**数据结构实验**

**实验5：树的基本实现和应用**

**一、实验目的**

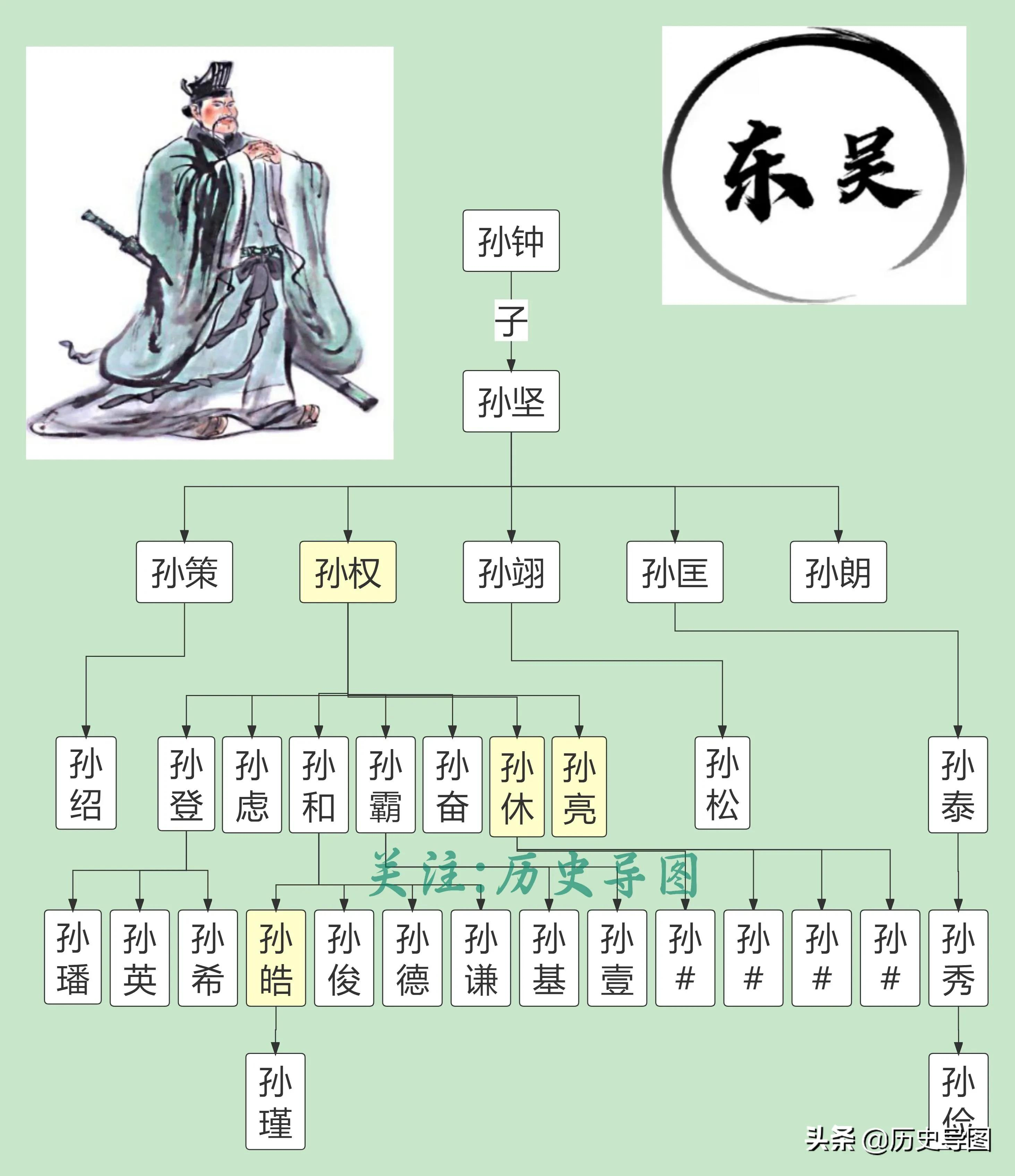
1. 能够理解树的存储结构及遍历操作，并能编码实现；
2. 能够利用树结构解决实际问题。

**二、实验内容**

1. 树的存储和遍历（必做题）
2. 编码实现树的两种存储结构：双亲表示法和孩子表示法；
3. 基于树的孩子表示法实现树的前序遍历、后序遍历和层序遍历。
4. 找亲属（必做题）

在家谱中找到某个人的所有直系亲属，如下图的家谱中给定“孙登”，找出直系祖先依次为孙权、孙坚、孙钟，直系子孙为孙璠、孙英、孙希。其中，直系亲属包括直系祖先和直系后代，直系祖先包括父亲、祖父、曾祖三代，直系子孙包括儿子以及孙子。

请选用合适的存储结构存储家谱，并设计算法实现找亲属功能。

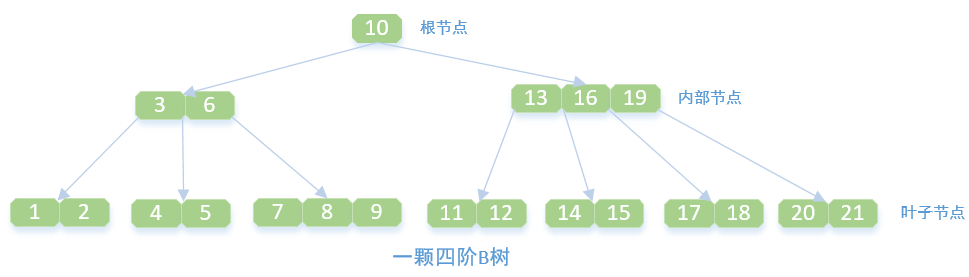


3、B树和B+树（选做题）

【参考网址：https://blog.csdn.net/weixin\_43156699/article/details/117216784】

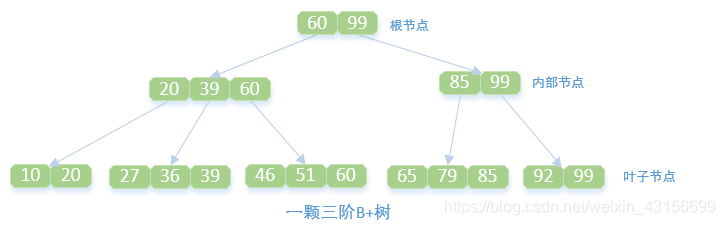
（1）一颗m阶的B树满足如下条件：

* 每个节点最多只有m个子节点。
* 除根节点外，每个非叶子节点具有至少有 m/2（向下取整）个子节点。
* 非叶子节点的根节点至少有两个子节点。
* 有k颗子树的非叶节点有k-1个键，键按照递增顺序排列。
* 叶节点都在同一层中。



（2）一颗m阶的B+树满足如下条件：

* 每个节点最多只有m个子节点。
* 除根节点外，每个非叶子节点具有至少有 m/2（向下取整）个子节点。
* 非叶子节点的根节点至少有两个子节点。
* 有k颗子树的非叶节点有k个键，键按照递增顺序排列。
* 叶节点都在同一层中。



请编程实现B树和B+树的存储即基本操作。

**三、实验要求**

1、两道必做题必须完成，学有余力的同学可以尝试完成选做题；

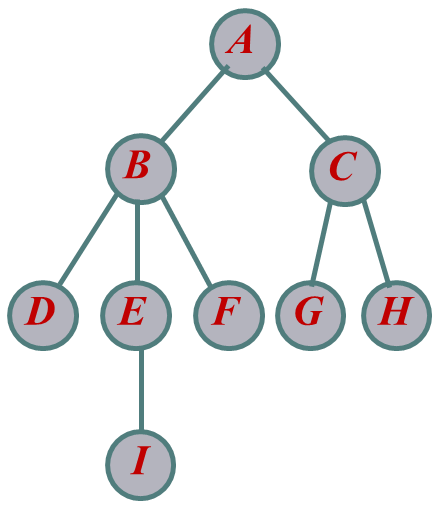
2、严格按要求、按时间提交代码和实验报告；

3、选做题先自己做，完成后再参考网上解法。

**四、实验提示**

1、树的存储

（1）输入格式：第一行输入树的所有结点，以空格分隔；下面若干行，每行以“双亲结点 孩子结点1 孩子结点2 …”格式表示某个双亲结点有哪些孩子节点，每个结点之间以空格分隔，最后一行以“-”符号结束。



如上图所示树结构输入为：

A B C D E F G H I

A B C

B D E F

C G H

E I

-

（2）双亲表示法和孩子表示法的结点定义如教材所示；

（3）基于孩子表示法的前序遍历思路是每棵子树都要先遍历根结点，然后从左至右依次遍历每一棵子树，具体如下：

PreOrderTree(int root) //递归算法，以Tree[root]为根的子树

{

If(root>=0 && root<NodeNum) //存在该结点

输出root结点

p=root结点的第一个孩子结点 //Tree[root]->firstChild

While(p!=NULL) //依次前序遍历root结点的每一棵子树

PreOrderTree(p->child)；

p=p->next

}

（4）基于孩子表示法的前序遍历思路是每棵子树都要先从左至右依次遍历每一棵子树，然后遍历根结点，具体如下：

PostOrderTree(int root) //递归算法，以Tree[root]为根的子树

{

If(root>=0 && root<NodeNum) //存在该结点

p=root结点的第一个孩子结点 //Tree[root]->firstChild

While(p!=NULL) //依次前序遍历root结点的每一棵子树

PostOrderTree(p->child)；

p=p->next

输出root结点

}

（5）基于孩子表示法的层序遍历思路是从树根开始，逐层向下，每层从左到右依次遍历每个结点，具体如下：

LevelOrderTree(int root)

{

if(root>=0 && root<NodeNum)

Q.Equeue(root);

While(!Q.Empty())

p=Q.Dequeue();

输出结点p；

将p结点的每个孩子结点依次入队列；

}

2、找亲属

（1）分析：操作主要是查找某个结点的双亲结点（向上找祖先）、查找某个结点的孩子结点（向下找孩子）；

（2）存储结构：要能够支持快速找到双亲和孩子；

（3）注意：查找双亲操作最多重复执行三次（父亲、祖父、曾祖三代），查找孩子操作最多执行两次（儿子、孙子两代）但需要找出所有孩子结点。

（4）族谱可以写入文件中，格式参考第一题

3、从屏幕读入字符

cin读入一个字符char或者字符串string；

getchar()每次读入单个字符【头文件为stdio.h】;

getline()每次读入一行【头文件为string.h】