**Thực hành Cây nhị phân tìm kiếm**

Sinh viên biết cách cài đặt một node trong cây: 1 node trong cây gồm 3 biến

struct node

{ int data; // biến để chứa giá trị của node

node\* left; // biến con trỏ để trỏ tới (chứa địa chỉ) của nút con trái

node\* right; // biến con trỏ để trỏ tới (chứa địa chỉ) của nút con phải

};

***Yêu cầu:***

* ***Sinh viên tham khảo code ở bên dưới và gọi được các hàm trong main()***
* ***Sau khi hiểu code, sinh viên tự mình viết lại:***

1. Viết hàm thêm một nút mới vào cây BTS
2. Viết hàm duyệt cây và hiển thị giá trị của các nút theo thứ tự trước, sau, giữa
3. Viết hàm xóa toàn bộ cây BTS
4. Viết hàm trả về số nút trong cây BTS
5. Viết hàm trả về số nút lá trong cây BTS
6. Viết hàm trả về số nút cha có đủ 2 con trong cây BTS
7. Viết hàm tính chiều cao của cây BTS
8. Viết hàm xóa nút có giá trị lớn nhất, nút có giá trị nhỏ nhất

#include <iostream>

using namespace std;

struct node

{ int data;

node\* left;

node\* right;

};

// them 1 nut moi vao cay. Con tro root tro den nut goc cua cay

void insert\_tree(node \* & root, int e)

{

if (root == NULL)

{

root=new node;

root->data=e;

root->left=NULL;

root->right=NULL;

}

else {

if (e < root->data) insert\_tree(root->left,e);

else if (e > root->data) insert\_tree(root->right,e);

else return;

}

}

//duyet cay theo thu tu truoc

void duyet\_truoc (node \* & root)

{ if (root==NULL) return;

else

{ cout<<root->data<<" ";

duyet\_truoc(root->left);

duyet\_truoc(root->right);

};

}

//duyet cay theo thu tu sau

void duyet\_sau (node \* & root)

{ if (root==NULL) return;

else

{

duyet\_sau(root->left);

duyet\_sau(root->right);

cout<<root->data<<" ";

};

}

//duyet cay theo thu tu giua

void duyet\_giua (node \* &root)

{ if (root==NULL) return;

else

{ duyet\_giua (root->left);

cout<<root->data<<" ";

duyet\_giua (root->right);

}

}

// tim kiem nut x tren cay. Neu tim thay thi tra ve dia chi cua nut do.

//Neu khong tim thay thi tra ve NULL

node \* timkiem(node \* & root, int x)

{ if (root==NULL) return NULL;

else

{ if ( root->data==x) return root;

else if (x< root->data) timkiem (root->left, x);

else timkiem(root->right, x);

}

}

//ham dem va tra ve so nut trong cay

int sonut(node \* & root)

{

if (root==NULL) return 0;

else

return (sonut(root->left) + sonut(root->right) +1);

}

//ham dem va tra ve so nut la trong cay

int sonutla(node \* & root)

{ if (root==NULL) return 0;

else

if((root->left==NULL) && (root->right==NULL) )

return 1;

else return ( sonutla(root->left) + sonutla(root->right));

}

//xoa tat ca cac nust trong cay (sau khi xoa, root =NULL)

void xoacay(node \* & root)

{ if (root==NULL) return ;

else if (root->left==NULL && root->right==NULL)

{ delete root; // xoa nut goc dang duoc tro boi bien root;

root=NULL; // gan gia tri NULL cho bien root

}

else

{ xoacay(root->left);

xoacay(root->right);

}

}

//dem so nut cha co du 2 con trong cay

int sonutchadu(node \* & root)

{ if (root==NULL || ((root->left==NULL && root->right==NULL))) return 0;

else if (root->left!=NULL && root->right!=NULL)

return sonutchadu(root->left) + sonutchadu(root->right) +1;

else if (root->left==NULL && root->right!=NULL)

return sonutchadu(root->right);

else return sonutchadu(root->left);

}

int main()

{

//tao 1 cay ban dau rong (root=NULL)

node \* root; //root là biến con trỏ trỏ đến nút gốc của cây

root=NULL;

//yêu cầu:

//1. Gọi hàm để tạo 1 cây BTS có 9 nút

//2. Gọi hàm để hiển thị các nút của cây theo thứ tự trước, sau, giữa

//3. Gọi hàm để kiểm tra xem cây có nút nào có giá trị 5 ko?

//4. Gọi hàm để kiểm tra xem cây có nút nào có giá trị 10 ko?

//5. Gọi hàm để đếm số nút lá của cây

//6. Gọi hàm để đếm số nút cha có đủ 2 con

//7. Gọi hàm để xóa cây

return 0;

}