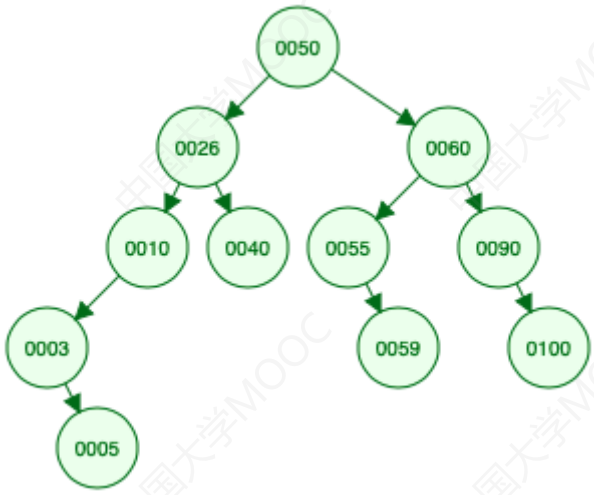
《考前补充文档》数据结构部分

应用题潜在考点：二叉排序树、平衡二叉树

1 自己设计一个例子，给出不少于 10 个关键字序列，按顺序插入一棵初始为空的二叉排序树，画出每一次插入后的样子

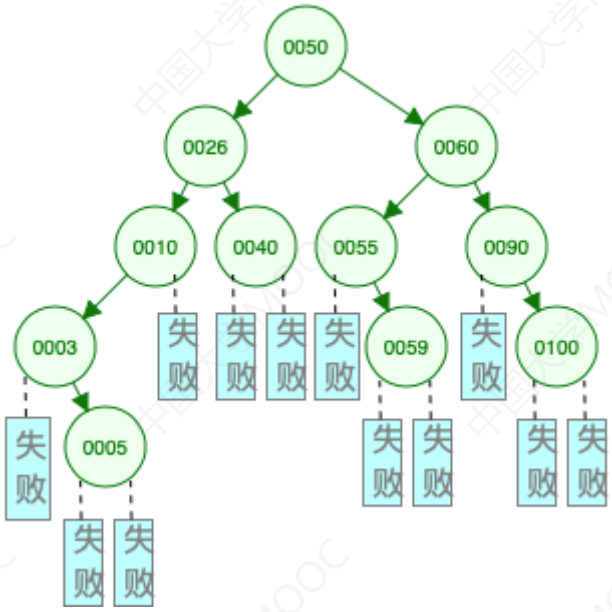
注：自己设计插入序列，并在“408 快乐小网站”模拟执行
Binary Search Trees
例：从一棵初始为空的 BST 开始，依次插入：50、26、10、3、5、60、90、40、55、100、59，最终结果如下



插入过程自己用网站模拟，可使用网站的“Pause、Step Forward”两个按钮单步执行。

2 基于你设计的例子，计算二叉排序树在查找成功和查找失败时的 ASL

计算“树形查找”的 ASL 时，需要补充“失败结点”，再进行计算

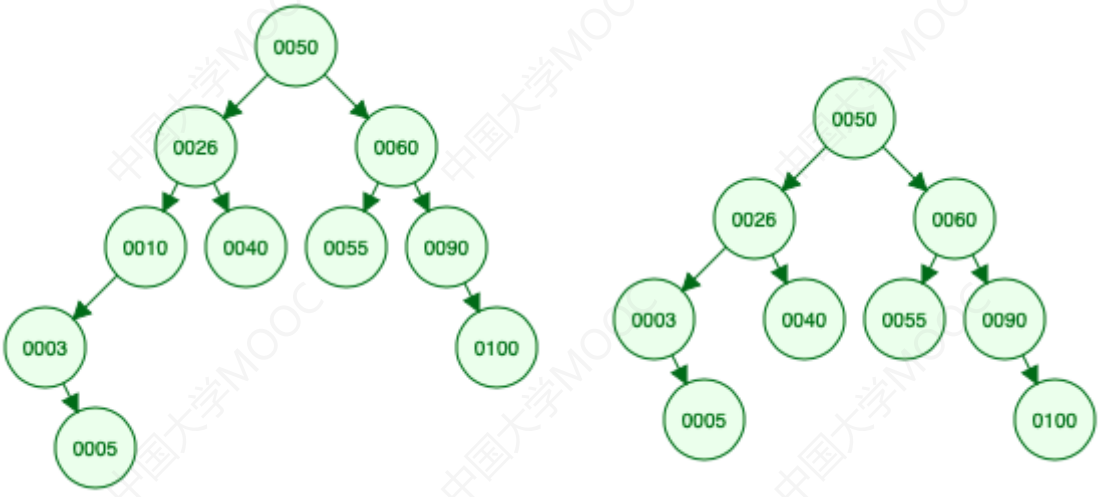


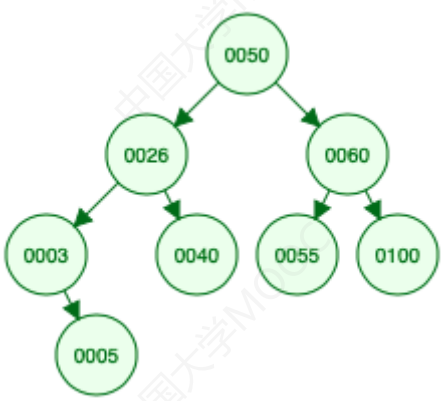
$$ASL_{成功} = \frac{1}{11}(1 \times 1 + 2 \times 2 + 3 \times 4 + 4 \times 3 + 5 \times 1) = \frac{34}{11}$$

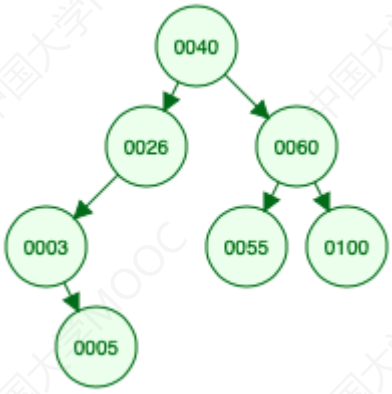
$$ASL_{失败} = \frac{1}{12}(3 \times 5 + 4 \times 5 + 5 \times 2) = \frac{45}{12} = 3.75$$

3 基于你设计的例子，依次删除不少于 4 个元素，画出每一次删除之后的样子（需要包含四种删除情况——删一个叶子结点、删一个只有左子树的结点、删一个只有右子树的结点、删一个既有左子树又有右子树的结点）

例：基于 3.6.1 的例子，依次删除：59（叶子）、10（仅有左子树）、90（仅有右子树）、50（拥有左右子树）。4 次删除后，二叉排序树的状态分别如下：

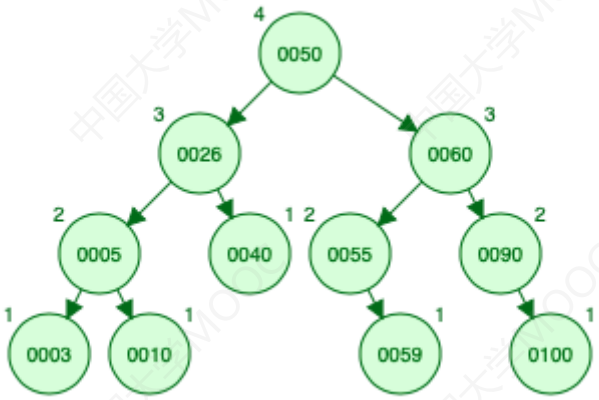






4 自己设计一个例子，给出不少于 10 个关键字序列，按顺序插入一棵初始为空的平衡二叉树，画出每一次插入后的样子（你设计的例子要涵盖 LL、RR、LR、RL 四种调整平衡的情况）

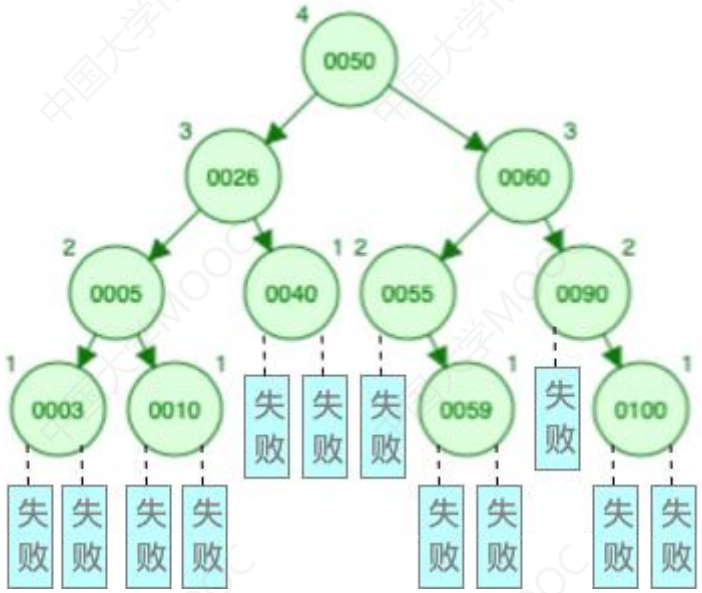
注：自己设计插入序列，并在“408 快乐小网站”模拟执行
AVL Trees (Balanced binary search trees)
例：从一棵初始为空的 AVL Trees 开始，依次插入：50、26、10（LL）、3、5（LR）、60、90（RR）、40、55、100、59（RL），最终结果如下



插入过程自己用网站模拟，可使用网站的“Pause、Step Forward”两个按钮单步执行。上述例子中，插入元素 10、5、90、59 时分别发生了 LL、LR、RR、RL 四种“调整平衡”的情况。

5 基于你设计的例子，计算平衡二叉树在查找成功和查找失败时的 ASL

计算“树形查找”的 ASL 时，需要补充“失败结点”，再进行计算



$$ASL_{成功} = \frac{1}{11}(1 \times 1 + 2 \times 2 + 3 \times 4 + 4 \times 4) = 3$$

$$ASL_{失败} = \frac{1}{12}(3 \times 4 + 4 \times 8) = \frac{44}{12}$$

应用题潜在考点：树、森林的定义、画图

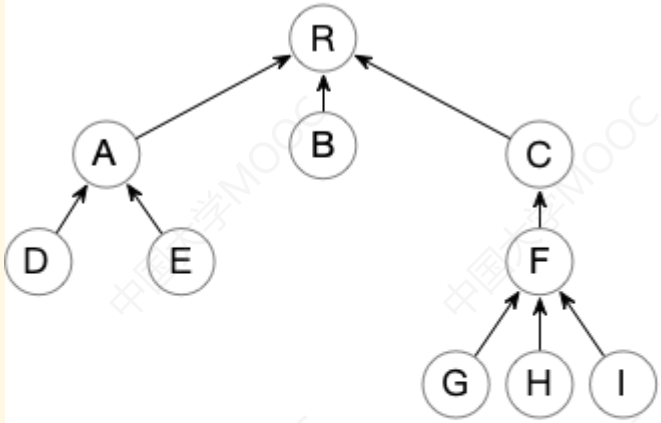
1 使用“双亲表示法”定义顺序存储的树（以及森林），并画出示意图

参考王道书 5.4.1 双亲表示法的代码描述

双亲表示法的存储结构描述如下：

```
#define MAX_TREE_SIZE 100           //树中最多结点数
typedef struct{                     //树的结点定义
    ElemType data;                  //数据元素
    int parent;                     //双亲位置域
}PTNode;
typedef struct{                     //树的类型定义
    PTNode nodes[MAX_TREE_SIZE];    //双亲表示
    int n;                          //结点数
}PTree;
```

双亲表示法可以表示“多叉树”，如下所示：



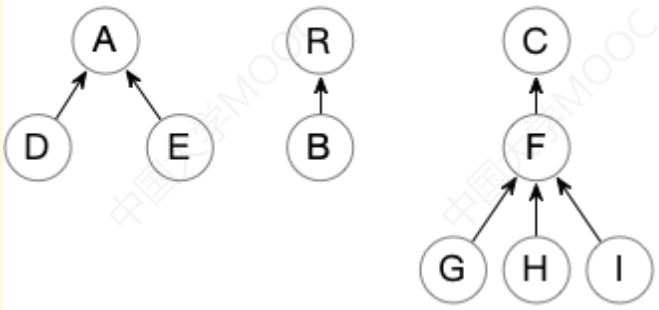
由10个结点组成的树

	data	parent
0	R	-1
1	A	0
2	B	0
3	C	0
4	D	1
5	E	1
6	F	3
7	G	6
8	H	6
9	I	6

双亲表示法表示“树”

显然，双亲表示法也可以表示 “森林”

只需令森林中的每棵树根结点的 parent 值为 -1 即可，如下所示：



由10个结点组成的森林，共计3棵树

	data	parent
0	R	-1
1	A	-1
2	B	0
3	C	-1
4	D	1
5	E	1
6	F	3
7	G	6
8	H	6
9	I	6

双亲表示法表示“森林”

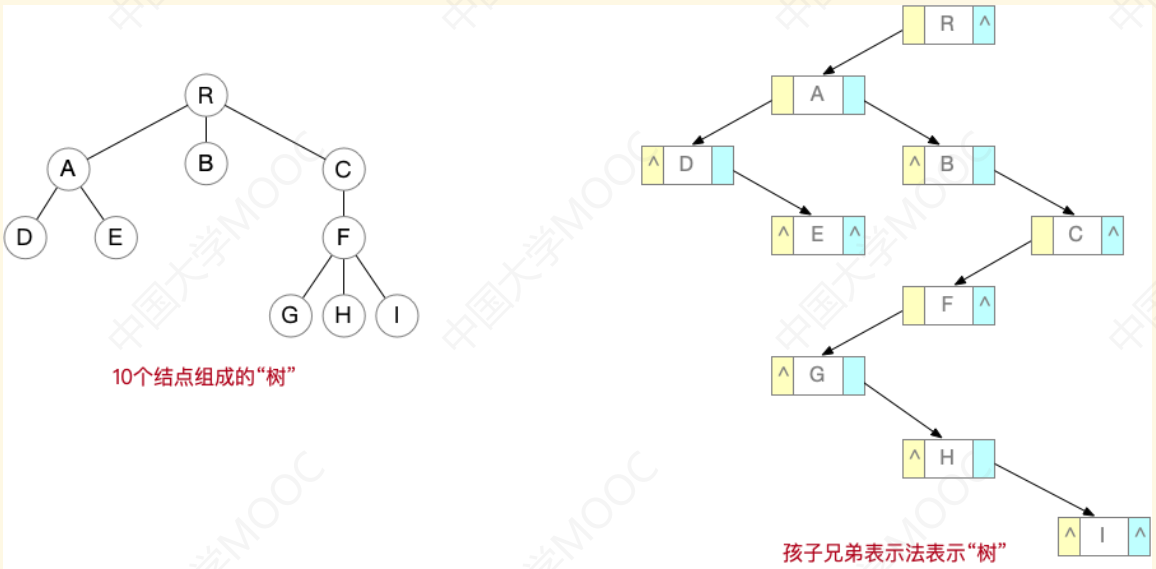
注意上面　这个例子，如果未来考 “森林” 的应用题，很可能考双亲表示法的数据结构定义+画图

2 使用“孩子表示法”，定义链式存储的树（以及森林），并画出示意图

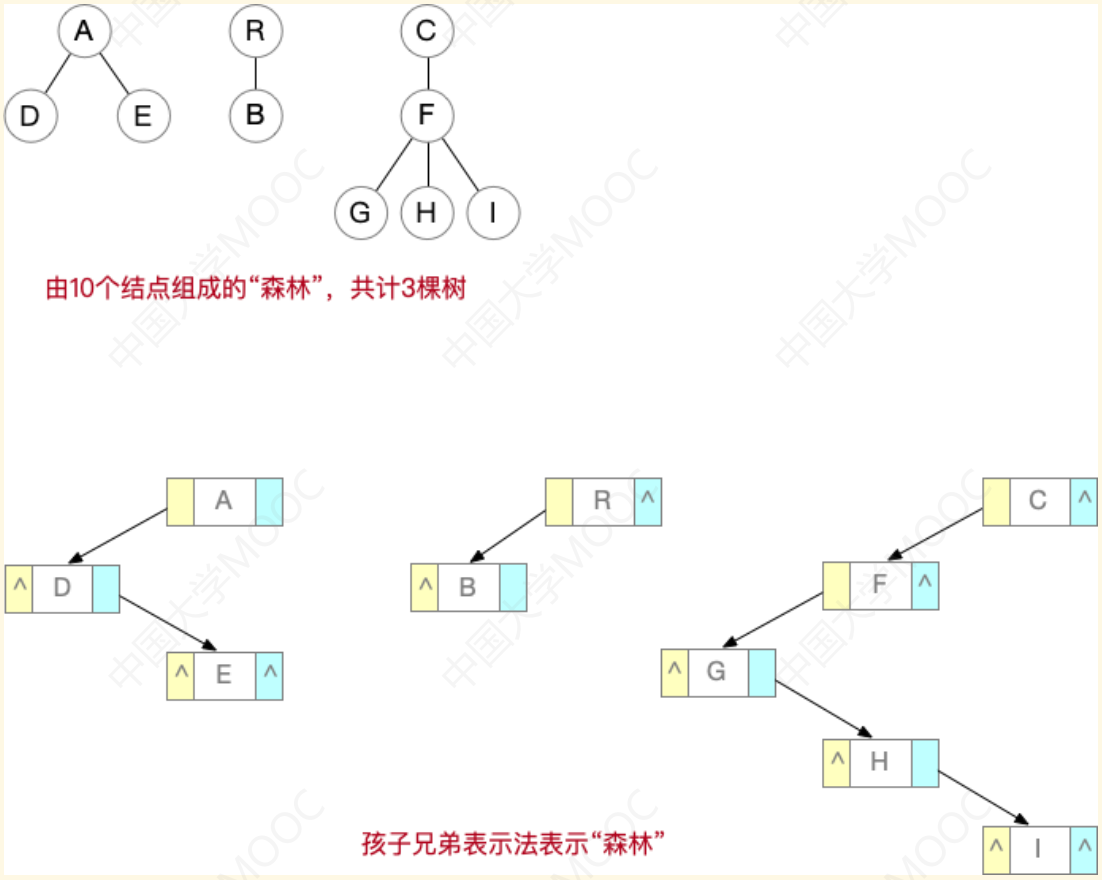
```
//Child 表示下一个孩子的信息
typedef struct Child{
    int index;              //孩子编号
    struct Child * next;    //下一个孩子
} Child;

//TreeNode 用于保存结点信息
typedef struct TreeNode {
    char data;              //结点信息
    Child * firstChild;     //指向第一个孩子
} TreeNode;

TreeNode tree[10];    //定义一棵拥有 10 个结点的树（孩子表示法）
```

```
typedef struct CSNode{
    ElemType data;                                  //数据域
    struct CSNode *firstchild,*nextsibling;     //第一个孩子和右兄弟指
针
}CSNode,*CSTree;
```

孩子表示法可表示“多叉树”，如下所示：



显然，孩子表示法可表示“森林”，如下所示：



特别注意：应用题中很可能考双亲表示法、孩子表示法、孩子兄弟表示法的数据结构定义+画图