代码：（截图在后面）

#include <iostream>

#include <iomanip>

using namespace std;

typedef struct Node

{

int data;

int count;

struct Node \*left;

struct Node \*right;

}Node;

class BinaryTree

{

private:

Node \*root;

inline Node\* getNode(int num); //内置输入

Node\* add(Node \*p , int num); //二叉排序树增加节点，内置在Create中

void show(Node \*p); //输出二叉树，内置在Print中

int Find(Node \*p , int target); //查找target在树中是否存在,存在返回1，不存在返回0 ，内置于Search函数

public:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BinaryTree(int num = 0); //构造函数,构造时同步输入根节点数据

void Create(int count = 0); //创建二叉排序树,参数count表示除跟节点之外，需要创建的节点个数

void Print(); //输出中序遍历结果

void Search(int target); //查找target是否存在若存在输出提示，若不存在，则将该数据插入二叉排序树中

};

BinaryTree::BinaryTree(int num)

{

root = new Node;

root->data = num;

root->count = 1;

root->left = NULL;

root->right = NULL;

}

inline Node\* BinaryTree::getNode(int num)

{

Node \*t = new Node;

t->left = NULL;

t->right = NULL;

t->data = num;

t->count = 1;

return t;

}

Node\* BinaryTree::add(Node \*p , int num)

{

if(p == NULL)

{

return this->getNode(num);

}

if(num == p->data)

{

p->count++;

return p;

}

if(num < p->data)

{

if(p->left == NULL)

{

p->left = this->getNode(num);

return p->left;

}

else

{

add(p->left , num);

}

}

else if(num > p->data)

{

if(p->right == NULL)

{

p->right = this->getNode(num);

return p->right;

}

else

{

add(p->right , num);

}

}

}

void BinaryTree::Create(int count)

{

int number = 0;

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

{

cout << "请输入该节点的数据：";

cin >> number;

this->add(root , number);

}

}

void BinaryTree::show(Node \*p)

{

if(p->left != NULL)

{

show(p->left);

}

cout << setw(3) << p->data << "\t" << setw(3) << p->count << endl;

if(p->right != NULL)

{

show(p->right);

}

}

void BinaryTree::Print()

{

cout <<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "数据\t" << "出现次数" << endl;

show(root);

cout <<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

int BinaryTree::Find(Node \*p , int target)

{

if(target == p->data)

{

return 1;

}

if(target < p->data)

{

if(p->left != NULL)

{

if(Find(p->left , target))

{

return 1;

}

else

{

return 0;

}

}

else

return 0;

}

else if(target > p->data)

{

if(p->right != NULL)

{

if(Find(p->right , target))

{

return 1;

}

else

{

return 0;

}

}

else

return 0;

}

}

void BinaryTree::Search(int target)

{

if(Find(root , target))

{

cout << "--->已找到" << target << endl;

}

else

{

cout << "--->没有找到" << target << ",已成功将其插入该二叉树" << endl;

add(root , target);

}

}

int main()

{

BinaryTree tree(45);

tree.Create(5);

tree.Print();

tree.Create(1);

tree.Print();

tree.Search(37);

tree.Print();

tree.Search(20);

tree.Print();

}

截图：

