**Bài 7: Cây nhị phân tìm kiếm**

#include"stdio.h"

#include"conio.h"

#include"string.h"

typedef struct sinhvien

{

char hoten[40];

int tuoi;

}sv;

typedef struct node

{

sv data;

node \*left;

node \*right;

}node;

typedef struct node \*TREE;

TREE root;

void khoitao(TREE &root)

{

root=NULL;

}

void chennode(TREE &root, sv x)

{

if(root!=NULL)

{

if(strcmp(root->data.hoten,x.hoten)==0) return ;

if(root->data.tuoi>x.tuoi)

chennode(root->left,x);

else

chennode(root->right,x);

}

else

{

root=new(node);

root->data=x;

root->left=root->right=NULL;

}

}

void taocay(TREE &root)

{

int n;

sv x;

printf("\n nhap so sv:");

scanf("%d",&n);

printf("\n nhap thong tin cho sv:");

for(int i=1;i<=n;i++)

{

printf("\n nhap ho ten:");

fflush(stdin);

gets(x.hoten);

printf("\n nhap tuoi:");

scanf("%d",&x.tuoi);

chennode(root,x);

}

}

void NLR(TREE root)

{

if(root!=NULL)

{

printf("\n ho ten:%s tuoi:%d",root->data.hoten,root->data.tuoi);

NLR(root->left);

NLR(root->right);

}

}

void LRN(TREE root)

{

if(root!=NULL)

{

LRN(root->left);

LRN(root->right);

printf("\n ho ten:%s tuoi:%d",root->data.hoten,root->data.tuoi);

}

}

void LNR(TREE root)

{

if(root!=NULL)

{

LNR(root->left);

printf("\n ho ten:%s tuoi:%d",root->data.hoten,root->data.tuoi);

LNR(root->right);

}

}

int max(int a,int b)

{

if(a>b)

return a;

else

return b;

}

int chieucao(TREE root)

{

if(root==NULL)

return 0;

else

return max(chieucao(root->left),chieucao(root->right))+1;

}

node \*tim(TREE root,sv x)

{

if(root!=NULL)

{

if(root->data.tuoi==x.tuoi)

return root;

if(root->data.tuoi>x.tuoi)

return tim(root->left,x);

return tim(root->right,x);

}

return NULL;

}

void timthaythe(TREE &p,TREE &q)

{

if(q->right!=NULL)

timthaythe(p,q->right);

else

{

p->data=q->data;

p=q;

if(q->left!=NULL)

q=q->left;

else

q=q->right;

}

}

void huy(TREE &root,sv x)

{

if(root==NULL)

return;

if(root->data.tuoi>x.tuoi)

huy(root->left,x);

if(root->data.tuoi<x.tuoi)

huy(root->right,x);

if(root->data.tuoi==x.tuoi)

{

node \*p=root;

if(root->left==NULL)

root=root->right;

else

if(root->right==NULL)

root=root->left;

else

timthaythe(p,root->left);

delete(p);

}

}

void xoacay(TREE &root)

{

if(root!=NULL)

{

xoacay(root->left);

xoacay(root->right);

huy(root,root->data);

}

}

int main()

{

TREE t;

sv x,y,z;

khoitao(t);

taocay(t);

/\*printf("\n duyet cay theo thu tu truoc:");

NLR(t);

printf("\n nhap sinh vien moi:");

printf("\n nhap ho ten:");

fflush(stdin);

gets(x.hoten);

printf("\n nhap tuoi:");

scanf("%d",&x.tuoi);

chennode(t,x);\*/

printf("\n duyet cay theo thu tu truoc:");

NLR(t);

printf("\n duyet cay theo thu tu sau:");

LRN(t);

printf("\n duyet cay theo thu tu giua:");

LNR(t);

printf("\n chieu cao cay la:%d",chieucao(t));

/\* printf("\n sv can tim co tuoi la:");

scanf("%d",&y.tuoi);

printf("\n sv can tim la:");

node \*p=tim(t,y);

printf("\n ho ten:%s tuoi:%d",p->data.hoten,p->data.tuoi);\*/

printf("\n sv can xoa co tuoi la: ");

scanf("%d",&z.tuoi);

huy(t,z);

/\*printf("\n cay sau khi xoa la:");

NLR(t);\*/

printf("\n cay sau khi xoa het la:");

// xoacay(t);

NLR(t);

getch();

}