哈尔滨工业大学

**计算机科学与技术学院/国家示范性软件学院**

**2020年秋季学期**

**《软件过程与工具》课程**

**实验报告**

**Lab 1：增量模型应用+结对编程实践**

|  |  |  |
| --- | --- | --- |
| **姓名** | **学号** | **联系方式** |
| 邓昆昆 | 1183710129 | 15804877376 |
| 梅智敏 | 1183710118 | 13385658102 |

目 录

[1 项目概述 1](#_Toc54460542)

[1.1 总体需求分析 1](#_Toc54460543)

[1.2 软件系统总体设计 3](#_Toc54460544)

[1.3 开发任务计划 4](#_Toc54460545)

[1.4 开发环境与工具 5](#_Toc54460546)

[2 第1轮迭代：基本系统 5](#_Toc54460547)

[2.1 迭代目标 5](#_Toc54460548)

[2.2 需求分析 5](#_Toc54460549)

[2.3 设计方案 5](#_Toc54460550)

[2.4 结对编程过程记录 11](#_Toc54460551)

[2.5 单元测试 12](#_Toc54460552)

[2.6 集成测试 12](#_Toc54460553)

[3 第2轮迭代：增量1 13](#_Toc54460554)

[3.1 迭代目标 13](#_Toc54460555)

[3.2 需求分析 13](#_Toc54460556)

[3.3 设计方案 13](#_Toc54460557)

[3.4 结对编程过程记录 17](#_Toc54460558)

[3.5 单元测试 18](#_Toc54460559)

[3.6 集成测试 18](#_Toc54460560)

[4 第3轮迭代：增量2 18](#_Toc54460561)

[4.1 迭代目标 18](#_Toc54460562)

[4.2 需求分析 18](#_Toc54460563)

[4.3 设计方案 19](#_Toc54460564)

[4.4 结对编程过程记录 21](#_Toc54460565)

[4.5 单元测试 22](#_Toc54460566)

[4.6 集成测试 22](#_Toc54460567)

[5 项目总结 22](#_Toc54460568)

[5.1 增量模型应用总结 22](#_Toc54460569)

[5.2 结对编程总结 23](#_Toc54460570)

# 项目概述

（1）项目名称：二叉树遍历算法教学演示小软件

（2）项目简要介绍：（200字之内）

二叉树是非常重要的一种数据结构，遍历是[二叉树](https://baike.baidu.com/item/%E4%BA%8C%E5%8F%89%E6%A0%91/1602879)上最重要的运算之一，是二叉树上进行其它运算之基础。对于初学者而言，理解二叉树的三种遍历方式（前序、中序、后序）是十分必要的。

本项目基于JSwing进行可视化编程，让用户可以通过图形化界面实现以下功能：

* 添加节点
* 展示当前的二叉树
* 选择想要展示的遍历方式
* 以动画的方式展示三种遍历算法的执行过程

通过图形化界面让用户直观地了解二叉树的建立和遍历过程，会给他们留下更深的印象，从而帮助他们进一步学习。

## 总体需求分析

（1）项目要开发的软件的目的和意义（回答为什么要开发该项目）。

目的：

对于数据结构初学者而言，理解二叉树的遍历算法非常重要。因此，我们想要开发出一款拥有图形化界面的小软件，能够让用户直观地理解二叉树的建立和三种遍历过程。

意义：

本软件简易小巧、操作简单且为图形化界面。一方面，此软件非常适合学生个人在课下学习时使用，它能够让学生在动手的过程中去深入理解二叉树的建立和遍历过程；另一方面，由于此软件简易小巧，也很适合教师的课堂教学使用。

（2）软件系统的涉众分析（列出该系统跟哪些相关人员有关，其关心的核心内容）。

用户（学生、老师）关心的核心内容：

* 是否简单易用
* 功能是否齐全
* 添加节点过程是否清晰易懂
* 二叉树的结构展示是否正确、清晰、美观
* 演示遍历过程是否清晰易懂

开发者（2名软件开发人员）关心的核心内容：

* 软件开发过程是否简单
* 软件是否支持不同的操作系统（如Window、Mac、Linux）
* 软件的模块分层是否清晰

监督人员（应用商店管理员）关心的核心内容：

* 软件是否安全可靠
* 软件是否免费

（3）软件系统的功能需求：可从用户视角来描述功能需求（即用户故事）。

从用户的角度来看，我希望此软件应该可以完成如下功能：

* 可以人为控制节点的添加，从而完成二叉树的建立
* 可以实时展示当前二叉树的结构图
* 可以人为选择三种遍历方式
* 能够以动画的方式在二叉树的结构图上演示所选泽的遍历方式执行流程
* 可以人为控制节点及后续子树的删除，同样需要实时展示结构
* 可以打印出遍历的order

（4）软件系统的非功能需求：包括性能要求、界面要求、技术指标等。

此软件不仅能够满足基本的功能需求，还可以满足一些非功能需求：

性能要求：

* 用户的所有操作不需要使用键盘，只需要通过鼠标选择即可
* 软件可以实时展示二叉树的结构
* 演示遍历过程应该以通俗易懂的动画方式展示

界面要求：

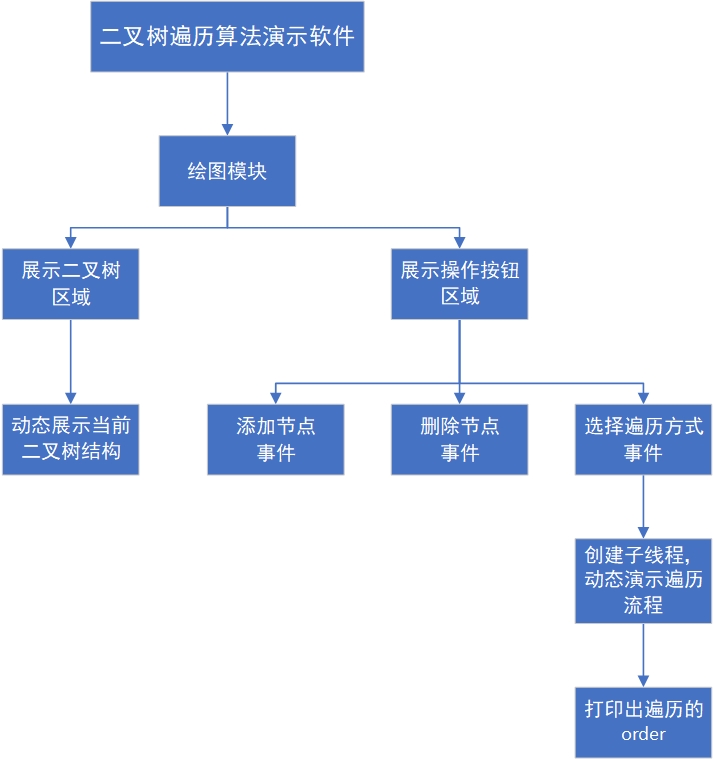
* 用户的选择区域应该和二叉树的展示区域分开
* 能够正确、清晰、美观地展示二叉树的结构图

技术指标：

* 软件应该支持多种操作系统
* 有较好的可拓展性

## 软件系统总体设计

（1）软件系统功能结构设计方案：给出软件功能模块及模块间的关联关系。



（2）软件系统相关的数据结构设计：给出必要的数据结构设计，包括配置参数、缓冲数据结构等。

* 线性表：二叉树为了存储节点、“允许添加子节点的序号”以及遍历时的序号order，使用了线性表。
* 二叉树：本次迭代使用的核心数据结构就是二叉树，我们使用自定义的Node类和BinaryTree类。Node类拥有左右孩子；BinaryTree类拥有节点列表，且拥有增加节点和删除节点方法。
* 栈Stack：为了实现节点及子树的删除，需要使用Stack。

## 开发任务计划

迭代过程的划分，各迭代计划完成的增量任务，进度安排。

（1）第1轮迭代的任务：

主要任务：

* A：创建ADT：Node和BinaryTree，允许添加节点以完成二叉树的建立
* B：能够在可视化界面中展示二叉树的结构
* C：能够在可视化界面中显示各种操作按钮
* D：允许用户通过鼠标点击操作按钮来实现节点的添加，并实时展示当前二叉树的结构
* E：将Java源程序打包成exe文件

进度安排：

|  |  |
| --- | --- |
| 时间段 | 完成任务 |
| 2020.10.21 上午 | A |
| 2020.10.21 下午  2020.10.21 晚上 | B、C  D、E |

（2）第2轮迭代的任务：

主要任务：

* A：在二叉树上实现3种遍历算法
* B：用户可以在图形化界面上选择遍历方式，随后以动画的方式展示遍历的流程
* C：限制二叉树的高度，从而保证二叉树结构的展示美观

进度安排：

|  |  |
| --- | --- |
| 时间段 | 完成任务 |
| 2020.10.22 上午 | A |
| 2020.10.22 下午 | B、C |

（3）第3轮迭代的任务：

主要任务：

* A：实现删除节点及其子树的功能
* B：能够打印出遍历的顺序order

进度安排：

|  |  |
| --- | --- |
| 时间段 | 完成任务 |
| 2020.10.23 上午 | A |
| 2020.10.21 晚上 | B |

## 开发环境与工具

开发语言：java

开发环境：JDK 1.8

工具：eclipse 2019

# 第1轮迭代：基本系统

## 迭代目标

简要介绍本轮迭代的增量任务目标，所需完成的任务。

（1）任务1：

创建ADT：Node和BinaryTree，允许添加节点以完成二叉树的建立

（2）任务2：

能够在可视化界面中展示二叉树的结构

（3）任务3：

能够在可视化界面中显示各种操作按钮

（4）任务4：

允许用户仅通过鼠标点击操作按钮来实现节点的添加，并实时展示当前二叉树的结构

## 需求分析

本轮迭代所开发模块的详细需求分析。

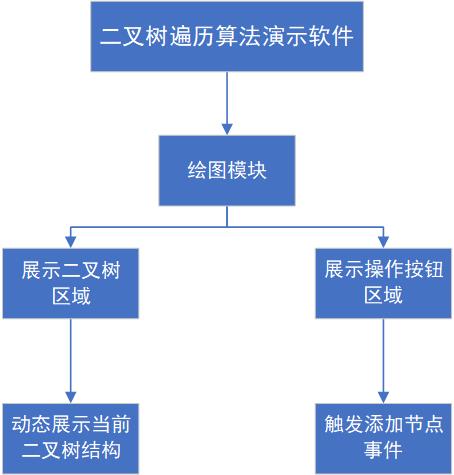
* 用户可以通过图形化界面选择“父节点”和“方向”，从而完成添加节点的操作
* 软件可以实时展示当前二叉树的结构

## 设计方案

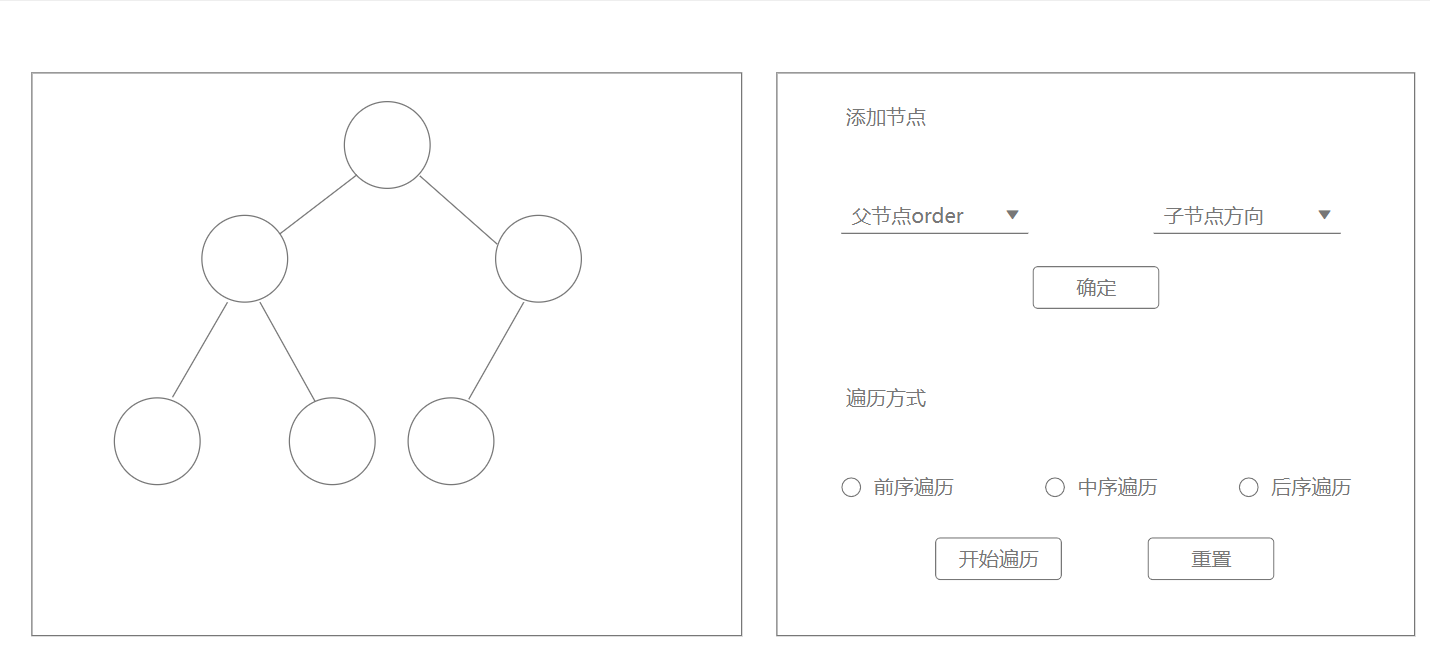
针对本轮迭代的开发内容给出设计方案：

（1）详细设计方案：模块功能结构设计，界面原型设计等。

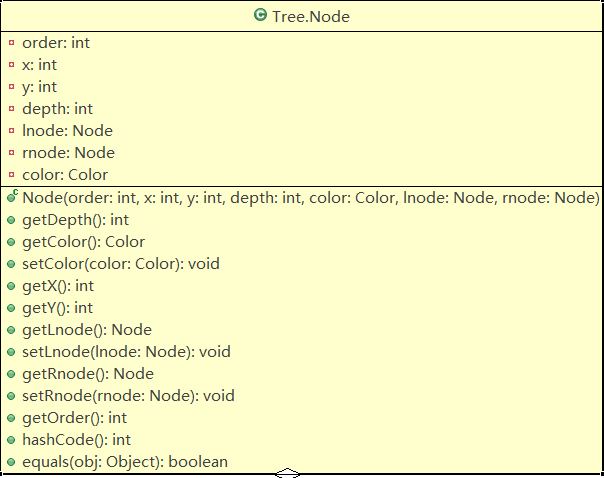
模块功能结构设计如下



界面原型设计如下



（2）模块相关的设计类（Design Class）：给出每个类的名字、属性、方法。

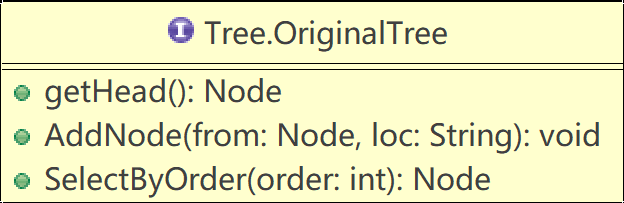


|  |  |
| --- | --- |
| 类名 | 解释说明 |
| Node | 代表二叉树中的节点 |

|  |  |
| --- | --- |
| 属性名 | 解释说明 |
| order  x  y  depth  lnode  rnode  color | 节点的序号  节点在界面中的x坐标  节点在界面中的y坐标  节点在二叉树中所处的深度  节点的左孩子  节点的右孩子  节点的颜色 |

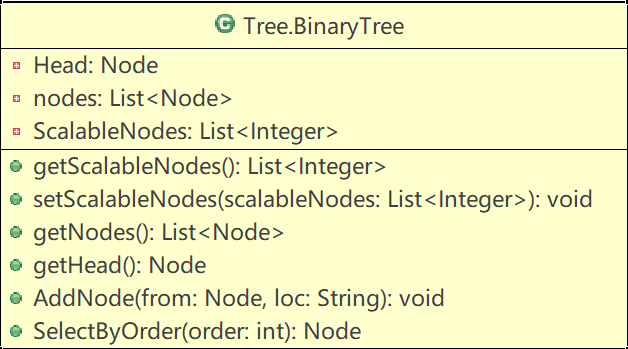
注意：下面对方法的介绍不包含对field的Getter和Setter方法

|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| Node | 构造方法，用于构建一个Node对象 |
| equals | 重写的equals方法 |
| hashCode | 重写的hashcode方法 |



|  |  |
| --- | --- |
| 类名 | 解释说明 |
| OriginalTree | 二叉树interface |

|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| getHead | 得到树的根节点 |
| AddNode | 向某一个节点添加左孩子或右孩子 |
| SelectByOrder | 根据序号得到相应节点 |

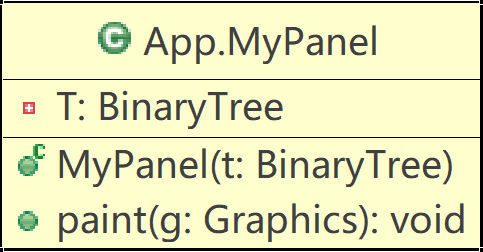


|  |  |
| --- | --- |
| 类名 | 解释说明 |
| BinaryTree | 二叉树class，实现了interface |

|  |  |
| --- | --- |
| 属性名 | 解释说明 |
| Head  nodes  ScalableNodes | 根节点  树中的节点列表  “可添加sub-Node”的节点序号list |

注意：下面对方法的介绍不包含对field的Getter和Setter方法

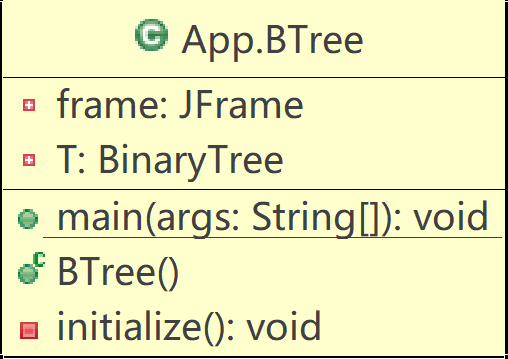
|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| AddNode | 向某一个节点添加左孩子或右孩子 |
| SelectByOrder | 根据序号得到相应节点 |
| getHead | 得到树的根节点 |



|  |  |
| --- | --- |
| 类名 | 解释说明 |
| MyPanel | 继承Panel类，是自定义画板类 |

|  |  |
| --- | --- |
| 属性名 | 解释说明 |
| T | 要在画板中展示的二叉树 |

|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| paint | 将二叉树的结构画出来 |



|  |  |
| --- | --- |
| 类名 | 解释说明 |
| BTree | 二叉树遍历算法演示软件App |

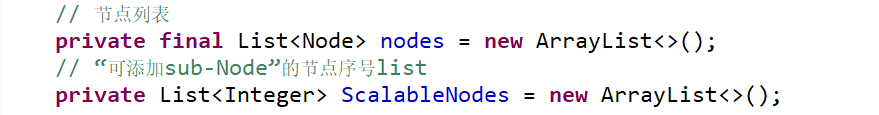
|  |  |
| --- | --- |
| 属性名 | 解释说明 |
| Frame  T | JFrame对象，作为可视化界面  二叉树对象 |

注意：下面对方法的介绍不包含对field的Getter和Setter方法

|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| BTree | 构造方法 |
| initialize | 初始化JFrame中的内容 |
| main | Main方法，程序的入口 |

（3）所需的数据结构。

* 二叉树结构：本次迭代使用的核心数据结构就是二叉树，我们使用自定义的Node类和BinaryTree类。
* 线性表：二叉树为了存储节点以及“允许添加子节点的序号”，使用了线性表。



## 结对编程过程记录

**（1）角色切换与任务分工**

表1-1 第1轮迭代过程结对编程角色与任务分工

| 日期 | 时间(HH:MM - HH:MM) | 驾驶员角色 | 领航员角色 | 本段时间的任务 |
| --- | --- | --- | --- | --- |
| 2020.10.21 | 9:00 ——11:30 | 邓昆昆 | 梅智敏 | 创建ADT：Node和BinaryTree，允许添加节点以完成二叉树的建立 |
| 2020.10.21 | 14:00——16:00 | 梅智敏 | 邓昆昆 | 能够在可视化界面中展示二叉树的结构 |
| 2020.10.21 | 16:00——16:30 | 邓昆昆 | 梅智敏 | 能够在可视化界面中显示各种操作按 |
| 2020.10.21 | 19:00——21:00 | 梅智敏 | 邓昆昆 | 允许用户仅通过鼠标点击操作按钮来实现节点的添加，并实时展示当前二叉树的结构 |

**（2）工作日志**

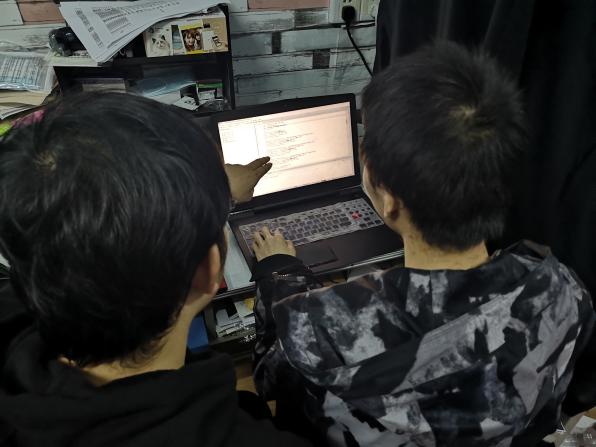
由领航员负责记录，记录结对编程期间的遇到的问题、两人如何通过交流合作解决每个问题的。

表1-2 第1轮迭代过程结对编程工作日志

| 日期/时间 | 问题描述 | 最终解决方法 | 交流过程 |
| --- | --- | --- | --- |
| 2020.10.21 | “节点”的下拉列表框的响应是元素状态的改变，第一个默认元素不会触发响应“方向”下拉框content的改变，导致后者content为null | 直接在“方向”下拉列表框初始化item | 经过双方讨论 |
| 2020.10.21 | 直接编写Java可视化代码太过繁杂 | 使用window builder插件帮助我们进行可视化 | 上网查阅博客 |
| 2020.10.22 | Graphics里面画图问题 | 通过自定义Mypanel继承JPanel，重写paint方法 | 上网查博客 |

**（3）结对编程工作现场照片**

请其他同学帮助拍摄结对编程现场照片至少2张。

第1次迭代结对编程现场照片1 第1次迭代结对编程现场照片2

## 单元测试

对本轮迭代所开发的每个类进行单元测试。

表1-3 第1轮迭代过程的单元测试记录

| 日期/时间 | 类名：方法名 | 输入数据 | 输出数据 | 结论 |
| --- | --- | --- | --- | --- |
| 2020.10.22 | Node：getOrder  Node：getX  Node：getY  Node：getDepth  Node：getColor  Node：getLnode  Node：getRnode | 1 | 1 | True |
| 2020.10.22 | 20 | 20 | True |
| 2020.10.22 | 20 | 20 | True |
| 2020.10.22 | 0 | 0 | True |
| 2020.10.22 | BLACK | BLACK | True |
| 2020.10.22 | Null | Null | True |
| 2020.10.22 | Null | Null | True |
| 2020.10.22 | BinaryTree：getHead  BinaryTree：getNodes  BinaryTree：getScalableNodes  BinaryTree ：SelectByOrder | Null | Head | True |
| 2020.10.22 | Null | List<Node> | True |
| 2020.10.22 | Null | List<Integer> | True |
| 2020.10.22 | 2 | Node[2] | True |

## 集成测试

对本轮迭代所开发功能模块进行测试。

表1-4 第1轮迭代过程的集成测试记录

| 日期/时间 | 功能名称 | 测试用例 | 测试结果描述 | 结论 |
| --- | --- | --- | --- | --- |
| 2020.10.23 | 添加节点 | root节点增加左孩子 | 添加成功 | True |
| 2020.10.23 | 二叉树的建立 | root节点增加左孩子，2号节点添加右孩子…… | 建立成功 | True |

# 第2轮迭代：增量1

## 迭代目标

简要介绍本轮迭代的增量任务目标，所需完成的任务。

（1）任务1：在二叉树上实现3种遍历算法

（2）任务2：用户可以在图形化界面上选择遍历方式，随后以动画的方式展示遍历的流程

（3）任务3：限制二叉树的高度，从而保证二叉树结构的展示美观

## 需求分析

本轮迭代所开发模块的详细需求分析。

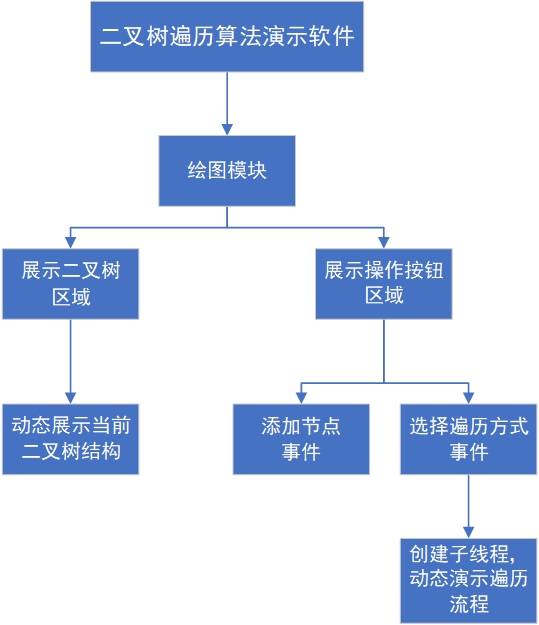
* 用户可以用鼠标选择：“前序遍历”、“中序遍历”、“后序遍历”其中一种
* 点击确定之后软件会以动画形式演示当前二叉树的遍历流程

## 设计方案

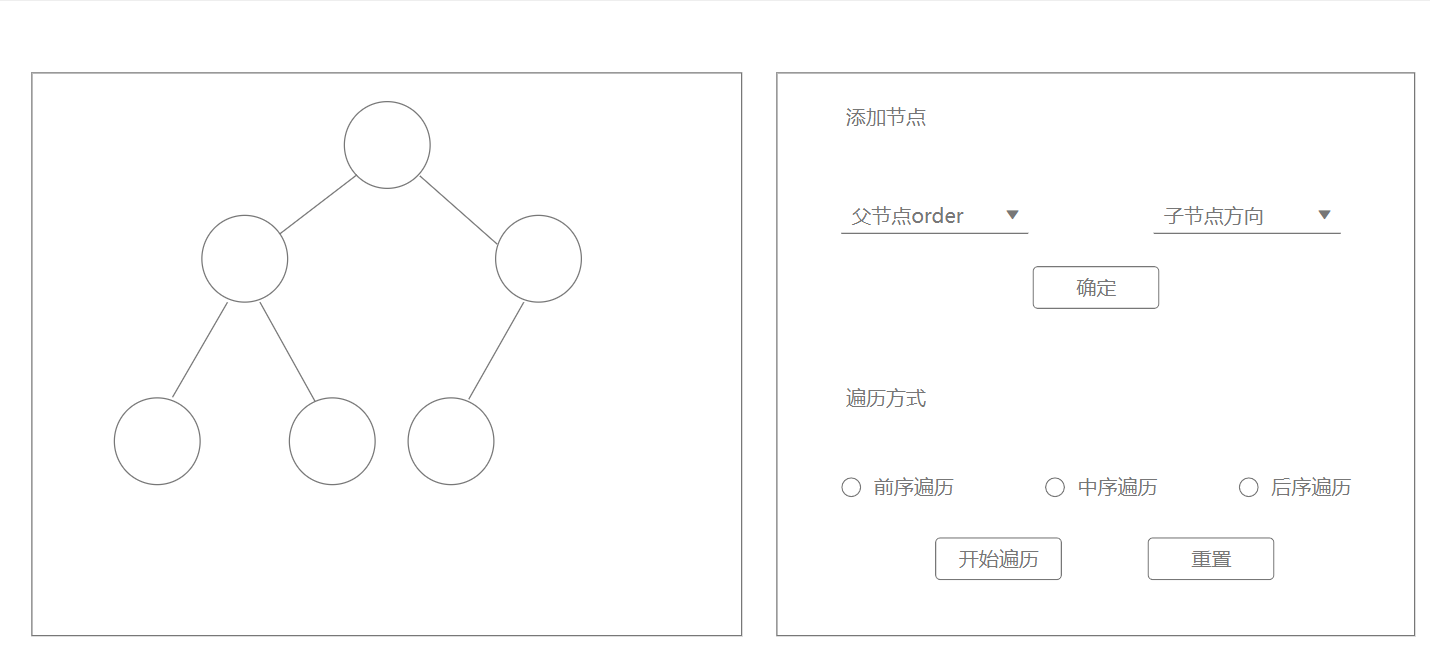
针对本轮迭代的开发内容给出设计方案：

（1）详细设计方案：模块功能结构设计，界面原型设计等。

模块功能结构设计如下

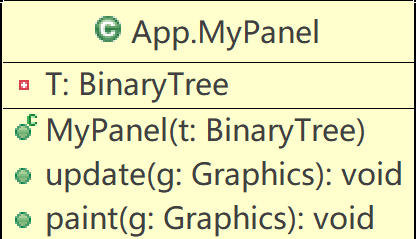


界面原型设计如下



（2）模块相关的设计类（Design Class）：给出每个类的名字、属性、方法。

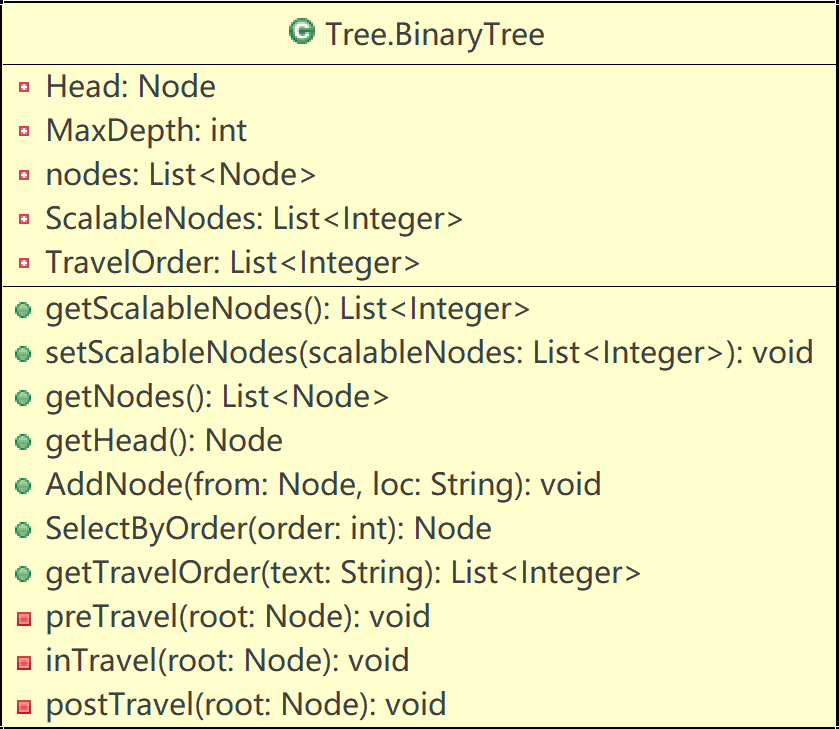
注意：和基本系统相同的部分不在此展示



|  |  |
| --- | --- |
| 类名 | 解释说明 |
| MyPanel | 自定义的画板，继承JPanel |

|  |  |
| --- | --- |
| 属性名 | 解释说明 |
| T | 二叉树对象 |

|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| update  MyPanel  paint | 更新Graphics对象  构造器  将当前二叉树结构画出来 |



注意：此处只介绍新增的方法和属性

|  |  |
| --- | --- |
| 类名 | 解释说明 |
| BinaryTree | 二叉树class |

|  |  |
| --- | --- |
| 属性名 | 解释说明 |
| MaxDepth  TravelOrder | 最大树高  遍历到的节点order |

|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| preTravel  inTravel  postTravel  getTravelOrder | 前序遍历  中序遍历  后序遍历  根据text来获取不同的遍历顺序 |

（3）所需的数据结构。

* 线性表：除了在基本系统中使用的ScalableNodes外，在此迭代中我们使用List<Interger> TravelOrder来记录遍历顺序的order
* 二叉树：和基本系统一致

## 结对编程过程记录

**（1）角色切换与任务分工**

表2-1 第2轮迭代过程结对编程角色与任务分工

| 日期 | 时间(HH:MM - HH:MM) | 驾驶员角色 | 领航员角色 | 本段时间的任务 |
| --- | --- | --- | --- | --- |
| 2020.10.22 | 9:00——10:00 | 梅智敏 | 邓昆昆 | 限制树高，并在二叉树上实现3种遍历算法 |
| 2020.10.22 | 18:00——20:00 | 邓昆昆 | 梅智敏 | 以动画形式演示3种遍历算法的流程 |

**（2）工作日志**

由领航员负责记录，记录结对编程期间的遇到的问题、两人如何通过交流合作解决每个问题的。

表2-2 第2轮迭代过程结对编程工作日志

| 日期/时间 | 问题描述 | 最终解决方法 | 交流过程 |
| --- | --- | --- | --- |
| 2020.10.22 | 绘图的时候会有“闪烁”问题 | 将Panel换成新版的JPanel | 和其他同学讨论 |
| 2020.10.22 | 不能动态展示遍历流程，只能给出最终的结果 | 创建子线程，并间歇性sleep，从而产生动画效果 | 通过查阅博客 |
| 2020.10.22 | 在展示遍历流程的过程中，若点击操作按钮，则会引发线程的冲突 | 在展示遍历流程的子线程没有结束时，将操作按钮设置为“不可用”状态 | 通过查阅网络资料 |

**（3）结对编程工作现场照片**

请其他同学帮助拍摄结对编程现场照片至少2张。

第2次迭代结对编程现场照片1 第2次迭代结对编程现场照片2

## 单元测试

对本轮迭代所开发的每个类进行单元测试。

表2-3 第2轮迭代过程的单元测试记录

| 日期/时间 | 类名：方法名 | 输入数据 | 输出数据 | 结论 |
| --- | --- | --- | --- | --- |
| 2020.10.23 | BinaryTree:getTravelOrder | 节点序号1-3的二叉树的前序遍历 | [1, 2, 3] | true |
| 2020.10.23 | BinaryTree:getTravelOrder | 节点序号1-3的二叉树的中序遍历 | [2, 1, 3] | true |
| 2020.10.23 | BinaryTree:getTravelOrder | 节点序号1-3的二叉树的后序遍历 | [2, 3, 1] | true |

## 集成测试

对本轮迭代所开发功能模块进行测试。

表2-4 第2轮迭代过程的集成测试记录

| 日期/时间 | 功能名称 | 测试用例 | 测试结果描述 | 结论 |
| --- | --- | --- | --- | --- |
| 2020.10.23 | 二叉树的遍历 | 节点序号1-3的二叉树 | 遍历成功 | true |
| 2020.10.23 | 二叉树动画的遍历 | 任意二叉树 | 动画演示效果成功 | true |

# 第3轮迭代：增量2

## 迭代目标

* 实现删除节点及后续子树功能
* 能够打印出遍历的顺序order

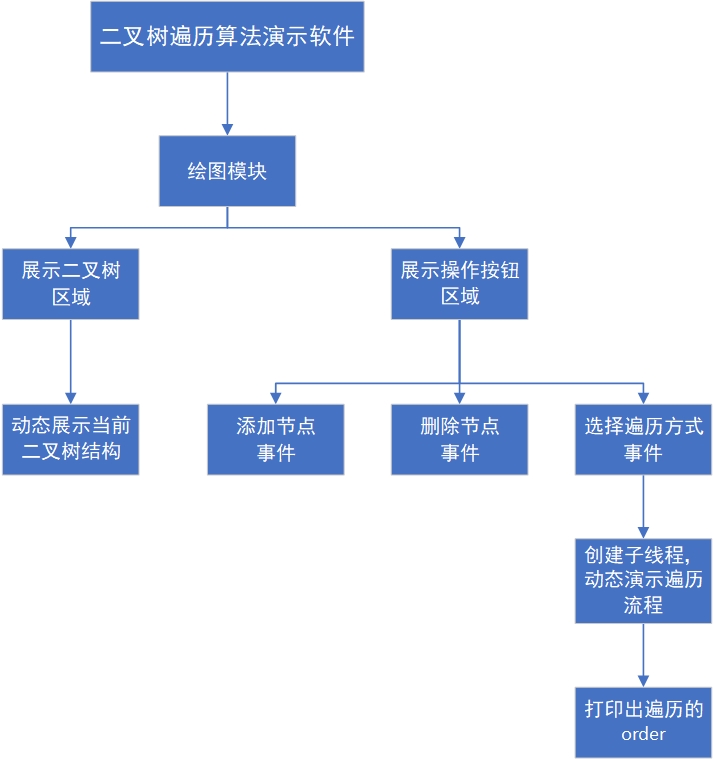
## 需求分析

* 用户可以通过选择节点进行删除，同样实时展示当前二叉树结构
* 选择遍历方式后，可以将遍历结果打印在图形化界面中

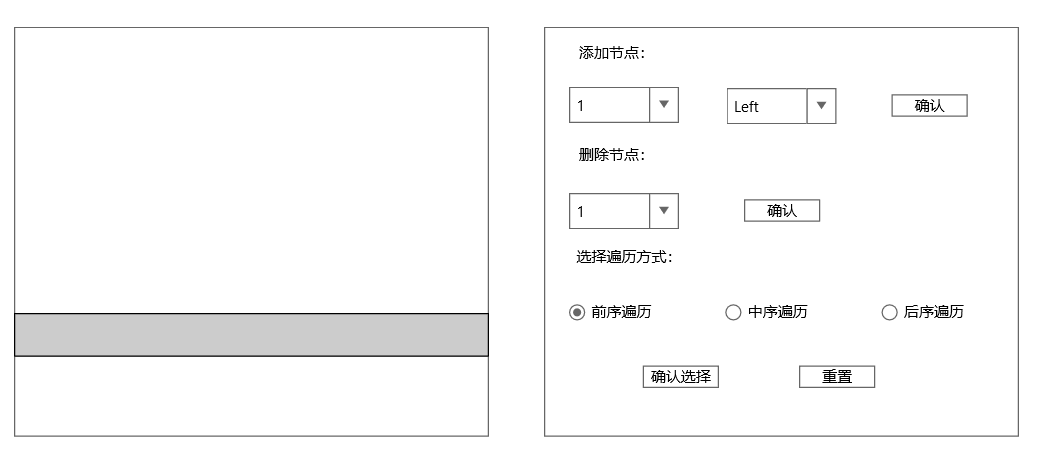
## 设计方案

（1）详细设计方案：模块功能结构设计，界面原型设计等。

模块结构设计

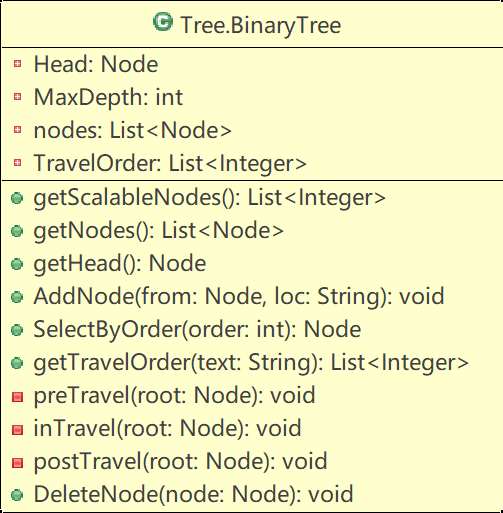


界面原型设计



（2）模块相关的设计类（Design Class）：给出每个类的名字、属性、方法。

注意：和基本系统or增量1相同的部分不在此展示



|  |  |
| --- | --- |
| 类名 | 解释说明 |
| BinaryTree | 二叉树class |

|  |  |
| --- | --- |
| 方法名 | 解释说明 |
| DeleteNode | 利用Stack实现删除节点及后续子树 |

（3）所需的数据结构

栈Stack：在实现删除节点及后续子树功能的时候，利用了一个Stack



## 结对编程过程记录

**（1）角色切换与任务分工**

表2-1 第2轮迭代过程结对编程角色与任务分工

| 日期 | 时间(HH:MM - HH:MM) | 驾驶员角色 | 领航员角色 | 本段时间的任务 |
| --- | --- | --- | --- | --- |
| 2020.10.23 | 9:00——10:00 | 梅智敏 | 邓昆昆 | 实现删除节点及后续子树 |
| 2020.10.23 | 18:00——19:00 | 邓昆昆 | 梅智敏 | 实现打印遍历order |

**（2）工作日志**

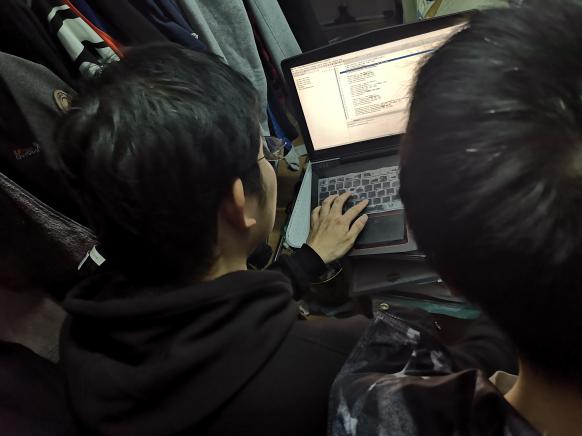
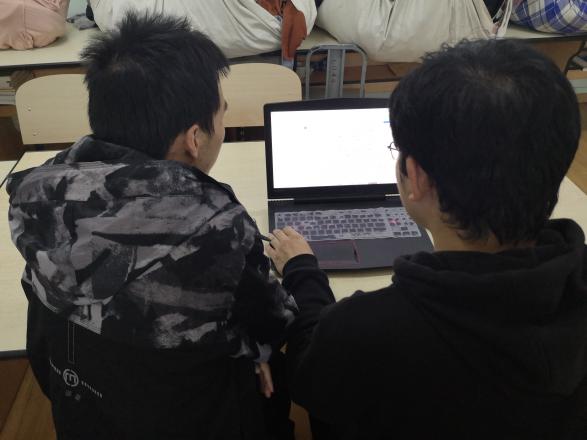
由领航员负责记录，记录结对编程期间的遇到的问题、两人如何通过交流合作解决每个问题的。

表2-2 第2轮迭代过程结对编程工作日志

| 日期/时间 | 问题描述 | 最终解决方法 | 交流过程 |
| --- | --- | --- | --- |
| 2020.10.23 | 打印遍历的order时数据超出Jlabel长度而没有换行导致部分数据被遮掩 | JLabel换成JTextArea | 和其他同学讨论 |
| 2020.10.23 | 删除节点时，某些叶节点无法删除 | 利用eclipse的调试功能，为DeleteNode方法增加判断是否为null的逻辑 | 和其他同学讨论 |

**（3）结对编程工作现场照片**

请其他同学帮助拍摄结对编程现场照片至少2张。

第2次迭代结对编程现场照片1 第2次迭代结对编程现场照片2

## 单元测试

对本轮迭代所开发的每个类进行单元测试。

表2-3 第2轮迭代过程的单元测试记录

| 日期/时间 | 类名：方法名 | 输入数据 | 输出数据 | 结论 |
| --- | --- | --- | --- | --- |
| 2020.10.23 | BinaryTree:DeleteNode | 节点序号1-5的二叉树，删除节点3 | 二叉树中节点3为null | true |
| 2020.10.23 | BinaryTree:DeleteNode | 节点序号1-5的二叉树，删除节点2 | 二叉树中节点2，4，5为null | true |

## 集成测试

对本轮迭代所开发功能模块进行测试。

表2-4 第2轮迭代过程的集成测试记录

| 日期/时间 | 功能名称 | 测试用例 | 测试结果描述 | 结论 |
| --- | --- | --- | --- | --- |
| 2020.10.23 | 二叉树某个节点（包括其子树）的删除 | 节点序号1-5的二叉树 | 删除正确 | true |
| 2020.10.23 | 二叉树遍历结果的显示 | 任意二叉树 | 正确显示 | true |

# 项目总结

## 增量模型应用总结

对增量模型应用过程进行总结，包括但不限于以下几方面的内容：

* 增量划分是根据工作量划分还是根据用户需求来划分？

两个都要考虑。

从用户需求而言：得先通过添加节点来建立二叉树；再在此基础上实现3种遍历算法并动态演示；最后应该对系统进行优化（如:新增删除节点功能以及打印遍历的顺序order）。

从工作量划分而言：在一开始就该确定好图形化界面的大致布局，并写好Node和二叉树ADT以便于后续使用；再在此基础上实现遍历算法和删除节点功能。

* 第1个增量你们团队是如何确定的？

在基本系统中，我们已经确定了图形化界面的基本布局，并完成添加节点功能、建立二叉树功能、实时显示二叉树结构功能。根据用户需求，我们在基本系统的基础上还需要完成3种遍历算法并以动画形式演示。根据工作量划分，有了基本系统，我们实现遍历算法并不困难。

故我们确定的增量1为：在二叉树上实现3种遍历算法；并能以动画方式演示遍历流程

* 你认为增量模型的好处在哪里？有什么不好之处吗？它的适用场合是什么？

增量模型的好处：可以将一个大的任务划分为多个小任务并逐次实现，降低开发难度。

不好之处：

1. 由于各个构件是逐渐并入已有的软件体系结构中的，所以加入构件必须不破坏已构造好的系统部分，这需要软件具备开放式的体系结构。在开发过程中，需求的变化是不可避免的。
2. 增量模型的灵活性可以使其适应这种变化的能力大大优于瀑布模型和快速原型模型，但也很容易退化为边做边改模型，从而是软件过程的控制失去整体性。

适用场合：

用户急需部分核心功能，后续延伸功能暂时不急需。总的来说，适合于快速迭代的小项目。

## 结对编程总结

对结对编程过程进行总结，包括但不限于以下几方面的内容：

* 与2人分别编程相比，是否体验到编程效率的提高、编程质量的提高？

确实，双方轮流担任领航员和驾驶员，可以有效提高编程的效率和编程质量。

* 你认为，你们2人本次合作是否真的提高了效率和质量？有什么成功的体验和需要反省的不足或教训？

我认为我们确实提高了效率和质量，

成功的体验主要是：

1. 双方轮流当领航员和驾驶员十分有效
2. 结对编程的另一大好处是遇到问题可以随时交流

需要反省的不足主要是：增量划分还需要更加清晰

* 你认为结对编程的优势在哪里、有什么不适应之处？它的适用场合是什么？

优势：领航员可以帮助驾驶员确定方向，并及时发现程序中的错误然后查阅资料，反馈给驾驶员。

不适应：以往的编程实践都是个人独自编程，现在有领航员和驾驶员的身份互换，有些不太适应。

适合场合：双方的配合比较默契，适合快速迭代的小项目。