

**2021年春季学期  
计算学部《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 童彦澎 |
| 学号 | 1190201701 |
| 班号 | 1903011 |
| 电子邮件 | [1308091501@qq.com](mailto:1308091501@qq.com) |
| 手机号码 | 18959076997 |

**目录**

[1 实验目标概述 1](#_Toc71131534)

[2 实验环境配置 1](#_Toc71131535)

[3 实验过程 1](#_Toc71131536)

[3.1 Magic Squares 1](#_Toc71131537)

[3.1.1 isLegalMagicSquare() 1](#_Toc71131538)

[3.1.2 generateMagicSquare() 1](#_Toc71131539)

[3.2 Turtle Graphics 1](#_Toc71131540)

[3.2.1 Problem 1: Clone and import 2](#_Toc71131541)

[3.2.2 Problem 3: Turtle graphics and drawSquare 2](#_Toc71131542)

[3.2.3 Problem 5: Drawing polygons 2](#_Toc71131543)

[3.2.4 Problem 6: Calculating Bearings 2](#_Toc71131544)

[3.2.5 Problem 7: Convex Hulls 2](#_Toc71131545)

[3.2.6 Problem 8: Personal art 2](#_Toc71131546)

[3.2.7 Submitting 2](#_Toc71131547)

[3.3 Social Network 2](#_Toc71131548)

[3.3.1 设计/实现FriendshipGraph类 2](#_Toc71131549)

[3.3.2 设计/实现Person类 2](#_Toc71131550)

[3.3.3 设计/实现客户端代码main() 2](#_Toc71131551)

[3.3.4 设计/实现测试用例 3](#_Toc71131552)

[4 实验进度记录 3](#_Toc71131553)

[5 实验过程中遇到的困难与解决途径 3](#_Toc71131554)

[6 实验过程中收获的经验、教训、感想 3](#_Toc71131555)

[6.1 实验过程中收获的经验和教训 3](#_Toc71131556)

[6.2 针对以下方面的感受 3](#_Toc71131557)

# 实验目标概述

本次实验通过求解三个问题，训练基本 Java 编程技能，能够利用 Java OO 开发基本的功能模块，能够阅读理解已有代码框架并根据功能需求补全代码，能够为所开发的代码编写基本的测试程序并完成测试，初步保证所开发代码的正确性。另一方面，利用 Git 作为代码配置管理的工具，学会 Git 的基本使用方法。  
⚫ 基本的 Java OO 编程  
⚫ 基于 Eclipse/IDEA IDE 进行 Java 编程  
⚫ 基于 JUnit 的测试  
⚫ 基于 Git 的代码配置管理

# 实验环境配置

简要陈述你配置本次实验所需开发、测试、运行环境的过程，必要时可以给出屏幕截图。

特别是要记录配置过程中遇到的问题和困难，以及如何解决的。

本次实验需要通过git提交至GitHub远程仓库，首先先安装并配置了git，建立了git与GitHub账户的连接，无遇到大问题。在登陆GitHub的过程中，发现有时候请求超时，通过CSDN查询相关解决办法得知：GitHub服务器的ＩＰ会不时更换，较好的解决办法是实时查询服务器的ＩＰ并在修改本地host文件实现访问。本次实验的开发测试运行均通过eclipse完成，开发过程无遇到大问题，测试时发现未导入ｊｕｎｉｔ库，从项目中手动添加后解决。

在这里给出你的GitHub Lab1仓库的URL地址。

https://github.com/ComputerScienceHIT/HIT-Lab1-1190201701

# 实验过程

请仔细对照实验手册，针对四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（但无需把你的源代码全部粘贴过来！）。

为了条理清晰，可根据需要在各节增加三级标题。

## Magic Squares

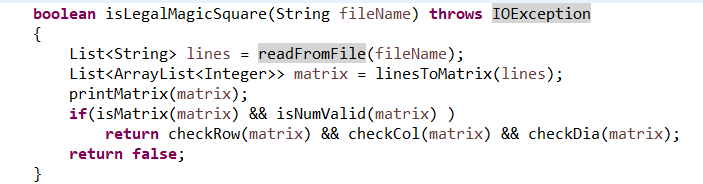
在这里简要概述你对该任务的理解。

幻方即为各行各列以及两条对角线上的数字之和均相等的n阶矩阵，本任务要求在能处理各类异常的情况下判断一个矩阵是否为n阶幻方，并测试及扩展给出的产生幻方的函数。

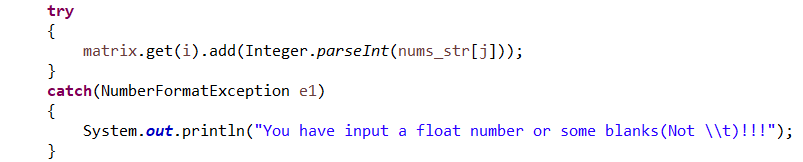
### isLegalMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

主函数由以下几个部分构成：

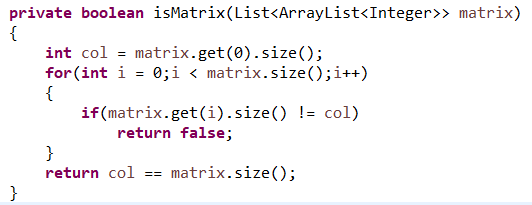


1. 将文件逐行读入，处理IO异常
2. LinesToMatrix函数将字符串转为二维数组，数组用容器实现，可变长。



这里处理了数字格式异常，遇到浮点数或是文件中误打了空格，未用tab键对齐，均会输出提示信息，并跳过该数据的读入。

1. 是n阶幻方首先需是n\*n的矩阵，通过isMatrix实现



每一行的列数应相等，且等于行数

1. 根据手册提示，还应检查每个数字是否为正整数，通过isNumValid实现
2. 若满足上述条件才有可能是幻方，再依据定义检查行列对角线
3. 否则直接输出false

### generateMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

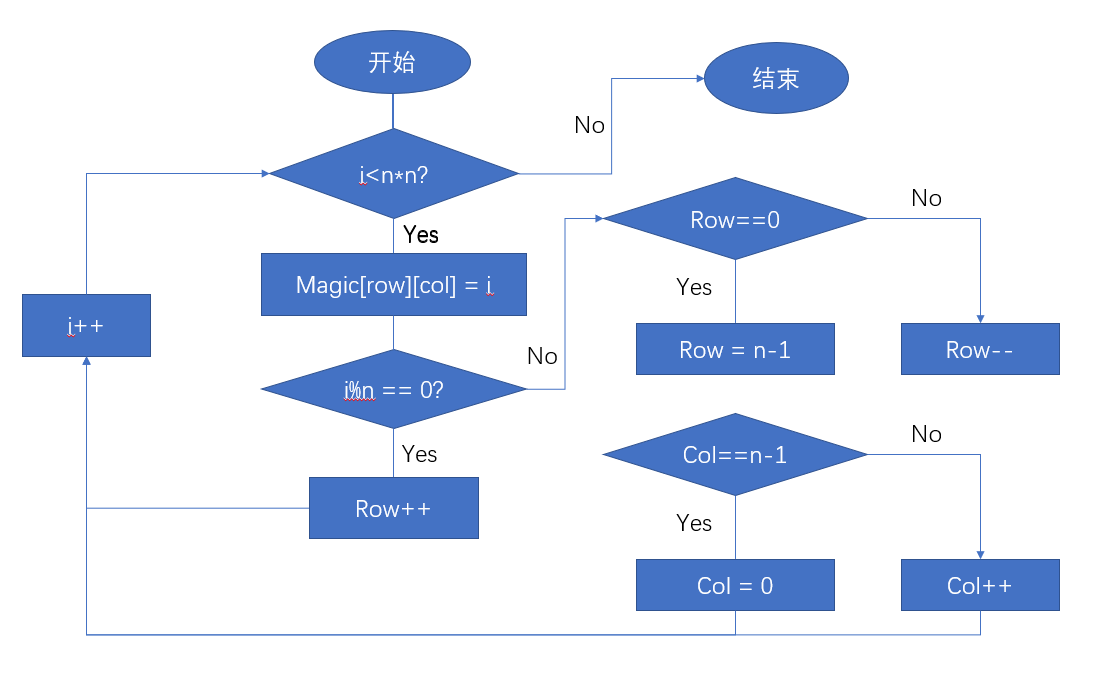
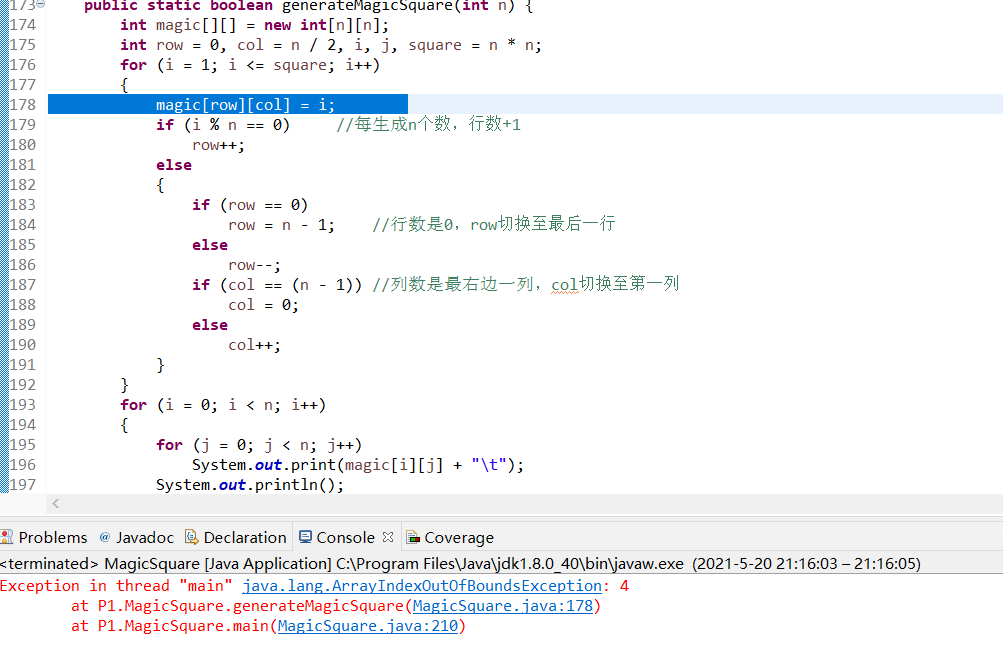


图 ‑1

流程图如图3.1.2-1所示。

一次将1-ｎ\*ｎ这ｎ\*ｎ个数按照流程图策略填入矩阵中构造幻方。

异常１：

根据trace跟踪到178行发现：若ｎ是偶数，以4为例跟踪，当ｉ＝9时，ｒｏｗ为4造成数组越界，进而抛出异常。

异常2：

输入ｎ为负数，在创建负数大小的二维数组时就会抛出数组大小负数异常。

针对上述异常利用try－catch “实现优雅的退出”并返回ｆａｌｓｅ。

如图2，3所示：

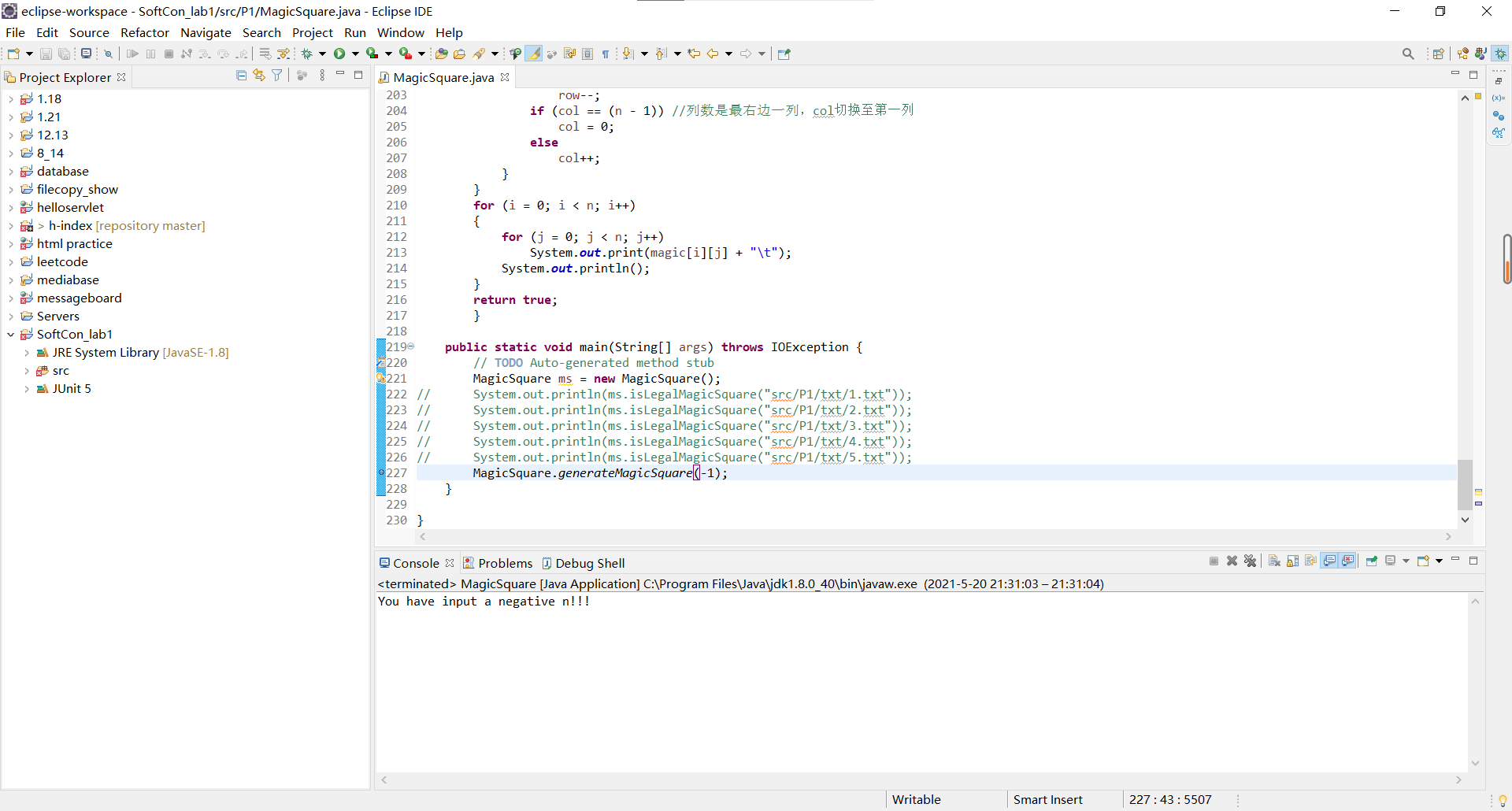


图 ‑2

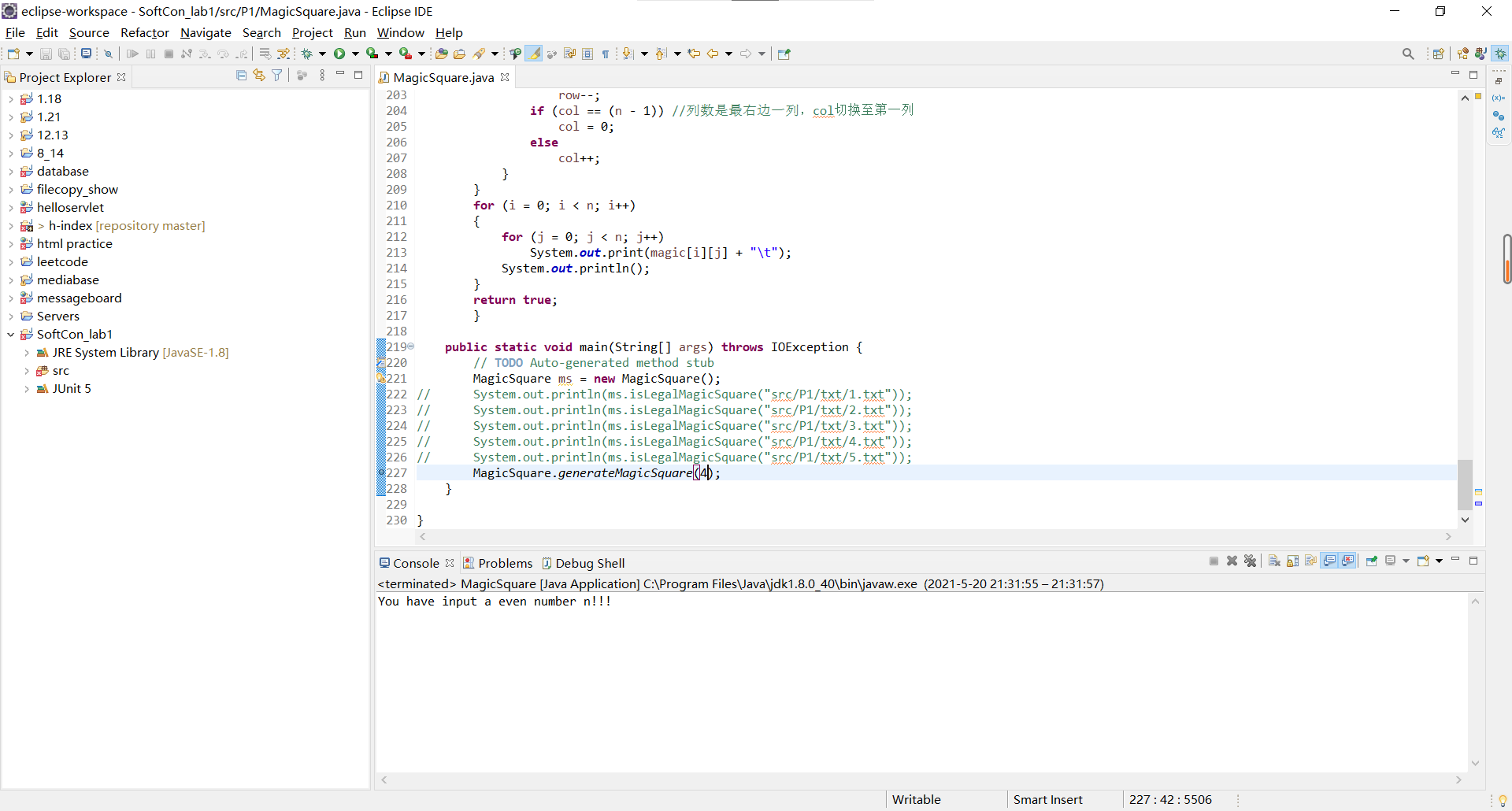
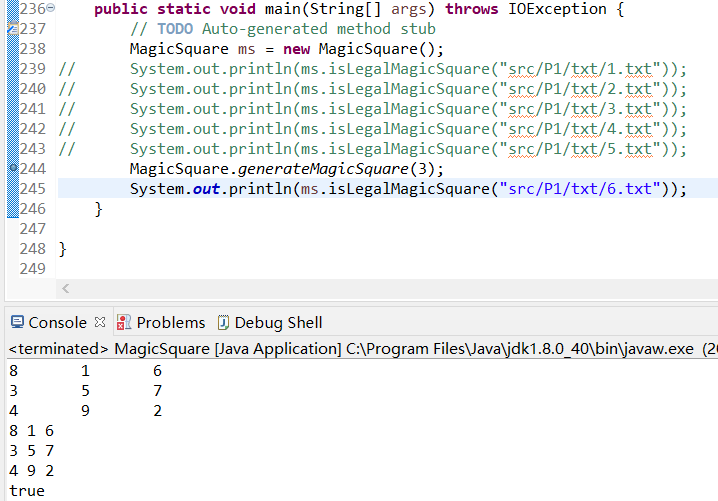


图 ‑3

扩展generateMagicSquare实现写入文件功能，并调用isLegal函数判断：



将matrix写入 “6.txt”，调用islegal进行测试：



## Turtle Graphics

在这里简要概述你对该任务的理解。

该任务不断完善turtle对象，实现turn与forward基本方法，并构造各种复杂方法；从计算两个点之间的偏移开始，最终到计算平面上点集的凸包，体现了模块化及OOP。

### Problem 1: Clone and import

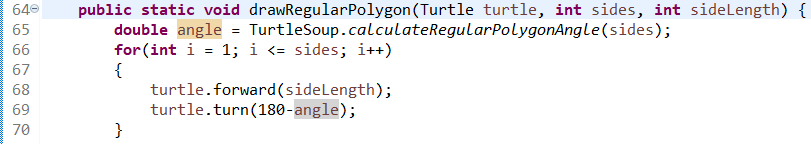
如何从GitHub获取该任务的代码、在本地创建git仓库、使用git管理本地开发。

通过git clone命令从指定仓库中将P2克隆至本地开发。

### Problem 3: Turtle graphics and drawSquare

四次forward与turn即绘制正方形，for循环实现即可。

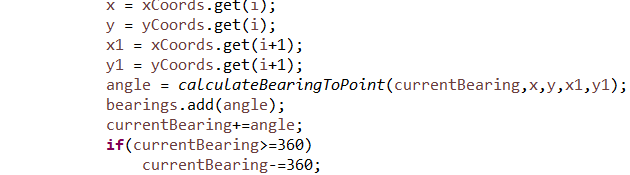
### Problem 5: Drawing polygons



调用calculateRegularPolygonAngle计算出正多边形内角，用for循环实现绘制正多边形即可，turn 的角度为180-内角。

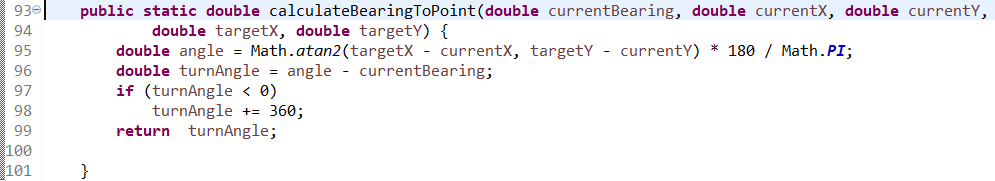
Calculate函数的实现为简单的(n-2)\*180/sides，需要强转一次double

### Problem 6: Calculating Bearings



不断调用BearingToPoint函数，计算出每两个点之间的偏移，修改currentBearing，若与y轴正方向的偏移大于360度，减去360

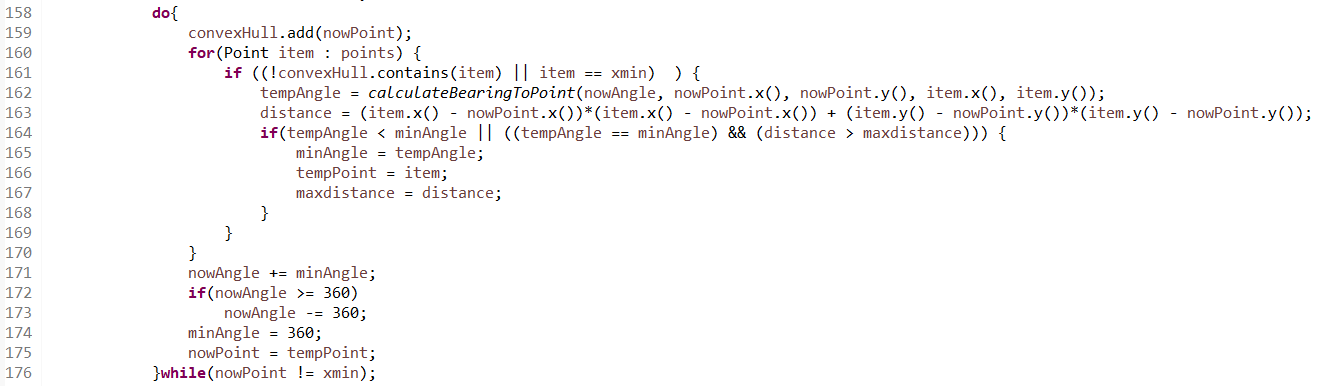
CalculateBearingToPoint函数实现的关键代码：



通过atan2函数算出极坐标下的偏转角，减去现在与y轴的偏转角即可，角度小于0加360即可（只允许顺时针转动）

### Problem 7: Convex Hulls

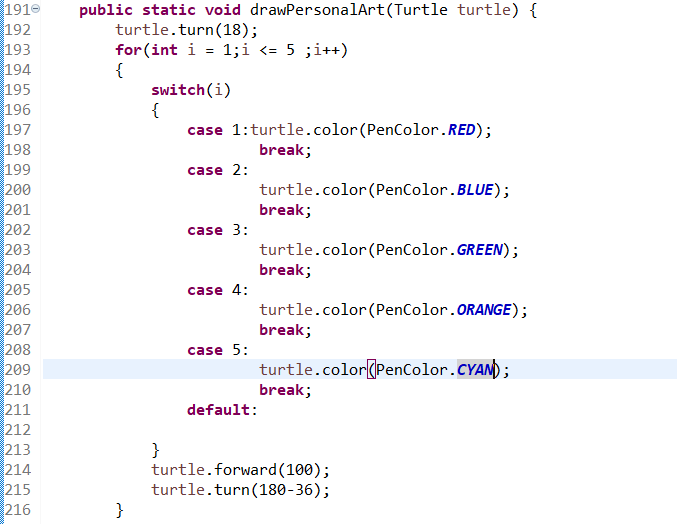
关键代码：



思路为:首先找到x最小的点作为起始点，该点一定在凸包上，接着调用calculateBearingToPoint计算与其余每一个点的偏转角，偏转角最小的即在凸包上，若偏转角相同，根据凸包定义，选取最远距离的点加入凸包，循环直到凸包内的点又回到xmin为止。

### Problem 8: Personal art

采用for循环仅利用turn与forward不断变换颜色绘制一个五角星。



### Submitting

如何通过Git提交当前版本到GitHub上你的Lab1仓库。

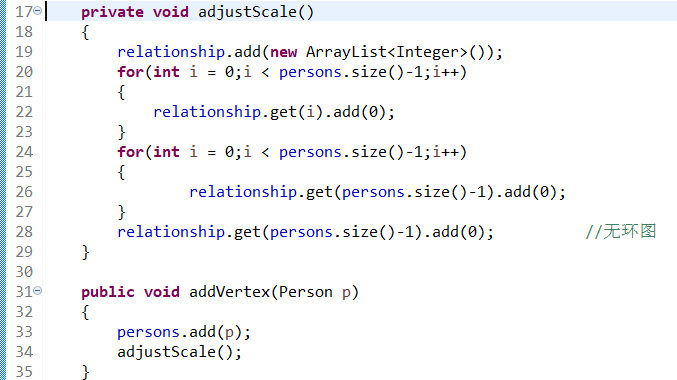
## Social Network

在这里简要概述你对该任务的理解。

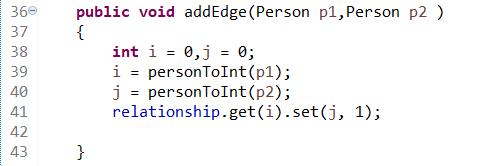
通过设计Person与FriendshipGraph两个类，模拟社交关系网络。

### 设计/实现FriendshipGraph类

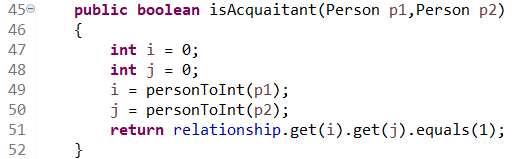
FriendshipGraph类维护一个persons数组，存放person对象，将person映射到非负整数。还应维护一个relationship矩阵，表示社交关系。



AddVertex再添加一个person后，相应的初始化relationship数组。社交关系网络设为无环图，故自己与自己不认识。



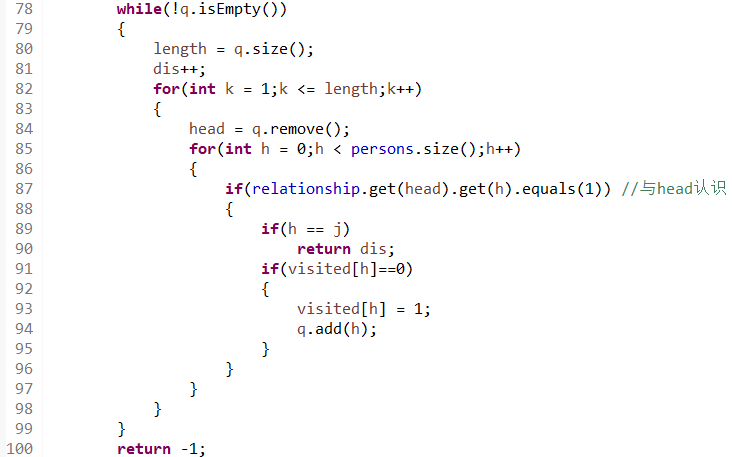
Addedge方法找到相应的两个人，relationship矩阵设为1即可，由于类可能要扩展到有向图，这里添加的关系是单向的，仅为p1认识p2。



设计了一个判断是否认识的方法，找到relationship矩阵中的关系返回即可。

GetDistance方法的实现：

按照提示用BFS实现，维护一个队列存放节点，从起始节点开始做广度优先搜索，每一轮搜索维护一个dis距离变量，每一轮加一，代表距离逐渐增加，直到搜索到终止（目标）节点为止。若搜索失败则队列为空，代表二者之间没有路径，返回-1.关键代码如下图所示：



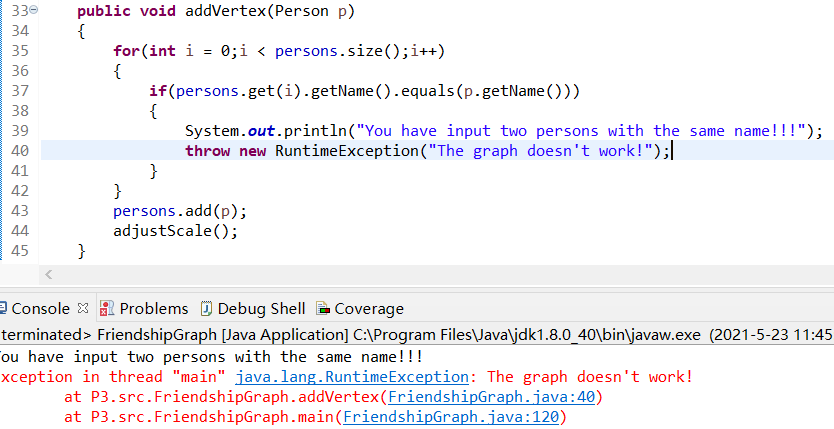
### 设计/实现Person类

在本实验中，Person仅根据人名区分，故只要由基本的String name以及相应的getter、setter即可。

### 设计/实现客户端代码main()

给出你的设计和实现思路/过程/结果。

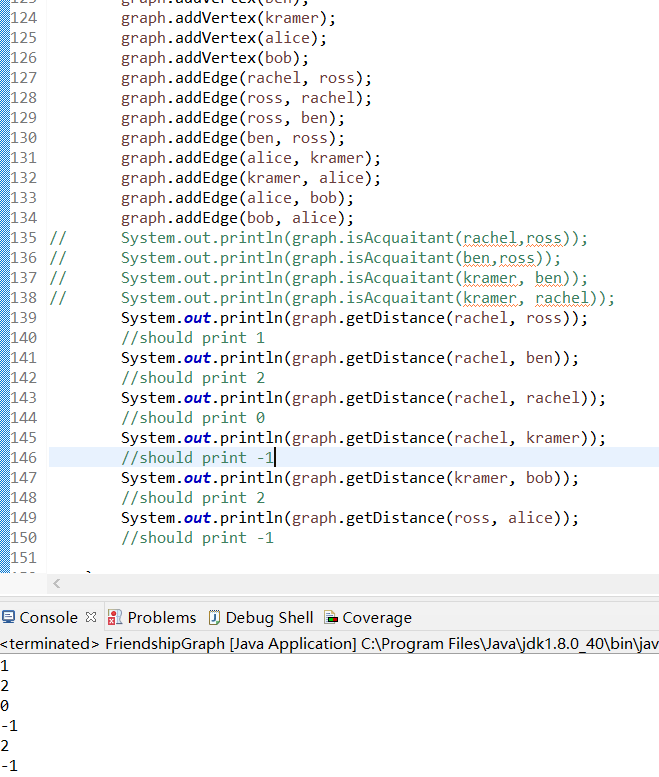
若将第十行注释，即Rachel不认识Ross，社交关系是单向的，14-17行应输出：-1，-1，0，-1，与程序结果一致。若输入两次Rachel，程序输出结果为：0，1，0，-1，在将人名映射至顶点的过程中出现了问题，导致图结构不正确。

处理该异常：

在addVertex之前检查是否有重名，若有抛出异常终止执行。

再添加个别测试用例即测试在两个连通分量中的两个顶点距离是否正确：

故添加alice与bob两个person，与Kramer在同一个连通分量上，测试正确。



### 设计/实现测试用例

给出你的设计和实现思路/过程/结果。

FriendshipGraphTest类的实现：

addVertex类的测试：

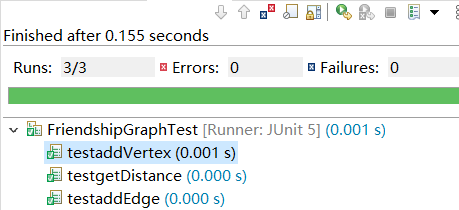
添加了Alice，Bob两个person，测试两个person是否被放入persons数组中。

addEdge类的测试：

添加Alice，Bob，Charlie三个person，Alice与Bob相互认识，为一个连通分量，Charlie自为一个连通分量，测试Alice与Bob是否有变，Alice与Charlie是否有边。

注：为了不破坏OOP的封装性，在获取graph的persons及relationship属性时采用getter，权限设为private

Junit测试结果均通过：



# 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 | 任务 | 实际完成情况 |
| 2021-05-11 | 18:30-20：00 | 编写问题1的isLegalMagicSquare函数并进行测试 | 按计划完成 |
| 2021-05-13 | 19：00-21：00 | 完成Ｐ2相关内容 | 按计划完成 |
| 2021-05-18 | 18：30-20：00 | 完成Ｐ3相关内容 | 按计划完成 |
| 2021-05-20 | 20：00-22：00 | 对Ｐ1进行异常处理及测试 | 按计划完成 |
| 2021-05-23 | 9：00-12：00 | 设计getDistance函数及补全相应报告 | 按计划完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的困难 | 解决途径 |
|  |  |
|  |  |
|  |  |

# 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

## 针对以下方面的感受

1. Java编程语言是否对你的口味？
2. 关于Eclipse IDE；
3. 关于Git和GitHub；
4. 关于CMU和MIT的作业；
5. 关于本实验的工作量、难度、deadline；
6. 关于初接触“软件构造”课程；