7

**2019年春季学期  
计算机学院《软件构造》课程**

**Lab 2实验报告**

|  |  |
| --- | --- |
| 姓名 | 杜钰欣 |
| 学号 | 1170300320 |
| 班号 | 1703003 |
| 电子邮件 | 1485422651@qq.com |
| 手机号码 | 18745718808 |

**目录**

[1 实验目标概述 1](#_Toc1988214)

[2 实验环境配置 1](#_Toc1988215)

[3 实验过程 1](#_Toc1988216)

[3.1 Poetic Walks 1](#_Toc1988217)

[3.1.1 Get the code and prepare Git repository 1](#_Toc1988218)

[3.1.2 Problem 1: Test Graph <String> 1](#_Toc1988219)

[3.1.3 Problem 2: Implement Graph <String> 1](#_Toc1988220)

[3.1.3.1 Implement ConcreteEdgesGraph 2](#_Toc1988221)

[3.1.3.2 Implement ConcreteVerticesGraph 2](#_Toc1988222)

[3.1.4 Problem 3: Implement generic Graph<L> 2](#_Toc1988223)

[3.1.4.1 Make the implementations generic 2](#_Toc1988224)

[3.1.4.2 Implement Graph.empty() 2](#_Toc1988225)

[3.1.5 Problem 4: Poetic walks 2](#_Toc1988226)

[3.1.5.1 Test GraphPoet 2](#_Toc1988227)

[3.1.5.2 Implement GraphPoet 2](#_Toc1988228)

[3.1.5.3 Graph poetry slam 2](#_Toc1988229)

[3.1.6 Before you’re done 2](#_Toc1988230)

[3.2 Re-implement the Social Network in Lab1 2](#_Toc1988231)

[3.2.1 FriendshipGraph类 2](#_Toc1988232)

[3.2.2 Person类 3](#_Toc1988233)

[3.2.3 客户端main() 3](#_Toc1988234)

[3.2.4 测试用例 3](#_Toc1988235)

[3.2.5 提交至Git仓库 3](#_Toc1988236)

[3.3 Playing Chess 3](#_Toc1988237)

[3.3.1 ADT设计/实现方案 3](#_Toc1988238)

[3.3.2 主程序ChessGame设计/实现方案 3](#_Toc1988239)

[3.3.3 ADT和主程序的测试方案 3](#_Toc1988240)

[3.4 Multi-Startup Set (MIT) 4](#_Toc1988241)

[4 实验进度记录 4](#_Toc1988242)

[5 实验过程中遇到的困难与解决途径 4](#_Toc1988243)

[6 实验过程中收获的经验、教训、感想 4](#_Toc1988244)

[6.1 实验过程中收获的经验和教训 4](#_Toc1988245)

[6.2 针对以下方面的感受 4](#_Toc1988246)

# 实验目标概述

本次实验训练抽象数据类型（ADT）的设计、规约、测试，并使用面向对象编程（OOP）技术实现ADT。具体来说：

* 针对给定的应用问题，从问题描述中识别所需的ADT；
* 设计ADT规约（pre-condition、post-condition）并评估规约的质量；
* 根据ADT的规约设计测试用例；
* ADT的泛型化；
* 根据规约设计ADT的多种不同的实现；针对每种实现，设计其表示（representation）、表示不变性（rep invariant）、抽象过程（abstraction function）
* 使用OOP实现ADT，并判定表示不变性是否违反、各实现是否存在表示泄露（rep exposure）；
* 测试ADT的实现并评估测试的覆盖度；
* 使用ADT及其实现，为应用问题开发程序；
* 在测试代码中，能够写出testing strategy并据此设计测试用例。

# 实验环境配置

按照实验报告所给网址下载eclemma压缩包即可。

在这里给出你的GitHub Lab2仓库的URL地址（Lab2-学号）:

https://github.com/ComputerScienceHIT/Lab2-1170300320.git

# 实验过程

请仔细对照实验手册，针对三个问题中的每一项任务，在下面各节中记录你的实验过程、阐述你的设计思路和问题求解思路，可辅之以示意图或关键源代码加以说明（但千万不要把你的源代码全部粘贴过来！）。

## Poetic Walks

此实验已经给定了ADT的结构只需要依次填充好内容就可以，给定的graph即是接口而边和点作为两个实现接口的类各自都可以完成添加，删除以及得到起始边，终点边和权重的功能，最后通过一个具体的实验来对这个graph进行使用，利用功能完成类似扩句的功能，总体难度不是很难但要对ADT有基本的理解才能顺利的完成这个实验

### Get the code and prepare Git repository

git clone +所给资源地址

即可在本地复制到远程仓库的文件然后就可以使用了

### Problem 1: Test Graph <String>

由于接口graph部分一开始是由泛型L实现的所以在之后的使用环节中要将泛型L具体化为String类型，这样才能使用

### Problem 2: Implement Graph <String>

#### Implement ConcreteEdgesGraph

该类中有两个变量分别为

这两个变量是不可以被改变的immutable型的变量。

实现方法：

Add:向图中新添加点，直接向vertices中调用set本身具有的添加方法即可，在添加时要考虑到添加的点已经存在的情况在这种情况下只需return false然后直接退出，不需重复添加。

Set：设置一条边的所有信息，包括起始点终结点权重，在这个set方法中我调用在edges类中的三个set方法，分别设置起点，终点，以及权重，在edges中详细说明

Remove：移除一个点，当明确移除一个点时，要遍历这个图中的所有边凡是起点或是终点有所要删除的点时就要直接把这条边删除，最后从点的set中删除目标点，这时关于这个点的全部信息才算全部被清除。

Vertices：为了确保代码的保密性和安全性，需要新定义一个新变量只不过它的内容实际上和vertices都一样，再返回这个新变量，即可保护好不想被用户破坏的信息。

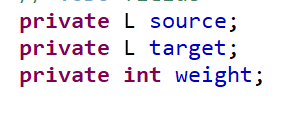
Edges，与上一个点的集合相同也需新定义一个变量用来返回给用户。

Sources:该方法其实就是给定一个终点的点来寻找所有他的起点集合以及这些边对应的权重，构成一个一一对应的map，返回这个map即可，在实现此方法的时候我还是用到了遍历，在所有的边中找到所有终点为目标点的点，记录下这条边的起点和权重，把他们存到map中，遍历完成后这个map即为所求。

Targets：targets需要返回一个给定起点的所有终点以及对应边的权重，与上述实现方法是相同的。

To string：将本图中的边信息转换成字符串形式，有注释信息可以被他人读懂，我将这种形式规范为起点+终点+权重，还是遍历全图输出即可。

Class Edge



Get source ; get target ; get weight:三个简单的获取图的基本信息的方法，与图中获取信息的方法类似

Set source ; set target ; set weight:三个设置图信息的基本方法，在这里edges类就是针对一个图中的一条边，所以在设置起点终点和权重的时候都只能有一个值

Set edge :一次性将一条边的所有信息都设置好

To string:将此条边的信息打印出来显示给用户，只需要将起点终点和权重这几项信息罗列出来即可

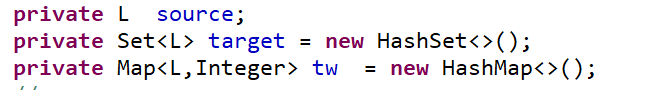
#### Implement ConcreteVerticesGraph

在此类中依旧是实现graph接口

但在这个类中的变量只有

在此类中实现的方法思路基本与上述concrete Edges相同但在class vertex有区别所以在此详细讲述一下vertex的实现思路

Class vertex

变量如下：

实现方法：

Get Vertex:由于在本方法中只有可使用的点，没有成套的边信息所以获取点直接返回一个新变量赋值当前点给用户即可

Set Vertex:设置一个新的点，在点的集合中先添加一项

Set Source:，将当前点设置为起点

Set Target:：将当前点设置为中终点

Set Weight: 在设置权重信息的时候需要提供终点和权重两项信息才具备设置条件，在这时在tw即终点和权重组成的map中添加这项新添加或是修改的信息即可。

Get Source: 默认当前点的vertex类是由它本身为起点定义的，所以得到起点即为返回当前点

get Target: 得到终点集合即新定义一个变量复制target的内容然后返回

get Weight:得到权重需要提供终点作为参数，然后再tw的map中找到与此终点对应的权重，返回这个值即可

remove:，清除当前点的信息，将该点从集合中删除，将他的target，tw清空

to string: 输出当前点，当前点所有的终点以及对应的权重信息。

### Problem 3: Implement generic Graph<L>

#### Make the implementations generic

将所有的string类型替换为L，这样在使用的时候就可以定义任何类型的顶点

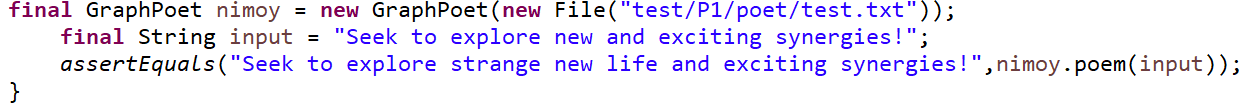
#### Implement Graph.empty()

初始化一个空图，新建一个空顶点集合一个空边集合

### Problem 4: Poetic walks

#### Test GraphPoet

我使用了mit官网的例子来作为一个辅助例子测试graphpoet



#### Implement GraphPoet

这个任务就是利用bufferreader stringbuffer,以及getpath等得到文本中实际的string以便之后利用

#### Graph poetry slam

实现方法：这个类使对graph类的实际调用，将一句给定的话，通过丰富的语料库将他拓展。

首先将给定的需要改进的话分割成字符数组依次存进字符数组

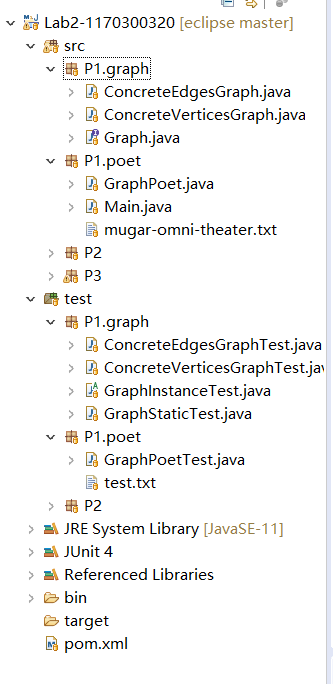
将语料库中的句子也按顺序存进字符数组，确定待改进语句的前后两个单词，在语料库中查找是否有中间可以插进单词的情况，若有，则将中间可插入单词加入f以flag为权值的map中，出现一次flag加一。最终输出时将map排序将权重最大的单词作为最终的选择插入到语句中去。

### Before you’re done

请按照[http://web.mit.edu/6.031/www/sp17/psets/ps2/#before\_youre\_done](http://web.mit.edu/6.031/www/sp17/psets/ps2/" \l "before_youre_done)的说明，检查你的程序。

如何通过Git提交当前版本到GitHub上你的Lab2仓库。

在这里给出你的项目的目录结构树状示意图。



## Re-implement the Social Network in Lab1

这个问题其实就是利用第一个问题的另外一个实际应用，这个应用恰好是实验一所做的人际交往网络，通过将每一个人抽象成点，每一个有向关系抽象成有向边，然后形成一个图，再利用边的权重来计算某人到某人的最短路径

### FriendshipGraph类

与实验一类似，添加新的关系图中的人物时就添加一个点，要添加一种关系时就添加一条边，同时将这条边的权重设为1.

### Person类

设置一个具有名字的类即可，同时在添加新的人物时检测一下是否和之前的姓名重复，如果重复就提示错误并且不添加重复信息。

### 客户端main()

我依旧使用了实验一所给的例子来验证我的程序是否运行正确，这是在测试前的一个辅助检验

### 测试用例

测试添加点：每添加一个新的任务，返回值为1；若添加的人物姓名与之前重复则返回0

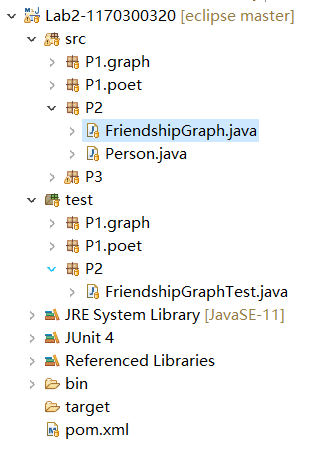
测试添加边：每添加一条新的边返回值为1；

测试距离：给出一个完整的人物关系图，与手算出的距离相比较

### 提交至Git仓库

如何通过Git提交当前版本到GitHub上你的Lab3仓库。

在这里给出你的项目的目录结构树状示意图。



## Playing Chess

### ADT设计/实现方案

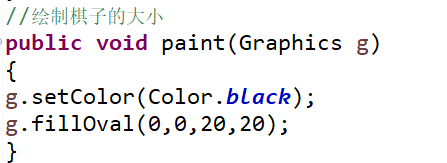
接口：Piece，Board , Player

Piece的实现：Chess\_Piece（由于在国际象棋中，棋子在开局时就已经摆放好，所以关于国际象棋的棋子我统一写在棋盘所在类中）

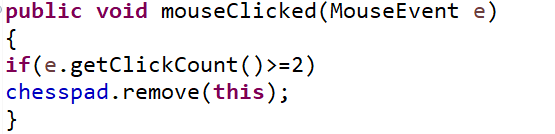
AF : includes white and black

RI : fixed size , color , mutable location

white\_piece , black\_piece：在这里规定好棋子的大小



对棋子进行监听

当左键单击时放子，在双击或更多次点击时将此棋子吃掉，棋子消失

Board：



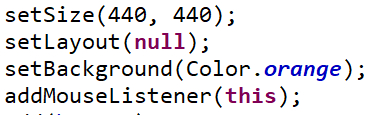
Board的实现：chess\_board

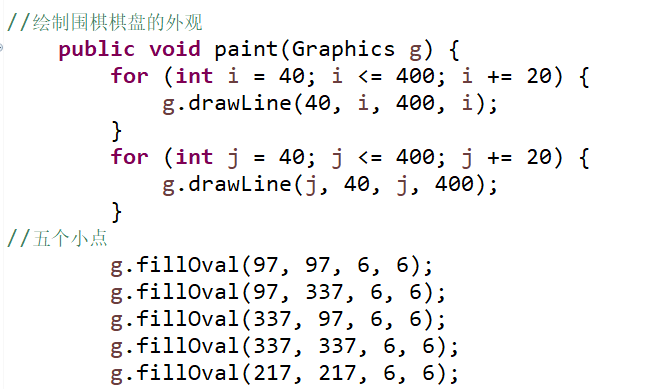


添加两个操作按钮

添加两个提示玩家下棋的文本框

为棋盘设置大小，背景颜色并设置监听



绘制围棋中的条线格

此类中围棋棋子的添加和删除可由用户随意改变，但棋子本身的属性如大小形状颜色等均为不可改变量

国际象棋  

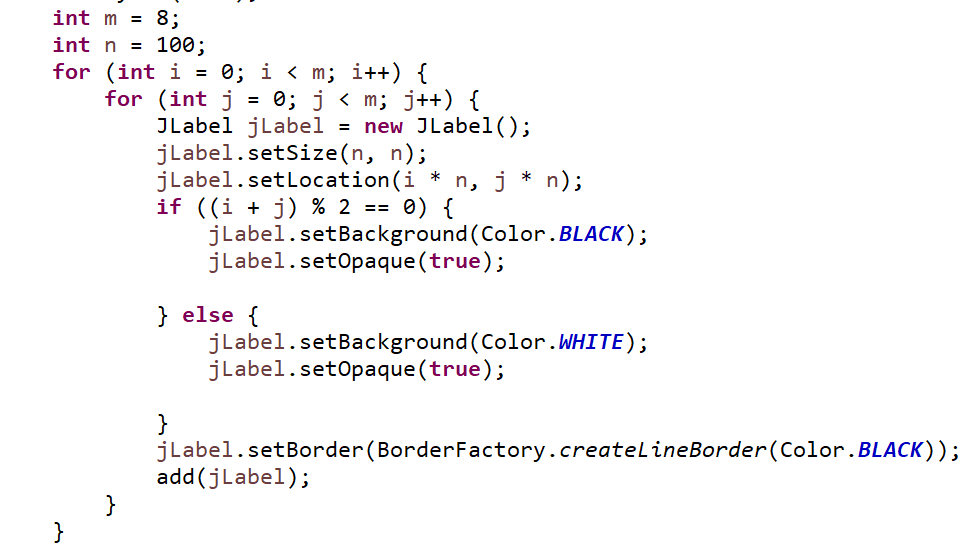

在建立国际象棋的棋盘时由于我将棋子和棋盘格作为两个图层有上下的区分所以无法继续继承Board只能作为JFrame的子类进行编写

首先初始化两个字符串作为历史纪录，然后依旧是新建两个功能按钮





绘制棋盘格



接着是添加初始化的国际象棋棋子位置

对于棋子的操作：首先将图片转换为JLabel格式易于添加到整个框架之中

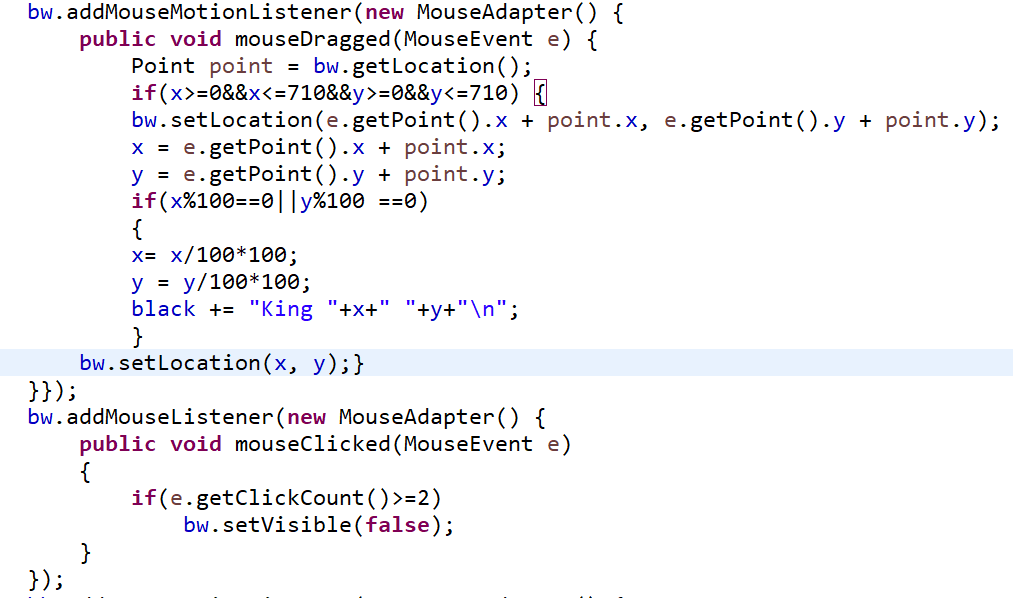


设置大小和初始位置

将棋子放置在图层的最上方



对于每个棋子限定他活动的范围，在操作是选择拖拽来进行移动，记录鼠标拖拽时的横纵坐标来作为历史纪录最后进行显示，当双击某个棋子时意味着这个棋子已经被吃掉，应该消失



国际象棋中，棋子时存在或消失时随时会被用户操纵的，但关于棋子的属性如大小颜色上面显示的图片等是用户不能改变的。

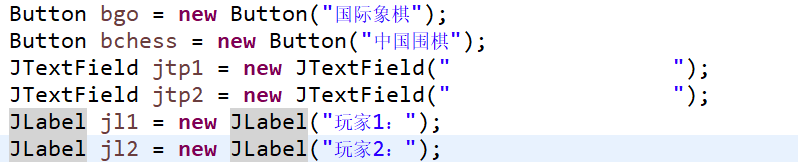
Player的实现：Chess\_Player;Go\_Player

这里的Player只是显示在刚开始进入游戏时输入两个玩家姓名，对于中国围棋来说，Player类中包括了历史纪录的保存，国际象棋则没有

玩家名字可以由用户随意更改，历史纪录随着游戏的进行一直在更新

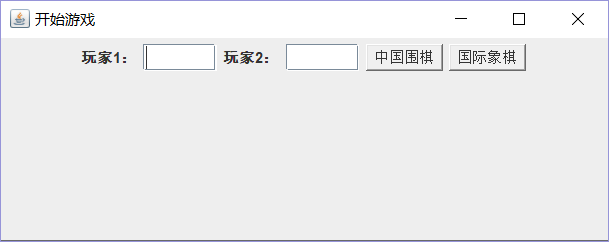
### 主程序MyChessAndGoGame设计/实现方案

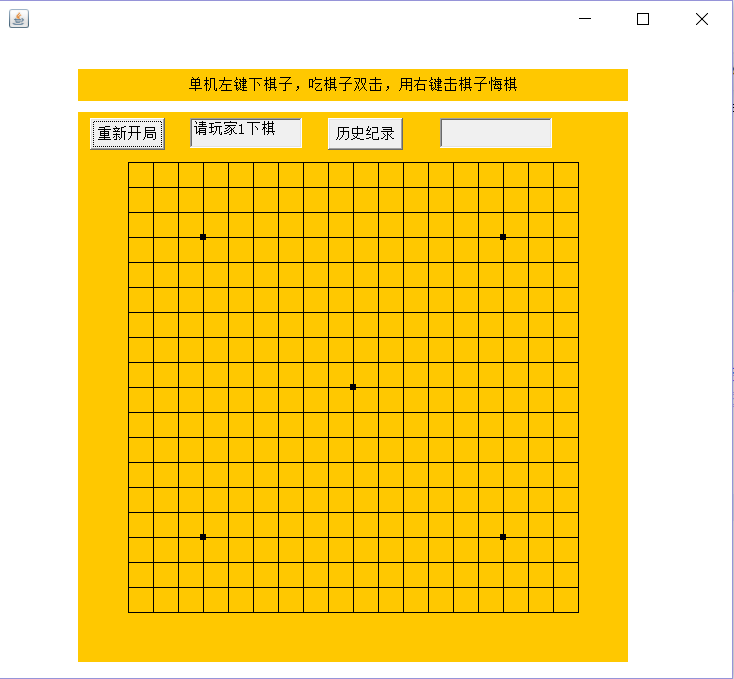
进入游戏时出现窗口，窗口标题为开始游戏，出现两个按钮供玩家选择，在这时还可以输入两个玩家的姓名

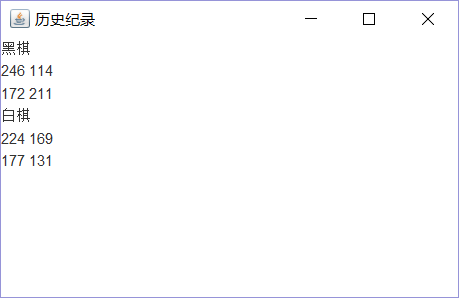


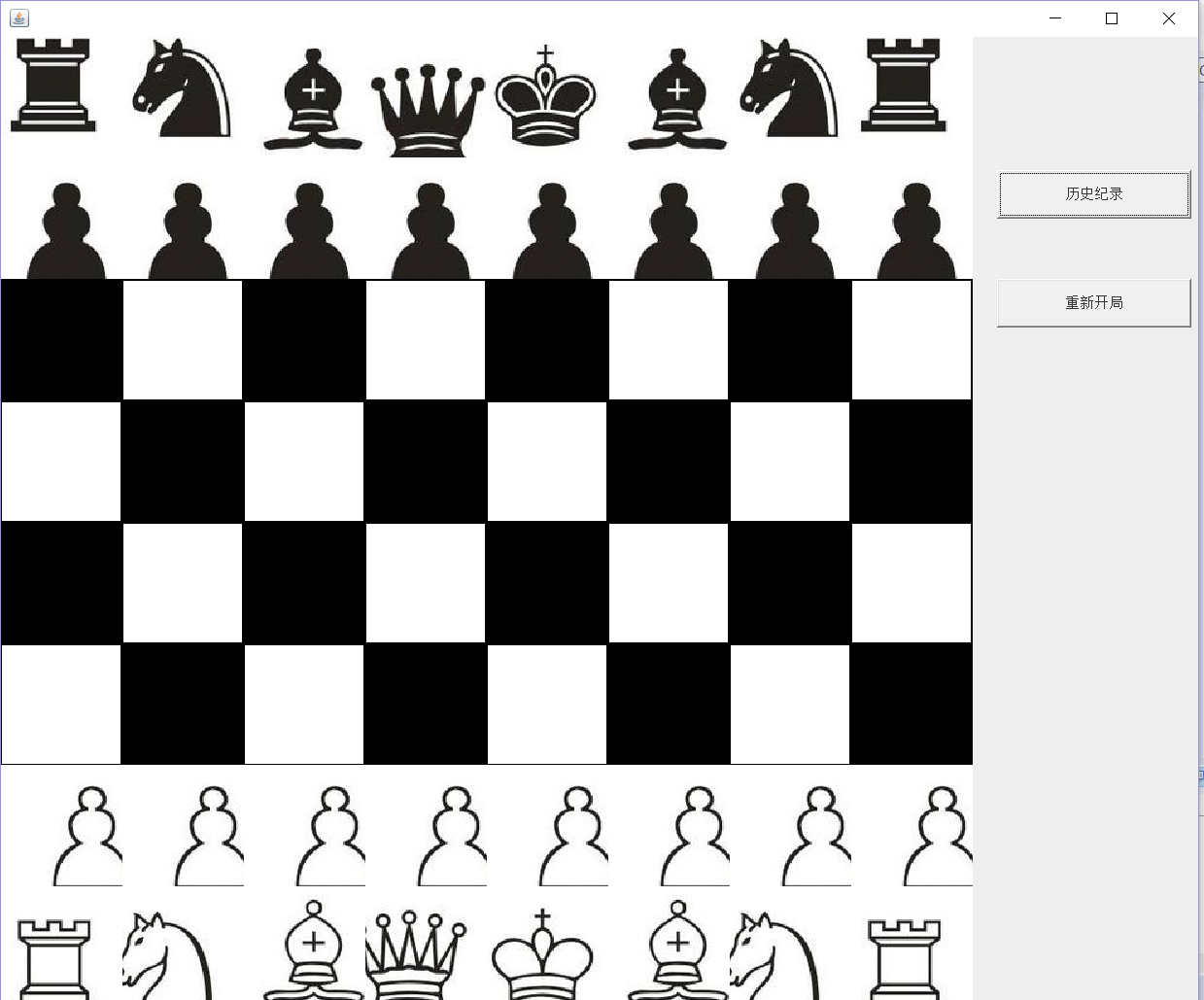
将这些插入到主框架中即可

游戏操作的流程，图片展示











### ADT和主程序的测试方案

围棋测试：

在某处放置棋子后返回1，测试时判断该处是否正常返回1，进行完毕

在某处点击取消棋子后返回2，测试时判断该处是否正常返回2，进行完毕

象棋测试：

在成功拖拽棋子到某个位置时，测试判断是否正常返回1，进行完毕

在吃子时判断吃掉棋子操作进行完后是否正常返回2，进行完毕

## Multi-Startup Set (MIT)

请自行设计目录结构。

注意：该任务为选做，不评判，不计分。

# 实验进度记录

请使用表格方式记录你的进度情况，以超过半小时的连续编程时间为一行。

每次结束编程时，请向该表格中增加一行。不要事后胡乱填写。

不要嫌烦，该表格可帮助你汇总你在每个任务上付出的时间和精力，发现自己不擅长的任务，后续有意识的弥补。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 计划任务 | 实际完成情况 |
| 3.20 | 8：00-11：00 | P1edges和vertices部分 | 基本完成 |
| 3.25 | 17：00-21：22 | P1完成 | 除了注释全部完成 |
| 3.26 | 19：00-21：00 | P2完成 | 完成 |
| 3.27 | 14：00-23：00 | 建立P3基本ADT，添加内容 | 完成 |
| 3.28 | 21：00-23：00 | 绘制中国围棋棋盘 | 发现之前的内容书写有错误，修改，未完成计划 |
| 3.29 | 13：00-23：00 | 中国围棋相关操作完成 | 完成 |
| 3.30 | 14：00-23：00 | 将国际象棋棋子添加到棋盘上 | 只切割好图片，但不显示 |
| 3.31 | 17：00-23：00 | 将棋子添加到棋盘上，赋予监听 | 完成 |
| 4.1 | 20:00-22:00 | 修改P3中对于拖拽棋子设置边界 | 完成 |
| 4.2 | 20：00-23：00 | 最后修改P3中的吃子操作 | 完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的难点 | 解决途径 |
| 对ADT理解不明确，书写P1看不懂实验要求 | 上课自习听，向同学求教 |
| P3国际象棋部分图片添加不到棋盘上去，切割好的图片不显示 | 上网搜索发现原因时图层被掩盖 |
| 拖拽时显示的历史纪录过于详细，导致底层记录根本拉不到，看不了 | 将记录规定为整百显示，这是只显示鼠标在格中停留的记录，不显示其他无用记录 |
| 想要将国际象棋的棋盘与中国围棋一样继承同一个父类，但一直报错 | 发现是因为国际象棋中两个图层叠加显示最上层的操作在父类中并没有，所以不能继承同一个父类，最后国际象棋继承JFrame |

# 实验过程中收获的经验、教训、感想

## 实验过程中收获的经验和教训

在书写代码时要先定好整个的框架然后再分一部分一部分的详细填充，这样的代码在书写时会更有规划性，心中有数，不会每天想要完成的任务没有完成时产生一种对截止日期的恐慌感，在书写时随时随地的补充注释，在写完某一部分时及时补充测试，这样一边写一边测试能及时找到代码发生错误的地方

## 针对以下方面的感受

1. 面向ADT的编程和直接面向应用场景编程，你体会到二者有何差异？

ADT更加广泛使用，只书写一个抽象数据行就将一类问题统统解决

1. 用泛型和不使用泛型的编程，对你来说有何差异？

每次利用泛型时都要加一个<L>在书写时很容易忘记

1. 在给出ADT的规约后就开始编写测试用例，优势是什么？你是否能够适应这种测试方式？

先对框架进行测试，提高了检测的准确性，也防止后来出现错误的时候，不知道从哪下手，找bug艰难

1. P1设计的ADT在多个应用场景下使用，这种复用带来什么好处？

使代码更加简洁，也更有整体性

1. P3要求你从0开始设计ADT并使用它们完成一个具体应用，你是否已适应从具体应用场景到ADT的“抽象映射”？相比起P1给出了ADT非常明确的rep和方法、ADT之间的逻辑关系，P3要求你自主设计这些内容，你的感受如何？

不是很适应自己设计ADT所以在设计时非常的艰难，多次返回P1观察所给的ADT的格式以及思路，个人对于ADT的掌握还不是很成熟，所以才出现这种现象，希望日后情况会越来越好

1. 为ADT撰写specification, invariants, RI, AF，时刻注意ADT是否有rep exposure，这些工作的意义是什么？你是否愿意在以后编程中坚持这么做？

不得不说写注释是非常麻烦的一件事情，而且尽量用英文书写更加大了在写代码时的烦躁感，但这些要求都时对整个程序负责人的一种表现，毕竟代码最终时服务于所有人的，我会尽量在以后自主书写代码时养成这样良好的习惯

1. 关于本实验的工作量、难度、deadline。

本实验的任务量在完成P1P2时觉得还算轻松，但进行到P3时开始紧张觉得截止日期在即，本次实验整个都围绕着ADT展开难度时逐渐攀升的，较上次实验而言已经高了一个等级，截止日期在最终完成时觉得还算合理

1. 《软件构造》课程进展到目前，你对该课程有何体会和建议？

希望老师在上课授课时能夹带一些实验的内容，某些难点能在课堂上就简单的讲述一下，这样会给实验的完成带来很大的作用，因为网上的很多搜索到的解决办法是没有作用的，尝试过之后失败，再摸索也很难解决。