

**2020年春季学期  
计算机学院《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 梁书育 |
| 学号 | 1180300208 |
| 班号 | 1803002 |
| 电子邮件 | [951009723@qq.com](mailto:951009723@qq.com) |
| 手机号码 | 18845773527 |

**目录**

[1 实验目标概述 1](#_Toc34488037)

[2 实验环境配置 1](#_Toc34488038)

[3 实验过程 2](#_Toc34488039)

[3.1 Magic Squares 2](#_Toc34488040)

[3.1.1 isLegalMagicSquare() 2](#_Toc34488041)

[3.1.2 generateMagicSquare() 4](#_Toc34488042)

[3.2 Turtle Graphics 6](#_Toc34488043)

[3.2.1 Problem 1: Clone and import 6](#_Toc34488044)

[3.2.2 Problem 3: Turtle graphics and drawSquare 7](#_Toc34488045)

[3.2.3 Problem 5: Drawing polygons 9](#_Toc34488046)

[3.2.4 Problem 6: Calculating Bearings 12](#_Toc34488047)

[3.2.5 Problem 7: Convex Hulls 13](#_Toc34488048)

[3.2.6 Problem 8: Personal art 14](#_Toc34488049)

[3.2.7 Submitting 16](#_Toc34488050)

[Social Network 16](#_Toc34488051)

[3.2.8 设计/实现FriendshipGraph类 16](#_Toc34488052)

[3.2.9 设计/实现Person类 18](#_Toc34488053)

[3.2.10 设计/实现客户端代码main() 19](#_Toc34488054)

[3.2.11 设计/实现测试用例 20](#_Toc34488055)

[4 实验进度记录 21](#_Toc34488056)

[5 实验过程中遇到的困难与解决途径 21](#_Toc34488057)

[6 实验过程中收获的经验、教训、感想 22](#_Toc34488058)

[6.1 实验过程中收获的经验和教训 22](#_Toc34488059)

[6.2 针对以下方面的感受 22](#_Toc34488060)

# 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 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

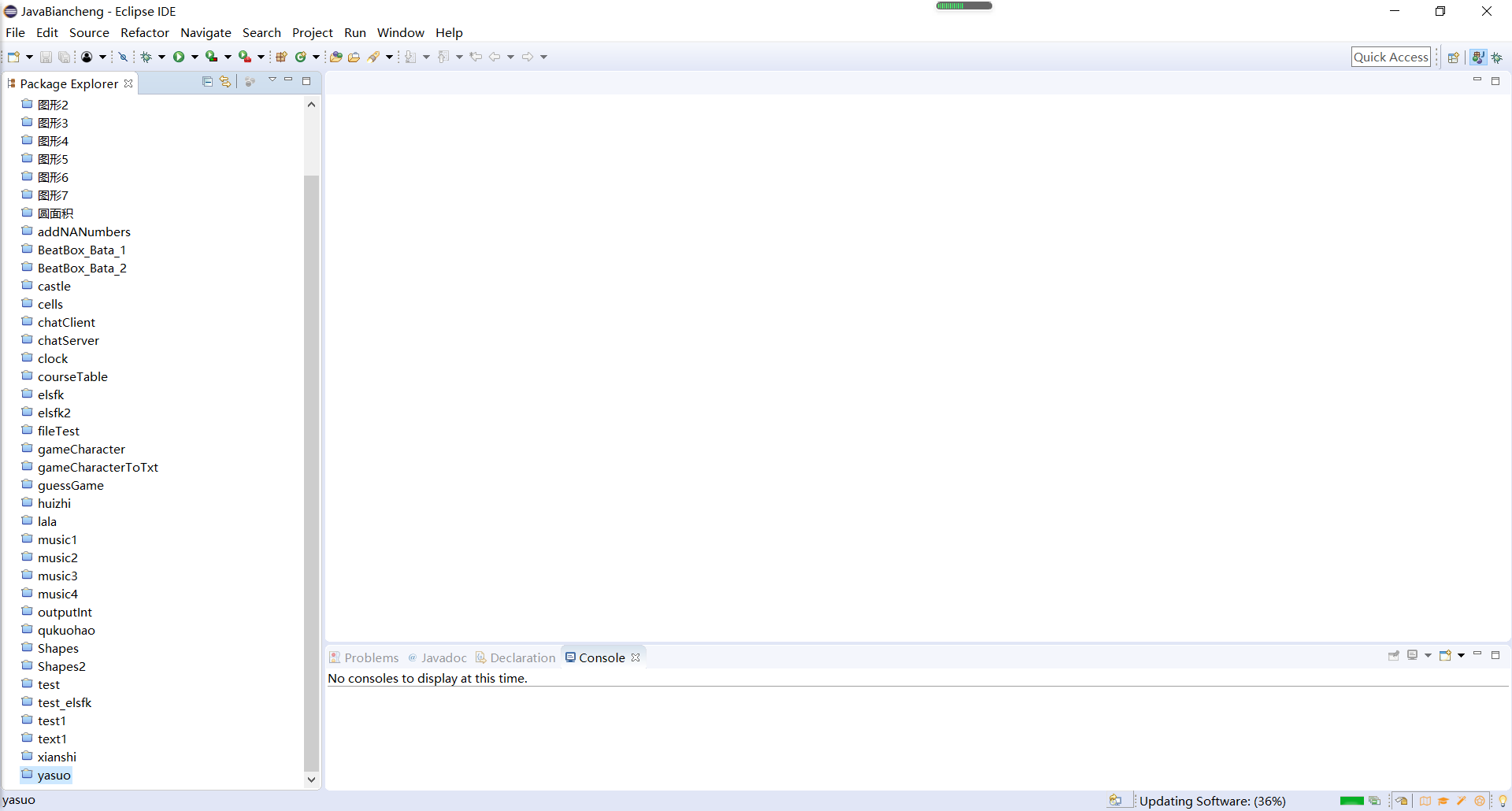
简要陈述你配置本次实验所需开发、测试、运行环境的过程，必要时可以给出屏幕截图。

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。

由于之前选修过Java编程的课程，所以JDK以及eclipse都已经安装完成。同时git在之前也有安装过。所以在这部分就直接使用之前的环境，没有遇到问题和困难。

下面展示一下我的实验环境。

Eclipse：



Git：



在这里给出你的GitHub Lab1仓库的URL地址（Lab1-学号）。

[https://github.com/ComputerScienceHIT/Lab1-1180300208](https://github.com/ComputerScienceHIT/Lab1-1180300208.git).git

# 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

该任务主要分为两步：一是完成isLegalMagicSquare()这一方法，二是拓展给定的generateMagicSquare()方法。实验的基础是了解magic square（幻方）构成的基本规则：每行每列以及对角线的和都一样。

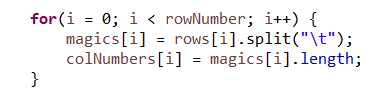
### isLegalMagicSquare()

首先明确实验的要求。实验要求读入包含了矩阵信息的txt文件，之后对该文件进行解析，之后判断该矩阵是否满足构成幻方的条件。

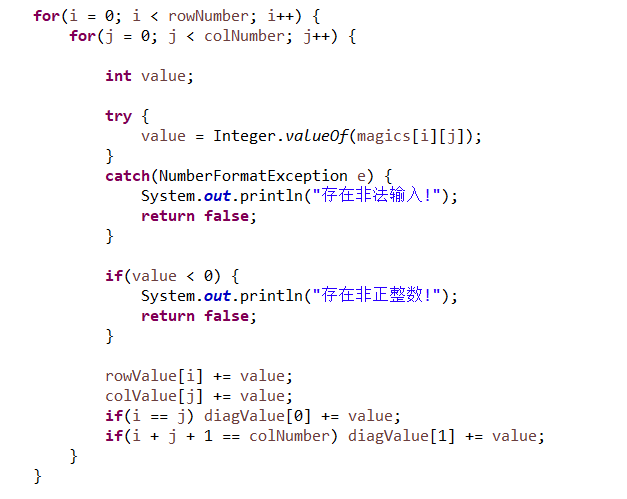
进行文件信息读入。将文件中的每一行作为一个字符串分别存放在字符串数组中，同时记录下行数备用，等待之后步骤中对每一行的数据进行解析。



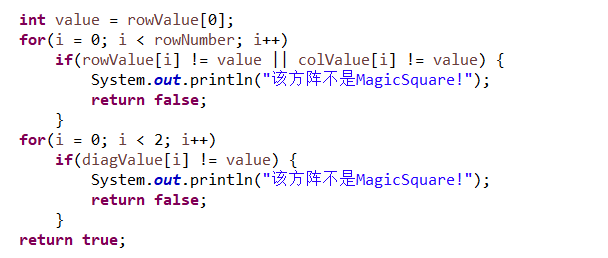
解析矩阵数据。使用split函数进行将每一行中由 ”\t” 隔开的每个数字存放在字符串数组中，同时记录下每一行的数字数，方便下一步判断矩阵是否为方阵。



对分离出来的数字进行解析，将其转换成正整数，如果出现了非正整数或者非法输入的情况，“优雅”地提示出现了异常情况，并返回false。与此同时，求出需要的和。



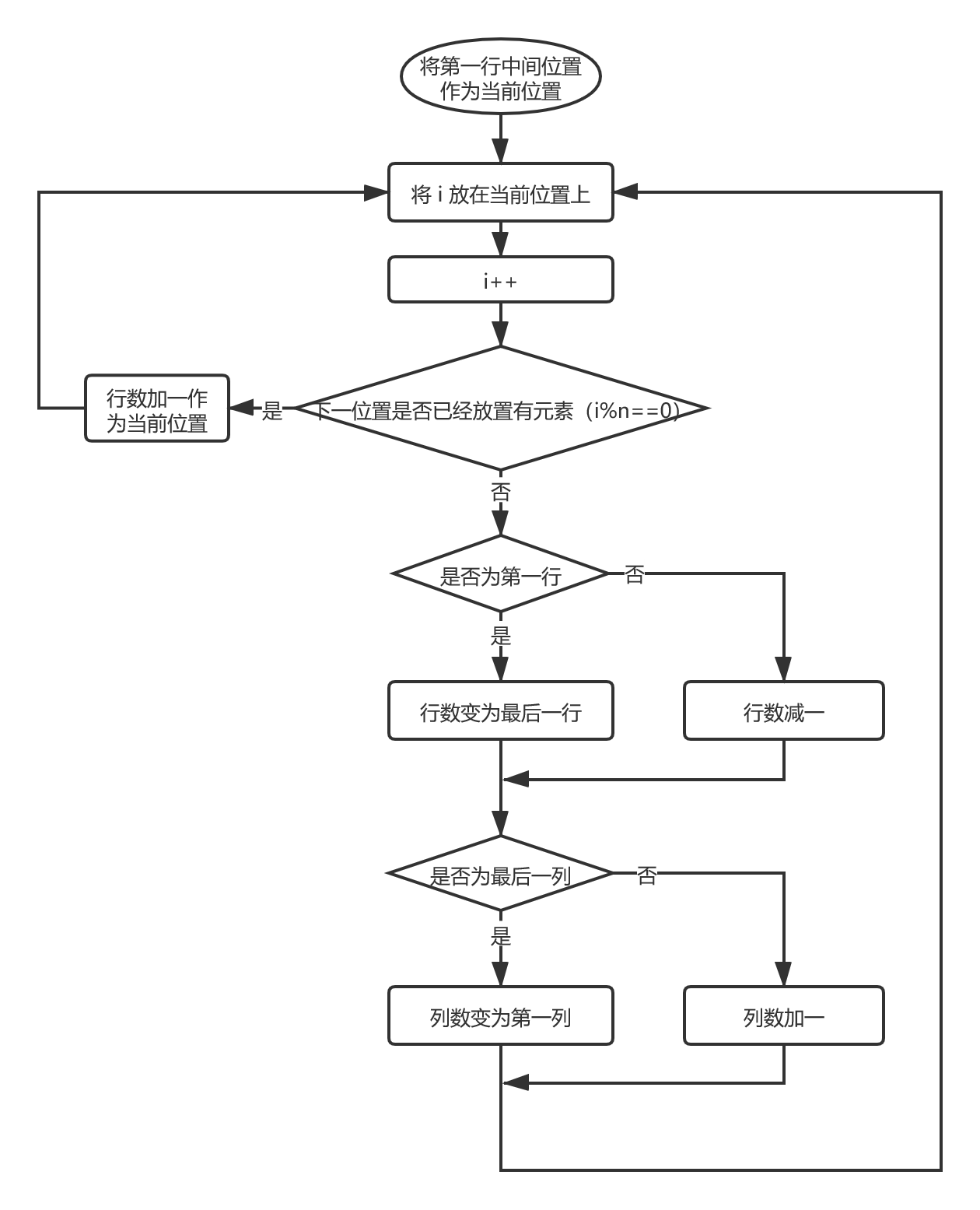
判断方阵是否为幻方。如果程序成功进行到了这一步，那么将正式进行幻方的判断。对每一行每一列及对角线求和并且比较每一个和是否相等，若满足幻方的规则，则返回true，否则返回false。



### generateMagicSquare()

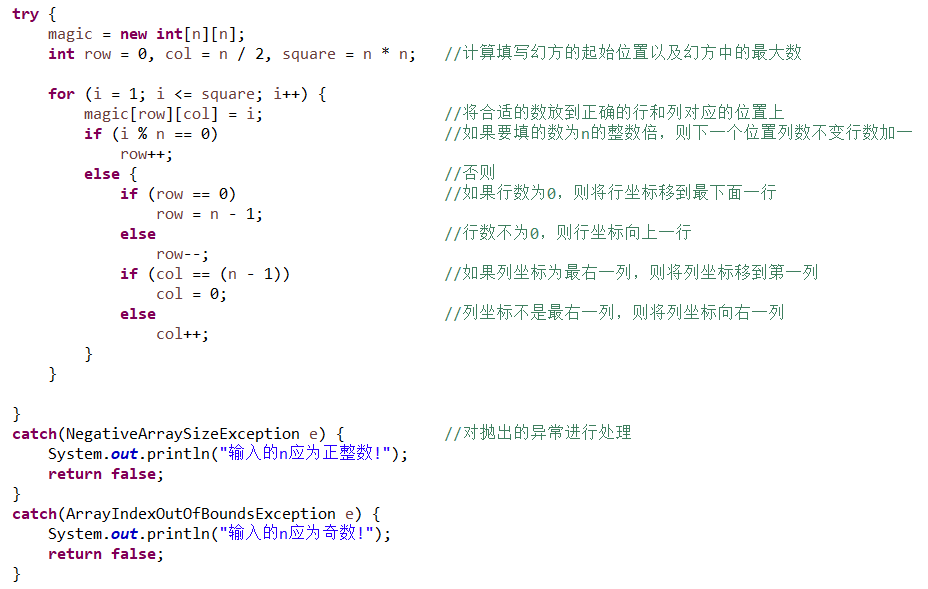
按步骤给出你的设计和实现思路/过程/结果。

由参数n构造幻方的流程：

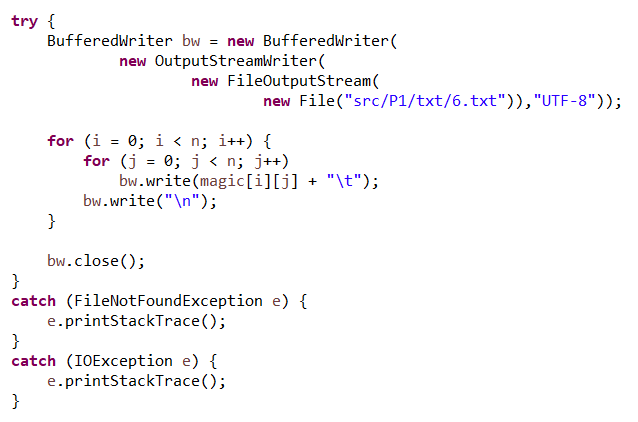
**

根据参数n实现幻方的方法：

首先确定从1开始的第一个数放在第一行的中间，之后将每一个待放置的数放在前一个数的右上的位置，如果当前位置为最上一行则下一次放在最后一行，当前位置为最右一列，则将下一次放在第一列，如果按照上述方式选取的位置已经有元素存在，则将其放在该位置的下一行同一列的位置（判断i是否被n整除的意义也在于此）。



之后用类似之前文件读取的方法将生成的幻方存储到文件6.txt中。

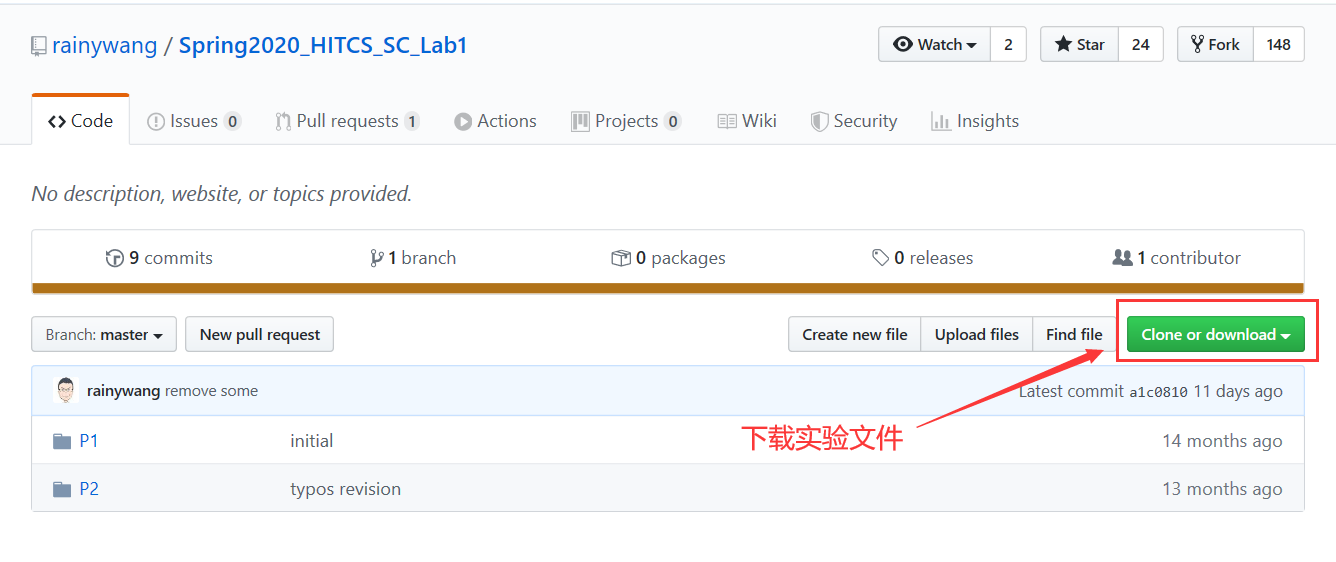


## Turtle Graphics

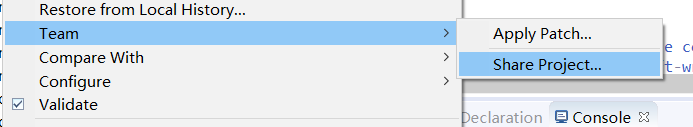
该任务主要是补全给出的空白方法，计算多边形角和边的关系，在画纸上画出给定边长的正方形，画出给定边数和边长的正多边形，给定两个或者多个点的坐标和初始偏移角计算所需要偏移的角度，计算一系列点的凸包等。最后需要对完成的方法进行测试。

### Problem 1: Clone and import

关于从GitHub获取该任务的代码。进入提供的GitHub仓库（URL：<https://github.com/rainywang/Spring2020_HITCS_SC_Lab1/tree/master/P2> ），再进入到上一级目录Spring2020\_HITCS\_SC\_Lab1中，下载该实验的实验文件，保存到本地。

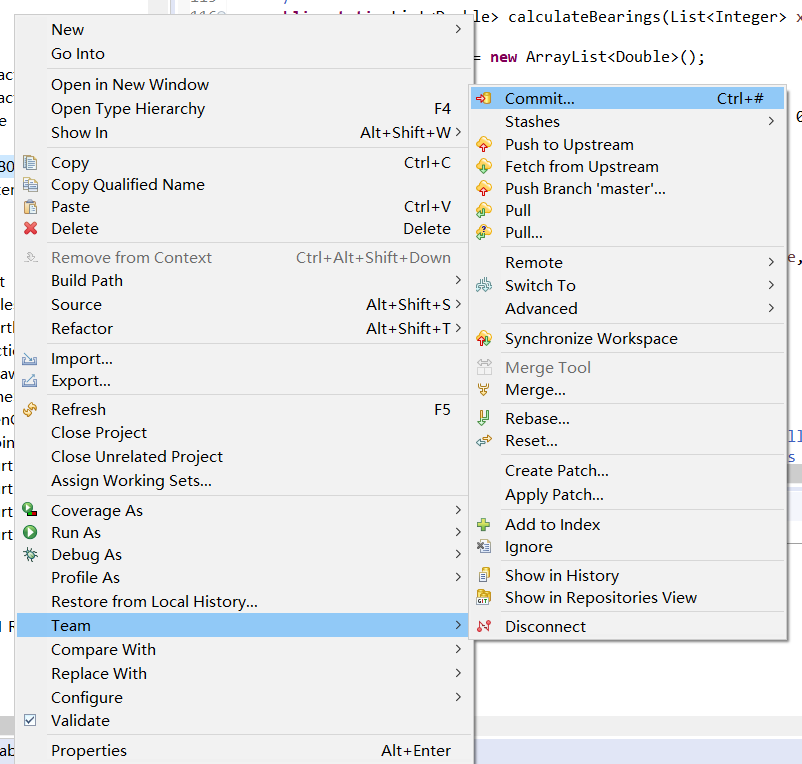


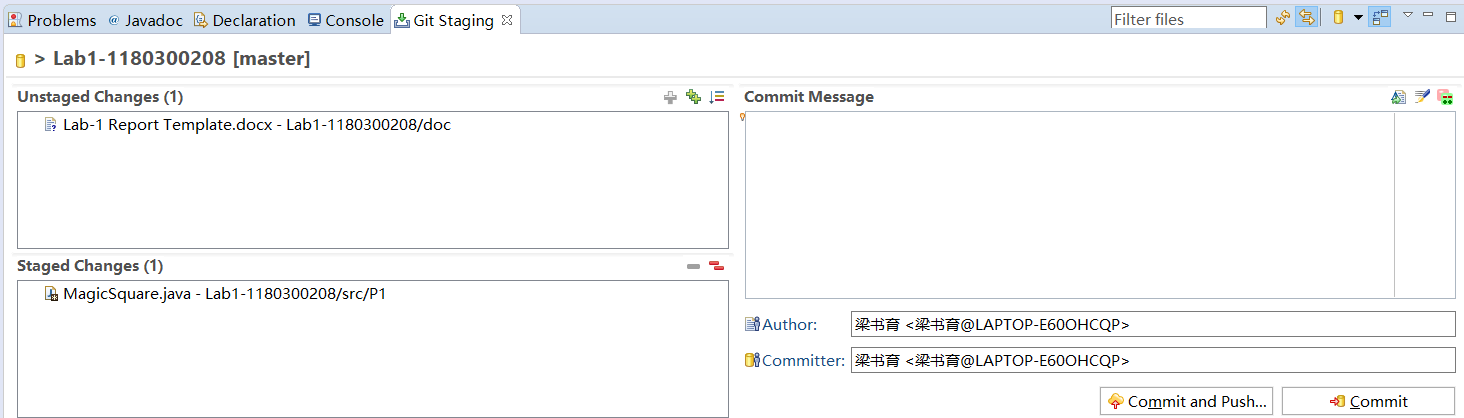
关于在本地创建git仓库。我使用了eclipse中的可视化操作方法，在工程项目上右键，选择Team 🡪 Share Project… ，如下图。



随后创建本地git仓库，将代码存放在仓库中，至此，本地git仓库创建成功。

关于使用git管理本地开发。每次要将代码提交至仓库中时，依然到Team选项中，选择Commit…，如下图。





之后选择要更新的文件，填写提交信息即可。如果要提交到本地git仓库点击Commit，如果提交到远程GitHub仓库则点击Commit and Push

### Problem 3: Turtle graphics and drawSquare

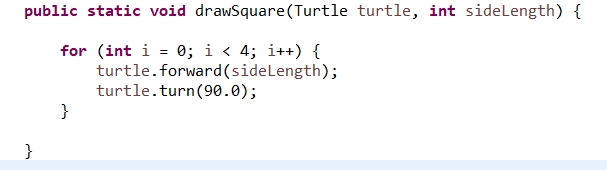
使用以及提供的forward(units)和turn(degrees)两个方法绘制正方形。首先需要明确两个方法的作用。

forward(units)：向当前方向前进units距离，其中units为int类型的数据

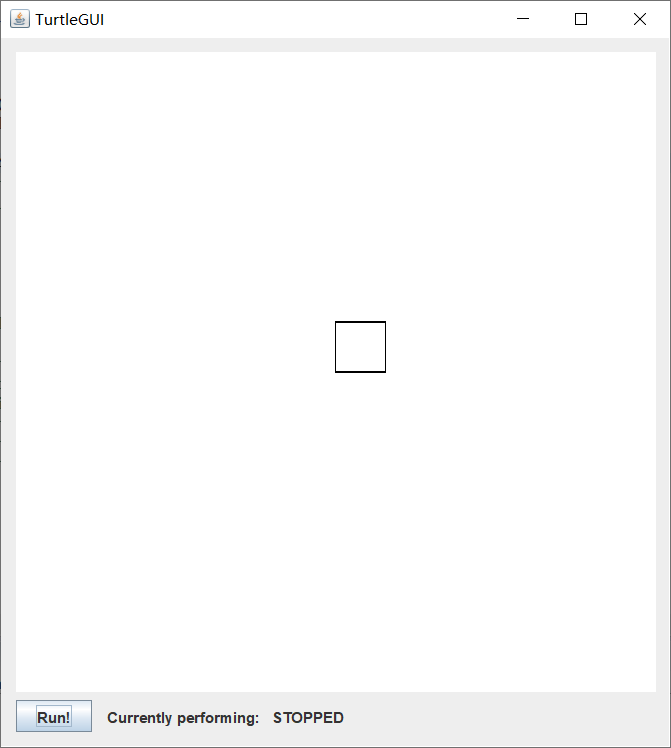
turn(degrees)：将当前方向顺时针旋转degrees°，其中degrees为double类型的数据。

明确了两个方法的作用，我们就可以使用其来绘制正方形。

设正方形的边长为sideLength，即需要将线条前进sideLength长度，之后旋转90°，再向前进sideLength长度，如此往复四次即可。简单编写代码如下图：



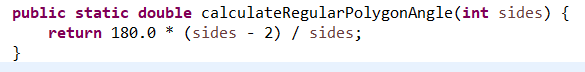
运行结果如下图：



### Problem 5: Drawing polygons

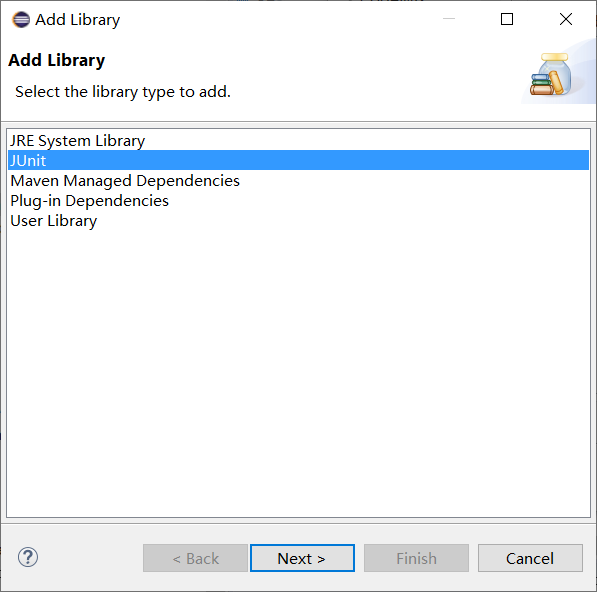
同Problem 4类似，只是我们需要计算出每次需要偏移的角度。即完成calculateRegularPolygonAngle(int sides)方法，其中参数sides为要绘制的正多边形的边数。完成这一方法需要的数学基础为多边形内角和为180.0 \* (sides - 2) / sides

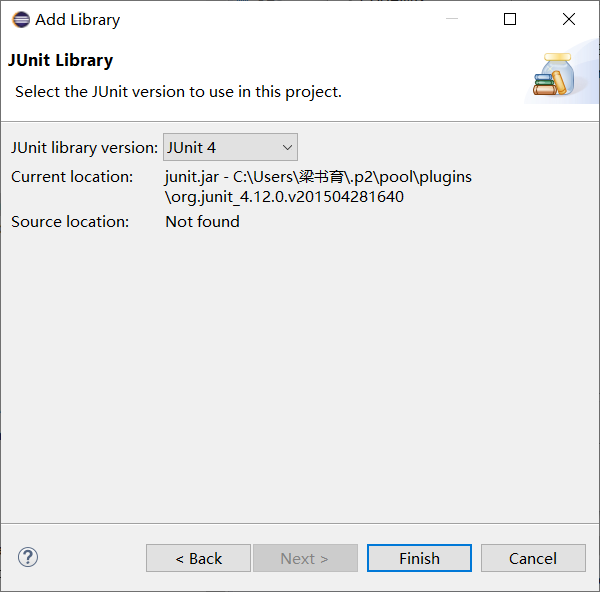
代码如下图：



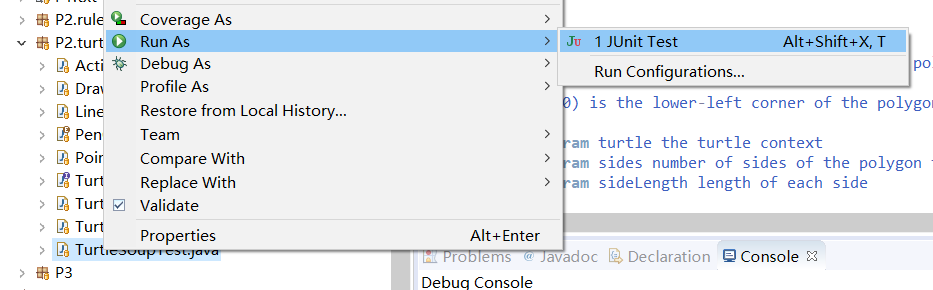
之后需要对该方法进行测试。

需要向工程中添加Junit Library，方法如下图：

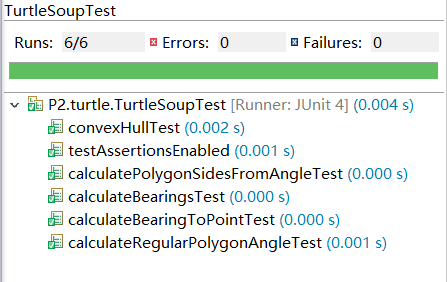


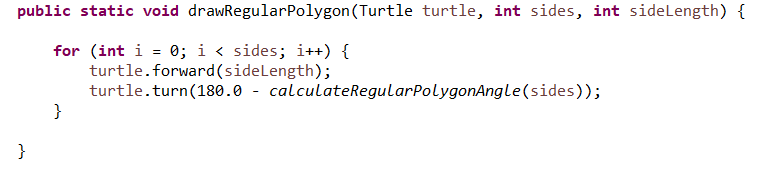


Finish后即成功添加。之后运行TurtleSoupTest，如下图：

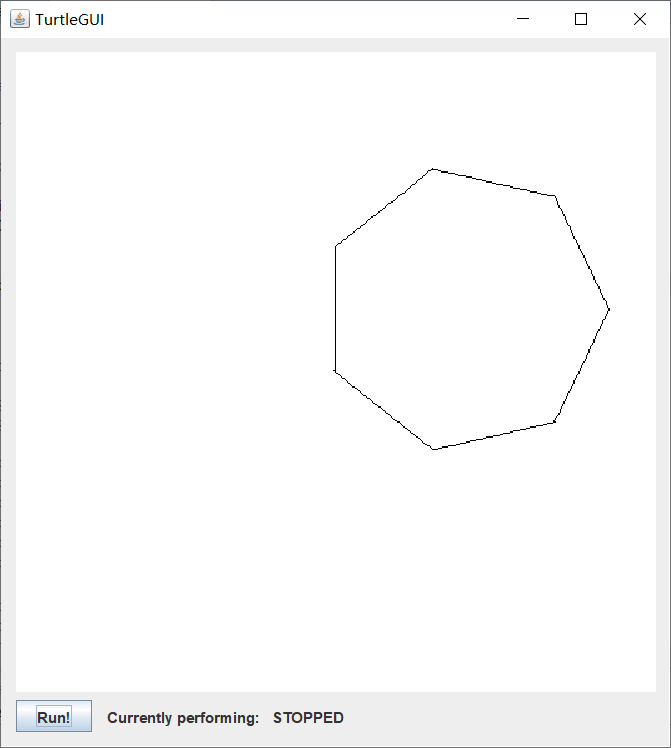


运行结果如下：

（全是绿的，不要介意，我实验都完成才完成的实验报告，下次不会了~）

测试通过说明该方法没有问题，之后类似Problem 4绘制多边形即可，代码如下：

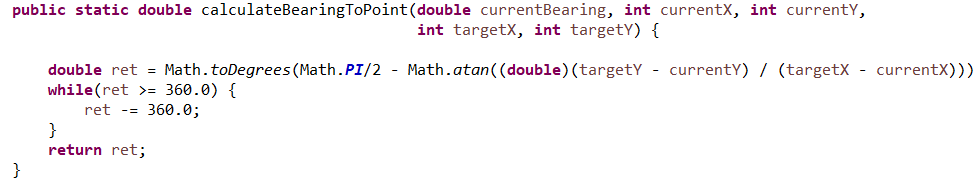
运行结果如下：



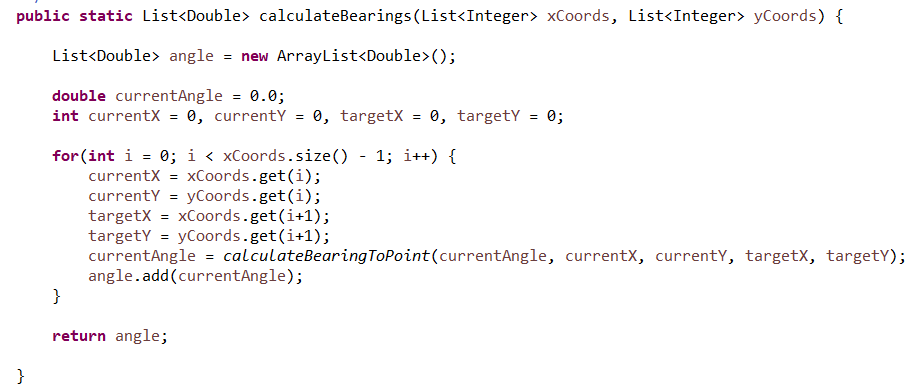
### Problem 6: Calculating Bearings

该问题为计算从一个点到另一个点所需要旋转的角度以及从一个点经过一系列点需要的旋转的一系列角度。方法即为根据两点之间的横坐标差和纵坐标差计算出偏转角的正切值，再通过反三角函数算出需要的角度，再减去已经有的当前角度即可。

代码如下所示：



calculateBearings(List<Integer> xCoords, List<Integer> yCoords)方法使用类似的思路，调用calculateBeringToPoint方法即可，代码入下：



经过测试，通过。

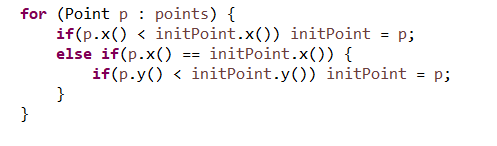
### Problem 7: Convex Hulls

该问题计算凸包问题，使用卷包裹算法可以很好解决问题。思路为先选取一定在凸包上的点，之后不断寻找最外侧的边，即将该点与其他点的连线不断向顺时针方向转动，最终找到最靠外侧的点。而判断是是否为顺时针旋转用到了向量的叉乘的概念。具体思路方法借鉴了该视频：

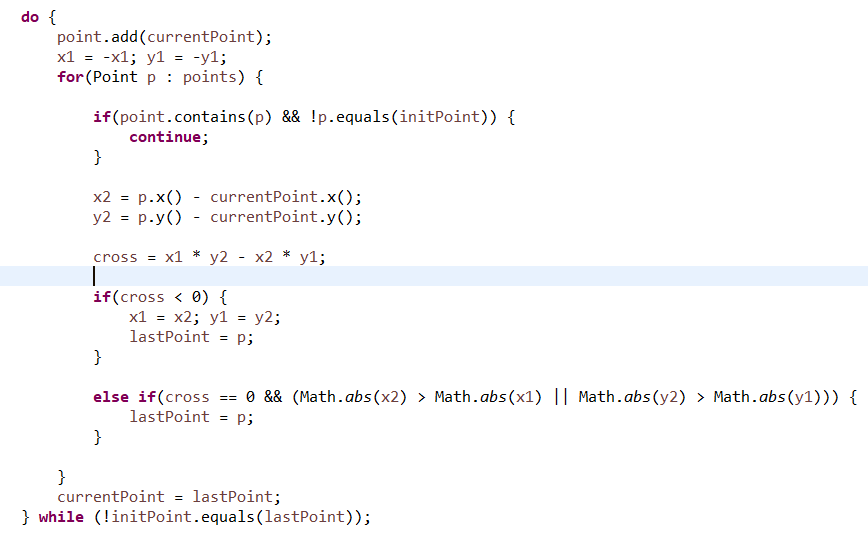
<https://www.bilibili.com/video/av59283385>

下面介绍代码细节：

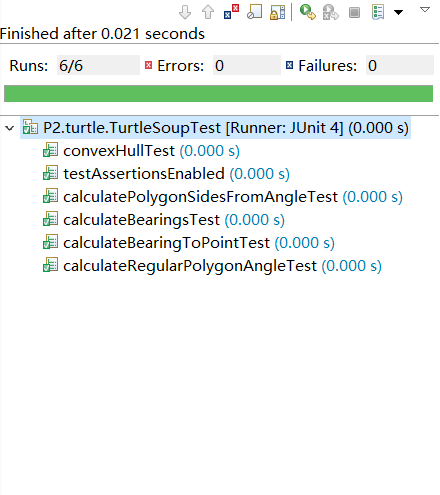
1、寻找一定在凸包上的点，即寻找横坐标最小的点，如果有横坐标最小的几个点，则选取纵坐标也相同的一个，如下图：



2、将当前的点的初始方向设为与y轴正方向同向，随后计算连线到达剩余是否为顺时针，如不是则舍弃，如是则暂时将当前连线作为凸包边界。最终找到凸包边界后，将当前点置为当前连线的终点，继续进行，直到当前点与初始点相同为止，代码如下：

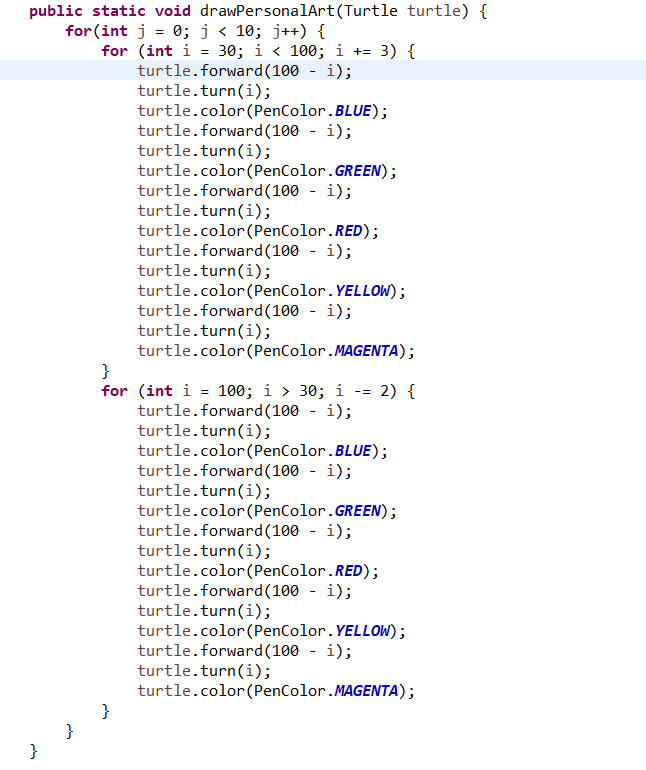


最终的集合即为凸包的边界点。值得注意的是，当有多个点在边界上时，需要选择距离当前点远的点作为边界点。进行测试后，通过。至此，所有的六个测试全部通过。

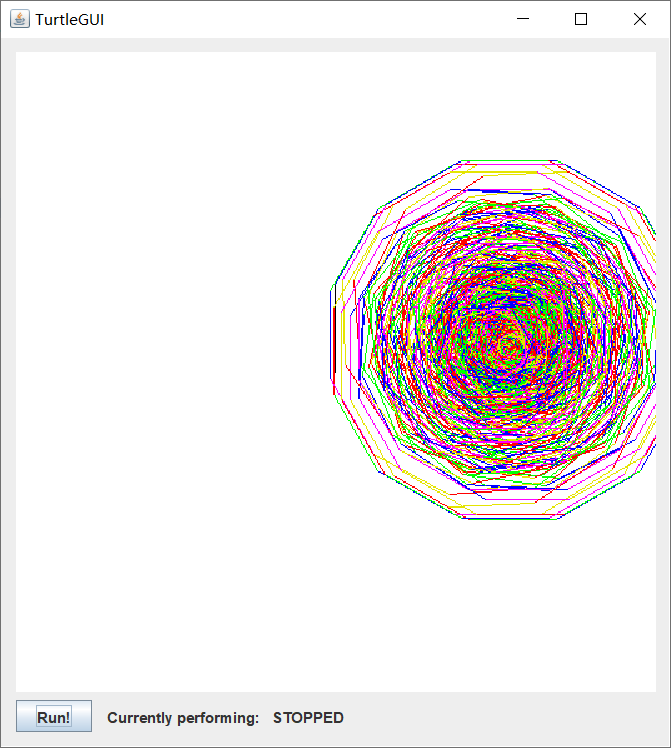


### Problem 8: Personal art

自由发挥即可。



运行结果：



### Submitting

使用Team 🡪 Commit 🡪 Commit and Push提交即可。

## Social Network

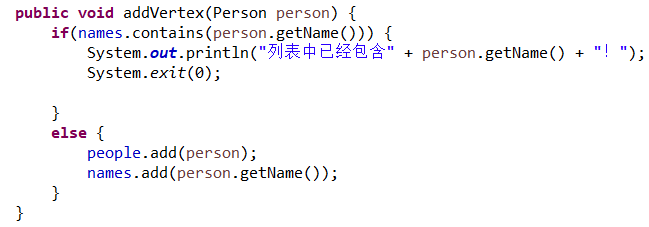
该任务主要为向图中添加“person”和“friend”的关系，形成friendship关系网络，同时计算人与人之间的最短关系层数，即图中两顶点的最短生成路径的长度。

### 设计/实现FriendshipGraph类

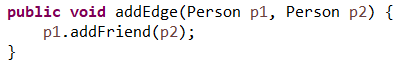
在FriendshipGraph类中声明了List<Person> people和List<String> names两个ArrayList容器，分别用来存放人的信息和名字。

声明了addVertex(Person person)、addEdge(Person p1, Person p2)、getDistance(Person p1, Person p2)三个方法。

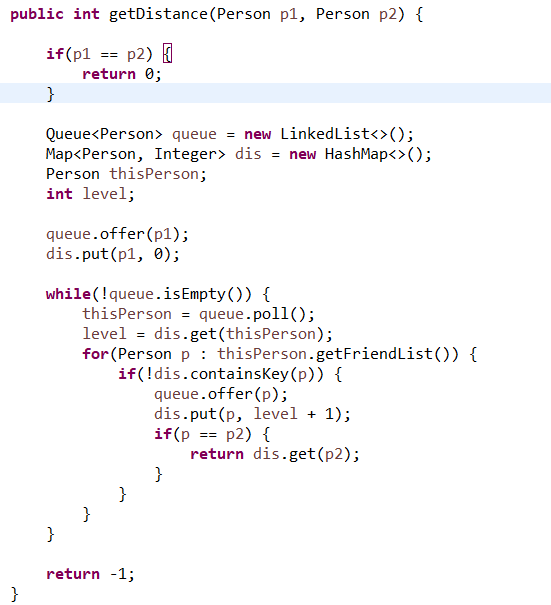
addVertex为将人加入列表，同时注意列表中不能有相同的人，否则程序给出提示并退出，代码如下图：



addEdge为将p2作为p1的friend加入到p1的friend列表中，代码如下图：



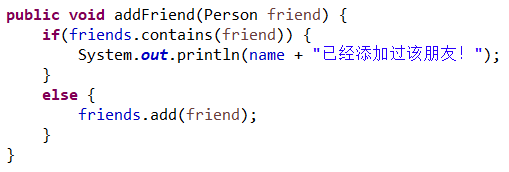
getDistance为计算人与人之间最短的朋友路径长度，使用了队列进行广度优先搜索，用hashmap存储每一个经过的person的路径长度，每经过一个节点，存储的路径长度加一，直到找到最终的目标为止，这时目标返回目标的路径长度，如果没有找到该目标，则返回-1说明这两人之间没有直接或者间接的friend关系，如果目标即为本身则直接返回0。



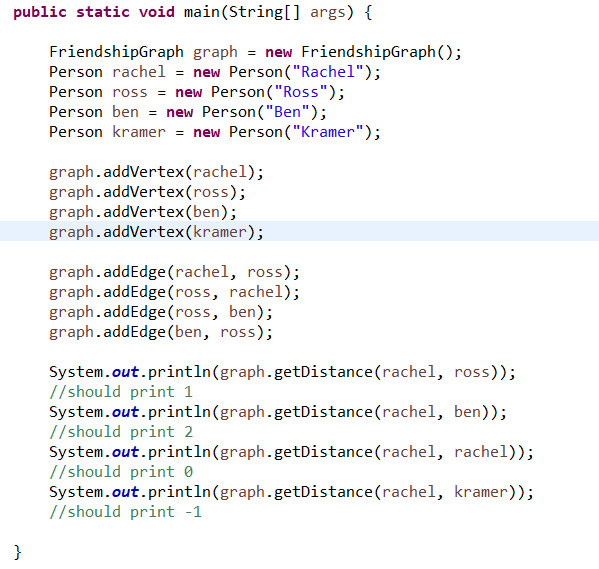
### 设计/实现Person类

Person中声明了String类型的name用来存储该Person的姓名信息，以及List<Person>类型的friends用来存储该Person的friend列表。

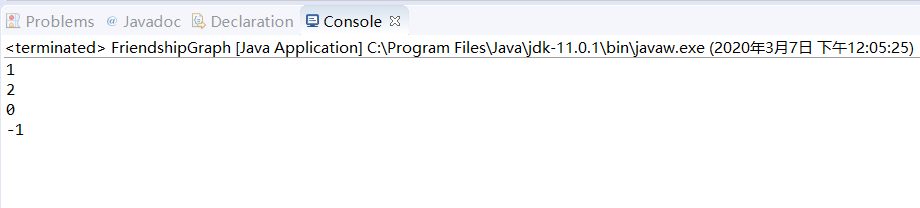
声明了addFriend(Person friend)方法和getName()、getFriendList()方法，后二者顾名思义，不再赘述。addFriend(Person friend)方法则是将传入的Person加入到该Person的friend列表当中，也要注意不能重复添加friend。代码如下：



### 设计/实现客户端代码main()

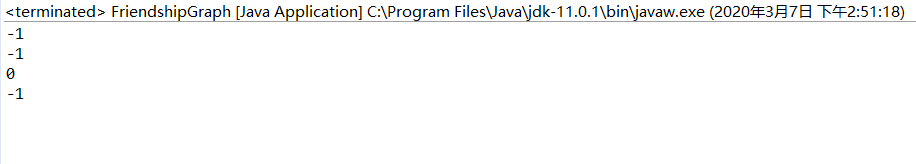


先按照实验手册中给出的上述的main方法运行，得到正确结果（如下图）。

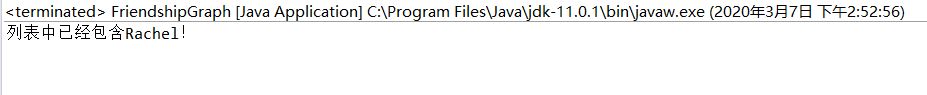


之后再按照实验指导书中将main方法进行改动。

1、使rachel和ross之间只存在单向的社交关系。运行结果如下：

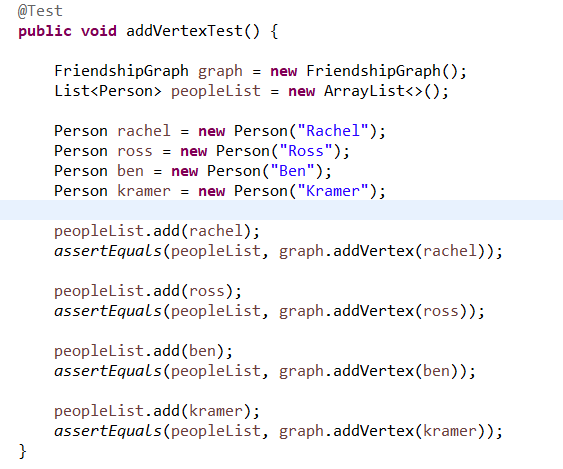


2、将命名阶段的“Ross”替换为“Rachel”。运行结果如下：



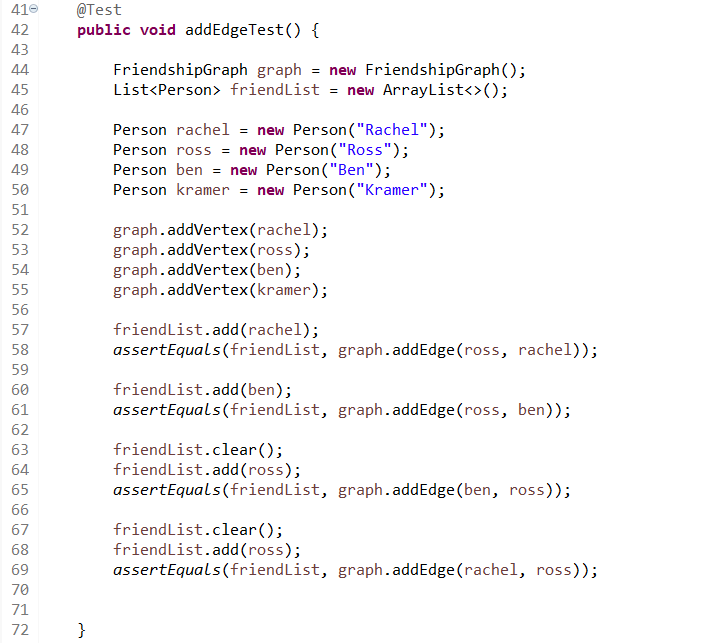
### 设计/实现测试用例

addVertex测试：



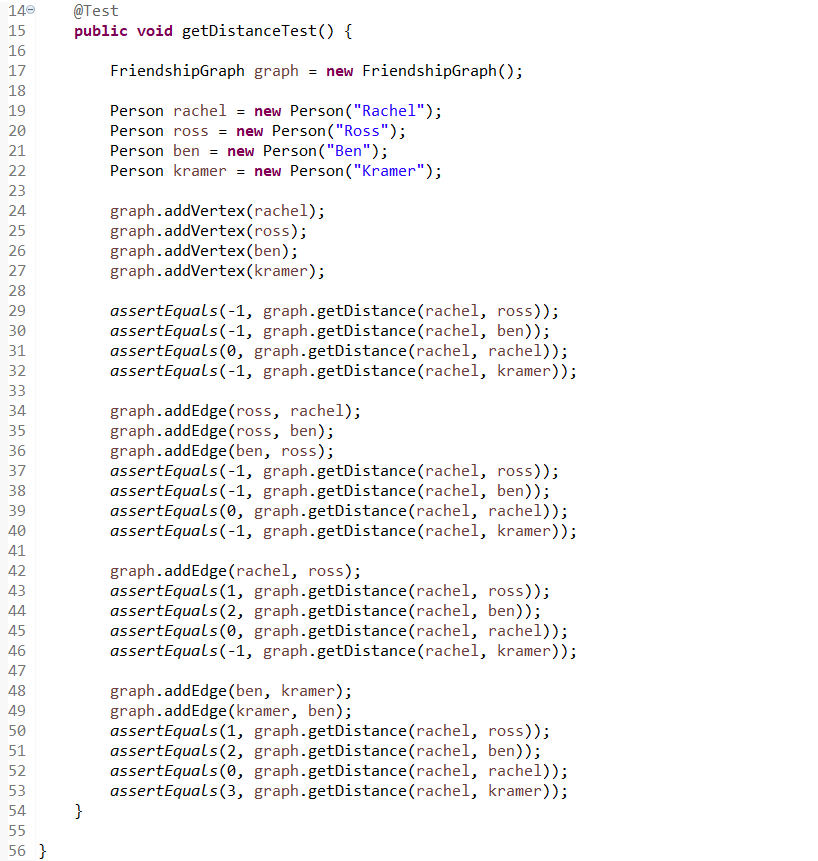
测试了加入不同姓名的Person后person列表的变化。

addEdge测试：



测试了对应的Person在加入friend后friend list的变化。

getDistance测试：

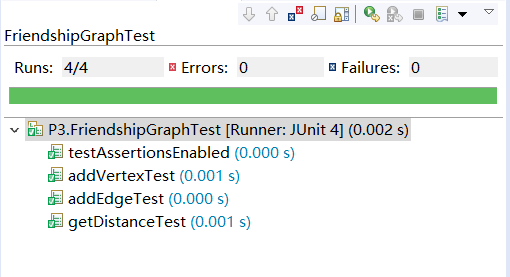


1、测试了只添加人的信息而不添加friend信息的情况。

2、测试了人与人之间单向连接的情况

3、测试了人与人之间的间接friend关系

测试通过，结果如下所示：



# 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。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 任务 | 实际完成情况 |
| 2020-02-29 | 21:00-23:30 | 编写问题1的函数并进行测试 | 按计划完成 |
| 2020-03-01 | 08:00-09:45 | 编写问题2的drawSquare、calculateRegularPolygonAngle、drawRegularPolygon函数并进行测试 | 按计划完成 |
| 2020-03-01 | 10:00-11:35 | 编写问题2的calculationBearingToPoint、calculateBearings方法并进行测试 | 按计划完成 |
| 2020-03-01 | 21:00-23:55 | 编写问题2的calculatePolygonSidesFromAngle、convexHull方法并进行测试 | 按计划完成， |
| 2020-03-02 | 10:50-11:11 | 编写drawPersonalArt方法并进行测试 | 按计划完成 |
| 2020-03-04 | 15:30-16:50 | 发现了convexHull存在的问题，重新调试修改并将junit5改为junit4 | 按计划完成 |
| 2020-03-05 | 20:00-21:30 | 编写完成问题3需要的类及方法 | 按计划完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的难点 | 解决途径 |
| 读取文件时遇到障碍 | 上网查找资料发现问题并成功解决 |
| 一开始将junit版本定为了5，在转为4的过程中出现错误 | 向老师请教，自己又回去查看设置发现虽然library改成了Junit 4但是测试文件使用的测试版本还是5，没有跟随library的改动调整为4 |
| 计算凸包问题时无从下手 | 上B站看到了一个详细的介绍“卷包裹”算法的视频，受益匪浅，视频链接：  <https://www.bilibili.com/video/av59283385> |

# 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

1、熟悉了Java的语法以及各种重要的容器类的使用。

2、熟悉eclipse的使用。

3、了解了Junit的使用方法和意义。

4、了解了git版本控制工具的重要作用以及基本用法。

## 针对以下方面的感受

1. Java编程语言是否对你的口味？

Java用起来还是非常方便的，上手也很容易。

1. 关于Eclipse IDE

功能非常强大，代码补全功能也很好用，内置的git工具非常节省时间，能够很好提高效率。

1. 关于Git和GitHub

第一次使用这样的版本控制工具，确实对多次开发和团队协作有很重要的作用。

1. 关于CMU和MIT的作业

非常有趣，将实验本身融入到实际的例子中，做起来不是很枯燥但是确实可以学到很多东西，一次实验下来虽然很艰苦但是确实掌握了很多技巧和知识。

1. 关于本实验的工作量、难度、deadline

工作量较大，难度本身不大但是由于对新的工具不熟悉所以还是占用了很多的时间，DDL适中，实验任务完成起来很从容。

1. 关于初接触“软件构造”课程

软件构造确实是一门实践课程，需要靠实践来提高自己的水平，同时也对今后的工作研究实践有着很大的帮助。