**CÂY TÌM KIẾM NHỊ PHÂN**

#include <stdio.h>

#include <stdlib.h>

//Dinh nghia cay

typedef struct Node \* NodeType;

struct Node{

int Key;

NodeType Left, Right;

};

typedef NodeType Tree;

//khoi tao cay rong

void MakenullTree(Tree \*T){

(\*T)=NULL;

}

//kiem tra cay rong

int ktRong (Tree T){

if (T==NULL) return 1;

return 0;

}

//Them khoa x vao Tree

void InsertNode(int x, Tree \*T){

if ((\*T)==NULL)

{

(\*T)=(struct Node\*)malloc(sizeof(struct Node));

(\*T)->Key=x;

(\*T)->Left=NULL;

(\*T)->Right=NULL;

}

else if (x>(\*T)->Key)

{

InsertNode(x,&(\*T)->Right);

}

else if (x<(\*T)->Key)

{

InsertNode(x,&(\*T)->Left);

}

}

//Tim kiem khoa x

Tree SearchNode(int x, Tree T){

if (T==NULL) return NULL;

if (T->Key==x) return T;

if (x>T->Key)

{

return SearchNode(x,T->Right);

}

if (x<T->Key)

{

return SearchNode(x, T->Left);

}

}

//Xoa va tra ve khoa nho nhat cua cay

int DeleteNodeMin (Tree \*T){

int k;

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

{

k=(\*T)->Key;

(\*T)=(\*T)->Right;

return k;

}

else return DeleteNodeMin(&(\*T)->Left);

}

//Xoa va tra ve khoa lon nhat cua cay

int DeleteNodeMax(Tree \*T){

int k;

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

{

k= (\*T)->Key;

(\*T)=(\*T)->Left;

}

else DeleteNodeMax(&(\*T)->Right);

}

//Xoa khoa x trong cay

void DeleteNode (int x, Tree \*T){

if((\*T)!=NULL)

{

if( (\*T)-> Key == x){

if ((\*T)->Left==NULL ) (\*T)=(\*T)->Right;

else if ((\*T)->Right==NULL) (\*T)=(\*T)->Left;

else (\*T)->Key=DeleteNodeMin(&(\*T)-> Right);

}

else if(x<(\*T)->Key) DeleteNode(x,&(\*T)->Left);

else if(x>(\*T)->Key) DeleteNode(x,&(\*T)->Right);

}

}

//Tra ve con phai

Tree RightChild(Tree T){

if (T->Right!=NULL)

{

return T->Right;

}

return NULL;

}

//Tra ve con trai

Tree LeftChild(Tree T){

if (T->Left!=NULL)

{

return T->Left;

}

return NULL;

}

//Tra ve con trai nhat

Tree LeftMaxChild(Tree T){

if (T->Left!=NULL)

{

Tree P=T->Left;

while (P!=NULL)

{

P=P->Left;

}

return P;

}

return NULL;

}

//Tra ve con phai nhat

Tree RightMaxChilde(Tree T){

if (T->Right!=NULL)

{

Tree P=T->Right;

while (P!=NULL)

{

P=P->Right;

}

return P;

}

return NULL;

}

//Kiem tra xem nut do co phai la nut la khong

int IsLeaf(Tree T){

if (T->Left==NULL && T->Right==NULL) return 1;

return 0;

}

//Duyet tien to

void PreOrder (Tree T){

if (T!=NULL)

{

printf ("%d ",T->Key);

PreOrder(LeftChild(T));

PreOrder(RightChild(T));

}

}

//Duyet trung to

void InOrder (Tree T){

if(T!=NULL)

{

InOrder(LeftChild(T));

printf("%d ",T->Key);

InOrder(RightChild(T));

}

}

//Duyet hau to

void PostOrder(Tree T){

if (T!=NULL)

{

PostOrder(LeftChild(T));

PostOrder(RightChild(T));

printf("%d ",T->Key);

}

}

//Tong so nut tren cay

int AllNodes(Tree T){

if (T==NULL) return 0;

else return 1+AllNodes(LeftChild(T))+AllNodes(RightChild(T));

}

//Tong so nut la cua cay

int AllLeafs(Tree T){

if (T!=NULL)

{

if (IsLeaf(T)) return 1;

else return AllLeafs(LeftChild(T))+AllLeafs(RightChild(T));

}

return 0;

}

//Tinh chieu cao cay

//Tim Max

int Max(int a, int b){

return a>b?a:b;

}

int hTree(Tree T){

if (T==NULL) return -1;

else if (IsLeaf(T)==1) return 0;

else return 1+Max(hTree(LeftChild(T)),hTree(RightChild(T)));

}

//Tim chieu cao nut

int hNode(int x,Tree T){

Tree P;

P=SearchNode(x,T);

return hTree(P);

}

//Tim do sau cua nut

int DepthNode(int x, Tree T){

Tree P;

P=SearchNode(x,T);

return hTree(T)-hTree(P);

}

//Diem so cay con cua cay

int CountTree(Tree T){

if (T->Left==NULL && T->Right==NULL) return 0;

return AllLeafs(T);

}

//Diem so cay con cua 1 nut

int CountNode (int x,Tree T)

{

Tree P;

P=SearchNode(x,T);

return CountTree(P);

}

int main (){

Tree T;

MakenullTree(&T);

int x,n;

printf ("1.NHAP CAC GIA TRI CUA CAY\n");

printf ("- Nhap so phan tu toi da trong cay:");

scanf ("%d",&n);

for (int i=0;i<n;i++)

{

printf("-Nhap vao phan tu thu %d:",i+1);

scanf("%d",&x);

InsertNode(x,&T);

}

printf("2.HIEN THI CAY:\n");

printf("- Gia tri duyet tien tu:\n");

PreOrder(T);

printf ("\n");

printf("- Gia tri duyet trung tu:\n");

InOrder(T);

printf ("\n");

printf("- Gia tri hau tien tu:\n");

PostOrder(T);

printf ("\n");

printf ("3. CAC HAM CO BAN:\n");

int k,t;

printf ("- Nhap khoa nut k:");

scanf ("%d",&k);

Tree K=T;

if (!SearchNode(k,K))

{

printf("\t+ Do cao cua nut %d trong cay la: %d(Not Exist)\n",k,hNode(k,K));

printf("\t+ Do sau cua nut %d trong cau la: %d(Not Exist)\n",k,DepthNode(k,K));

printf("\t+ Tong so nut la cua cay la: %d\n",AllLeafs(K));

printf("\t+ Chieu cao cua cay la: %d\n", hTree(K));

printf("\t+ Tong so nut cua cay: %d\n", AllNodes(K));

printf("\t+ So cay con cua cua cay la: %d\n",CountTree(K));

printf("--> Khoa %d khong tim thay trong cay\n",k);

InsertNode(k,&K);

printf("- Cay sau khi them khoa %d:\n",k);

printf("\t+ Do cao cua nut %d trong cay la: %d\n",k,hNode(k,K));

printf("\t+ Do sau cua nut %d trong cau la: %d\n",k,DepthNode(k,K));

printf("\t+ Tong so nut la cua cay la: %d\n",AllLeafs(K));

printf("\t+ Chieu cao cua cay la: %d\n", hTree(K));

printf("\t+ Tong so nut cua cay: %d\n", AllNodes(K));

printf("\t+ So cay con cua nut %d la: %d\n",k,CountNode(x,K));

printf("\t+ So cay con cua cua cay la: %d\n",CountTree(K));

printf("\t+ Duyet tien tu: ");

PreOrder(K);

printf("\n");

printf("\t+ Duyet trung tu: ");

InOrder(K);

printf("\n");

printf("\t+ Duyet hau tu: ");

PostOrder(K);

printf("\n");

}

else

{

printf("\t+ Do cao cua nut %d trong cay la: %d\n",k,hNode(k,K));

printf("\t+ Do sau cua nut %d trong cau la: %d\n",k,DepthNode(k,K));

printf("\t+ Tong so nut la cua cay la: %d\n",AllLeafs(K));

printf("\t+ Chieu cao cua cay la: %d\n", hTree(K));

printf("\t+ Tong so nut cua cay: %d\n", AllNodes(K));

printf("\t+ So cay con cua nut %d la: %d",k,CountNode(x,K));

printf ("-->Khoa %d co trong cay\n",k);

DeleteNode(k,&K);

printf("- Cay sau khi xoa khoa %d:\n",k);

printf("\t+ Do cao cua nut %d trong cay la: %d(Not Exist)\n",k,hNode(k,K));

printf("\t+ Do sau cua nut %d trong cau la: %d(Not Exist)\n",k,DepthNode(k,K));

printf("\t+ Tong so nut la cua cay la: %d\n",AllLeafs(K));

printf("\t+ Chieu cao cua cay la: %d\n", hTree(K));

printf("\t+ Tong so nut cua cay: %d\n", AllNodes(K));

printf("\t+ So cay con cua cay la: %d\n",CountTree(K));

printf("\t+ Duyet tien tu: ");

PreOrder(K);

printf("\n");

printf("\t+ Duyet trung tu: ");

InOrder(K);

printf("\n");

printf("\t+ Duyet hau tu: ");

PostOrder(K);

printf("\n");

}

}