《数据结构与算法》实验报告

年级、专业、班级	2021 级计卓 2 班	姓名	文红兵
实验题目	二叉树与二叉查找树实践		
实验时间	200/11/3	实验地点	线上
实验成绩		实验性质	☐验证性 √设计性 ☐综合性

教师评价：

☐算法/实验过程正确； ☐源程序/实验内容提交 ☐程序结构/实验步骤合理；

☐实验结果正确； ☐语法、语义正确； ☐报告规范；

其他：

评价教师签名：

实验目的

1. 掌握二叉树与二叉查找树的基本原理
2. 训练使用二叉树基本操作，通过编程解决不同难度问题的实践能力

二、实验项目内容

实验课题 1：

题目内容：

6-1 The Kth Largest in BST

解题思路：

降序遍历二叉搜索树，当遍历到第 K 个元素的时候返回

源代码：

报告创建时间：

```c
BinTree helper ( BinTree T, int *K_p )
{
    if(T==NULL) return NULL;                // 如果为空，直接返回
    BinTree Right = helper(T->Right,K_p);   // 遍历顺序 右->中->左，遍历结果为降序
    if(Right) return Right;                 // 若Kth在右子树，直接返回，否则，在左子树中寻找
    (*(K_p))--;                               // 没有找到的话 K 减一，直到找到
    if(*K_p == 0) return T;                 // 找到了就返回
    BinTree Left = helper(T->Left,K_p);
    if(Left) return Left;                   // 若Kth在左子树，直接返回
    return NULL;                            // Kth不符合要求，返回NULL
}
BinTree KthLargest ( BinTree T, int K )
{
    return helper(T,&K);  // 调用helper
}
```

时间与空间复杂度分析:

时间复杂度：O(n)

空间复杂度：O(n)

实验课题 2：

题目内容：

6-2　从下往上打印指定元素的所有祖先

解题思路：

递归遍历这棵树，遇到指定元素或者他的祖先就返回 tag = 1，并且打印该节点

源代码：

```c
int PrintAncestors(BiTree T,char ch)
{
    int tag1,tag2;
    if(T==NULL) return 0;
    else if(T->data==ch) return 1;      // 找到了这个节点，就返回1
    tag1=PrintAncestors(T->lchild,ch);  // 若节点在左子树，tag1 = 1
    tag2=PrintAncestors(T->rchild,ch);  // 若节点在右子树，tag2 = 1
    if(tag1 || tag2) {                  // 若左子树或者右子树有这个节点，那么当前节点为祖先节点
        printf("%c ",T->data);
        return 1;
    }
    return 0;
}
```

时间与空间复杂度分析：

时间复杂度：O(n)

空间复杂度：O(n)

实验课题 3：

题目内容：

7-1　构造二叉检索树

解题思路：

1、构造一颗二叉搜索树，并且添加先序遍历（preTravel）和插入操作（insert）。

2、再依次循环输入并且插入元素，输入完毕后先序遍历输出。

源代码：

```
#include<iostream>
using namespace std;

class TreeNode{
public:
    int val;
    TreeNode* left;
    TreeNode* right;
public:
    TreeNode(int _val):val(_val),left(NULL),right(NULL){}
    TreeNode* insert(TreeNode*root,int _val);     // 二叉搜索树的插入操作，返回根节点
    void preTravel(TreeNode* root);               // 先序遍历二叉搜索树并打印节点
};

TreeNode* insert(TreeNode*root ,int _val){
    if(root==NULL) return new TreeNode(_val);
    if(_val > root->val)
        root->right=insert(root->right,_val);
    else
        root->left=insert(root->left,_val);
    return root;
}

void preTravel(TreeNode* root){
    if(root==NULL)return;
    cout<<root->val<<" ";
    preTravel(root->left);
    preTravel(root->right);
}

int main(){
    TreeNode* root = NULL;
    int temp;
    while(cin>>temp){  // 输入二叉树的节点，输入0结束
        if(temp!=0) root = insert(root,temp);
        else break;
    }
    preTravel(root);  // 先序遍历打印二叉树
    system("pause");
    return 0;
}
```

时间与空间复杂度分析：

时间复杂度：O(nlogn)

空间复杂度：O(n)

三、思考题

遍历二叉树是按一定的顺序依次访问树中各结点并输出结点存放的数据。二叉树有四种遍历方式：前序、中序、后序遍历以及层序遍历。假设二叉树各结点的数据互异。如果出现如下的情况，即对二叉树用两种不同的方式遍历，生成的结点序列却相同，则该二叉树具有什么特征？

（1）前序遍历与中序遍历相同

答：空树或二叉树只有根节点或二叉树的非叶子节点只有右子树

（2）后序遍历与中序遍历相同

答：空树或二叉树只有根节点或二叉树的非叶子节点只有左子树

（3）前序遍历与后序遍历相同

答：空树或二叉树只有根节点

（4）前序遍历与层序遍历相同

答：

1、空树

2、二叉树只有根节点

3、二叉树的非叶子节点只有左子树

4、二叉树的每个节点的左子树节点个数<=1