桂林电子科技大学

**数据结构与算法**  实验报告

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验名称 | **实验五 二叉排序树的插入和检索** | | | | | | | |  | 辅导员意见：  成绩 辅导员  签 名 |
| 院 系 | 计算机与信息安全学院 | | | 专业 | | 计算机大类 | | |
| 学 号 | 1800300636 | | | 姓名 | | 奚锐 | | |
| 实验日期 | 2019 | 年 | 11 | | 月 | | 7 | 日 |
|  |  | | | | | | | |

**一、实验目的**

掌握二叉排序树的基本操作实现；

**二、实验具体内容**

构造一棵二叉排序树，并进行遍历，检索数据

要求：

1、输入数据构造二叉排序树

2、对构造好的二叉排序树进行中根序遍历

3、对构造好的二叉排序树进行数据插入和数据检索

4、实现二叉排序树的删除算法（选作）

上述要求中

第1、2、3点为基本要求，第4点为提高要求。

提示：

对二叉排序树进行中根序遍历的时候，获得的是排序序列

**（2）分析**（必须手写！！！）

**（3）实验代码**（源代码可以打印，算法描述必须手写！！！）

#include<bits/stdc++.h>

using namespace std;

int pos;

int mycount;

int charcount;

int numcount;

struct treeNode{

//这是结点的值

char data;

//这记录了该节点的层级

int layer;

//这是key

int key;

//这是左节点

treeNode\* left;

//这是右节点

treeNode \*right;

};

void middleacess(treeNode \*x){//递归中序遍历

if(x!=NULL){

middleacess(x->left);//中序访问左子树

cout<<x->key;//访问根节点

middleacess(x->right);//中序访问右子树

}

return;

}

void InsertBST(treeNode \*x,char t,int key){//插入数据

treeNode \*p=x;

treeNode \*q=x;

if(x==NULL){//根节点不存在，直接新建

cout<<"x is NULL"<<endl;

x=(treeNode\*)malloc(sizeof(treeNode));

x->data=t;

x->key=key;

x->left=NULL;

x->right=NULL;

}

else

{

while(p!=NULL){

q=p;

if(p->key<key){

p=p->left;

}

else if(p->key>key){

p=p->right;

}

else{

return;//一样的数字，没必要插入，直接返回

}

}

p=(treeNode\*)malloc(sizeof(treeNode));

p->data=t;

p->key=key;

p->left=NULL;

p->right=NULL;

if(p->key<q->key){

q->left=p;

}

else{

q->right=p;

}

}

return;

}

bool searchBST(treeNode \*p,int key){

treeNode \*x=p;

while(x!=NULL){

if(x->key==key){

return true;

}

else if(x->key>key)

{

x=x->right;

}

else if(x->key<key)

{

x=x->left;

}

}

return false;

}

void deleteBST(treeNode \*p,int key){

treeNode \*x=p;

treeNode\* t=x;

while(x!=NULL){

if(x->key==key){

if(x->right!=NULL&&x->left!=NULL){

treeNode \*maxpl=x->left;

treeNode \*xr=x->right;

while(maxpl->right!=NULL){

maxpl=maxpl->right;

}

maxpl->right=xr;

if(t->key>key){

t->left=x->left;

}

else{

t->right=x->right;

}

free(x);

}

return ;

}

else if(x->key>key)

{

t=x;

x=x->right;

}

else if(x->key<key)

{

t=x;

x=x->left;

}

}

return ;

}

int main(){

numcount=0;

charcount=0;

int a=0;

int num=0;

pos=0;

mycount=0;

cin>>a;

//cout<<a;

treeNode \*x=tree\_init("a##");

cin>>x->key;

for(int i=1;i<a;i++){

cin>>num;

//cout<<num;

InsertBST(x,'a',num);

}

middleacess(x);

return 0;

}

**三、实验小结**（必须手写！！！）

实验小结部分必须手写，主要是总结编程中出现的问题，解决的方法，有关数据结构的思考。