

**2021年春季学期  
计算学部《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 周凡 |
| 学号 | 1190201008 |
| 班号 | 1903011 |
| 电子邮件 | [434696317@qq.com](mailto:434696317@qq.com) |
| 手机号码 | 18217207114 |

**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() 2](#_Toc71131538)

[3.1.2 generateMagicSquare() 4](#_Toc71131539)

[3.2 Turtle Graphics 6](#_Toc71131540)

[3.2.1 Problem 1: Clone and import 6](#_Toc71131541)

[3.2.2 Problem 3: Turtle graphics and drawSquare 7](#_Toc71131542)

[3.2.3 Problem 5: Drawing polygons 8](#_Toc71131543)

[3.2.4 Problem 6: Calculating Bearings 10](#_Toc71131544)

[3.2.5 Problem 7: Convex Hulls 10](#_Toc71131545)

[3.2.6 Problem 8: Personal art 12](#_Toc71131546)

[3.2.7 Submitting 12](#_Toc71131547)

[3.3 Social Network 13](#_Toc71131548)

[3.3.1 设计/实现FriendshipGraph类 13](#_Toc71131549)

[3.3.2 设计/实现Person类 14](#_Toc71131550)

[3.3.3 设计/实现客户端代码main() 15](#_Toc71131551)

[3.3.4 设计/实现测试用例 16](#_Toc71131552)

[4 实验进度记录 17](#_Toc71131553)

[5 实验过程中遇到的困难与解决途径 18](#_Toc71131554)

[6 实验过程中收获的经验、教训、感想 18](#_Toc71131555)

[6.1 实验过程中收获的经验和教训 18](#_Toc71131556)

[6.2 针对以下方面的感受 18](#_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。

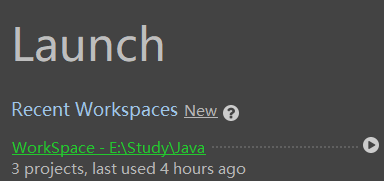
# 实验环境配置

根据Lab0中的指导安装JDK、Eclipse和Git。

在为JDK配置环境变量时，在系统环境变量中新建了一个名为“JAVA\_HOME”的环境变量，其值为JDK安装路径，然后在Path的环境变量中添加一个“%JAVA\_HOME%bin”的变量值。



安装完成Eclipse后，在“E:\Study\Java”路径中创建Workspace。



URL地址：https://github.com/ComputerScienceHIT/HIT-Lab1-1190201008.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

要求1：

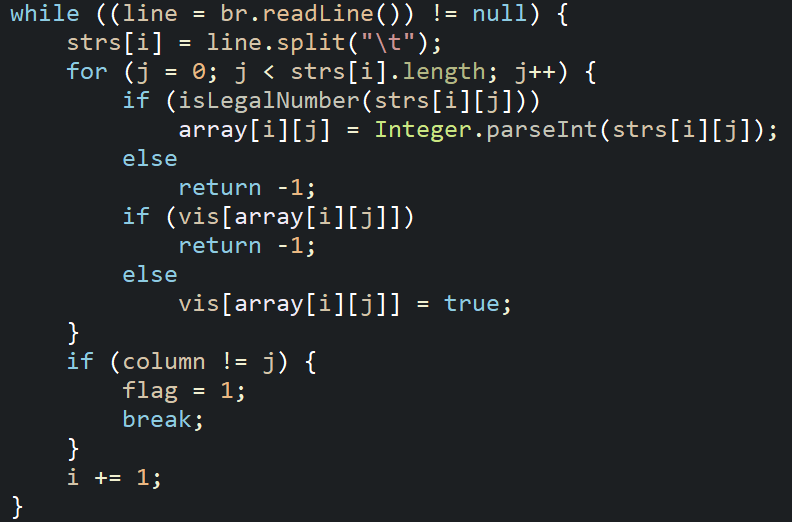
编写isLegalMagicSquare方法判断所给的矩阵是否为Magic Square，Magic Square满足以下条件：存在一个常数n，使得矩阵的每行、每列、两条对角线上的数之和分别等于n，同时也隐性地说明该矩阵的长和宽相等。

要求2：

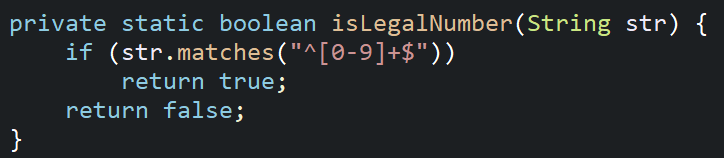
理解generateMagicSquare方法生成Magic Square的原理，并找出其中存在的缺陷：无法生成长宽为偶数的Magic Square。

### isLegalMagicSquare()

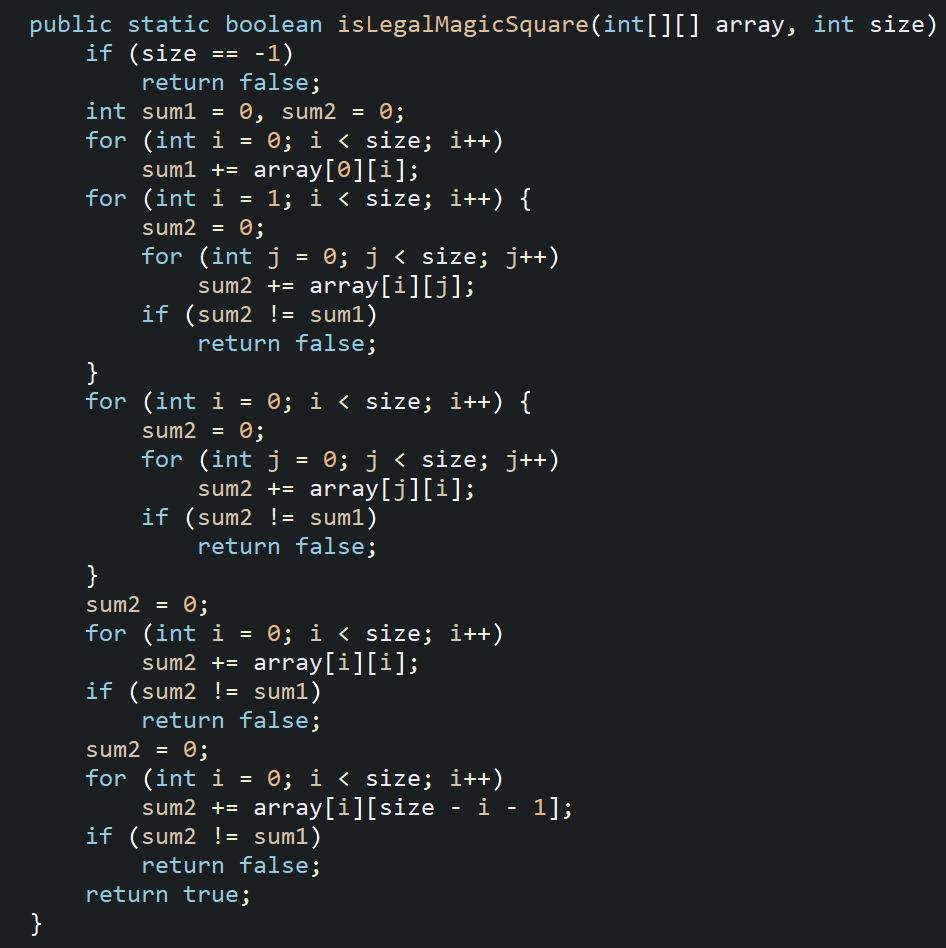
（1）首先编写了readFile方法从文件中读入矩阵，其中利用do-while循环不断将文件每行读入并根据“\t”将其分为多个元素，再存入String数组strs中。利用isLegalNumber函数判断二维数组strs中的每个元素是否合法，若不合法则返回-1以示异常结束，否则将其转换为整型并存入array数组中。接下来判断刚存入的数是否有重复，由于数据规模不大，此处利用了一个桶进行判断，若重复则同样返回-1以示异常结束，否则将桶中该数标记为true。每处理完一行，j都记录了该行元素的个数，即列数，将其与记录了第一行元素个数的column比较（第一行先用了一个if语句进行了do-while语句中类似的处理），不相等则该矩阵不合法，返回-1以示异常结束，否则继续。最终，若文件中是一个合法矩阵，则函数会返回矩阵的size。



（2）编写isLegalNumber函数用于判断输入的合法性，利用正则表达式“^[0-9]+$”判断str是否全是由0-9构成的串，是则返回true，否则返回false。



（3）编写isLegalMagicSquare方法，其输入为readFile函数最终将文件矩阵转化得到的整型二维数组array[][]以及矩阵的尺寸size。首先，若size==-1，说明在readFile中出现异常，文件输入不合法，返回false，否则说明文件输入合法。先将矩阵第一行求和得到一个基准值sum1，接下来将剩余行依次求和得到sum2，比较sum1和sum2，仅当两者相等程序继续，否则返回false。然后，在依次求每列的和，两条对角线的和，并和sum1进行比较，仅当两者相等程序继续，否则返回false。若程序进行完都有sum1==sum2，则说明该矩阵为Magic Square，返回true。



### generateMagicSquare()

（1）函数流程图如图3.1.2.1，其实现原理为从第0行中间的数开始赋值，值从1一直递增到矩阵的面积，确保矩阵元素的不重复性，并且方向是沿左下到右上斜对角线方向赋值，确保了矩阵每行、每列、两条对角线之和均相等。赋值顺序是从初始化的点开始，下一个点在上一个点的右上方。若已经到最上一行或最右一列，则下一个点取最下一行或最左一列，另一个坐标照常+1。

（2）n为负数报错是因为第一个点坐标（0, n/2）会出错；n为偶数会报错是因为此时，必然有一条斜线完全赋值之后，其结束于（x, n-1），接下来会执行row++使得row==n，再次执行magic[row][col]就会出现溢出的现象，导致报错。

（3）对函数的扩展：首先在开始判断n是否为正奇数，确保函数不会出现（2）中的错误。最后对生成的magic矩阵将其逐个写入文件6.txt，每个元素用“\t”隔开，每行间使用“\n”。

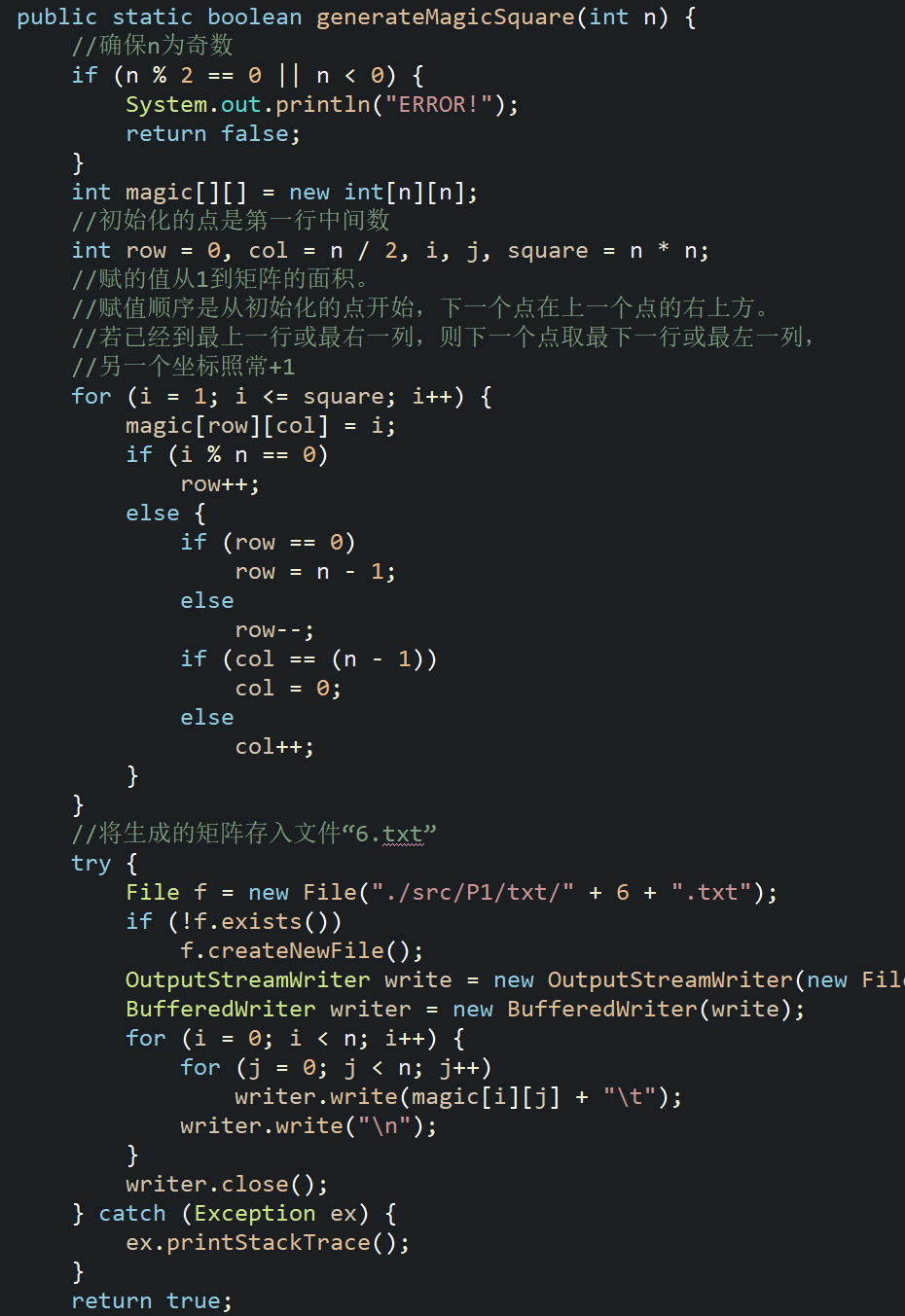


图3.1.2.1 对函数的注释以及扩展

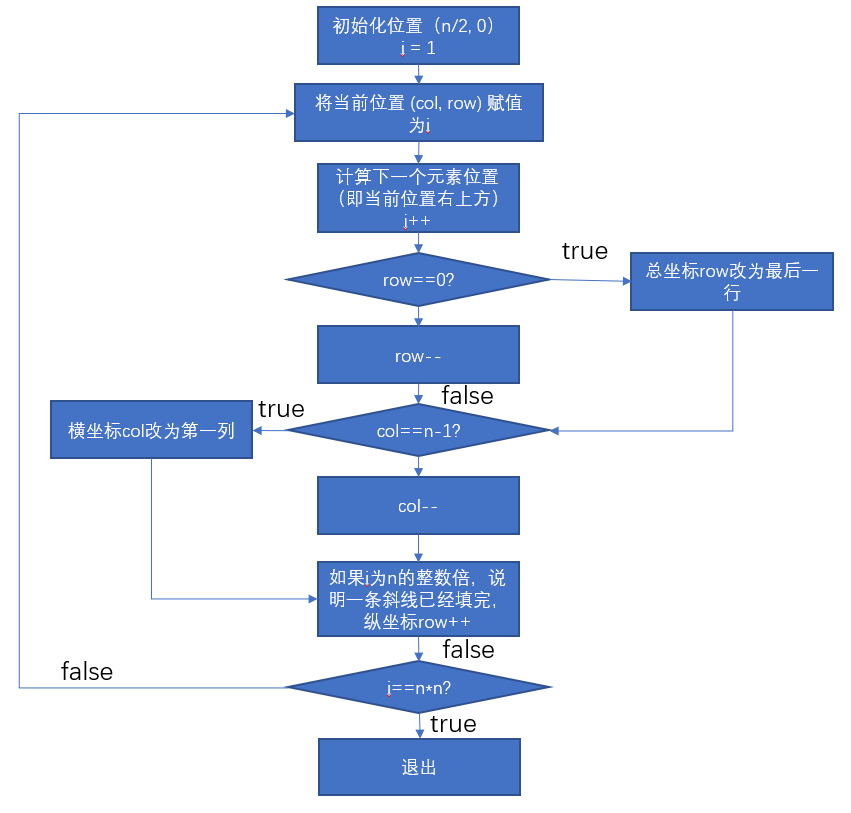


图3.1.2.2 函数流程图

## Turtle Graphics

要求1：从github上获取任务代码，在本地创建git仓库，并使用git管理本地开发。

要求2：利用turtle包中已经实现的实现的工具绘制正方形。

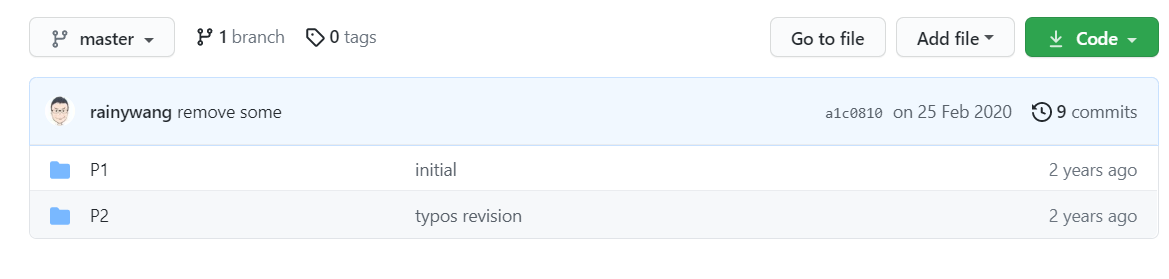
要求3：正多边形内角、外角、边数的求解和绘制的实现。

要求4：计算点到点的旋转角。

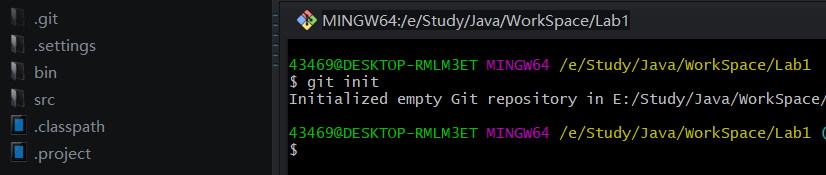
要求5：求凸包。

### Problem 1: Clone and import

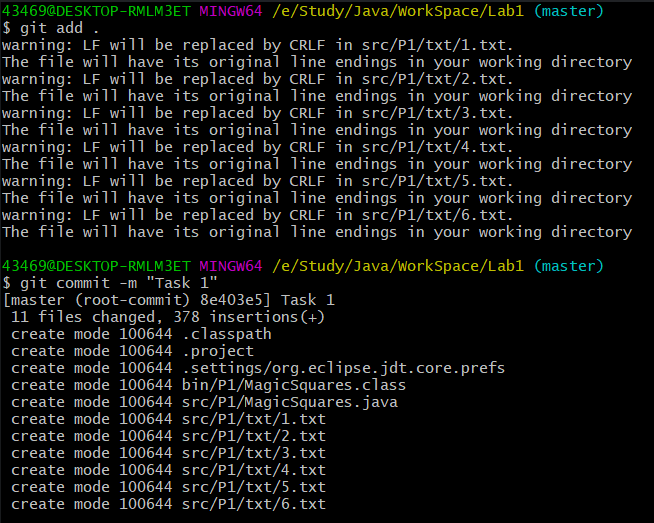
（1）从GitHub获取该任务的代码



（2）在本地创建git仓库：在E盘中创建repository文件夹，在文件夹中初始化git。

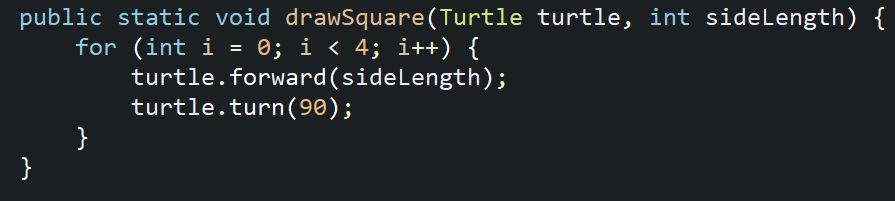


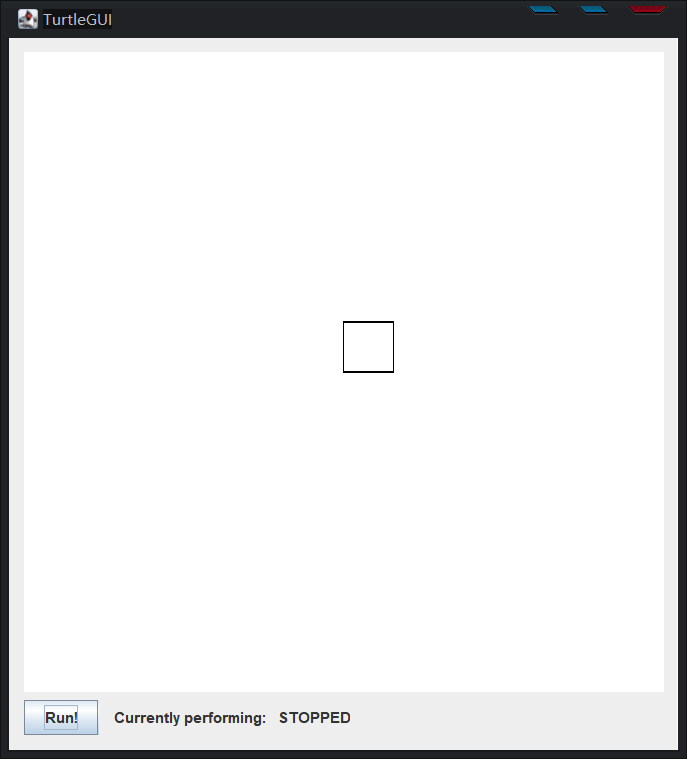
（3）使用git管理本地开发：将任务1的文件提交至git仓库当中，并进行注释。



### Problem 3: Turtle graphics and drawSquare

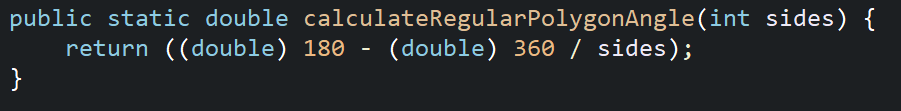
每次先走sideLength的长度，即正方形边长，再转90度，循环4次这样的操作即可得到一个以sideLength为边长的正方形。



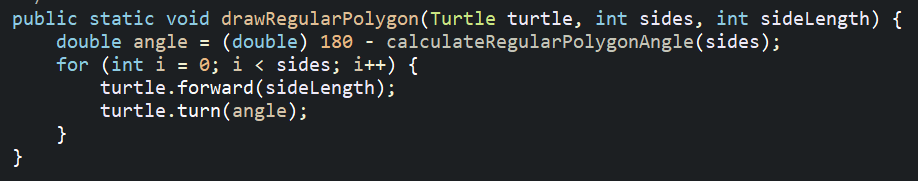


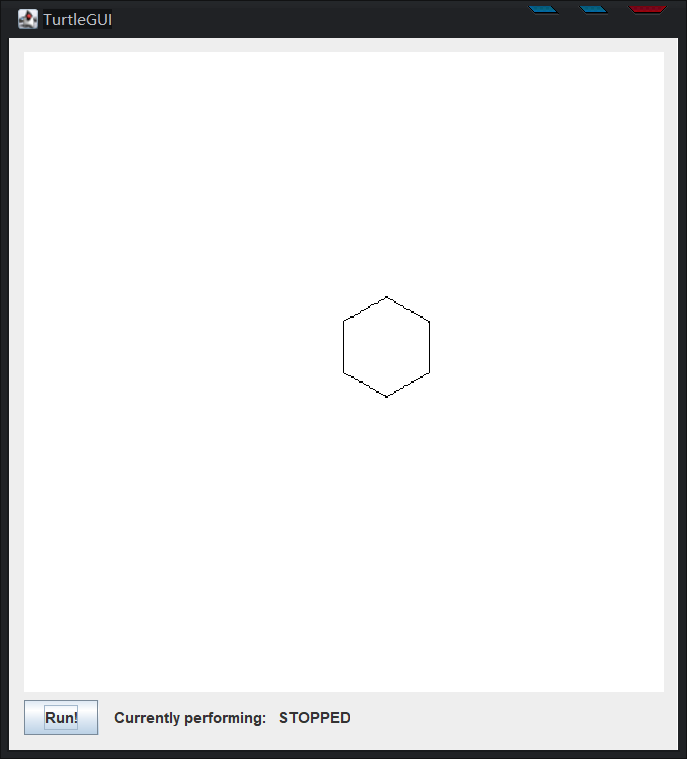
### Problem 5: Drawing polygons

（1）实现通过边数计算内角角度的calculateRegularPolygonAngle方法：由于正多边形外角和为360度且都相等，所以其外角为360/sides，故内角为180-360/sides。

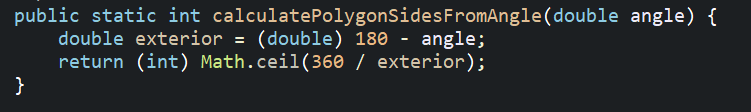


（2）实现绘制正多边形的drawRegularPolygon方法：对于绘制正n边形，每次先走sideLength的长度，然后旋转的角度应当等于外角，即180 - 内角（内角由（1）中函数计算得到），然后重复该操作n次即可。

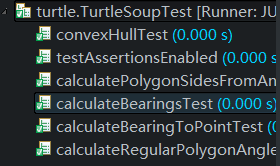




（3）实现通过角度计算边数的calculatePolygonSidesFromAngle方法：由（1）中内角度数angle和边数sides的关系可以推得sides = 360/(180-angle)，但考虑到double计算的误差，需要向上取整。

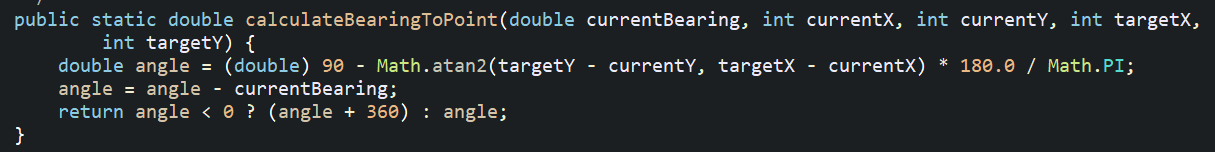


（4）测试结果

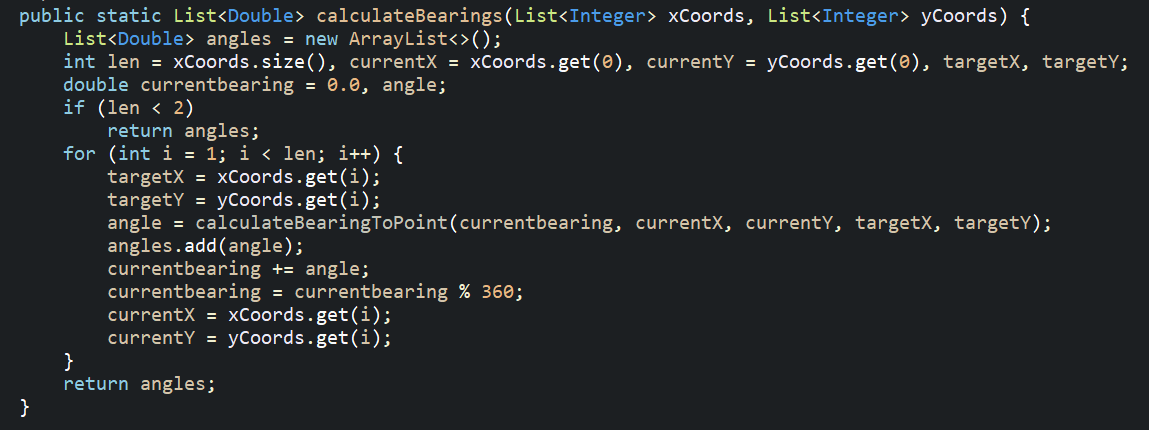


### Problem 6: Calculating Bearings

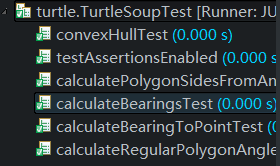
（1）实现calculateBearingToPoint方法：该方法的功能为计算当前点朝向与当前点到目标点的朝向之间的夹角。首先利用atan2函数计算处当前点与目标点构成的方向与x轴正向夹角，然后计算出其与y轴夹角，再与当前朝向作差，最后需要确保值在[0, 360)区间内。



（2）实现calculateBearings方法：利用for循环对列表中的每两个相邻点调用calculateBearingToPoint方法计算夹角，并保存到angles，最后返回angles。



（3）测试结果



### Problem 7: Convex Hulls

实现计算凸包的convexHull方法：

（1）首先找到点集中最左（若有多个区最下方）的点，显然其必在凸包中，以它为起点。

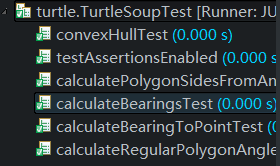


（2）然后从起点对其他点进行遍历，利用calculateBearingToPoint方法计算每个转向角，其中转向角最小的那个点必在凸包中，若有多个点转向角最小，则取离起点最远者，将其加入convexHull中。

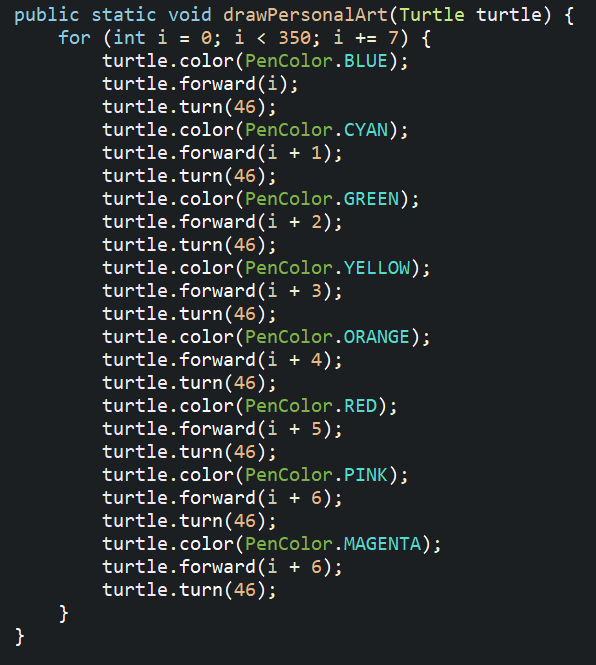


（3）最终，当下一点即为最初的起点时，说明线路已经实现回路，则此时的convecHull即为所求凸包。

（4）测试结果

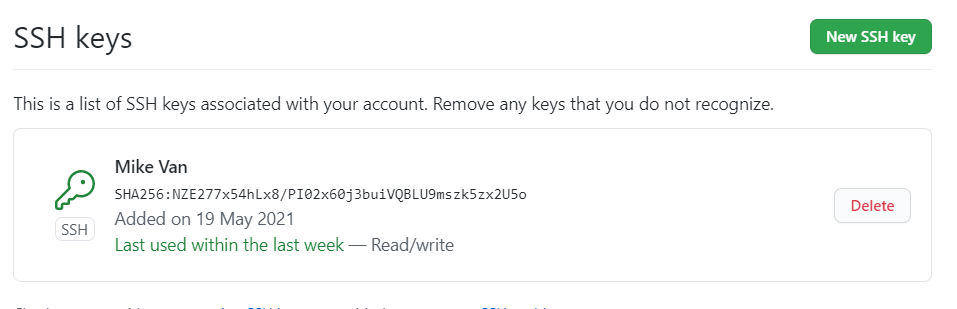


### Problem 8: Personal art

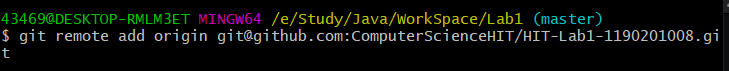
 

### Submitting

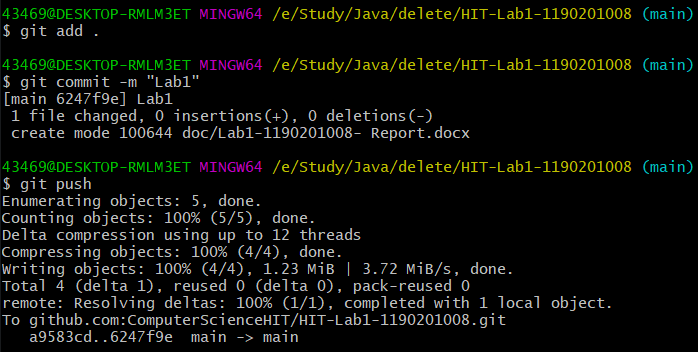
（1）首先在github账户中添加公用秘钥。



（2）连接到远程仓库



（3）将文件上传到本地仓库，再push到远程仓库。



## Social Network

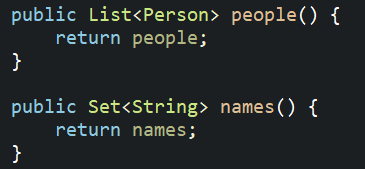
实现Person类和FriendshipGraph类，通过这两个类实现人际关系网的建立，并实现求出两人之间最短距离的算法。

### 设计/实现FriendshipGraph类

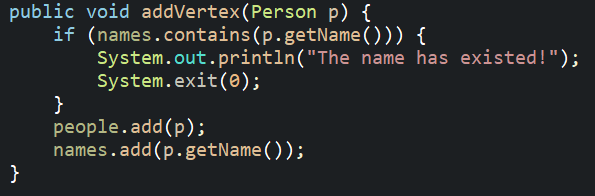
（1）首先申明两个类，people存储人际关系图中所有的人的信息，names存储所有人的名字（默认所有人名字不重复）。



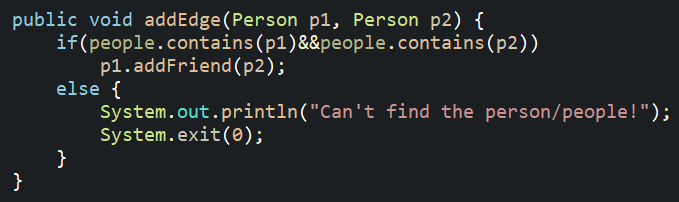
（2）通过方法people和names读取类中的people和names。



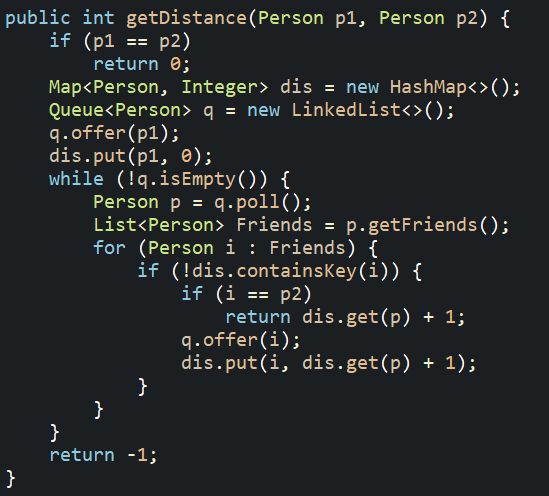
（3）实现addVertex方法：首先检查p的人名是否已经存在于图中，存在则结束程序，否则分别在paople和names中分别添加p的信息。



（4）实现addEdge方法：首先判断p1和p2是否都在图中，若有一个不在，则结束程序，否则添加一条从p1到p2的边（这是有向图）。



（5）实现求两人之间最短距离的getDistance方法：首先要考虑p1和p2是同一个人的情况。此处使用map构造一个Person到Integer的映射，这里的Integer为Person到p1的距离，然后申明一个队列，使用广度优先遍历，对同一深度的Person，都映射到同一个整型值，即到p1的距离，下一个深度距离+1，直到遍历到p2。

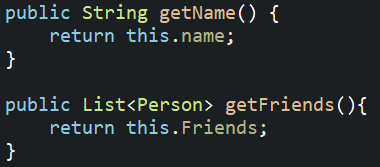


### 设计/实现Person类

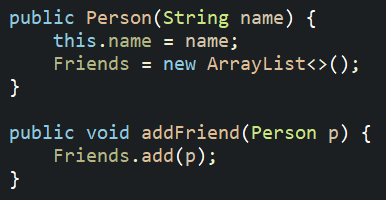
（1）首先，申明name存储人名，Friends存储从以该Person为起点的边的终点。



（2）通过getName函数和getFriends函数调用name和Friends。

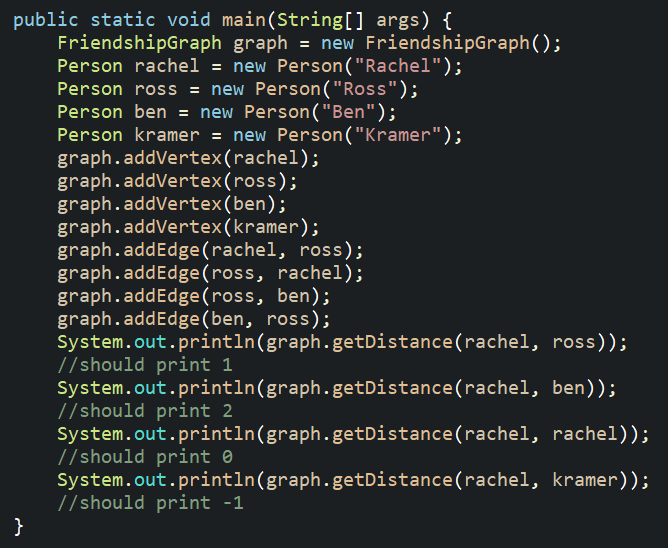


（3）Person中添加name和Friends的实现。



### 设计/实现客户端代码main()

（1）根据实验手册得到客户端代码：



（2）测试结果：

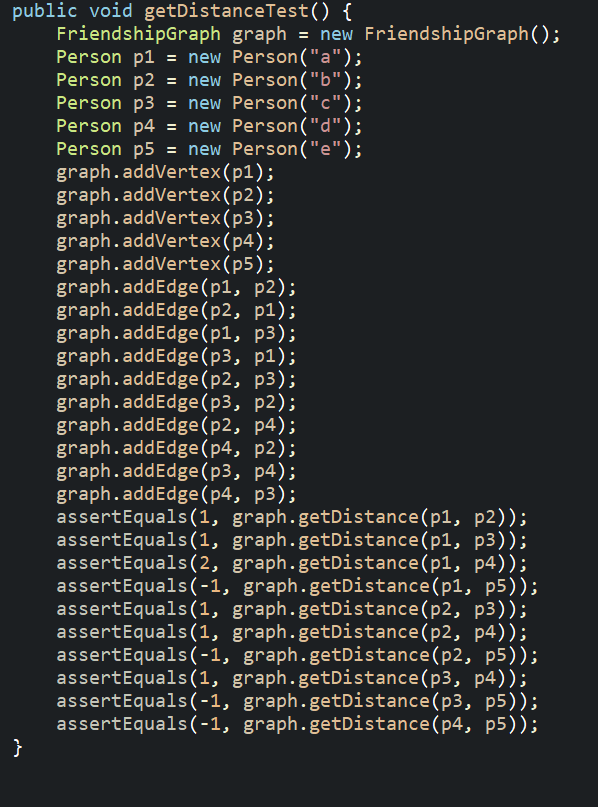


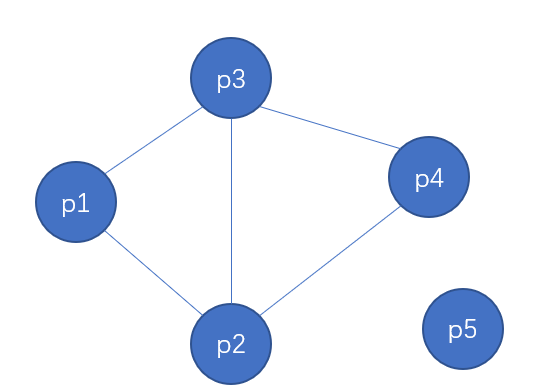
### 设计/实现测试用例

（1）对addVertex方法和addEdge方法的测试：



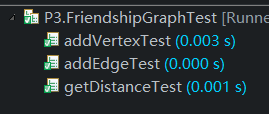
（2）对getDistance方法的测试（人际关系图为3.3.4.1）：





3.3.4.1 人际关系图

（3）测试结果：



# 实验进度记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 任务 | 实际完成情况 |
| 2021-05-11 | 18:30-20:30 | 编写问题1的isLegalMagicSquare函数并进行测试，实现了基本功能，通过了例1-例3，但暂时无法应对异常情况，例4、例5还未解决。 | 遇到困难，未完成 |
| 2021-05-12 | 15:00-16:30 | 完善了问题1的isLegalMagicSquare函数，解决了异常情况的处理，通过了例4、例5的测试；修改了generateMagicSquare函数，使其达到了题目要求。 | 按计划完成 |
| 2021-05-15 | 8:00-11:30 | 编写了问题2的drawSquare函数、calculateRegularPolygonAngle函数、calculatePolygonSidesFromAngle函数、drawRegularPolygon函数、calculateBearingToPoint函数、calculateBearings函数并进行了测试，通过了测试样例。 | 按计划完成 |
| 2021-05-15 | 15:00-17:00 | 编写了问题2的convexHull函数和 drawPersonalArt函数并进行了测试，通过了测试样例。 | 按计划完成 |
| 2021-05-16 | 18:30-20:00 | 编写了Person类和FriendshipGraph类，实现了问题3的addVertex函数、addEdge函数、getDistance函数并测试了题中所给的样例，与题目要求结果一致，同时编写了FriendshipGraphTest测试函数，测试结果正确。 | 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 |
| 任务1中，对矩阵中非法输入的判断和处理，如小数等。 | 利用字符串中的matches方法以及正则表达式"^[0-9]+$"判断读入的字符串是否能转化为整型。 |
| 任务2中，在计算转角时可能出现角度小于0或者大于360的情况。 | 当角度小于0时，添加angle += 360；并添加 angle %= 360式angle < 360。 |
| 任务3中，对getDistance方法的实现。 | 利用Map构造Person到距离的映射，然后使用广度优先遍历，对同一深度的Person距离相等，下一层深度距离+1，直到遍历到目标Person为止。 |

# 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

由于对Java语言比较陌生，对其中的方法、数据结构等都非常不了解，实验前期走了很多弯路。不过，这次实验让我对Java语言有了一个大致的了解，也让我感受到了它与C语言的不同。

## 针对以下方面的感受

1. 相比C语言，感觉Java编程更加方便，但是我对其底层机制还不够了解，不知道如何不优化语言可以提高运行效率。
2. 对于Eclipse，感觉这是一个功能比较齐全的IDE，在编程时为我提供了不少的方便。
3. 对于Git和GitHub是第一次使用，还不是很熟练，但是在实验过程中对其管理文件版本的功能还是印象深刻。
4. 此次实验工作量、难度都适中。
5. 初接触“软件构造”课程，感觉课程内容有点抽象，但是上手Java并使用Git之后，对老师讲解的内容就有了更好的理解。