# 实验三：二叉树的物理实现

计科1903陈旭 201914020128

**日期**：**2020.11.4**

**当天任务**：**实现树的结点。**

**产生问题**：在结点的实现类里面，是否应当在调用结点ADT中变量的时候引入作用域限定符。

**问题结果**：按照编译结果是需要的，virtual关键字的影响力只是在“运行时”，即在运行时根据引用或者指针的动态类型来绑定相应的函数调用。而在编译时，其与普通函数一样，要接受子类和父类作用域的限制

**日期：2020.11.5**

**当天任务**：**二叉树的ADT和它的实现。**

**产生问题**：



该问题并非在当天产生，而是在完全写好测试文件和实现类之后发现的。该处在调用node<E>

类型的指针it的element成员的时候出现错误，多加了一个括号。最终经调试、定位后发现

错误将其删除

**具体代码**：

                cout << it->element << ' '; //改正后

**日期：2020.11.6**

**当天任务**：**构建二叉树，将实验完成**

**产生问题**：建树函数理解不够深入。为贴合题目要求采用根据后序遍历次序和中序遍历次序

直接建树，但是出现很多逻辑上和语法上的问题。

**初始代码：**

void getpreorder(string s1, string s2)

{

    if (n==0)

        return;

    else if (n==1)

    {

        str=s2[0];

        return ;

    }

    int position=s2.find(s2[n-1]);

    str+=s1[n-1];

    getpreorder(s2.substr(0, position + 1),  s1.substr(0, position));

    getpreorder(s2.substr(position+1, n-position-

1), s1.substr(position, n-position-1));

}

其中，n在递归过程中始终未发生变化，导致整个递归过程陷入死循环。并且，字符串长度每次缩减一半，n保持不变，最终造成数组越界。

**第一次修改后的代码**：

node<char>\* build(string midorder, string postorder)

{

    node<char>\* it;

    char \_root = postorder.at(postorder.length() - 1);

    int position = midorder.find(\_root);

    int leftsize = position;

    int rightsize = midorder.length() - position - 1;

    it = new realisemybitreenode<char>;

    it->setelement(\_root);

    it->setleft(leftsize==0 ? build(midorder.substr(0, leftsize), postorder.substr(0, leftsize)) : NULL);

    it->setright(rightsize==0 ? build(midorder.substr(position + 1, rightsize), postorder.substr(leftsize, rightsize)) : NULL);

    return it;

}

最终发现树没有真正建立起来，建立到第一个元素之后函数直接返回了。

**产生原因**：

it->setleft(leftsize==0 ? build(midorder.substr(0, leftsize), postorder.substr(0, leftsize)) : NULL);

    it->setright(rightsize==0 ? build(midorder.substr(position + 1, rightsize), postorder.substr(leftsize, rightsize)) : NULL);

这两句，由于对基础语法掌握不熟练，判断条件rightsize==0取了反。?号后面跟着的是判断条件成立的情况下执行的语句，：后面是不成立的情况下执行的语句。

**改正方法**：

将rightsize==0和leftsize==0改为!=0。或是直接改成rightsize? …… 和leftsize?……，将其本身作为bool型变量。

**最终改正代码：**

node<char>\* build(string midorder, string postorder)

{

    node<char>\* it;

    char \_root = postorder.at(postorder.length() - 1);

    int position = midorder.find(\_root);

    int leftsize = position;

    int rightsize = midorder.length() - position - 1;

    it = new realisemybitreenode<char>;

    it->setelement(\_root);

    it->setleft(leftsize ? build(midorder.substr(0, leftsize), postorder.substr(0, leftsize)) : NULL);

    it->setright(rightsize ? build(midorder.substr(position + 1, rightsize), postorder.substr(leftsize, rightsize)) : NULL);

    return it;

}