天津大学

数据结构上机实验报告

题目:二叉树染色问题

学生姓名 王雨朦

学生学号 2016229082

学院名称 国际工程师学院

专业 计算机

时间 2016/11/20

目　　录

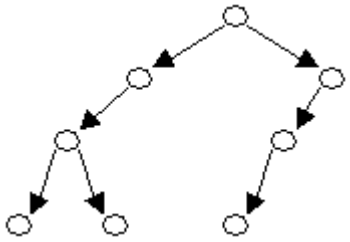
第一章　需求分析

1.1 原题表述

一棵二叉树可以按照如下规则表示成一个由 0、1、2 组成的字符序列，我们称之为"二叉树序列 S"：

$$S=\begin{cases}0 & \text{表示该树没有子节点}\\ 1S_1 & \text{表示该树有一个子节点，}S_1\text{为其子树的二叉树序列}\\ 2S_1S_2 & \text{表示该树有两个子节点，}S_1\text{和}S_2\text{分别表示其两个子树的二叉树序列}\end{cases}$$

例如，下图所表示的二叉树可以用二叉树序列 S=21200110 来表示



现在要对一棵二叉树的节点进行染色。每个节点可以被染成红色、绿色或蓝色。并且，一个节点与其子节点的颜色必须不同，如果该节点有两个子节点，那么这两个子节点的颜色也必须不相同。给定一棵二叉树的二叉树序列，请求出这棵树中最多和最少有多少个点能够被染成绿色。

1.2 解决方案

1) 按照要求，根据输入的二叉树序列构建相对应的二叉树，在构建二叉树的时候，直接对二叉树根节点赋颜色的值 1，左右子树颜色为 2，3 或者 3，2；
2) 递归左子树，再递归右子树；
3) 构建成功后，中序遍历二叉树，对 1，2，3 的数目进行记录；
4) 比较数据，根据要求对应三种颜色，比较计算得最大数目，最小数目。

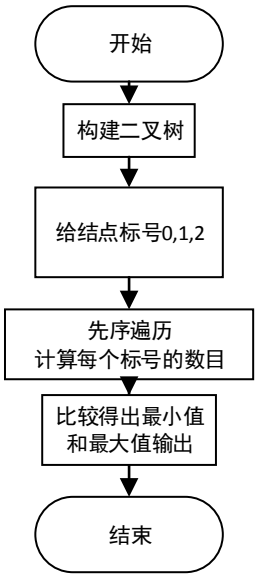
第二章　概要设计

2.1 抽象数据类型

二叉树

```
typedef struct BinTreeNode
{
    int data;
    struct BinTreeNode *lchild, *rchild;
}*BinTree;
CreateBinTree(BinTree & t)
```

2.2 算法



第三章　详细设计

3.1 程序代码

```
#include <iostream>

using namespace std;

static int b=3, c=3;
static int num[3];

typedef struct BinTreeNode
{
    int data;
    struct BinTreeNode *lchild, *rchild;
}*BinTree;

void CreateBinTree(BinTree & t)
{
    char a;
    cin >> a;
    t = new BinTreeNode;
    if(b!=c)
    {
        if((b==0&&c==2)||(c==0&&b==2))
            t->data = 1;
        else if((b==1&&c==0)||(c==1&&b==0))
            t->data = 2;
        else
            t->data = 0;
    }
    else
    {
        if(b==2)
            t->data = 1;
        else
            t->data = 2;
    }

    b=c=t->data;

    if(a=='2')
```

```
    {
        CreateBinTree(t->lchild);
        b=t->lchild->data;
        c=t->data;
        CreateBinTree(t->rchild);
    }
    else if(a=='1')
    {
        CreateBinTree(t->lchild);
        t->rchild = NULL;
    }
    else
    {
        t->lchild=NULL;
        t->rchild=NULL;
    }
}

void PreCount(BinTree t)
{
    if(t!=NULL)
    {
        num[t->data]++;
        PreCount(t->lchild);
        PreCount(t->rchild);
    }
}
int main()
{
    BinTree t;
    CreateBinTree(t);
    PreCount(t);
    int minC=num[0], maxC=num[0];
    for(int i=0; i<3; i++)
    {
        if(num[i]<minC)
            minC=num[i];
        else
            maxC=num[i];
    }
    cout << maxC << " " << minC << endl;
    return 0;
}
```

第四章　调试分析

4.1 调试过程

Bug 名称	逻辑错误
Bug 描述	计算结点树木的时候出现错误
Bug 原因	计算节点数目时候有多种情况没有考虑清楚
Bug 解决方案	把所有情况加入条件
Bug 总结	逻辑不清晰

Bug 名称	无法输出二叉树结点个数值
Bug 描述	运行程序时一直循环出错
Bug 原因	递归没有退出条件
Bug 解决方案	加入退出条件
Bug 总结	递归一定先写退出条件

Bug 名称	逻辑思维不清晰
Bug 描述	开始给根的情况分了三类，解决问题很复杂
Bug 原因	其实可以不用分类可以直接先混淆颜色，只区分是不同颜色着色的即可
Bug 解决方案	有了着色编号，比较累计计算出最大最小值就可以
Bug 总结	进行代码编写时，应该先理清楚思路再进行求解

第五章　测试结果

5.1 测试过程

测试编号	1
测试对象	着色功能
测试输入参数	1122002010
测试步骤	先输入二叉树结点编号，查看是否可以正常输出
测试预期结果	5 2
测试输出结果	5 2
测试分析	程序正确