

**重 庆 理 工 大 学**

**课程设计**

**课程 数据结构课程设计**

院系名称 计算机科学与工程学院

班 级 116030801班

学 号 11603080110

学生姓名 黄飞

指导教师 王森老师

评阅教师 王森老师

时 间 2018.01.07

[一、 哈夫曼 2](#_Toc24323)

[1.1题目要求 2](#_Toc32700)

[1.2设计思路 3](#_Toc11474)

[1.3测试 3](#_Toc5306)

[二、24点 5](#_Toc30906)

[2.2设计思路 5](#_Toc32262)

[2.3测试 6](#_Toc3306)

[三、十六个硬币问题 11](#_Toc11218)

[3.1题目要求 11](#_Toc13455)

[3.2 设计思路 12](#_Toc20845)

[3.3测试 12](#_Toc24384)

[四、 线性表、树、图的操作和演示 15](#_Toc11124)

[4.1、设计题目 15](#_Toc15618)

[4.2、设计思路 17](#_Toc24644)

[4.3测试 18](#_Toc32618)

[五、 农夫过河 25](#_Toc5414)

[5.1、题目要求 25](#_Toc4243)

[5.2、设计思路 27](#_Toc26076)

[5.3、测试 27](#_Toc22278)

[六、 迷宫问题 29](#_Toc6504)

[6.1题目要求 29](#_Toc5588)

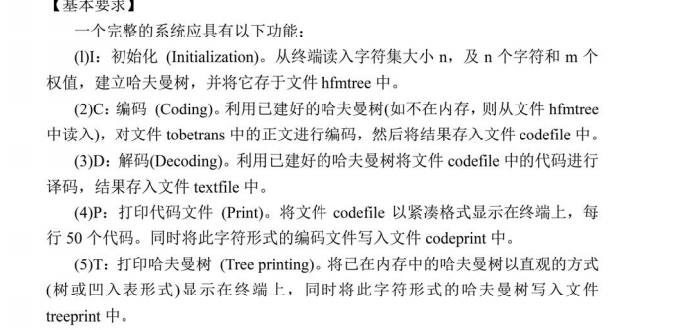
[6.2设计思路 31](#_Toc1610)

[6.3测试 31](#_Toc12900)

1. 哈夫曼

## 1.1题目要求

利用哈夫曼编码进行信息通讯新到利用率，缩短信息传输时间，降低传输成本。但是，这要求在发送端通过一个编码系统对待传数据预先编码；在接收端将传来的数据进行译码。对于双工信道（即可以双向传输信息的信道），每端都需要一个完整的编码系统。试对于任意的一段文字，也可能是保存在本地文件中或者网络上的，写一个哈夫曼码的编译码系统



## 1.2设计思路

本程序主要用到了三个算法

1. 哈夫曼编码
2. 串的匹配
3. 二叉树的遍历

## 1.3测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 3 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | 打印哈夫曼树 | | | | |
| 前提条件 | 编码之后 | | | | |
| 测试步骤 | 读取输入字符，建立哈夫曼树，展示编码 | | | | |
| 输入数据 | Hello world | | | | |
| 预期输出 |  | | | | |
| 实际输出 |  | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2017.12.20 | |
| 测试人 | 黄飞 | 测试日期 | | | 2017.12.20 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2017.12.20 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

# 二、24点

2.1题目说明

一副牌中抽去大小王剩下52张（如果初练也可只用1-10者40张牌），任意抽取4张牌（称牌组），用加、减、乘、除（可加括号）把牌面上的数算成24。每张牌必须使用一次且只能用一次，如抽出的牌是3、8、8、9，那么算式（9-8）\*3\*8或3\*8+（9-8）或（9-8%8）\*3等

## 2.2设计思路

1. 对栈，集合，数组，递归等知识熟悉，并且能熟练掌握
2. 首先在程序的设计部分由分为几个步骤：
3. 第一步：查阅有关归并排序算法的资料。
4. 第二步：设计这个项目的整体架构和算法。
5. 第三步：选择一门程序设计语言进行算法的描述。
6. 其次，进行程序的调试。

## 2.3测试

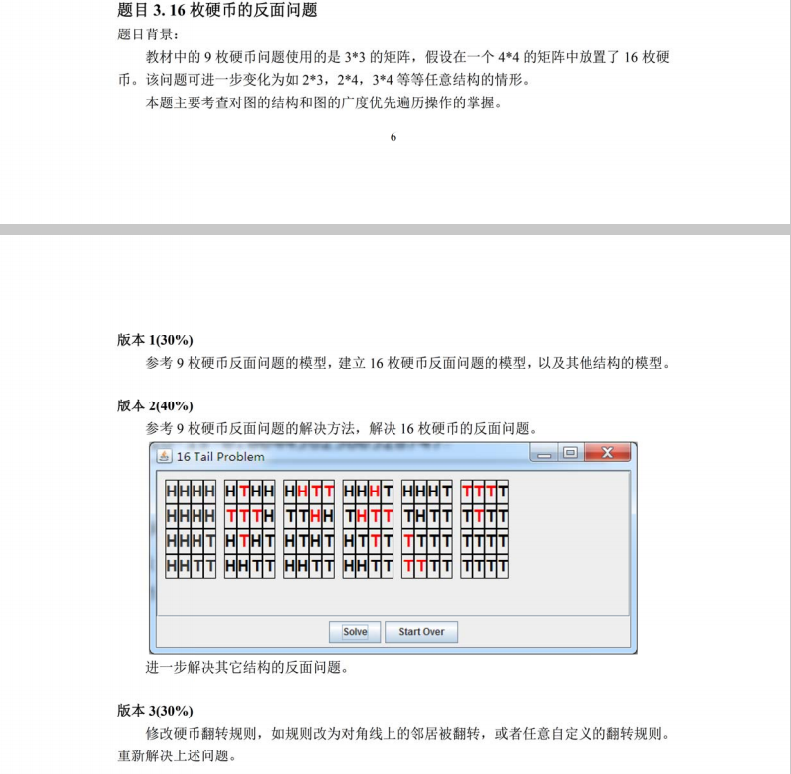
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 2 | | 版本号 | | 2 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | 24点第二个版本 | | | | |
| 前提条件 | 输入相应的text，点击相应分的按钮 | | | | |
| 测试步骤 | 10\*（8-6）+4 | | | | |
| 输入数据 | Value:23 index:1 | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2018.1.8 | |
| 测试人 | 黄飞 | 测试日期 | | | 2018.1.8 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2018.1.8 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |
| 测试用例编号 | 1 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | 24-point第三个版本 | | | | |
| 前提条件 | 输入相应的text，点击相应分的按钮 | | | | |
| 测试步骤 | 10\*（8-6）+4 | | | | |
| 输入数据 | Value:23 index:1 | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2018.1.8 | |
| 测试人 | 黄飞 | 测试日期 | | | 2018.1.8 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2018.1.8 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

Version3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 1 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | 线性表 | | | | |
| 前提条件 | 输入相应的text，点击相应分的按钮 | | | | |
| 测试步骤 | 10\*（8-6）+4 | | | | |
| 输入数据 | Value:23 index:1 | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2018.1.8 | |
| 测试人 | 黄飞 | 测试日期 | | | 2018.1.8 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2018.1.8 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

# 三、十六个硬币问题

## 3.1题目要求

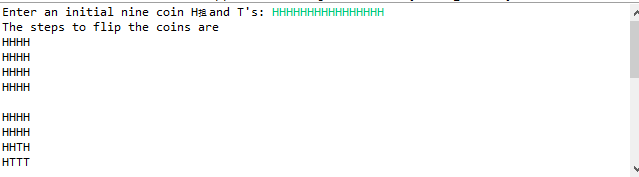


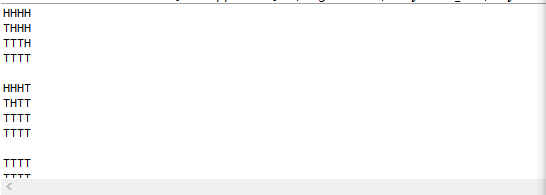
## 3.2 设计思路

本题主要采用图的广度优先算法，仿造九个硬币问题。储存结构为图，逻辑结构为非线性结构。主要的算法是图的最短路径等。

## 3.3测试

1. Version1：





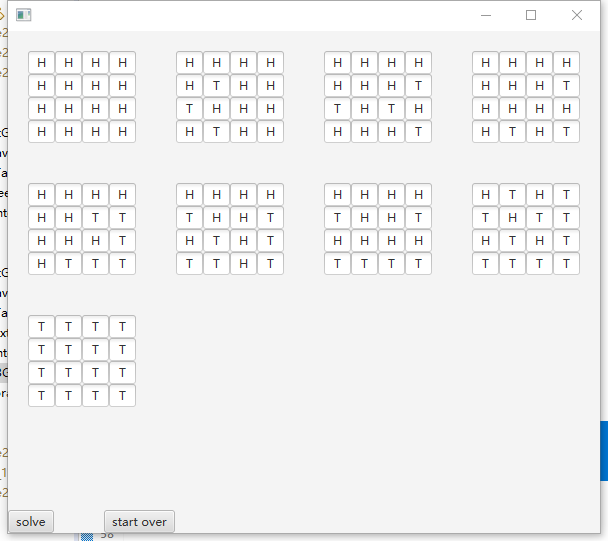
1. version2：

输入十六个硬币



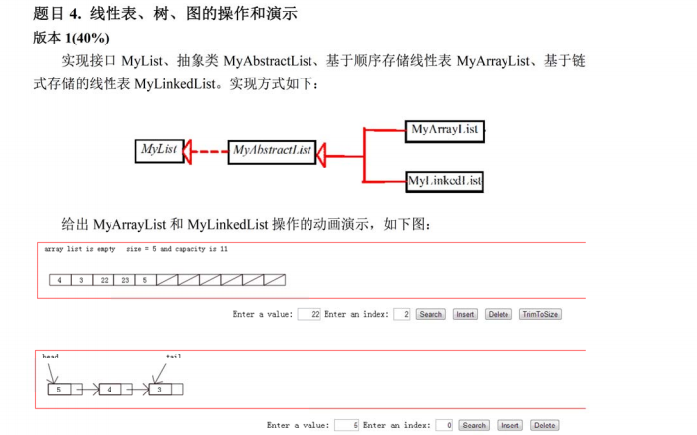
1. version3

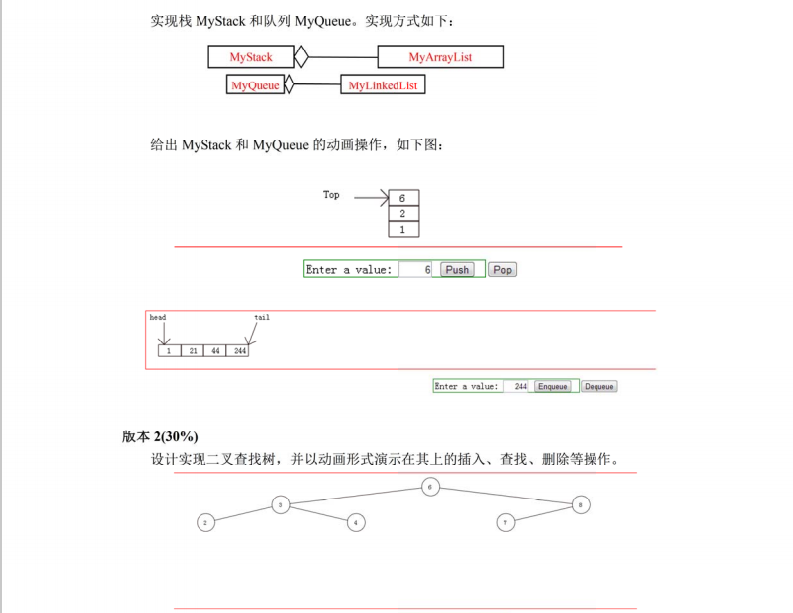


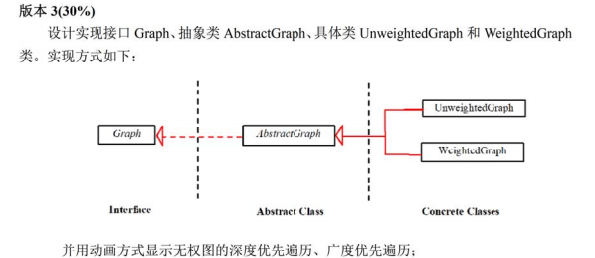


# 线性表、树、图的操作和演示

## 4.1、设计题目







## 4.2、设计思路

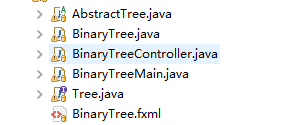
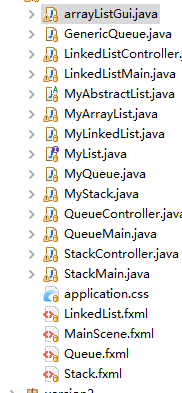
逻辑结构：题目中用到了线性表，队列等线性结构，和树，图等非线性结构

储存结构：1.数据元素自身值的表示（数据域）

1. 该节点与其它节点关系的表示（链表）

两种基本的存储方法：   
（1）顺序存储方法（顺序存储结构）   
（2）链接存储方法（链式存储结构）   
同一种逻辑结构可采用不同的存储方法（以上两种之一或组合），这主要考虑的是运算方便及算法的时空要求。

分别为version1，2的类图



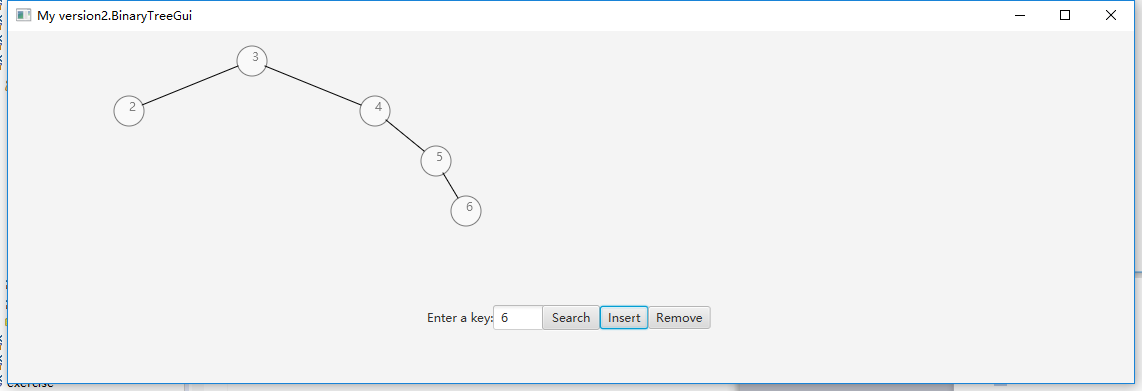
## 4.3测试

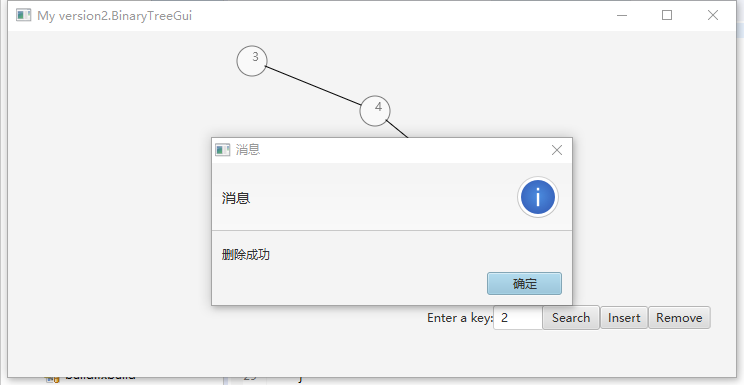
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 1 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | 线性表 | | | | |
| 前提条件 | 输入相应的text，点击相应分的按钮 | | | | |
| 测试步骤 | 输入相应的text，点击相应分的按钮 | | | | |
| 输入数据 | Value:23 index:1 | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2017.12.20 | |
| 测试人 | 黄飞 | 测试日期 | | | 2017.12.20 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2017.12.20 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

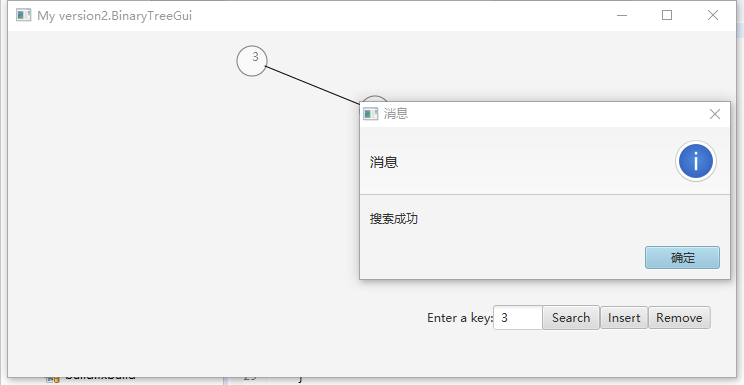
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 2 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | linkedList | | | | |
| 前提条件 | 输入相应的text，点击相应分的按钮 | | | | |
| 测试步骤 | 输入相应的text，点击相应分的按钮 | | | | |
| 输入数据 |  | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2017.12.20 | |
| 测试人 | 黄飞 | 测试日期 | | | 2017.12.20 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2017.12.20 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 3 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | StackGui | | | | |
| 前提条件 | 无 | | | | |
| 测试步骤 | 输入相应的text，点击相应分的按钮 | | | | |
| 输入数据 |  | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2017.12.20 | |
| 测试人 | 黄飞 | 测试日期 | | | 2017.12.20 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2017.12.20 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 4 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | Queue | | | | |
| 前提条件 | 无 | | | | |
| 测试步骤 | 输入相应的text，点击相应分的按钮 | | | | |
| 输入数据 |  | | | | |
| 预期输出 |  | | | | |
| 实际输出 |  | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2017.12.20 | |
| 测试人 | 黄飞 | 测试日期 | | | 2017.12.20 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2017.12.20 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

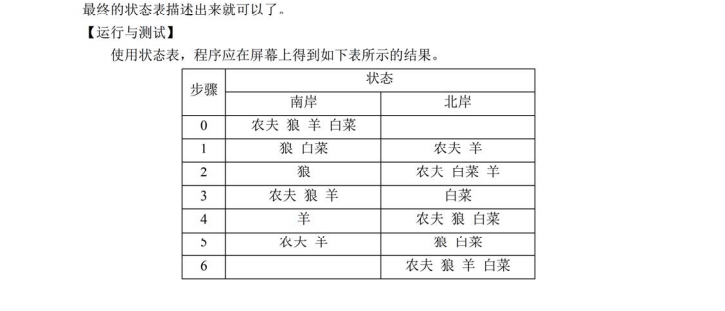


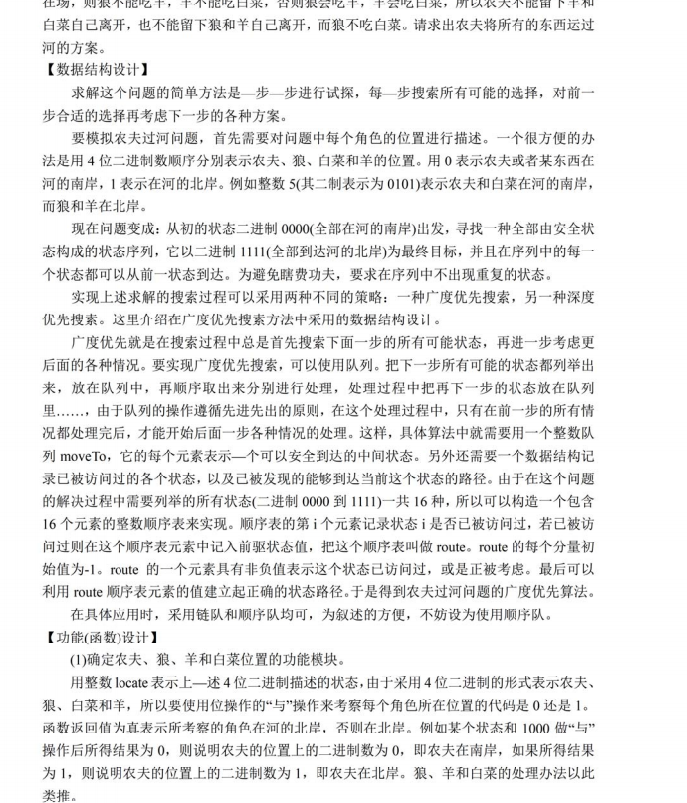




# 农夫过河

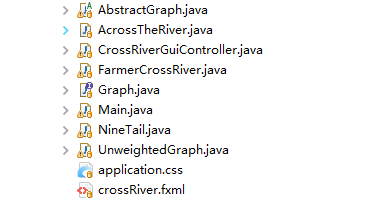
## 5.1、题目要求





## 5.2、设计思路

利用九个硬币的原理，设计农夫过河问题

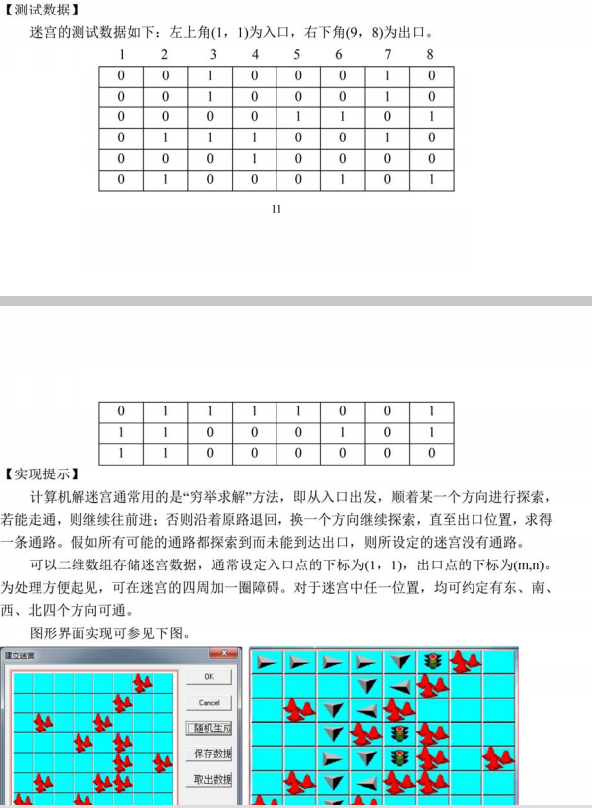
类图：

## 5.3、测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 1 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | 农夫过河 | | | | |
| 前提条件 | 输入状态值 | | | | |
| 测试步骤 | 输入状态值，点击下一步 | | | | |
| 输入数据 | The process | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2017.12.20 | |
| 测试人 | 黄飞 | 测试日期 | | | 2017.12.20 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2017.12.20 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

# 迷宫问题

## 6.1题目要求



## 6.2设计思路

（1）用‘穷举解法’方法解决迷宫问题 ，储存结构：用栈储存

（2） 建立一个二维数组表示迷宫的路径（0表示通道，1表示墙壁）；

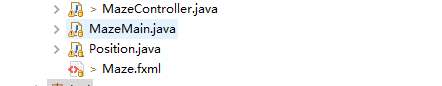
（3）创建一个栈，用来存储“当前路径”，即“在搜索过程中某一时刻所在图中某个方块位置”。

（4）创建一个Int类型的二维数组intmaze[n1][n2],用来存放0和1 ；

（5）创建一个结构体用来储存数组信息(数组的横坐标X，数组的纵坐标Y，方向C)

（5）创造一个栈包括(top表示栈顶元素)

类图：



## 6.3测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 1 | | 版本号 | | 1 |
| 测试环境 | Windows10 JDK1.8 IDEA2017.2 | | | | |
| 用例名称 | 迷宫问题 | | | | |
| 前提条件 | 无 | | | | |
| 测试步骤 | 无 | | | | |
| 输入数据 | The process | | | | |
| 预期输出 | IMG_256 | | | | |
| 实际输出 | IMG_256 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 黄飞 | 设计日期 | | 2018.1.8 | |
| 测试人 | 黄飞 | 测试日期 | | | 2018.1.8 |
| 再测试人 | 黄飞 | 再测试日期 | | | 2018.1.8 |
| 问题修改摘要 | 无 | | | | |
| 修改人 |  | | 修改日期 | |  |

课程设计总结

在同学以及老师的帮助下，通过自己的努力，终于完成了这次课程设计。我经过这段时间的编程，对其中的艰辛，我是深有体会。从刚开始的看题目，理解题目到后来的调试程序以及改进程序这个过程中，我遇到了各种各样的困难和挫折。甚至有时候问题解决不了也想过放弃，后来通过同学和老师们的点拨，我逐渐克服了困难，处理掉了bug，在这次课程设计中我对于我自己的短板劣势有了更加清晰的认识，我接受信息的能力和处理信息的能力较弱。但我认为了解到自己的弱点并不是什么坏事，因为只有接受它，才可能去改变。我还体会到了在进行编写一个程序之前，要有明确的目标和整体的设计思想，另外某些具体的细节内容也是相当的重要。

这些宝贵的编程思想和从中摸索到的经验都是在编程的过程中获得的宝贵财富对我以后的编程会有很大的帮助的，我要好好利用。