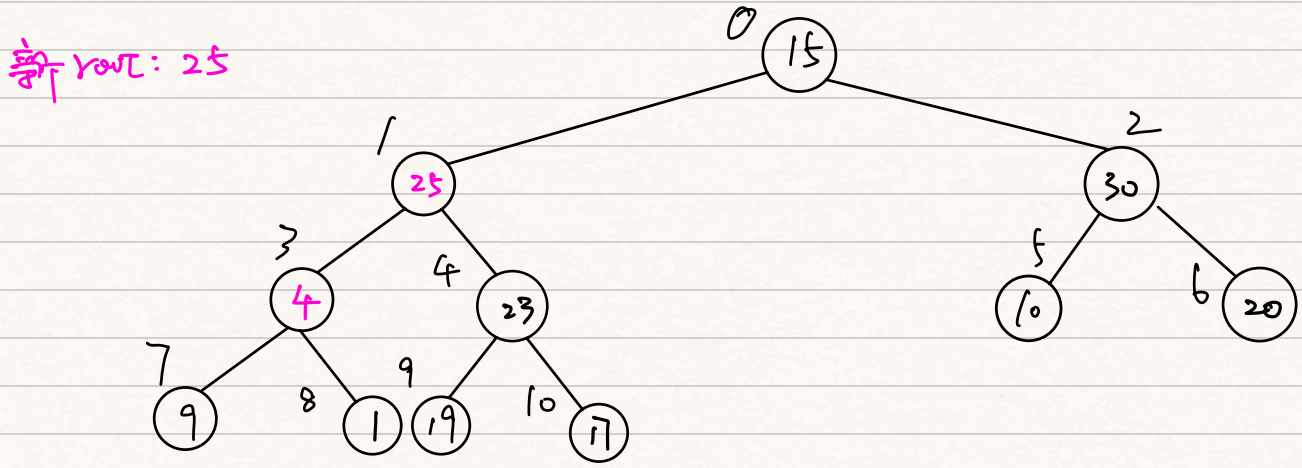
新root：23

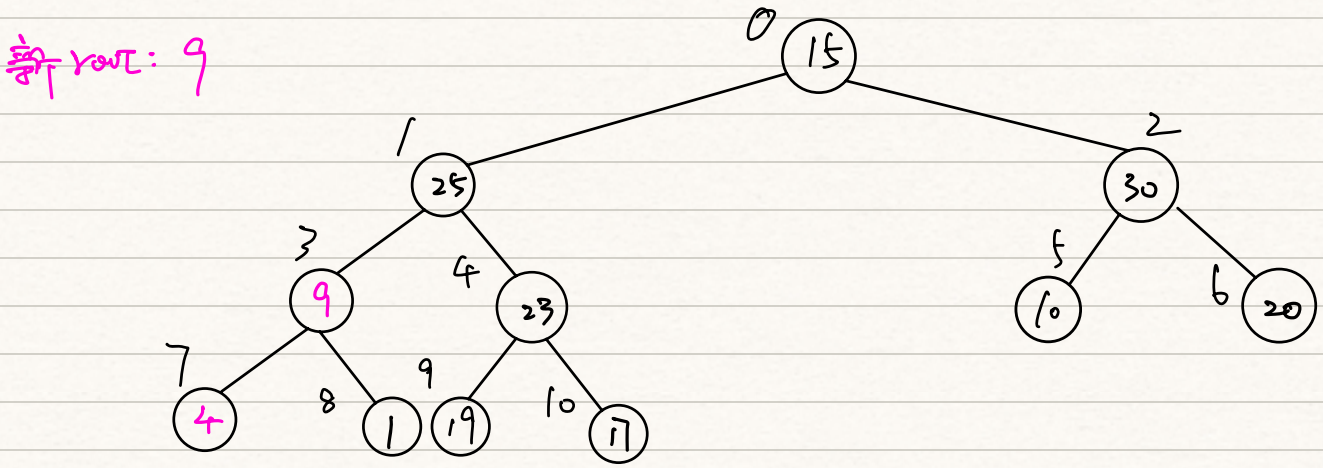


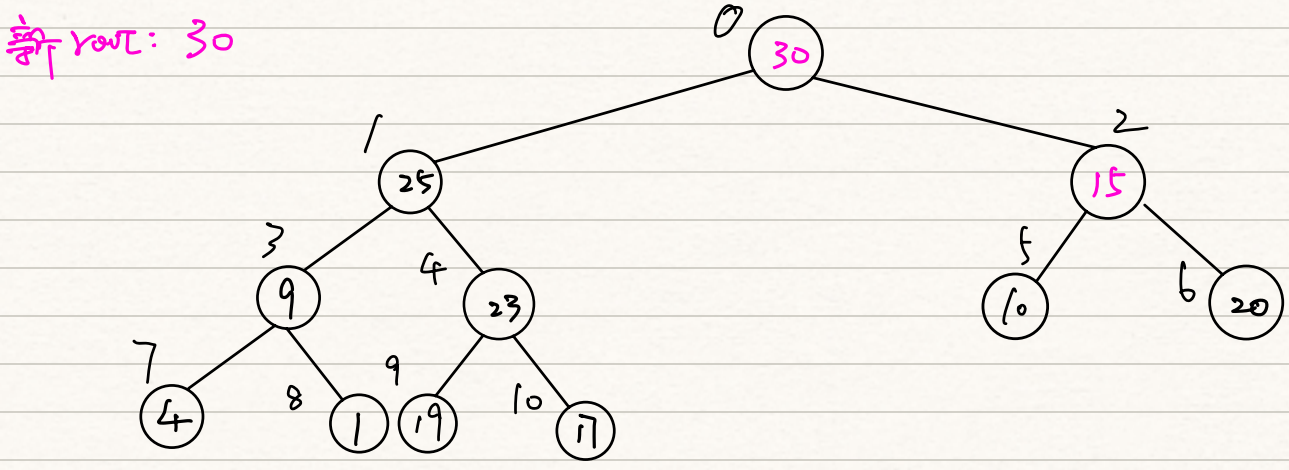
新root：25



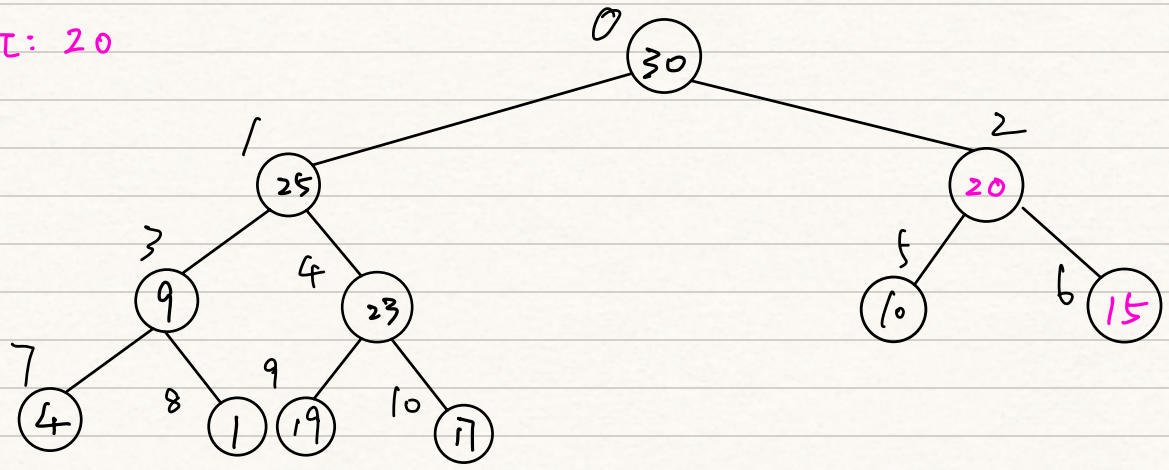

新root: 30
0
15
1
4
2
30
3
25
4
23
5
10
6
20
7
9
8
1
9
19
10
17


新root: 25
0
15
1
25
2
30
3
4
4
23
5
10
6
20
7
9
8
1
9
19
10
17


新root: 9
0
15
1
25
2
30
3
9
4
23
5
10
6
20
7
4
8
1
9
19
10
17


新root: 30
0
30
1
25
2
15
3
9
4
23
5
10
6
20
7
4
8
1
9
19
10
17

新root：20



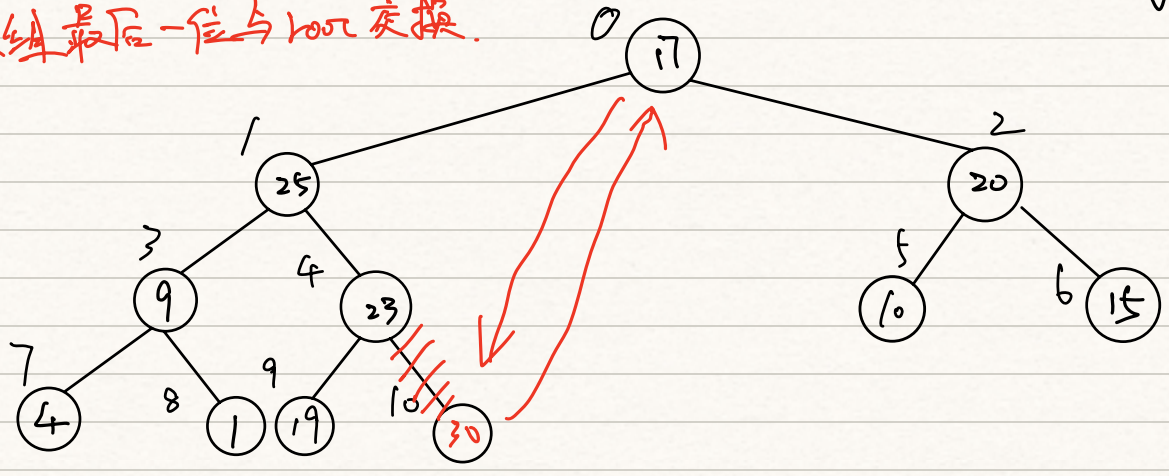
最终数组：

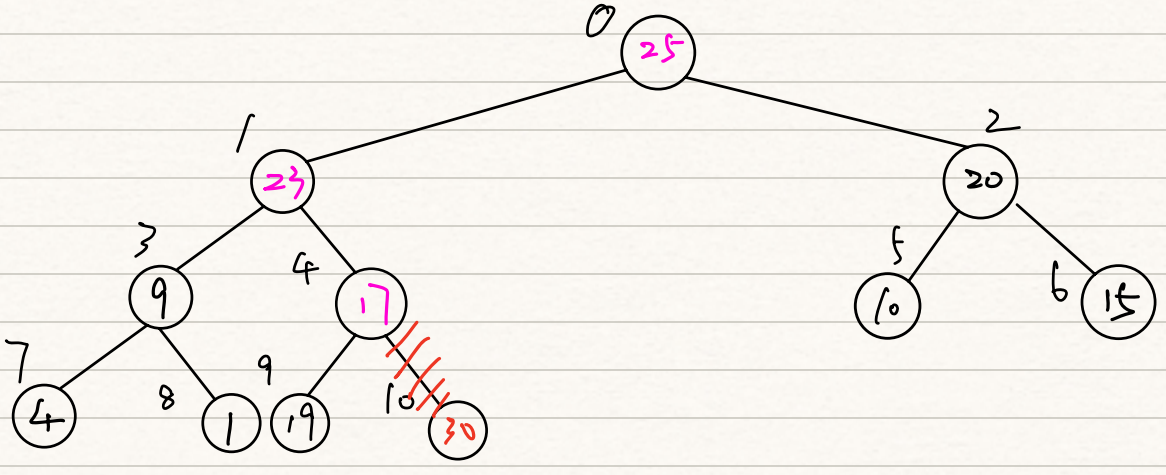
30	25	20	9	23	10	15	4	1	19	17
0	1	2	3	4	5	6	7	8	9	10

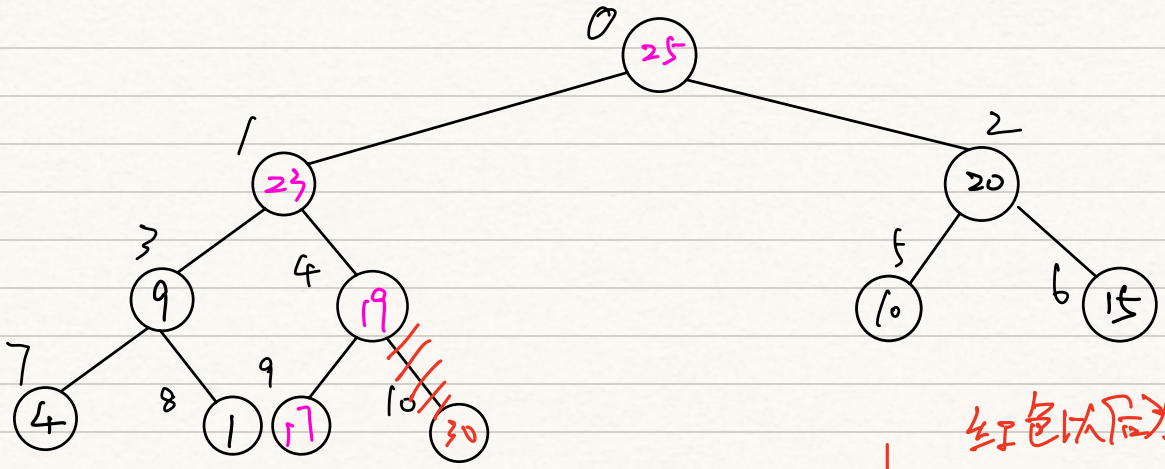
排序 (Heap Sort, in place!)

从root 删除，一直到最后，也是最小的，就会形成一个 ascending 的 array.

将数组最后一位与root交换.

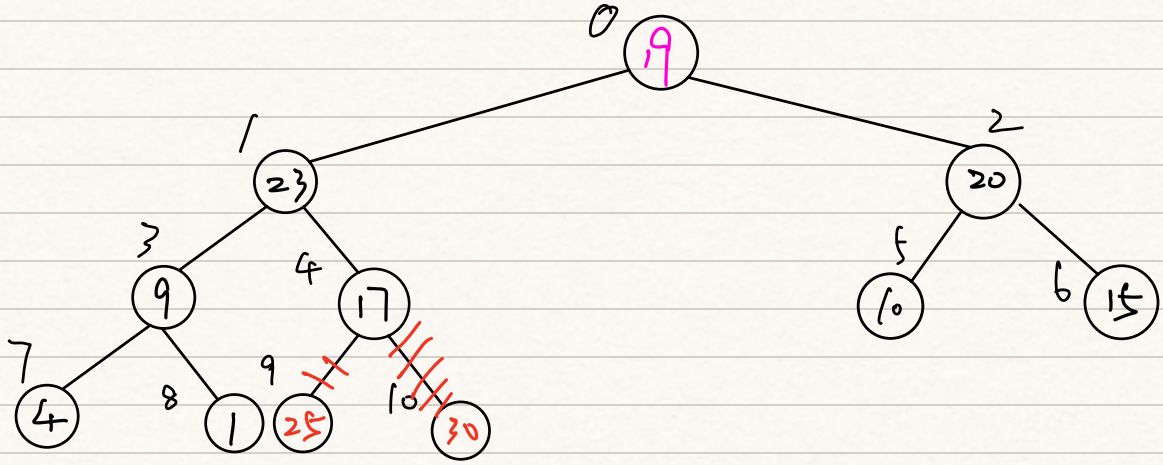


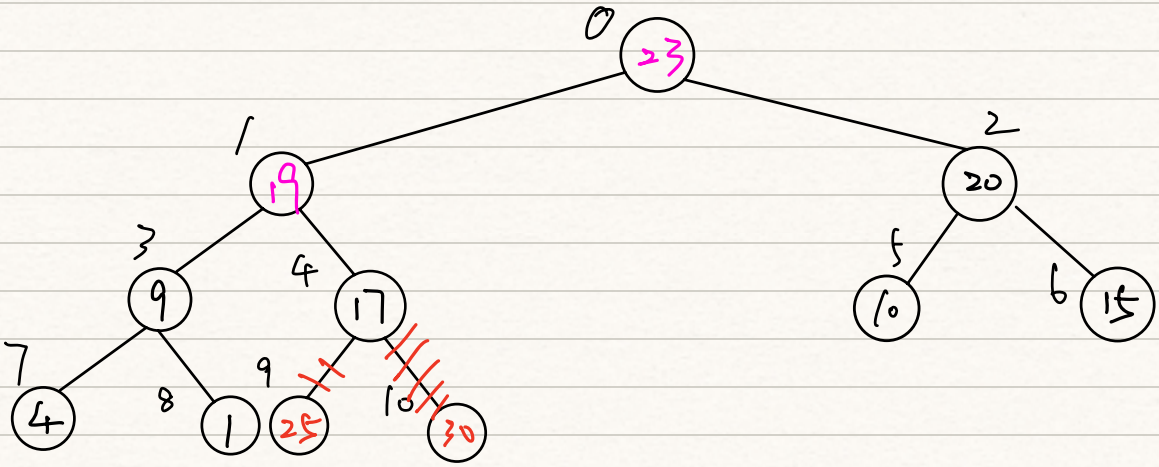




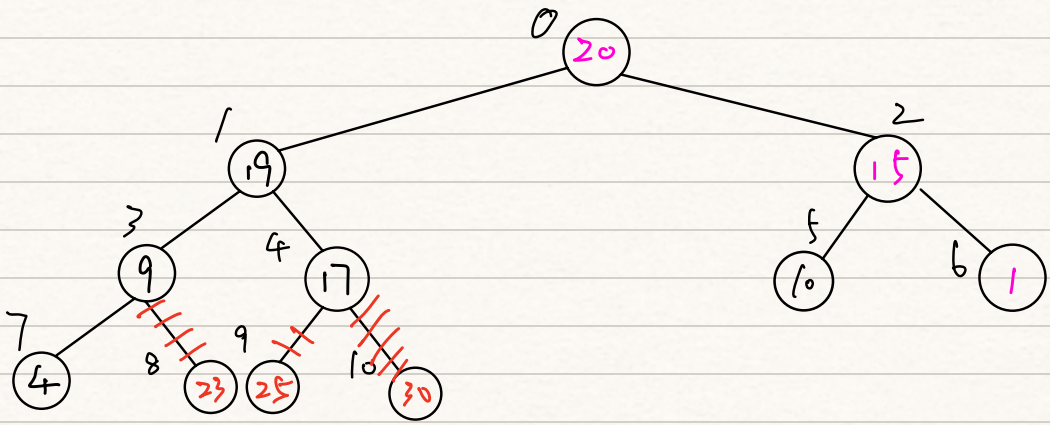
红色以后为 sorted

25	23	20	9	19	10	15	4	1	17	30
0	1	2	3	4	5	6	7	8	9	10

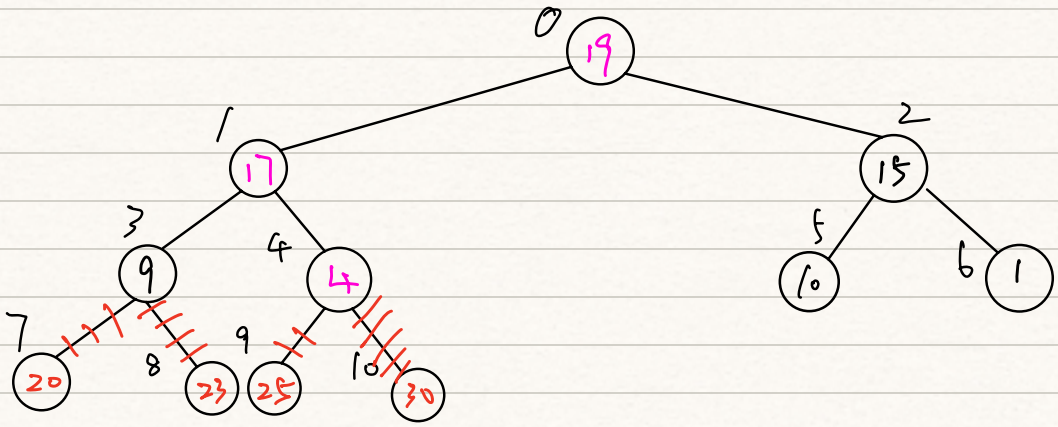




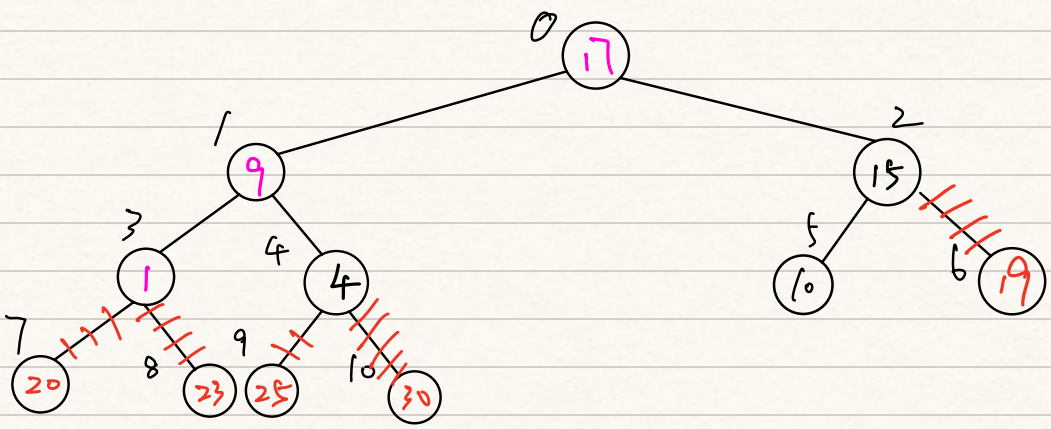
23	19	20	9	17	10	15	4	1	25	30
0	1	2	3	4	5	6	7	8	9	10



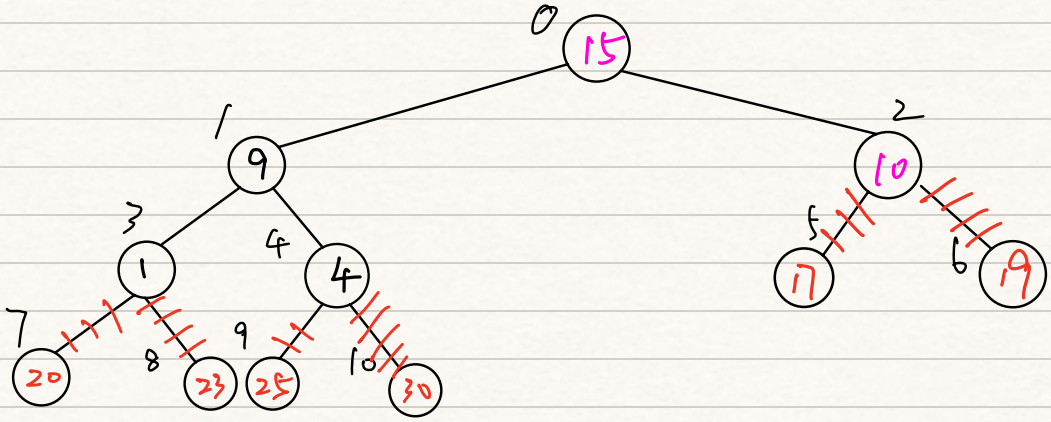
20	19	15	9	17	10	1	4	23	25	30
0	1	2	3	4	5	6	7	8	9	10



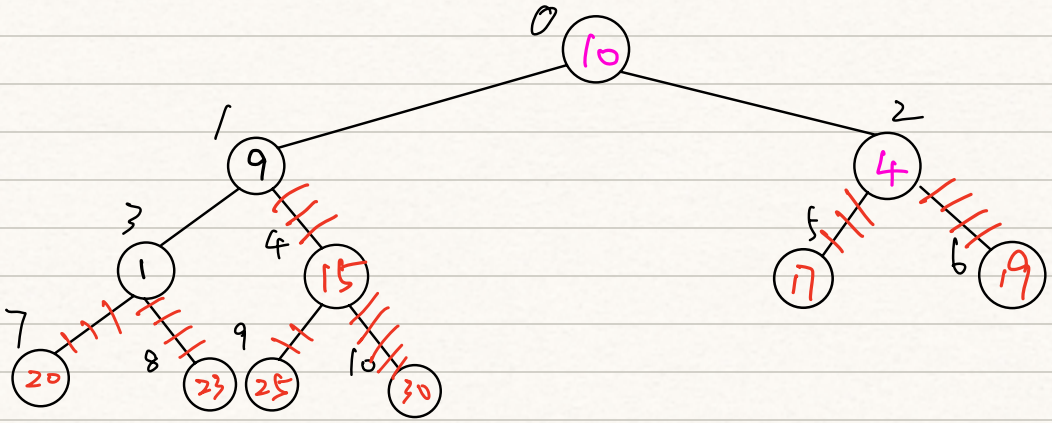
19	17	15	9	4	10	1	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10



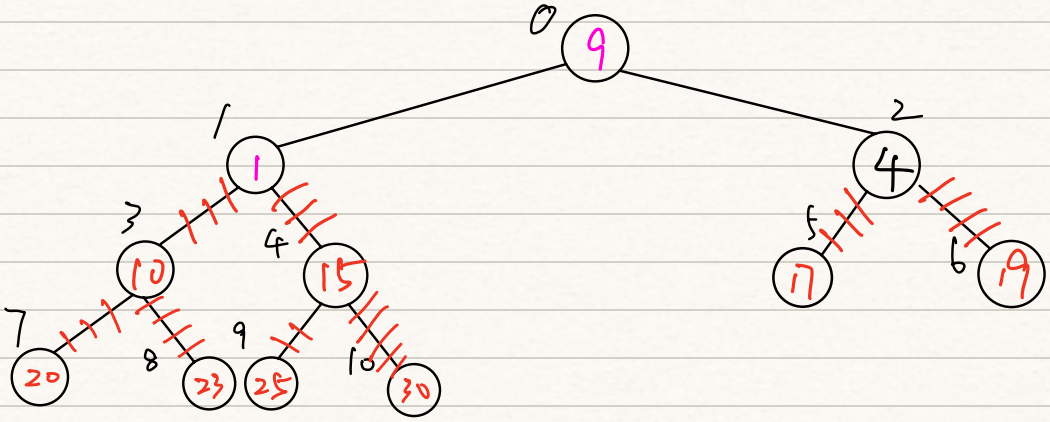
17	9	15	1	4	10	19	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10



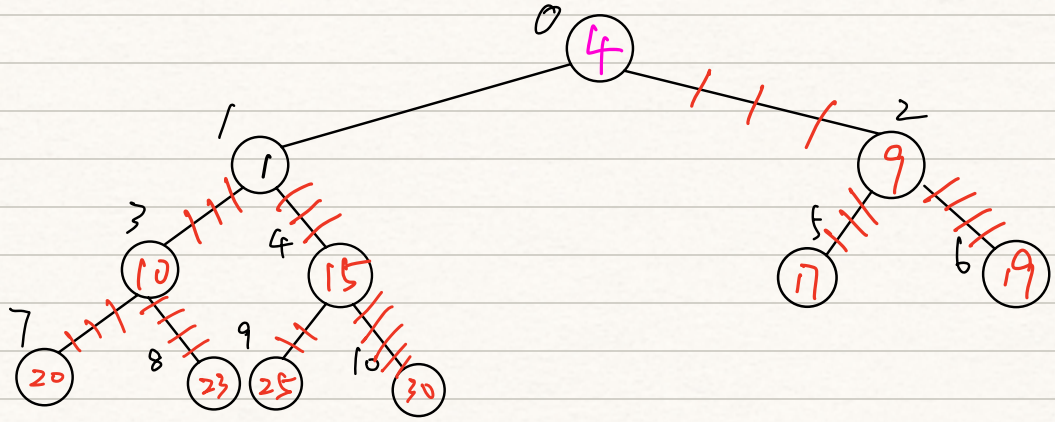
15	9	10	1	4	17	19	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10



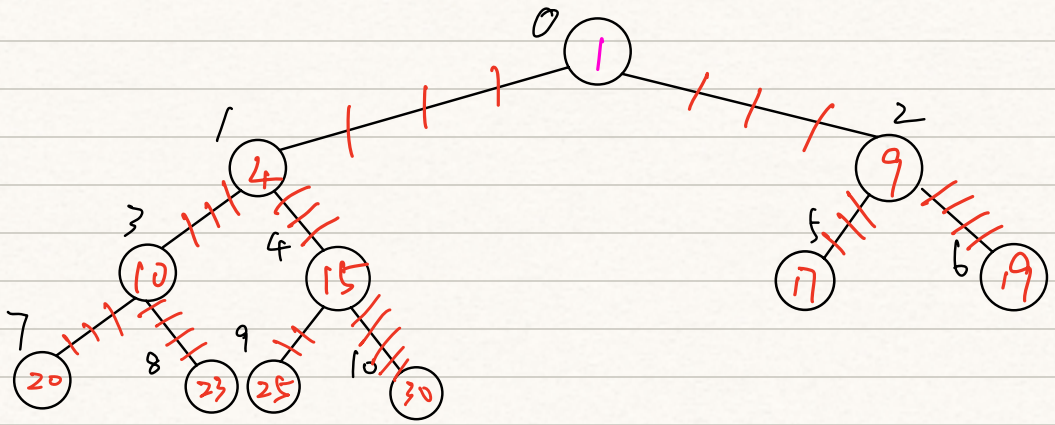
10	9	4	1	15	17	19	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10



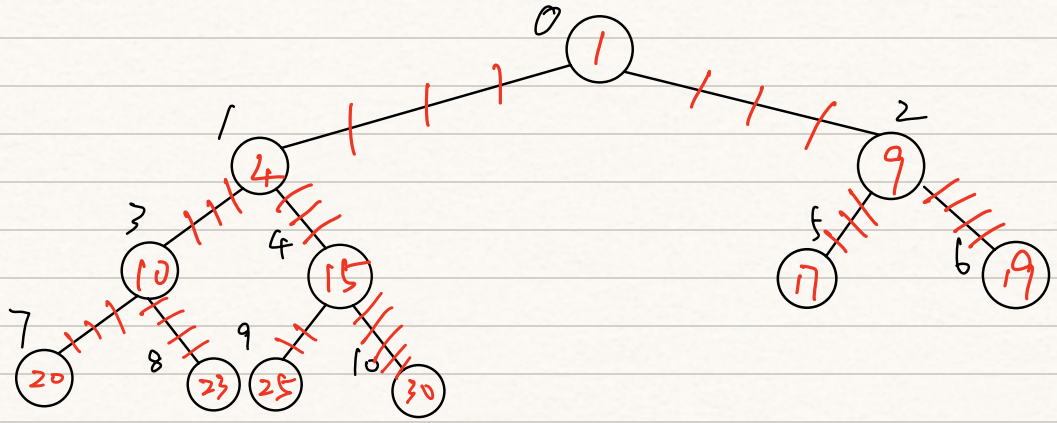
9	1	4	10	15	17	19	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10



4	1	9	10	15	17	19	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10



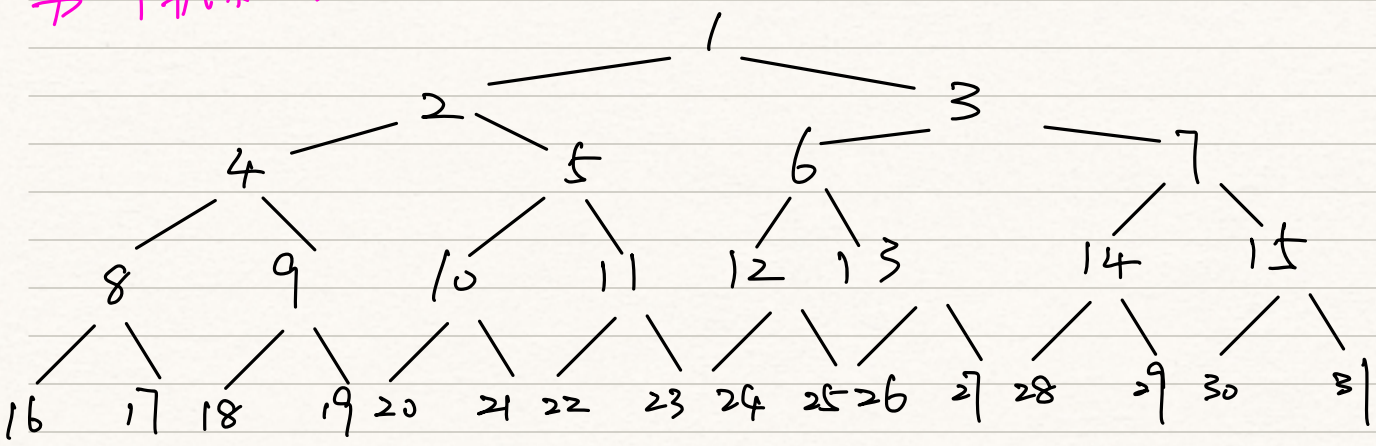
1	4	9	10	15	17	19	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10



1	4	9	10	15	17	19	20	23	25	30
0	1	2	3	4	5	6	7	8	9	10

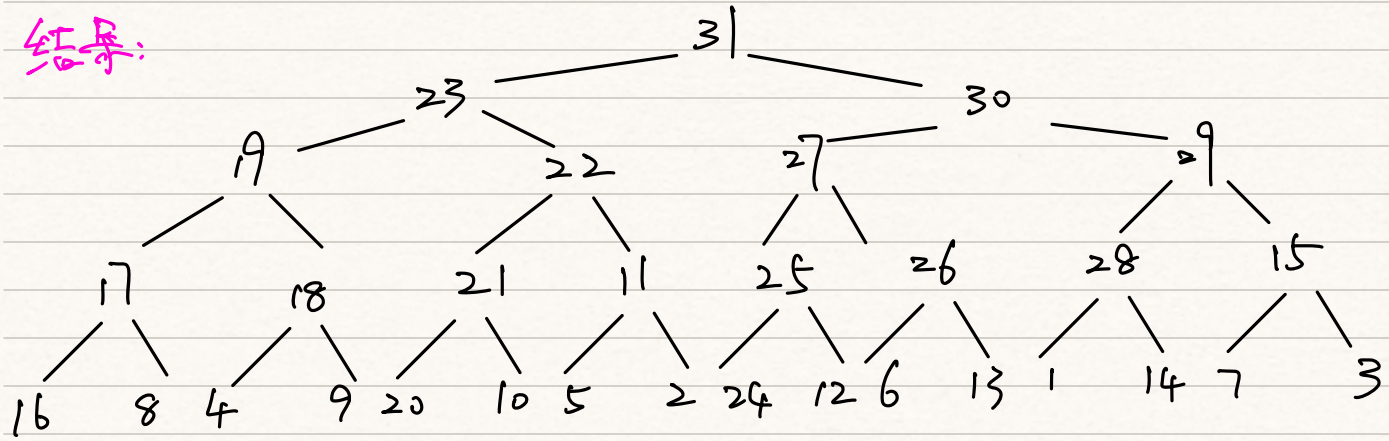
完成!!!

另一个疯狂的试验：



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

结果：



31	23	30	19	22	27	29	17	18	21	11	25	26	28	15	16	8	4	9	20	10	5	2	24	12	6	13	1	14	7	3
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30