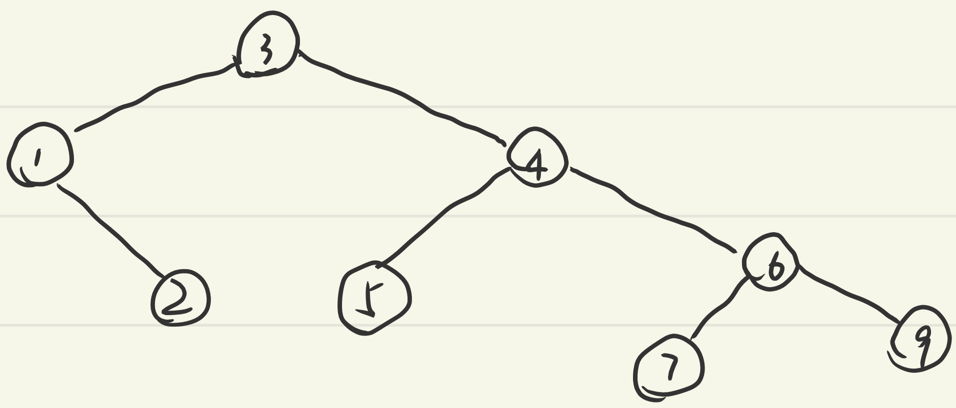
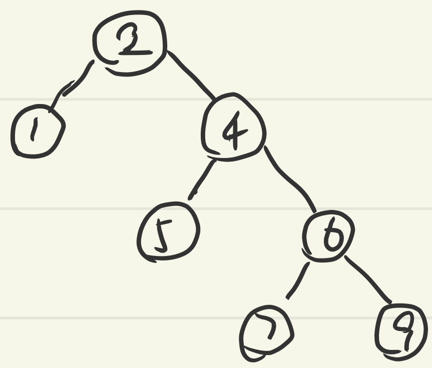
4.1

1. a.



b. 用左子树最大节点代替



2 Node 类中 属性 leftSize 存储左子树节点数+1

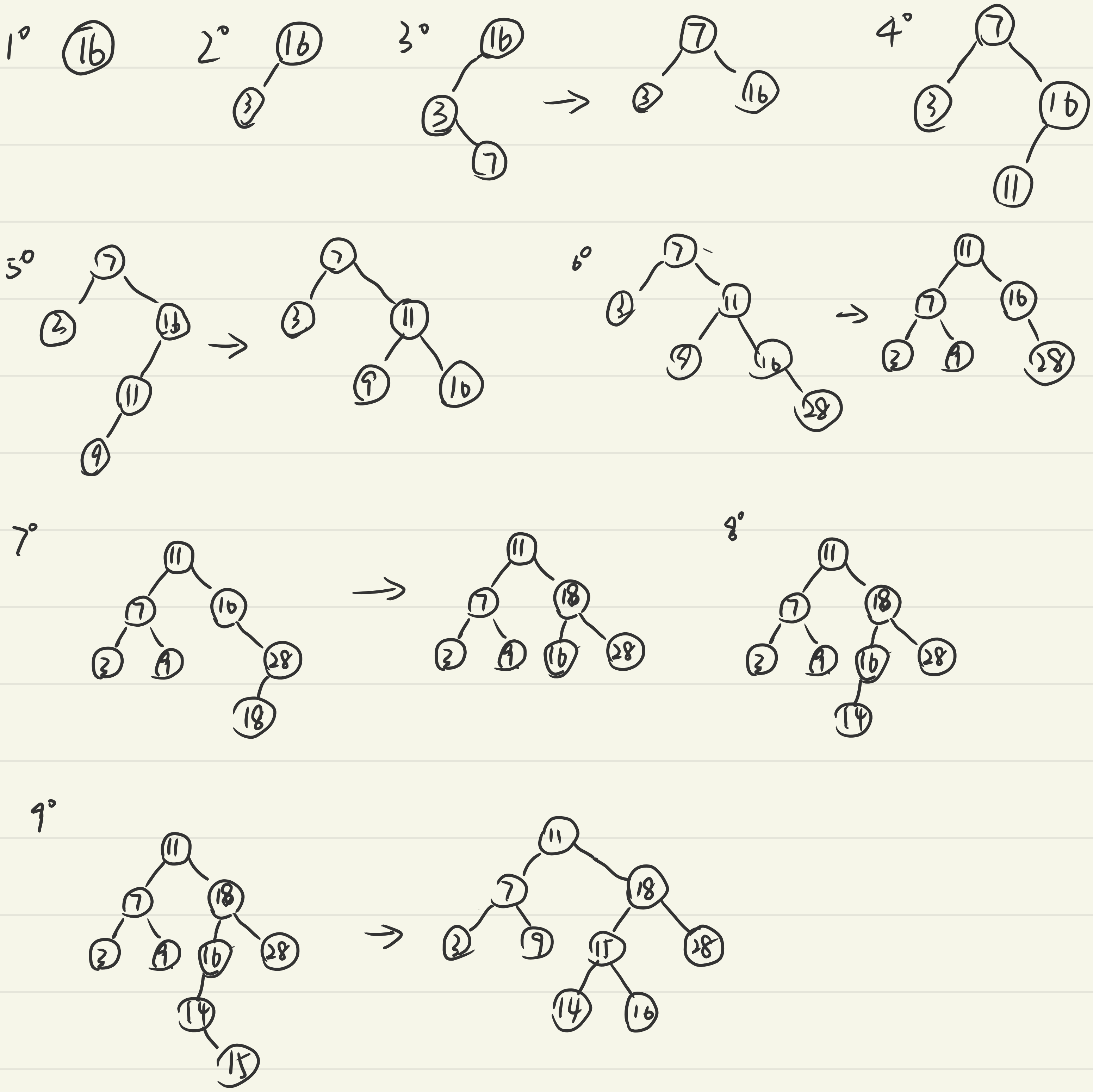
```
public static Node findkthMin(int k, Node root){
    if ( k == 0 ) return root;
    (if (root.left == null) return null;)

    return findkthMin(k-1, root.left);
}
```

```
if (k < root.leftSize) // 左子树
return findkthMin(k, root.left);
```

```
        return findKthMin(k, root.left);
    else if (k > root.leftsize)   //右子树
        return findKthMin(k - root.leftsize, root.right);
    else
        return root;
```

3.




```
4. public boolean isBST(BinaryNode root){
       if (root == null)   return true;
       boolean judgeLeft = true, judgeRight = true;
       if (root.left != null)
           judgeLeft = (root.left.element < root.element) && (isBST(root.left));
       if (root.right != null){
           judgeRight = (root.right.element > root.element) && (isBST(root.right));
```

judgeRight = (root.right.element > root.element) && (isBST(root.right));

reture judgeLeft && judgeRight;

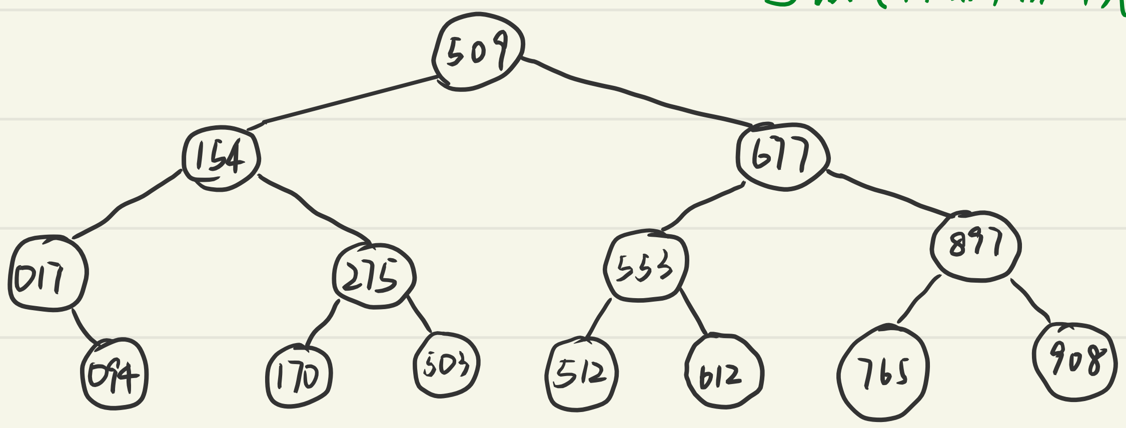
}

①得到中序遍历.检查是否为升序

② findMax (root.left) < root.element < findMin(root.right)　　递归

5 　二分查找判定树：　A[1, n] 是一个有序表，① 取 $m = (1+n)/2$，以为m为根，画出结点。

② 以 (1, m', m-1) 画左子树，以 (m+1, m*, n) 画右子树



$$\frac{\cancel{0}^{1} + \cancel{1}^{2}\times 2 + \cancel{2}^{3}\times 4 + \cancel{3}^{4}\times 7}{14} = \cancel{\frac{31}{14}}\ \frac{45}{14}$$

（根的搜索长度为1）

6. 不成立

对于 $\forall a \in A$ 在树中找到 a 在 S_2 上的最近祖先 f　则 a 在 f 的左子树上

b 在经过 f 到根结点的路径上　　因此有可能 f 在 b 的右子树上，因而 a 也在 b 的右子树上.

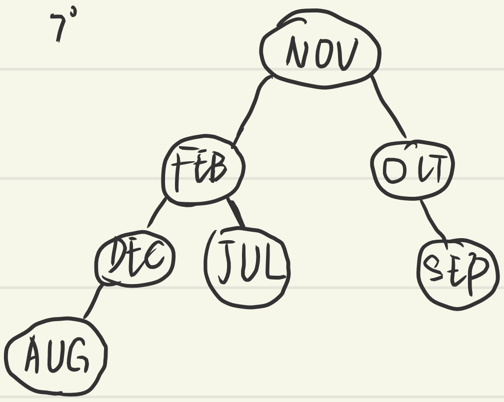
此时 $a > b$　因此 $a < b$ 不成立

同理 $b < c$ 不成立

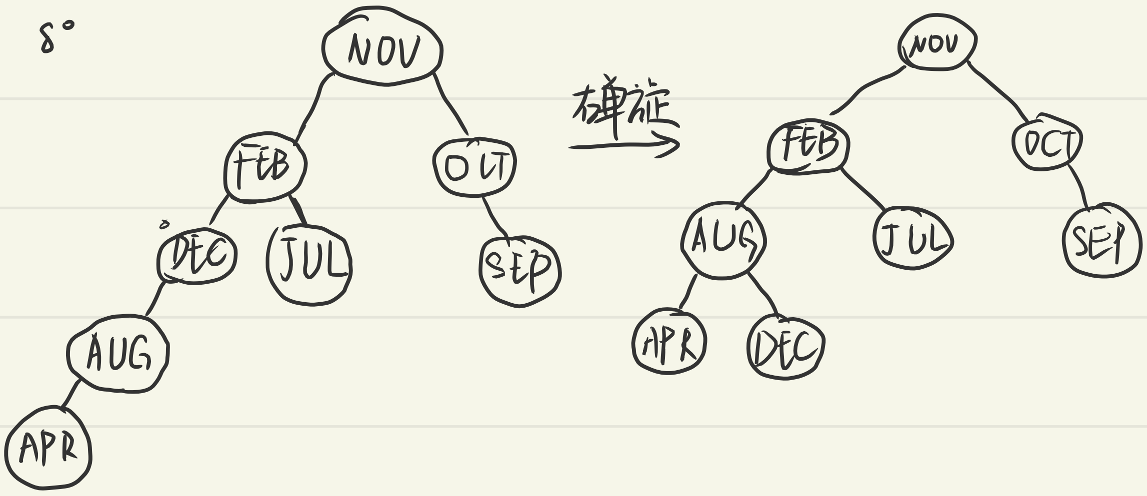
（而对于 $\forall a \in A$, $\forall c \in C$, 均有 $a < c$）

7.

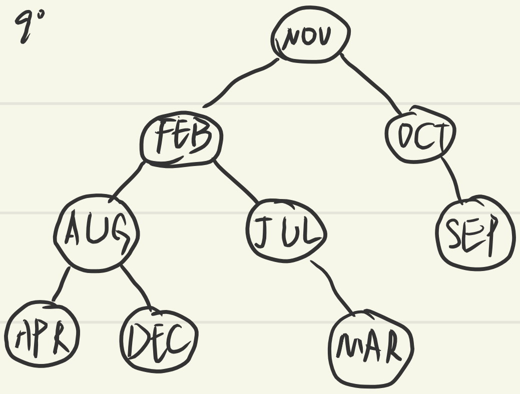
7°



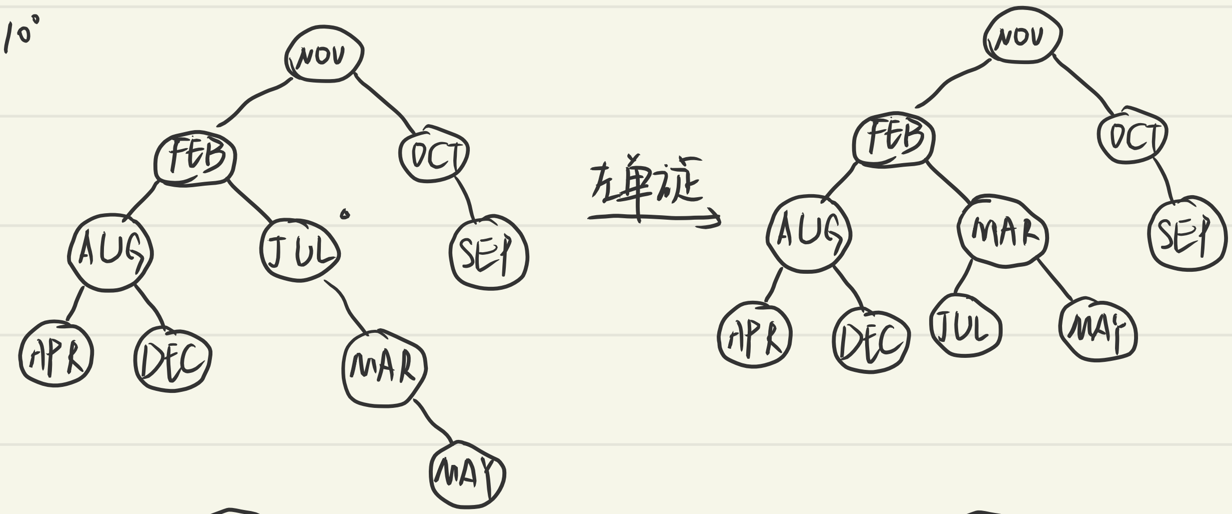
8°



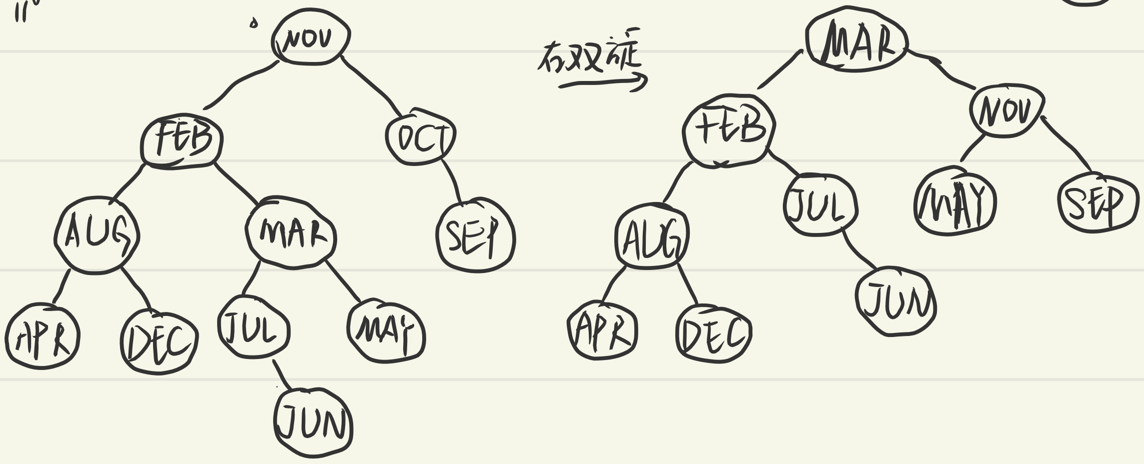
9°



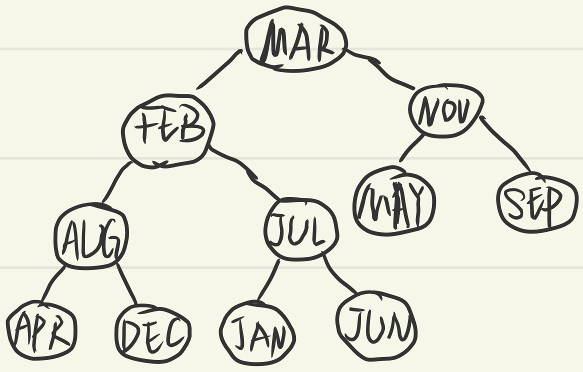
10°



11°



12°



8 设N_h是高度为h的AVL树的最小结点数。最坏情况下，树的一棵子树高度为h-1，另一棵子树高度为h-2，这两棵子树也是高度平衡的。因此有

空树 $N_0=0$，仅有根节点 $N_1=1$　　$N_h=N_{h-1}+N_{h-2}+1$ $(h>1)$

对于斐波那契数列 $F_0=0$ $F_1=1$ $F_n=F_{n-1}+F_{n-2}$　　$h\geq1$时有 $N_h=F_{h+2}-1$

F_n	0	1	1	2	3	5	8
N_n	0	1	2	4	7		

$$F_n=\frac{\phi_1^n-\phi_2^n}{\sqrt5}\qquad \phi_1=\frac{1+\sqrt5}{2}\qquad \phi_2=\frac{1-\sqrt5}{2}$$

$$\Rightarrow N_h=\frac{\phi_1^{h+2}-\phi_2^{h+2}}{\sqrt5}-1>\frac{\phi_1^{h+2}}{\sqrt5}-2\qquad \left|\frac{\phi_2^{h+2}}{\sqrt5}\right|<1$$ ⇐高度为h的AVL树的最少结点数

$$\Rightarrow \phi^{h+2}<\sqrt5(N_h+2)$$

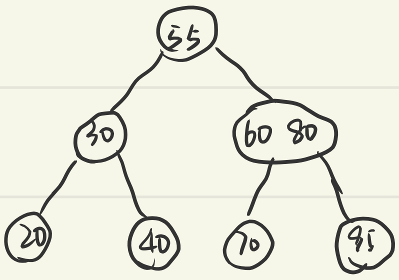
$$h+2<\log_\phi\sqrt5+\log_\phi(N_h+2)\qquad \log_2\phi=0.694$$

可得 $h<1.44\times\log_2(N_h+2)-0.328<1.44\times\log_2(n+2)$ (换底公式)

因此 有n个结点的AVL树，高度不超过 $1.44\times\log_2(n+2)$

设AVL树每一层结点个数都达到最多 (2^{i-1}, i为层号)　AVL树的高度最小, $h = \lceil \log_2(n+1) \rceil$

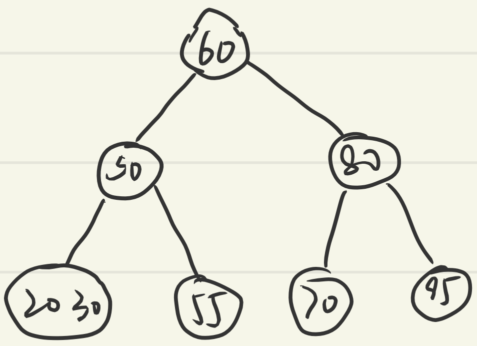
9. 1° delete 50



3阶. 子结点 [2, 3]

关键字 [1, 2]

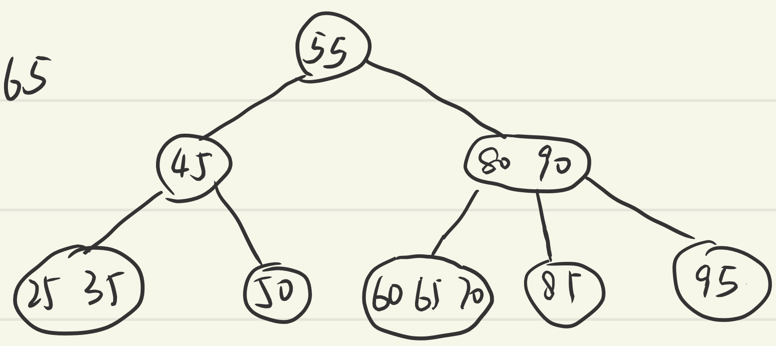
2° delete 40



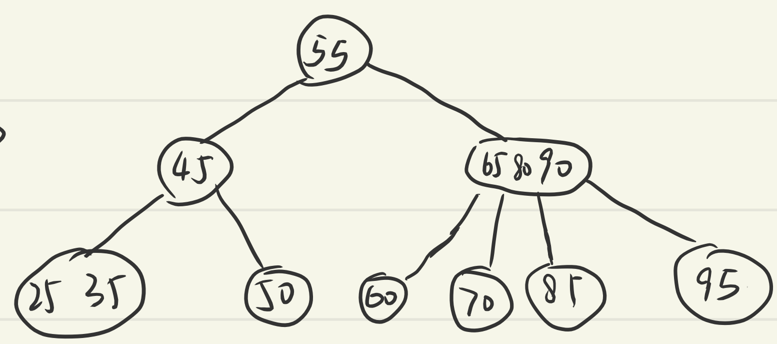
10. 1° insert 65

3阶

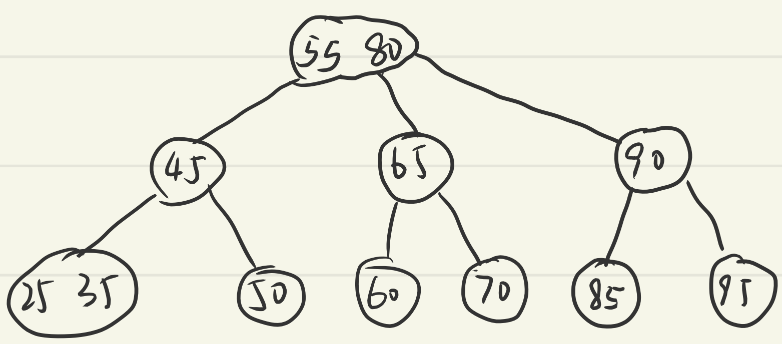
key 1~2个



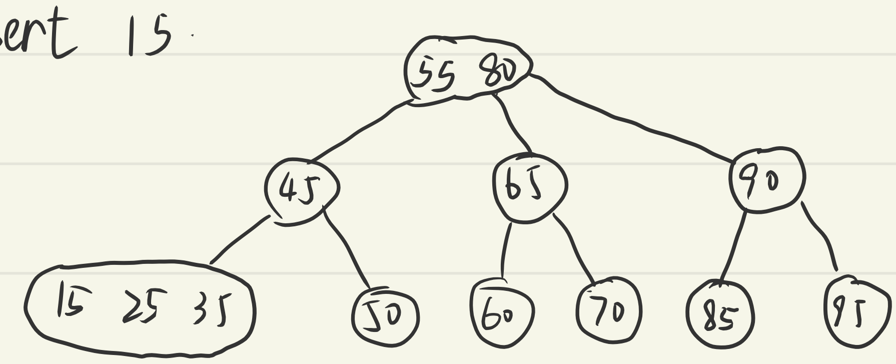
→



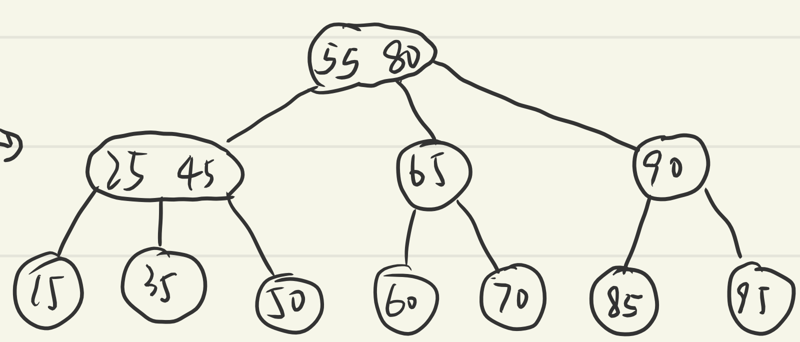
→



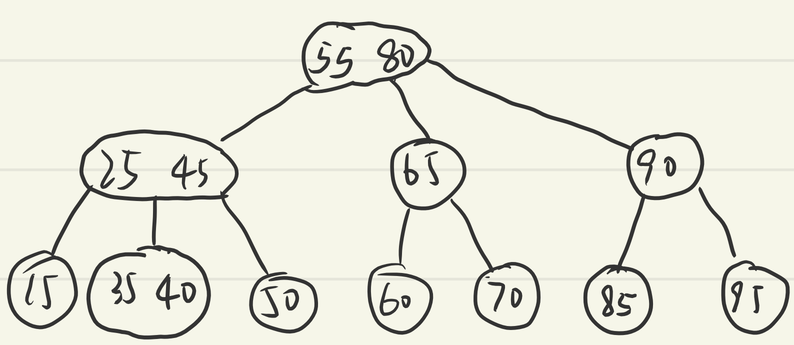
2° insert 15



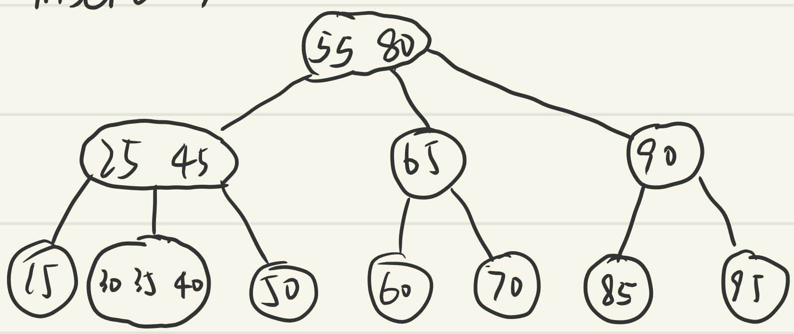
→

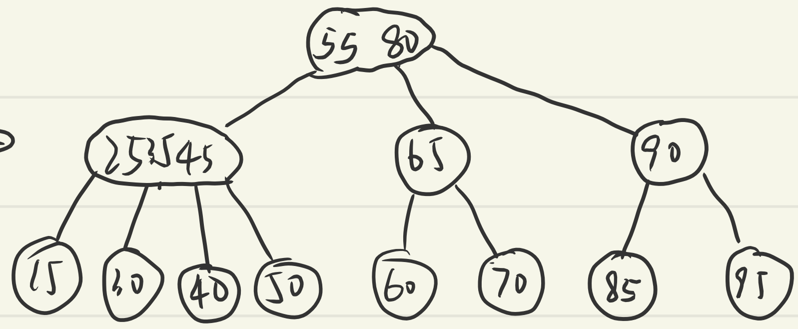


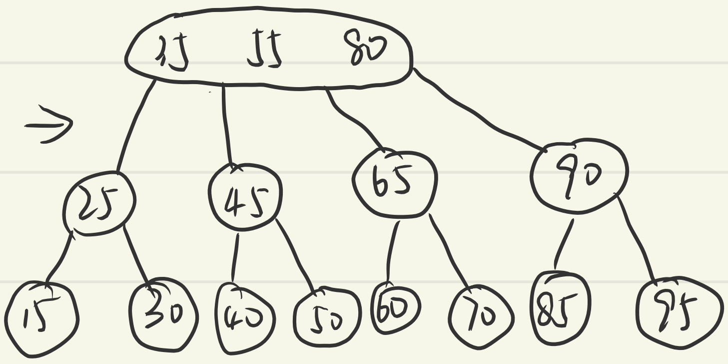
3° insert 40

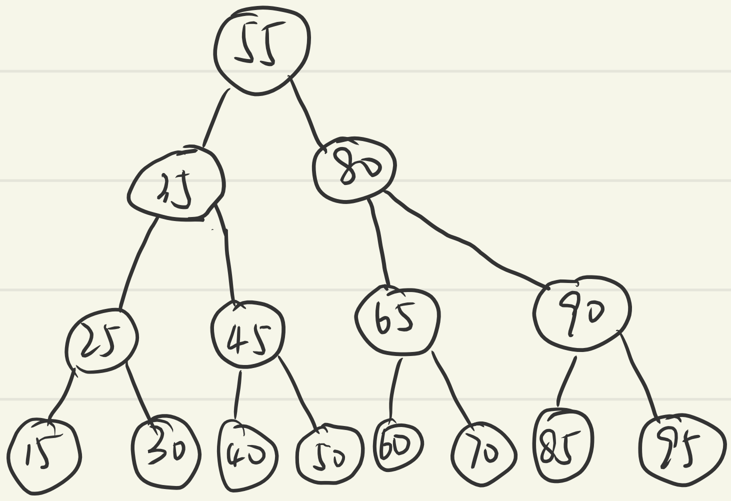


4° insert 30


55 80
25 45
65
90
15
30 35 40
50
60
70
85
95


55 80
25 35 45
65
90
15
30
40
50
60
70
85
95


35 55 80
25
45
65
90
15
30
40
50
60
70
85
95


55
35
80
25
45
65
90
15
30
40
50
60
70
85
95