哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

2021 年春季学期

计算学部《软件构造》课程

Lab 1 实验报告

姓名	袁野
学号	1190200122
班号	1903001
电子邮件	youngsc30@qq.com
手机号码	15831338797

目录

1 实验目标概述

- 基本的 Java OO 编程
- 基于 Eclipse IDE 进行 Java 编程
- 基于 JUnit 的测试
- 基于 Git 的代码配置管理

2 实验环境配置

2.1 下载并安装 JDK

A、到官网下载对应版本的 JDK4

B、安装 JDK。

C、修改环境变量

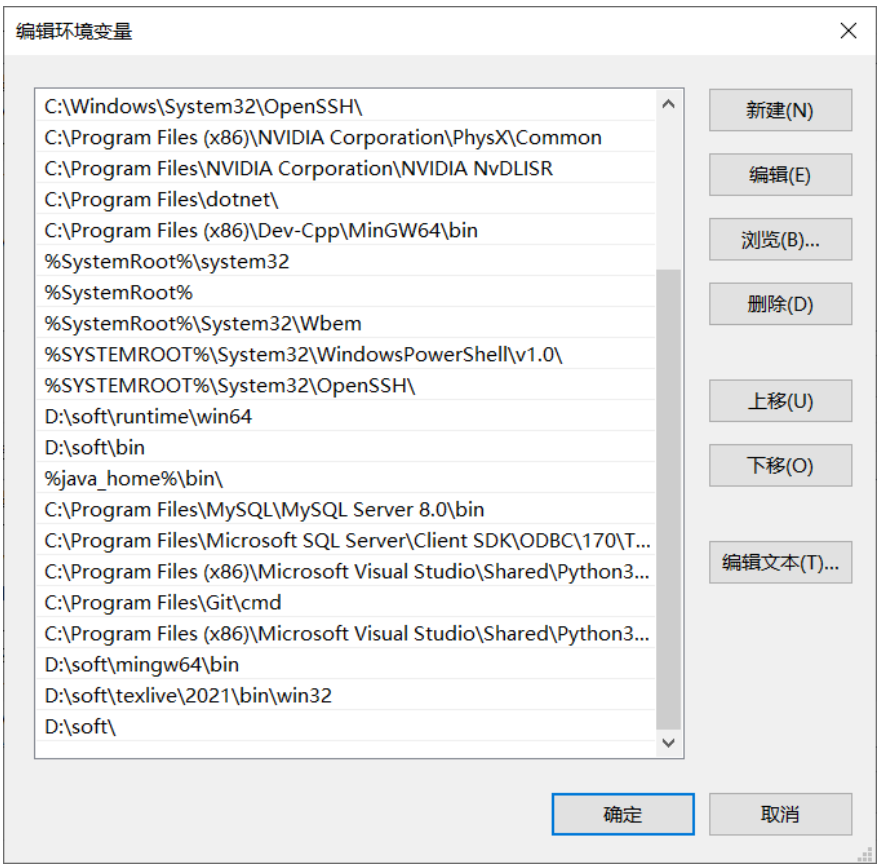

图 2-1 环境变量

D、cmd 测试

```
C:\Users\Youngsc>java -version
java version "1.8.0_251"
Java(TM) SE Runtime Environment (build 1.8.0_251-b08)
Java HotSpot(TM) 64-Bit Server VM (build 25.251-b08, mixed mode)

C:\Users\Youngsc>javac -version
javac 1.8.0_251
```

图 2-2 cmd 测试

2.2 下载安装 eclipse

按照步骤安装，没有遇到困难。

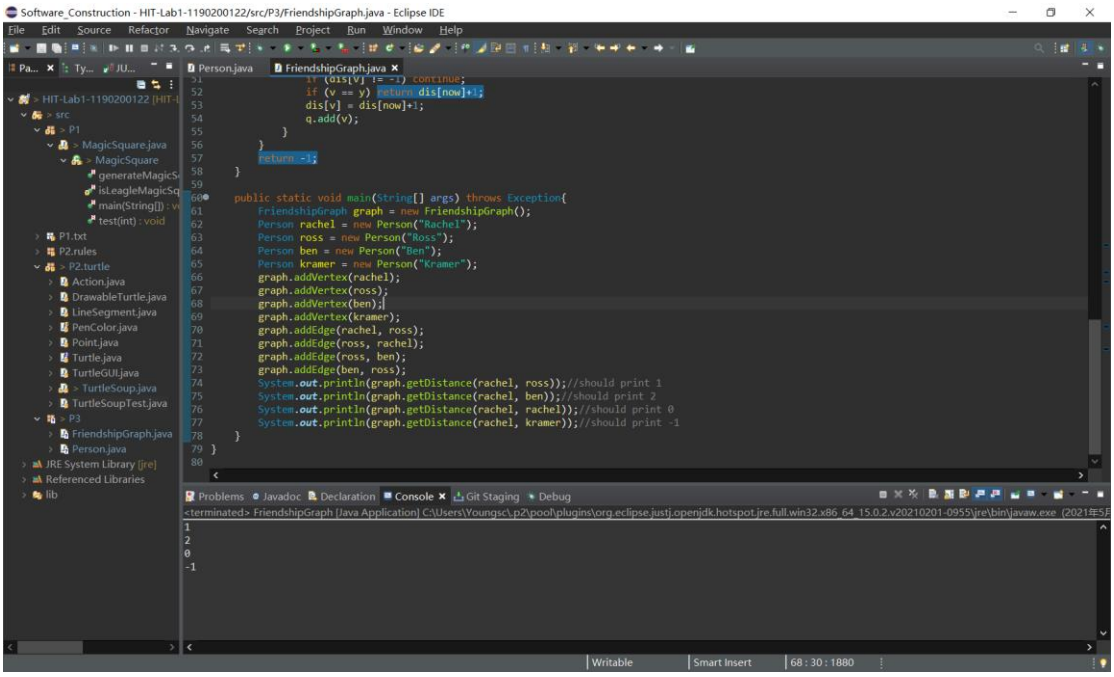

图 2-3 Eclipse 截图

2.3 下载安装 git

官网下载安装包并按照步骤安装。

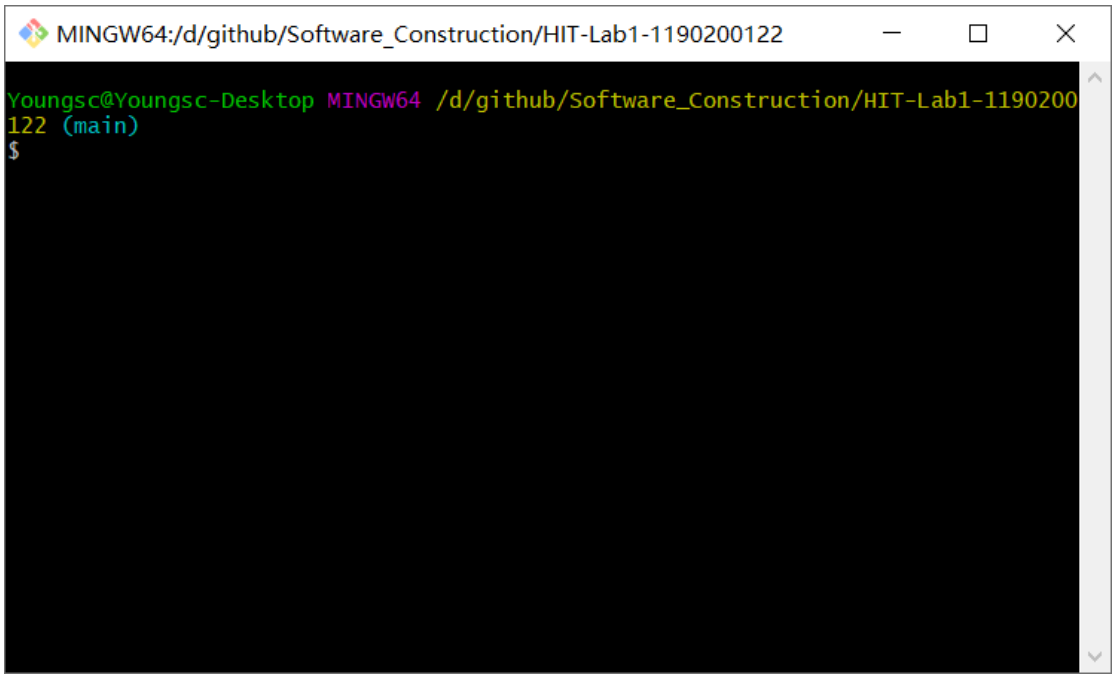

图 2-4 git bash 截图

2.4 Github Lab1 URL：

https://github.com/ComputerScienceHIT/HIT-Lab1-1190200122

3 实验过程

3.1 Magic Squares

本问题需要完成以下任务：1、判断一个文件中是否是一个合法幻方；2、生成一个奇数阶的幻方并输出到指定文件中。

3.1.1 isLegalMagicSquare()

作用：判断一个文件中是否是一个合法幻方。

实现：该函数使用文件读入流，通过两个嵌套的 Vector 实现一个动态二维数组，将文件内的数据进行分割处理后进行储存，同时判断是不是每一个数字都是整数或者都是用‘\t’分割的，然后判断每一行、每一列的数字个数是否相等，判断每一个数字是否是所有 1~n*n 的数字且互不相同，以及每一行每一列以及两条对角线的数字之和是否相等。

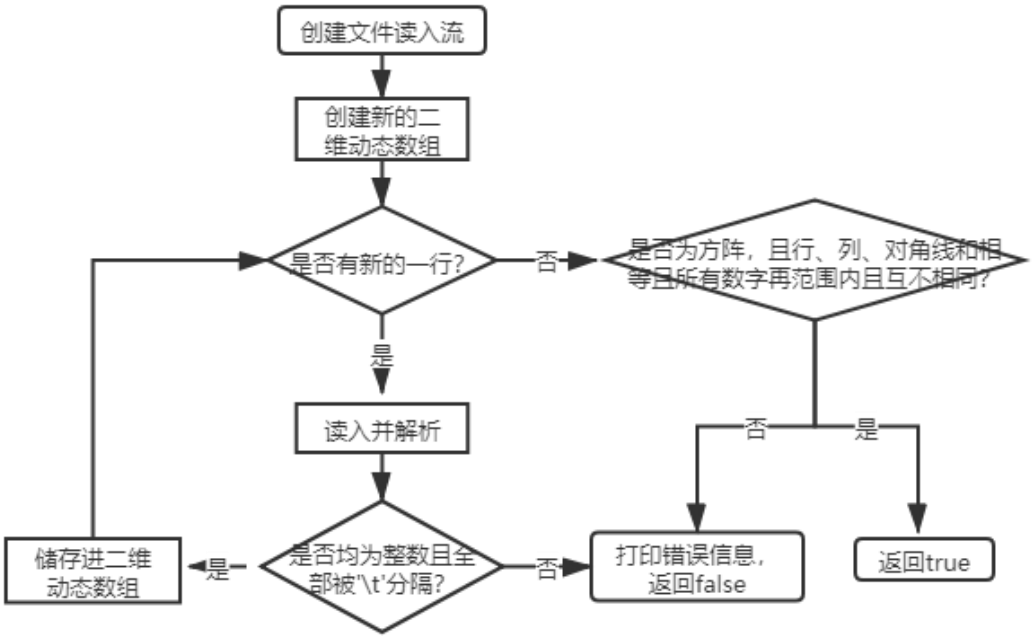

图 3-1 函数流程图

3.1.2 generateMagicSquare()

作用：生成一个奇数阶的幻方，并且写入到指定位置。

实现：该代码的具体注释以及思想已经注释在代码中，通过沿着对角线的方式构造幻方。

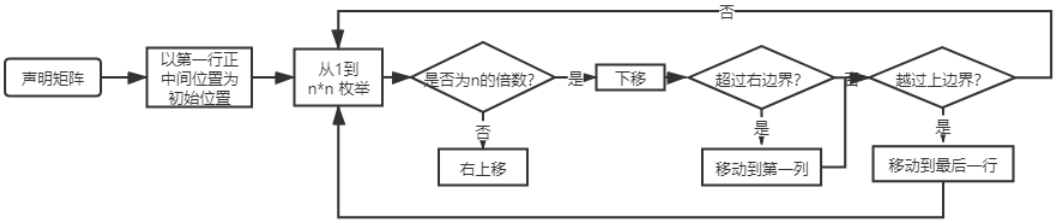

图 3-2 函数流程图

如果输入的为偶数，那么控制台输出：

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 12
    at MagicSquare.generateMagicSquare(MagicSquare.java:17)
    at MagicSquare.main(MagicSquare.java:121)
```

图 3-3 为偶数时抛出的异常

该异常为下标越界，具体为当 n 为偶数时，我们模拟过程会发现，程序总会访问到（n，0）导致下标越界。

如果输入的为负数，那么控制台输出：

```
Exception in thread "main" java.lang.NegativeArraySizeException
    at MagicSquare.generateMagicSquare(MagicSquare.java:11)
    at MagicSquare.main(MagicSquare.java:121)
```

图 3-4 n 为负数时抛出异常

当程序试图创建一个大小为负数的数组时，就会抛出一个对应的异常。

3.2 Turtle Graphics

该任务实现了一个绘制几何图形的工具，并且通过一些函数计算例如内角角度、凸包等一系列信息。

通过 test 结果得知正确。

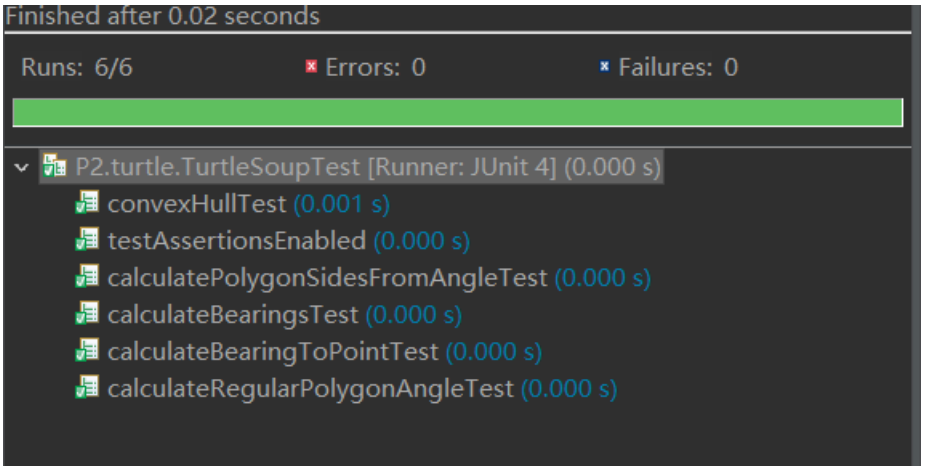

图 3-5 test 结果

3.2.1 Problem 1: Clone and import

首先将放置信息的仓库 clone 到本地并保存到工程文件夹。

```
Youngsc@Youngsc-Desktop MINGW64 /d/github
$ git clone https://github.com/rainywang/Spring2021_HITCS_SC_Lab1.git
Cloning into 'Spring2021_HITCS_SC_Lab1'...
remote: Enumerating objects: 85, done.
remote: Total 85 (delta 0), reused 0 (delta 0), pack-reused 85
Receiving objects: 100% (85/85), 570.53 KiB | 1.05 MiB/s, done.
Resolving deltas: 100% (24/24), done.
```

图 3-6 git bash 界面

将个人仓库克隆到本地，并且将所有的工程文件放置在该目录下等待提交。通过 ssh 将本地与远程仓库链接。

```
Youngsc@Youngsc-Desktop MINGW64 /d/github
$ git clone https://github.com/ComputerScienceHIT/HIT-Lab1-1190200122.git
Cloning into 'HIT-Lab1-1190200122'...
remote: Enumerating objects: 46, done.
remote: Counting objects: 100% (46/46), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 46 (delta 3), reused 38 (delta 1), pack-reused 0
Receiving objects: 100% (46/46), 417.26 KiB | 932.00 KiB/s, done.
Resolving deltas: 100% (3/3), done.
```

图 3-7 克隆个人仓库

3.2.2 Problem 3: Turtle graphics and `drawSquare`

绘制一个正方形，执行四次 forward 和 turn 旋转 270 度即可。

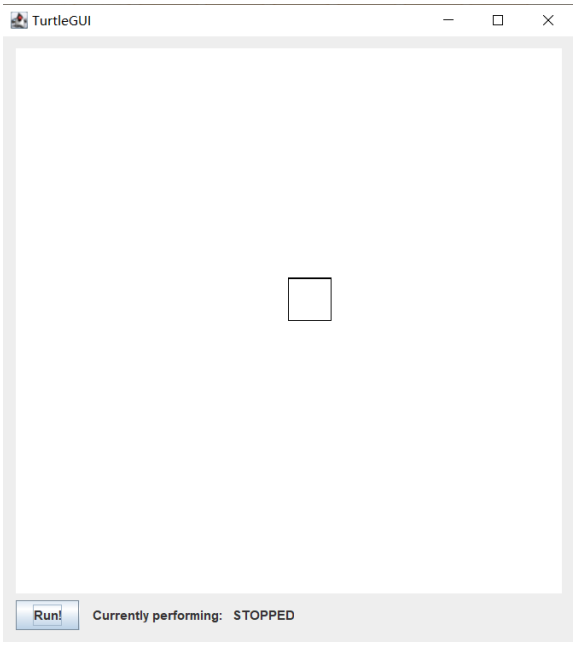

图 3-8 界面截图

3.2.3 Problem 5: Drawing polygons

该函数绘制一个正多边形。

我们通过 calculateRegularPolygonAngle 函数借助内角 $= 180^{o} - 360^{o} /$ 多边形边数 的知识来求得正多边形的内角。然后执行边数次 forward 和 turn 函数，每次偏转外角角度即可。

3.2.4 Problem 6: Calculating Bearings

该函数计算路径转角。

实现 calculateBearingToPoint 函数用于计算两点之间的需要的转角角度。首先计算两点之间的相对坐标，之后使用 Math.atan2 可以计算相对于 y 轴的角度，最后减去原本的角度就可以得到需要的转角。由于返回的角度必须是角度制而且在[0º,360º)间，所以需要转换。最后只需要逐步计算相邻两点之间的转角即可实现 calculateBearing 函数。

3.2.5 Problem 7: Convex Hulls

该函数计算平面点集的凸包。

我采用 graham 凸包算法（O（nlogn））。先挑出最左下方的点 A 作为初始点，按照 A 进行极角排序，将第二个点 B 压入栈中形成第一条边，依次将排序后的点入栈。栈中的点是我们想要的凸包。每加入一个点 C，就相当于加入了一条 C 与栈顶组成的边。利用叉积性质可以判断这条边相对于上一条边的旋转方向。如果这条边发生了逆时针旋转（Cross 为负），则不断弹出栈顶顶点直到满足顺时针旋转。最终所得的栈为凸包。

3.2.6 Problem 8: Personal art

我画了一个抽象的圆形 Windows XP 经典 LOGO 及配色，同时为了调整颜色的深度，再图形中点缀了不同密度的灰色。

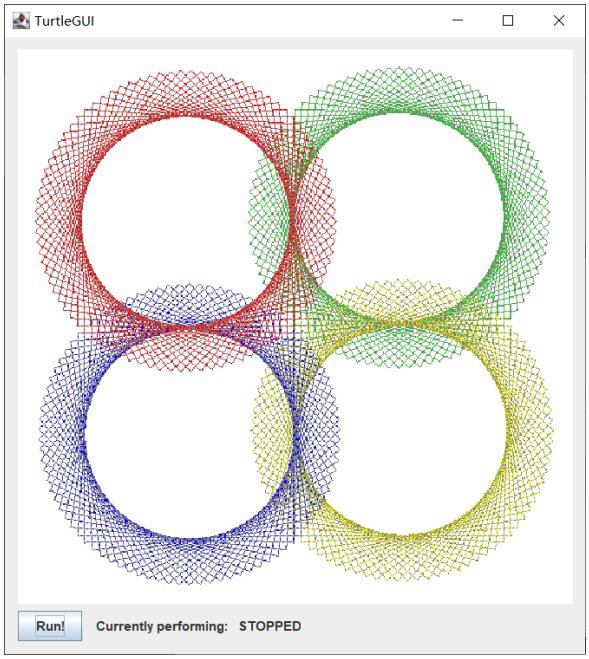

图 3-9 界面截图

3.2.7 Submitting

首先执行 git add .将修改文件放入缓存区，然后执行 git commit -m “”添加信息，最后采用 git push origin master 推送到 master 分支。

3.3 Social Network

该函数实现一个关系网络模型，支持节点和有向边的插入以及任意点对最短距离查询，从而熟悉 OOP 的编程思想和了解 JUnit 进行测试。

3.3.1 设计/实现 FriendshipGraph 类

函数内点与边的定义方式如下。

```
private Map <String, Integer> personID;
private Map<Integer, Set<Integer>> Friendship;

public FriendshipGraph() {
    this.Friendship = new TreeMap<Integer, Set<Integer>>();
    this.personID = new HashMap<String,Integer>();
}
```

图 3-10 函数信息定义

添加节点时，首先判断 HashMap 中有没有该姓名，如果有则抛回异常信息，否则将该姓名与 id 关联，同时初始化对应的关系 Set。

添加边时，首先获得两个节点对应的编号。如果有两个节点都存在，则在起始点在 TreeMap 中对应的 Set 中添加终止点。

计算点对间距离时使用 BFS 算法实现，起点终点相同返回 0，起点终点不连通返回-1。

3.3.2 设计/实现 Person 类

包含私有的 String 类型 name，以及一个公共的 Name（）函数返回 name。

3.3.3 设计/实现客户端代码 main()

将指导书中的代码粘贴到 main 之后，编译运行。

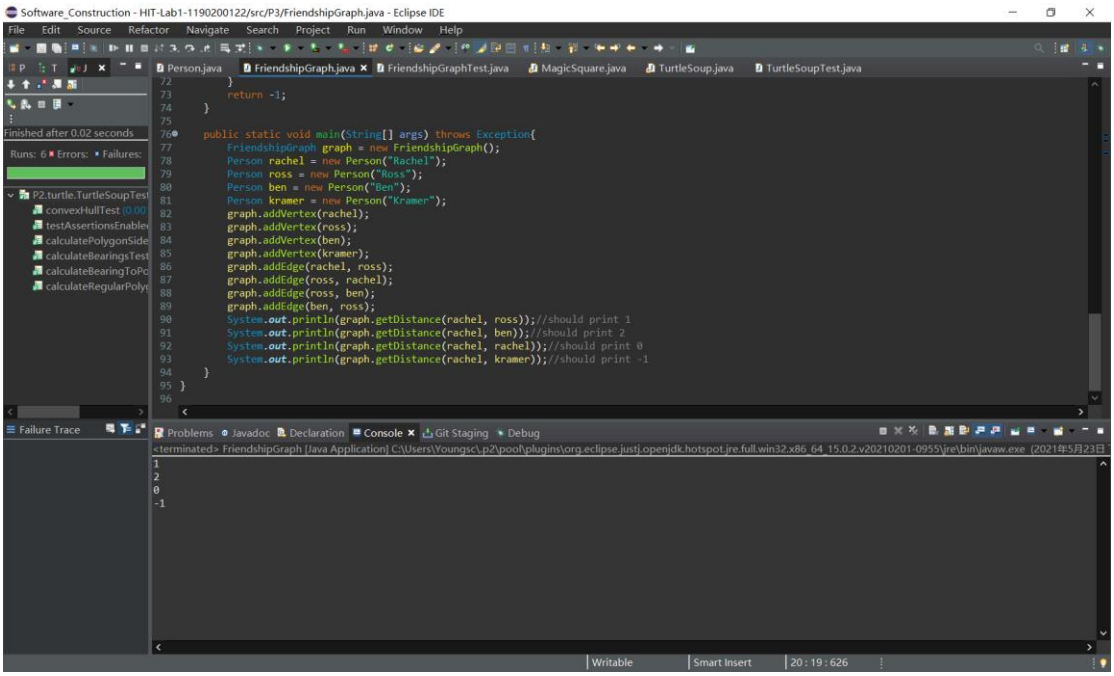

图 3-11 界面结果截图

3.3.4 设计/实现测试用例

构建辅助函数 containPerson 和 containsEdge 分别判断有无指定点和指定边，构造测试用例时测试一下相同名字的不同人是否会报错，有错误抛回异常信息。

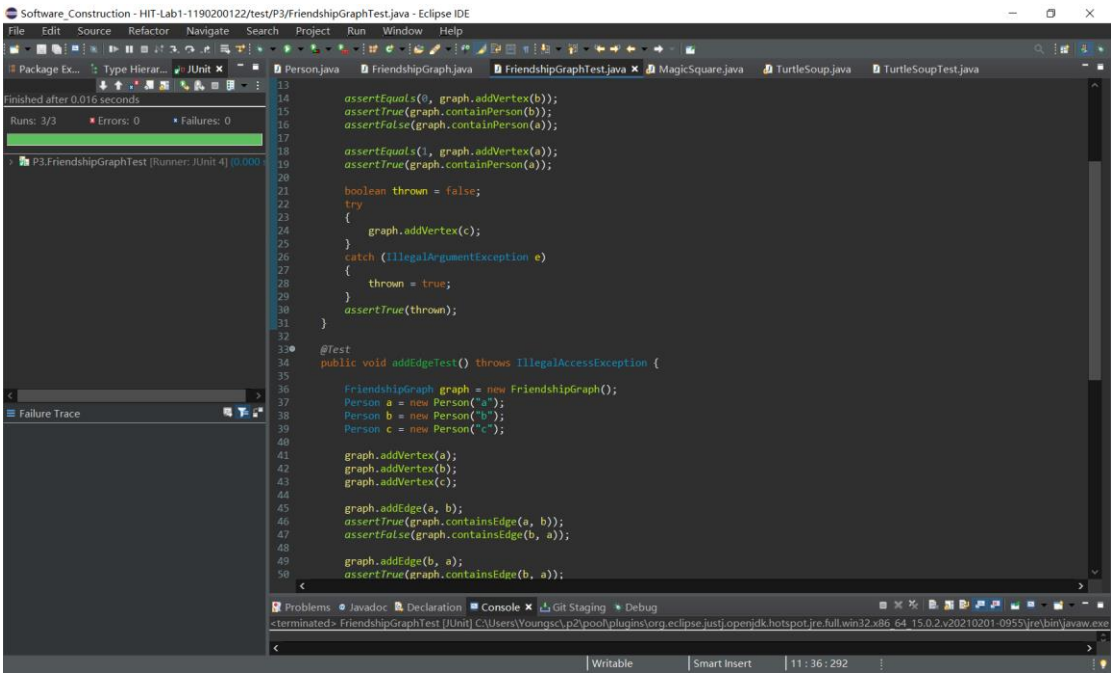

图 3-12 截图

4 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
2021-05-20	15:30-20:00	编写问题 1 的 `isLegalMagicSquare` 和 `generateMagicSquare` 函数并进行测试	按计划完成
2021-05-21	18:00-20:30	编写问题 2 的 `TurtleSoup` 类并进行测试	按计划完成
2021-05-22	18:00-19:30	编写问题 3 的 Person 类和 FriendshipGraph 类并进行测试	按计划完成

5 实验过程中遇到的困难与解决途径

遇到的困难	解决途径
如何从文件中读入并解析出幻方？	使用 ArrayList 套 int[]的方式存储数据，使用 String.split 方法分段，使用 Integer.parseInt 方法解析出 int。
如何计算出 calculateBearingToPoint 中需要的转角？	首先计算两点之间的相对座标，之后使用 Math.atan2 可以计算相对于 y 轴的角度，最后减去原本的角度就可以得到需要的转角。由于返回的角度必须是角度制而且在[0º,360º)间，所以需要转换。
如何测试重复添加时的异常？	使用 JUnit 的 @Test(expected = RuntimeException.class) 可以测试是否抛出 RuntimeException。

6 实验过程中收获的经验、教训、感想

6.1 实验过程中收获的经验和教训

在此次实验，我初步了解了如何 Java 编程。不过在实验过程中，我在编码花费的时间要少于自己思考设计的时间，编码不是编程的难点，如何设计程序才是重点。同时我在实验过程中查询手册花费的时间过多，一方面说明我对于 Java 还不够熟悉，另一方面说明我查询手册的方法还可以改进。

6.2 针对以下方面的感受

(1) Java 编程语言是否对你的口味？

还好，感觉比较繁琐，没有 C++自由，不过很多地方更加方便（比如 import 和 GC）。

(2) 关于 Eclipse IDE

感觉一般，可能是因为我自己平时使用的是 Emacs，更熟悉另一套工作流。个人认为 Eclipse 的工程管理比较迷，个人更加喜欢 Emacs+Maven。

(3) 关于 Git 和 GitHub

我自己之前使用过 git 和 GitHub，对于这两个工具比较熟悉。我认为将工业中比较流行和先进的工具加入教学可以提高学生的工程能力，同时这本身也是教育的进步。

(4) 关于 CMU 和 MIT 的作业

CMU 和 MIT 的作业比较成熟（毕竟他们的计算机教育体系迭代的时间更长），希望工大可以从他们的成果中学习，提高工大自己布置的作业水平。

开源界就不怕“抄袭”，但是新的“代码”应当是更好的更好的，是在巨人的肩膀上进步。

(5) 关于本实验的工作量、难度、deadline

此次实验本身的难度并不大。回顾整个实验如果足够认真，我应该可以在一个下午完成（但是总是摸鱼）。在此次实验中我遇到的花费时间较长的问题都不是实验本身，而是使用 Maven 进行测试、打包等。如果测试时使用的是 Maven 等测试工具，那么实验的代码结构应当与默认的设置相同，减少学生在这些问题上花费的时间。

(6) 关于初接触“软件构造”课程

初步了解了软件构造的过程，不过其实没有太大的感受，可能是因为是初接触。希望以后可以讲解一些具体的案例。