

## 实现代码

/\*\*

\* DeleteBST.C 二叉排序树的删除

\* @auther Darbuly 970073804@qq.com

\* @copyright 2018-2019

\*/

#include <stdio.h>

#include <stdlib.h>

typedef struct BiNode

{

int data;

struct BiNode \*lchild,\*rchild;

}BiNode,\*BiTree;

int str[] = {70,105,115,104,67,46,99,111,109,65535};

int top=0;

void insertChild(BiTree \*T,int num);

void CreateBT(BiTree \*T)

{

\*T = (BiNode\*)malloc(sizeof(BiNode));

(\*T)->lchild = NULL;

(\*T)->rchild = NULL;

int num;

//scanf("%d",&num);

num = str[top++];

if((\*T)!=NULL && 65535 !=num)

(\*T)->data = num;

while(65535!=num)

{

//scanf("%d",&num);

num = str[top++];

if((\*T) !=NULL && 65535 !=num)

insertChild(T,num);

}

}

void insertChild(BiTree \*T,int num)

{

if(num<(\*T)->data)

{

if((\*T)->lchild==NULL)

{

(\*T)->lchild = (BiNode\*)malloc(sizeof(BiNode));

(\*T)->lchild->lchild=NULL;

(\*T)->lchild->rchild=NULL;

(\*T)->lchild->data = num;

}

else

{

insertChild(&((\*T)->lchild),num);

}

}

else

{

if((\*T)->rchild==NULL)

{

(\*T)->rchild = (BiNode\*)malloc(sizeof(BiNode));

(\*T)->rchild->rchild=NULL;

(\*T)->rchild->lchild=NULL;

(\*T)->rchild->data = num;

}

else

{

insertChild(&((\*T)->rchild),num);

}

}

}

void visit(int c,int level)

{

printf(" %d located in %d layer\n",c,level);

}

void MidOrderTraverse(BiTree T,int level)

{

if(T)

{

MidOrderTraverse(T->lchild,level+1);

visit(T->data,level+1);

MidOrderTraverse(T->rchild,level+1);

}

}

int DeleteBST(BiTree \*T,int key)

{

if(!(\*T))

{

printf("\nThe BiTree is nofind or empty!\n");

return 0;

}

else

{

if(key == (\*T)->data)

{

return Delete(T);

}

else if(key<(\*T)->data)

{

return DeleteBST(&(\*T)->lchild,key);

}

else

{

return DeleteBST(&(\*T)->rchild,key);

}

}

}

int Delete(BiTree \*p)

{

BiTree q,s;

if((\*p)->rchild==NULL)

{

q = \*p;

\*p = (\*p)->lchild;

free(q);

}

else if((\*p)->lchild==NULL)

{

q=\*p;

\*p = (\*p)->rchild;

free(q);

}

else

{

q = \*p;

s = (\*p)->lchild;

while(s->rchild)

{

q = s;

s = s->rchild;

}

(\*p)->data = s->data;

if(q!=\*p)

{

q->rchild = s->lchild;

}

else

{

q->lchild = s->lchild;

}

free(s);

}

return 1;

}

int main()

{

printf("Search.h Test\n");

BiTree T,p;

int level=0;

int key;

CreateBT(&T);

MidOrderTraverse(T,level);

printf("Please enter what you want delete:\n");

scanf("%d",&key);

if(DeleteBST(&T,key))

{

printf("Delete successfully !\n");

MidOrderTraverse(T,level);

}

else

{

printf("Delete error!\n");

}

return 0;

}

