

**《计算机综合课程设计》**

**课程设计**

题 目：判断任意给定的二叉树是否为满二叉树

**学生姓名： 黄晨箬**\_\_\_\_\_\_\_\_\_\_\_\_

**学 号： 6109119066\_\_\_\_\_\_\_\_\_\_\_**

**专业班级： 计算机科学与技术193班**

**指导教师： 武友新\_\_\_\_\_\_\_\_\_\_\_\_**

**二零二一 年 六 月 二十八 日**

**目 录**

[1.课程设计目的 3](#_Toc76561768)

[2.课程设计题目描述和要求 3](#_Toc76561769)

[2.1课程设计题目描述 3](#_Toc76561770)

[2.2课程设计要求 3](#_Toc76561771)

[3.课程设计报告内容 3](#_Toc76561772)

[3.1程序流程图 3](#_Toc76561773)

[3.2 声明二叉树的结构 4](#_Toc76561775)

[3.3 设定创建二叉树的函数 4](#_Toc76561776)

[3.5 竖向打印二叉树 5](#_Toc76561777)

[3.6 主函数 6](#_Toc76561778)

[3.7 完整代码 7](#_Toc76561779)

[3.8 输出结果 10](#_Toc76561780)

[4. 结论 10](#_Toc76561781)

[参考书目 10](#_Toc76561782)

1.课程设计目的

2.课程设计题目描述和要求

2.1课程设计题目描述

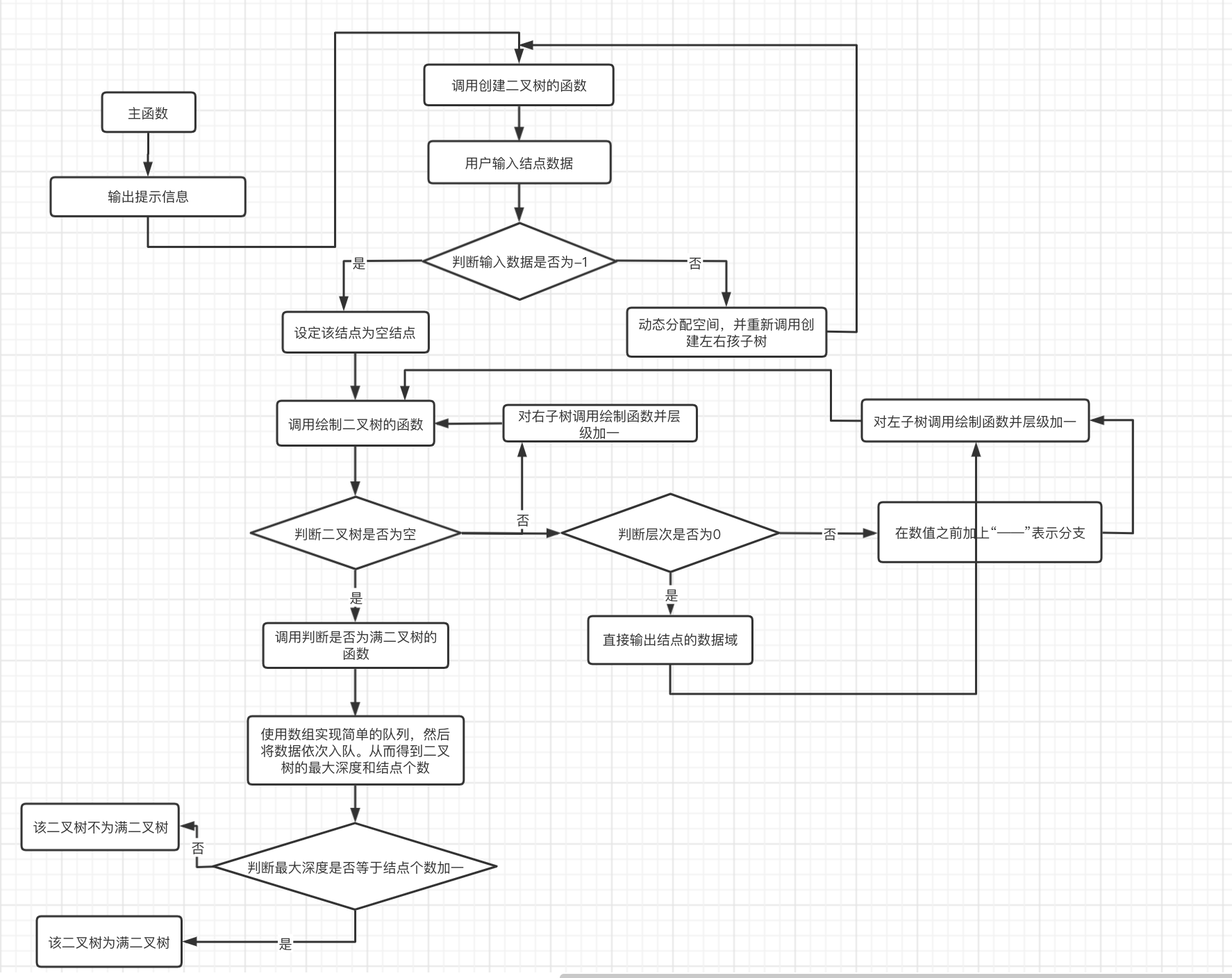
假设二叉树使用链式存储方式，编写一个程序，判断任意给定的二叉树是否为满二叉树。

2.2课程设计要求

通过课程设计能够独立完成所布置的任务，能够对一个较大的题目进行分析，给出解题的思路和方法，利用所学的专业知识解决相关的问题。

3.课程设计报告内容

3.1程序流程图



3.2 声明二叉树的结构

//声明二叉树结构

struct BiTree {

    int data;   //数值域

    BiTree \*Lchild;

    BiTree \*Rchild;     //指针域

};

3.3 设定创建二叉树的函数

//设定创建二叉树的函数

void CreateBiTree(BiTree \* &T) {

    int d;

    cin >> d;  //输入数据

    //当输入-1时为空结点

    if(d == -1) {

        T = NULL;

    }

    else {

        T = (BiTree \*)malloc(sizeof(BiTree));

    //使用sizeof计算字节数，使用malloc动态分配内存空间

        T->data = d;

    //该节点的数据域为输入的内容

        CreateBiTree(T->Lchild);    //递归创建左子树

        CreateBiTree(T->Rchild);    //递归创建右子树

    }

}

3.4 判断是否为满二叉树

//判断是否为满二叉树

bool isFull(BiTree \* T){

    struct Node {

        BiTree \*tree;

        int deep;        //深度

    };

    Node q[100];          //使用数组实现简单的队列

    if (T == NULL) return true;

    int rear = 0, front = 0;

    q[rear].tree = T;

    q[rear].deep = 0;

    int dMax = 0;          //表示最大深度

    int num = 0;          //表示结点个数

    while (rear >= front) {

        if (q[front].tree != NULL) {

            num++;

            q[++rear].tree = q[front].tree->Lchild;      //入队

            q[rear].deep = q[front].deep + 1;

            q[++rear].tree = q[front].tree -> Rchild;   //入队

            q[rear].deep = q[front].deep + 1;

            if (dMax < q[rear].deep)

                dMax = q[rear].deep;

        }

        front++;

    }

    cout << "最大深度为 = " << dMax << "  结点个数为 = " << num << endl;

    return 1 << dMax == num + 1;

}

3.5 竖向打印二叉树

/\*\*

    按竖向打印二叉树。若二叉树为空，则执行返回。

    否则，递归执行打印右子树，同时将层次加一，

    然后根据曾局打印空格数，输出根结点，最后递归打印左子树。

\*\*/

void PrintTree(BiTree \*T, int l) {

    if (!T) {

        return;

    }

    PrintTree(T->Rchild, l+1);

    for(int i = 0; i < l; i++) {

        cout << "   ";

    }

    if (l == 0) {

        cout << T->data << endl;

    }else {

        cout << "——" << T->data << endl;

    }

    PrintTree(T->Lchild, l+1);

}

3.6 主函数

//主函数

int main() {

    BiTree \*T;

    cout << "使用先序遍历的方法输入结点数据，当输入-1时表示该结点为空 ";

    CreateBiTree(T);

    cout << "建好的树按照横向输出如下 " << endl;

    PrintTree(T,0);

    string a = isFull(T) ? "是" : "不是";

    cout << "该二叉树 " << a << "满二叉树";

    return 0;

}

3.7 完整代码

//假设二叉树采用链式存储方式，编写一个程序，判断任意给定的二叉树是否为满二叉树。

//一个满二叉树总结点树等于2的最大深度次方-1。

//最大深度是通过层次遍历求出

#include <iostream>

using namespace std;

//声明二叉树结构

struct BiTree {

    int data;   //数值域

    BiTree \*Lchild;

    BiTree \*Rchild;     //指针域

};

//设定创建二叉树的函数

void CreateBiTree(BiTree \* &T) {

    int d;

    cin >> d;  //输入数据

    //当输入-1时为空结点

    if(d == -1) {

        T = NULL;

    }

    else {

        T = (BiTree \*)malloc(sizeof(BiTree));   //使用sizeof计算字节数，使用malloc动态分配内存空间

        T->data = d;             //该节点的数据域为输入的内容

        CreateBiTree(T->Lchild);    //递归创建左子树

        CreateBiTree(T->Rchild);    //递归创建右子树

    }

}

//判断是否为满二叉树

bool isFull(BiTree \* T){

    struct Node {

        BiTree \*tree;

        int deep;           //深度

    };

    Node q[100];            //使用数组实现简单的队列

    if (T == NULL) return true;

    int rear = 0, front = 0;

    q[rear].tree = T;

    q[rear].deep = 0;

    int dMax = 0;           //表示最大深度

    int num = 0;            //表示结点个数

    while (rear >= front) {

        if (q[front].tree != NULL) {

            num++;

            q[++rear].tree = q[front].tree->Lchild;     //入队

            q[rear].deep = q[front].deep + 1;

            q[++rear].tree = q[front].tree -> Rchild;   //入队

            q[rear].deep = q[front].deep + 1;

            if (dMax < q[rear].deep)

                dMax = q[rear].deep;

        }

        front++;

    }

    cout << "最大深度为 = " << dMax << "  结点个数为 = " << num << endl;

    return 1 << dMax == num + 1;

}

/\*\*

    按竖向打印二叉树。若二叉树为空，则执行返回。

    否则，递归执行打印右子树，同时将层次加一，

    然后根据曾局打印空格数，输出根结点，最后递归打印左子树。

\*\*/

void PrintTree(BiTree \*T, int l) {

    if (!T) {

        return;

    }

    PrintTree(T->Rchild, l+1);

    for(int i = 0; i < l; i++) {

        cout << "   ";

    }

    if (l == 0) {

        cout << T->data << endl;

    }else {

        cout << "——" << T->data << endl;

    }

    PrintTree(T->Lchild, l+1);

}

//主函数

int main() {

    BiTree \*T;

    cout << "使用先序遍历的方法输入结点数据，当输入-1时表示该结点为空 ";

    CreateBiTree(T);

    cout << "建好的树按照横向输出如下 " << endl;

    PrintTree(T,0);

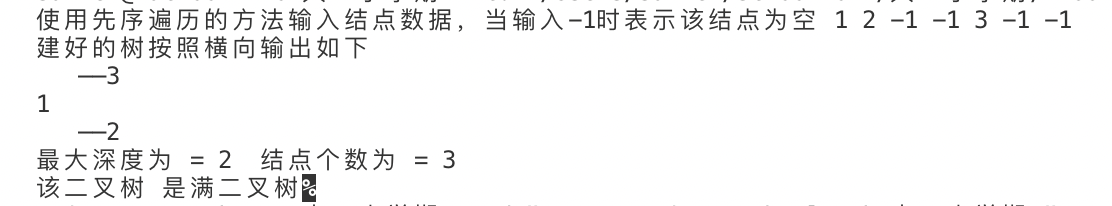
    string a = isFull(T) ? "是" : "不是";

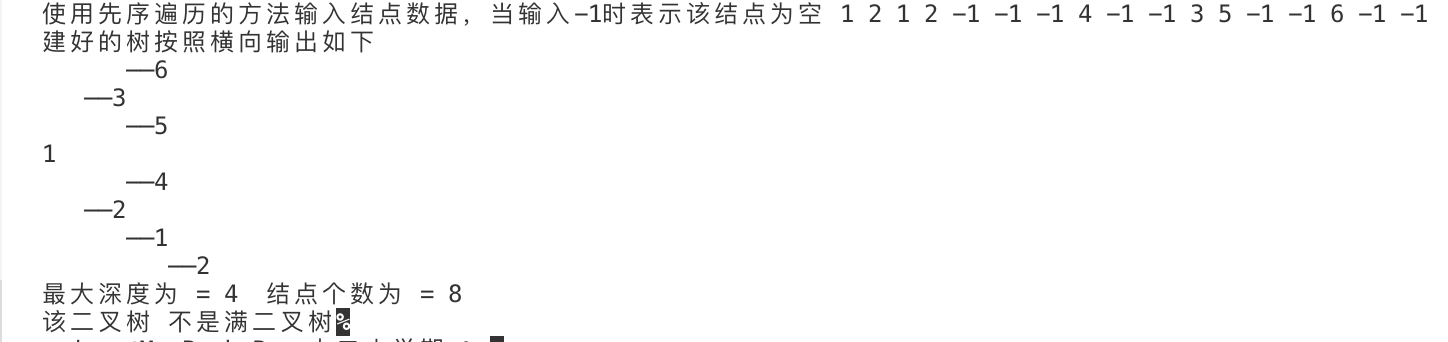
    cout << "该二叉树 " << a << "满二叉树";

    return 0;

}

3.8 输出结果





4. 结论

二叉树（英语：Binary tree）是每个节点最多只有两个分支（即不存在分支度大于2的节点）的[树结构](https://zh.wikipedia.org/wiki/%E6%A0%91%E7%BB%93%E6%9E%84" \o "树结构)。通常分支被称作“左子树”或“右子树”。二叉树的分支具有左右次序，不能随意颠倒。

二叉树的第{\displaystyle i}i层至多拥有{\displaystyle 2^{i-1}}2的i-1个节点；深度为{\displaystyle k}k的二叉树至多总共有{\displaystyle 2^{\begin{aligned}k+1\end{aligned}}-1}2的k+1次方加1个节点（定义根节点所在深度k=0）kkkkkkdasdasdf，而总计拥有节点数符合的，称为“满二叉树”。可以通过树的最大深度和结点个数之间的关系来判断一棵二叉树是否为满二叉树。

在绘制二叉树时，可以通过递归判断该结点的深度来将数据域放在指定的地点从而正确的画出该树。

**参考书目**

1. 塞奇威克 (Robert Sedgewick)/韦恩 (Kevin Wayne),《算法（第4版）》,人民邮电出版社,2012-10-1.

2. 斯基恩纳,《算法设计手册》,清华大学出版社,2009-9.

3. Thomas H.Cormen/Charles E.Leiserson/Ronald L.Rivest/Clifford Stein,《算法导论（原书第3版）》,机械工业出版社,2012-12.

4. 程杰,《大话数据结构》,清华大学出版社,2011-6.

5. Aditya Bhargava,《算法图解》,人民邮电出版社,2017-3.

6. Mark Allen Weiss,《数据结构和算法分析》,人民邮电出版社,2007-6.

7. [Stanley B. Lippman](https://book.douban.com/search/Stanley%20B.%20Lippman)/Josée Lajoie/Barbara E. Moo,《C++ Primer 中文版（第 5 版）》,电子工业出版社,2013-9-1.

8. Scott Meyers,《Effective C++》,电子工业出版社,2006-7.

9. Bjarne Stroustrup,《C++语言的设计和演化》,机械工业出版社,2002-1.

10. [Andrew Koenig](https://book.douban.com/search/Andrew%20Koenig)/Barbara Moo,《C++沉思录》,人民邮电出版社,2002-11-01.

11. [David Vandevoorde](https://book.douban.com/search/David%20Vandevoorde)/Nicolai M.Josuttis,《C++Templates中文版》,人民邮电出版社,2008-2.

12. avid Abrahams/Aleksey Gurtovoy,《C++模版源编程》,暂无出版社,2010-1.

13. Barbara Johnston,《现代C++程序设计》,机械工业出版社,2008-9.

14.

15.

…………………………………………………………………………..

(要求:五号字，宋体，单倍行距。按作者、书名、出版社、出版时间格式逐一列出，中间用逗号格开，不少于15篇)