

数据结构实验报告

张恒硕 2212266 智科 2 班

实验目标： 用顺序和二叉链表两种方法实现三种二叉树遍历

一、顺序二叉树

实验原理：

顺序二叉树，用数组存储。对于某个位置 i 的元素，其左子节点在位置 $2i$ 处，右子节点在位置 $2i+1$ 处。在中序遍历中，可以得到一个有序序列。中序遍历是先访问左子树，然后访问根节点，最后访问右子树。由于左子树的所有节点值都小于根节点的值，而右子树的所有节点值都大于根节点的值，所以中序遍历的结果就是一个升序的有序序列。在三种输出过程中，可以利用函数的迭代。

代码：

```
#include<iostream>
#include <cmath>
using namespace std;
class Tree { //顺序二叉树类
public:
    int size; //阶数
    int whole; //总元素数+1
    int* a; //顺序数组
    Tree() { //无参构造
        size = whole = 0;
        a = NULL;
    }
    Tree(int s) { //有参构造
        size = s;
        whole = pow(2, size) ;
        a = new int[whole]; //构造二叉树顺序表
        a[0] = 0; //将a[0]占住，方便运算
    }
    void Preorder(int i) { //前序输出
        cout << a[i] << " ";
        if (2 * i < whole) {
            Preorder(2 * i);
            Preorder(2 * i + 1);
        }
    }
    void Inorder(int i) { //中序输出
        if (2 * i < whole) {
            Inorder(2 * i);
        }
    }
}
```

```

        cout << a[i] << " ";
        if (2 * i < whole) {
            Inorder(2 * i + 1);
        }
    }
}

void Postorder(int i) { //后序输出
    if (2 * i < whole) {
        Postorder(2 * i);
        Postorder(2 * i + 1);
    }
    cout << a[i] << " ";
}

bool judge() { //判断二叉树逻辑是否正确，即当一节点为零，成为叶子结点，其叉出的两个
                结点不能有值
    for (int i = 1; i < whole; i++) {
        if (a[i] == 0 && (a[2 * i] != 0 || a[2 * i + 1] != 0)) {
            cout << "二叉树逻辑不对!" << endl;
            return 0;
        }
    }
    return 1;
}

};

int main() {
    int size;
    cout << "请输入二叉树的阶数: ";
    cin >> size;
    Tree tree(size);
    cout << "请由上至下、由左至右输入二叉树的元素: ";
    for (int i = 1; i < tree.whole; i++) {
        cin >> tree.a[i];
    }
    if (!tree.judge()) {
        return 0;
    }
    cout << "前序输出: ";
    tree.Preorder(1);
    cout << endl << "中序输出: ";
    tree.Inorder(1);
    cout << endl << "后序输出: ";
    tree.Postorder(1);
    cout << endl;
    return 0;
}

```

输入样例：

1、3 1 2 3 4 5 6 7

即

```
    1
   2  3
  4 5 6 7
```

2、3 1 0 3 4 5 6 7（0 代表该结点为叶子结点，本案例可检验判断树是否合理的函数）

即

```
    1
   0  3
  4 5 6 7
```

运行结果：

1、



```
Microsoft Visual Studio  x + v
请输入二叉树的阶数：3
请由上至下、由左至右输入二叉树的元素：1 2 3 4 5 6 7
前序输出：1 2 4 5 3 6 7
中序输出：4 2 5 1 6 3 7
后序输出：4 5 2 6 7 3 1

C:\Users\zhs20\Desktop\Study\数据结构（刘进超，大二上）\数据结构\顺序二叉树\x64\Debug\顺序二叉树.exe（进程 10636）已退出，
代码为 0。
按任意键关闭此窗口...
```

2、



实现操作:

无参构造

有参构造

前序输出

中序输出

后续输出

判断树是否合理

分析

本代码将树的第一个位置的数组下标记为 1，即 0 处由元素占位，这简化了根结点和其左右结点之间的数学关系，即根节点为 i ，左结点为 $2i$ ，右结点为 $2i+1$ 。而判断树是否合理是针对其输入数组形式数据而设计的，能有效甄别错误的数组结构。

顺序二叉树的优点包括：查找快速，插入和删除操作方便，易于实现。

二、二叉树链表

实验原理:

二叉树链表是一种存储二叉树的结构，它通过为每个节点增加左右指针，分别引用该节点的左子节点和右子节点，从而实现对二叉树的存储。在三种输出过程中，可以利用函数的迭代。

代码:

```
#include<iostream>
using namespace std;
class tree { //树类
public:
    int value; //元素
    tree* left; //左枝
```

```

tree* right; //右枝
tree() { //无参构造
    value = 0;
    left = right = NULL;
}
tree(int x) { //构造叶子结点
    value = x;
    left = right = NULL;
}
void set(int a, int x) { //引申结点，左为1，右为2
    if (a == 1) {
        left = new tree(x);
    }
    if (a == 2) {
        right = new tree(x);
    }
}
};

void Preorder(tree* root) { //前序输出
    if (root == NULL) {
        return;
    }
    cout << root->value << " ";
    Preorder(root->left);
    Preorder(root->right);
}

void Inorder(tree* root) { //中序输出
    if (root == NULL) {
        return;
    }
    Inorder(root->left);
    cout << root->value << " ";
    Inorder(root->right);
}

void Postorder(tree* root) { //后序输出
    if (root == NULL) {
        return;
    }
    Postorder(root->left);
    Postorder(root->right);
    cout << root->value << " ";
}

int main() {
    tree* root = new tree(1);

```

```

    root->left = new tree(2);
    root->left->left = new tree(3);
    root->left->right = new tree(4);
    root->right = new tree(5);
    root->right->left = new tree(6);
    root->right->right = new tree(7);
    cout << "前序输出: ";
    Preorder(root);
    cout << endl << "中序输出: ";
    Inorder(root);
    cout << endl << "后序输出: ";
    Postorder(root);
    cout << endl;
    return 0;
}

```

输入样例:

见上述代码主函数

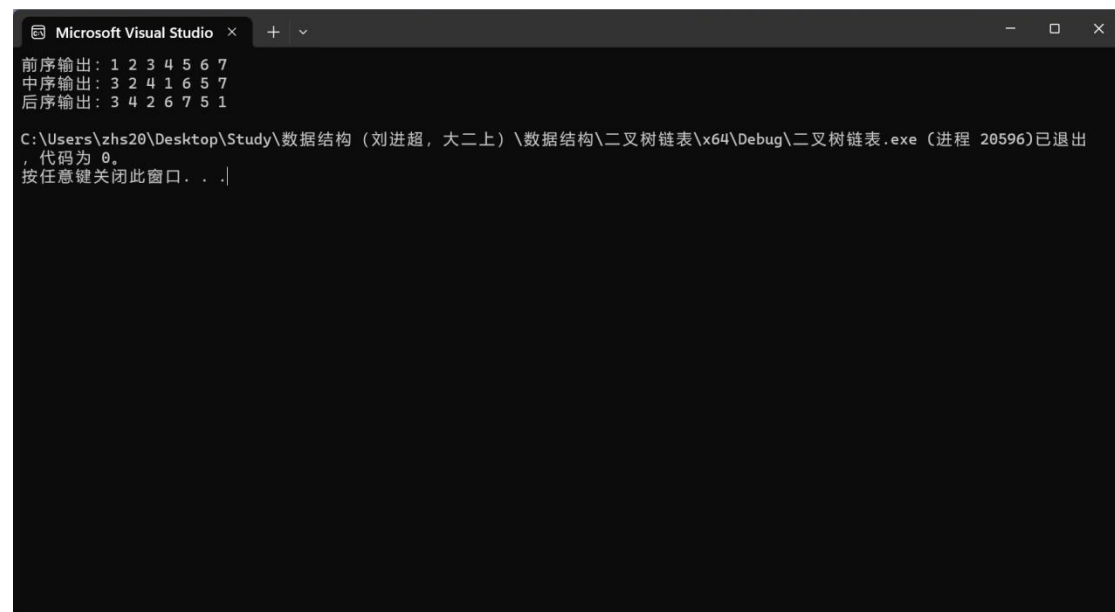
即 (前序输出为数字顺序)

```

    1
   2  5
  3 4 6 7

```

运行结果:



```

Microsoft Visual Studio
前序输出: 1 2 3 4 5 6 7
中序输出: 3 2 4 1 6 5 7
后序输出: 3 4 2 6 7 5 1

C:\Users\zhs20\Desktop\Study\数据结构 (刘进超, 大二上)\数据结构\二叉树链表\x64\Debug\二叉树链表.exe (进程 20596) 已退出, 代码为 0.
按任意键关闭此窗口. . .

```

实现操作:

无参构造

构造叶子结点

引申结点

前序输出

中序输出

后续输出

分析

二叉树链表可以很方便地找到节点的左右子节点，便于进行遍历、查找等操作。可以根据需要动态地分配内存空间，从而有效地利用存储空间。可以实现对二叉树的动态修改和维护，例如插入、删除等操作。