

**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 | 刘璟烁 |
| 学号 | 1180500301 |
| 班号 | 1803001 |
| 电子邮件 | 907599581@qq.com |
| 手机号码 | 13102886813 |

**目录**

[1 实验目标概述 1](#_Toc29325521)

[2 实验环境配置 1](#_Toc29325522)

[3 实验过程 1](#_Toc29325523)

[3.1 Magic Squares 1](#_Toc29325524)

[3.1.1 isLegalMagicSquare() 1](#_Toc29325525)

[3.1.2 generateMagicSquare() 1](#_Toc29325526)

[3.2 Turtle Graphics 1](#_Toc29325527)

[3.2.1 Problem 1: Clone and import 2](#_Toc29325528)

[3.2.2 Problem 3: Turtle graphics and drawSquare 2](#_Toc29325529)

[3.2.3 Problem 5: Drawing polygons 2](#_Toc29325530)

[3.2.4 Problem 6: Calculating Bearings 2](#_Toc29325531)

[3.2.5 Problem 7: Convex Hulls 2](#_Toc29325532)

[3.2.6 Problem 8: Personal art 2](#_Toc29325533)

[3.2.7 Submitting 2](#_Toc29325534)

[3.3 Social Network 2](#_Toc29325535)

[3.3.1 设计/实现FriendshipGraph类 2](#_Toc29325536)

[3.3.2 设计/实现Person类 2](#_Toc29325537)

[3.3.3 设计/实现客户端代码main() 2](#_Toc29325538)

[3.3.4 设计/实现测试用例 3](#_Toc29325539)

[4 实验进度记录 3](#_Toc29325540)

[5 实验过程中遇到的困难与解决途径 3](#_Toc29325541)

[6 实验过程中收获的经验、教训、感想 3](#_Toc29325542)

[6.1 实验过程中收获的经验和教训 3](#_Toc29325543)

[6.2 针对以下方面的感受 3](#_Toc29325544)

# 实验目标概述

在浏览完实验任务后，总结如下：

①这次实验以求解问题的方法训练包括利用JavaOO编程开发基本功能模块、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等来训练Java基本编程技能。

②同时，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

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。

1. 配置过程：

首先，安装JDK13作为java的开发环境，（但课程要求java8，因此每次进行实验任务时应注意将项目的compiler改为1.8版本）

其次，安装版本控制工具Git，由于之前未接触过版本控制工具，在安装后无所适从，为加深对其的了解与操作方法，我阅读了老师上传的Lab0中相关的Git手册，了解了分布式版本控制系统的概念与优点，并学习如何进行追踪、暂存、commit等命令行操作及其意义，同时将本地Git与我的Github相关仓库关联。

安装Eclipse作为IDE。（由于eclipse中自带Junit，无需另行安装Junit）

在这里给出你的GitHub Lab1仓库的URL地址（Lab1-学号）。

https://github.com/ComputerScienceHIT/Lab1-1180500301.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

在这里简要概述你对该任务的理解。

该实验任务通过对幻方矩阵的判断以及生成，帮助我们进一步学习java语言的基础用法，除了复习常见的数组、各种控制结构等，还应用了异常、调用对象方法、打开文件等用法。

### isLegalMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

3.1.1.1设计和实现思路：

首先，需要先将txt文档中字符串矩阵逐行读入，并将其存入数组中，过程中若出现非整数、负数、行列数目不相等或者未按tab分隔的情况，会报告错误，认为不是合法幻方矩阵。

其次，判断一个数字方阵是否为幻方矩阵，需要分别计算 其行和、列和、两条对角线和是否均相等。我为此实现了对应的三个函数分别用于计算三者。如果都相等，则可以认为是合法幻方矩阵。

3.1.1.2过程

a.最初，由于对java语言不够熟悉，并未掌握从文件读入数据的实现方法，遂从相关课程（浙大的javaMOOC）中了解到相关方法（同时也掌握了向文件写出的方法）：

读入：

BufferedReader in =**new** BufferedReader(

**new** InputStreamReader (

**new** FileInputStream(filename)));

PrintWriter out =**new** PrintWriter(

**new** BufferedWriter(

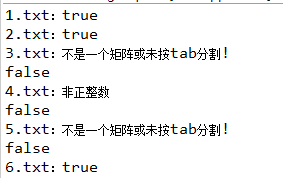
**new** OutputStreamWriter(

**new** FileOutputStream(filename))));

b.随后调用readline方法循环逐行读入，并按tab分割，调用方法将分割后生成的字符串数组转为数字数组，其中若出现非正整数或未按tab分割，则报错。同时，循环过程中用一变量保存读入的函数，判断其与列数是否相等，若不相等也会报错。

c.上述之后，得到合法正整数方阵，分别实现了设计思想中提到的三个方法，判断是否全相等，若相等返回true否则返回true。

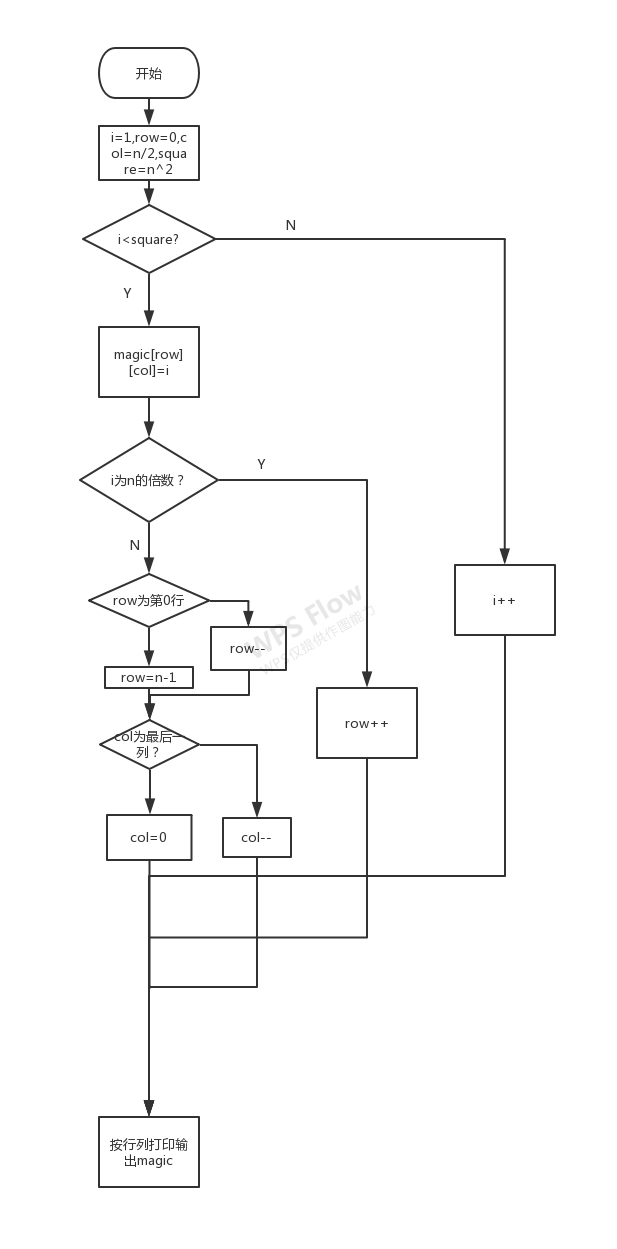
3.1.1.3结果



### generateMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

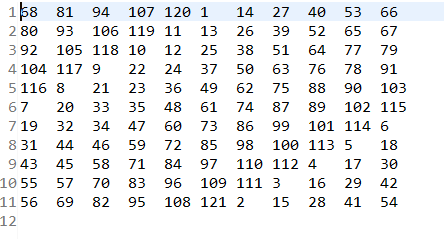
3.1.2.1流程图如下：



另外，对于第一个异常ArrayIndexOutOfBoundsException，官方说明为：抛出该错误指示数组已使用非法索引访问。索引为负或大于或等于数组的大小。当行数为偶数时，在判断i是否为n的倍数时，row可能已经为最后一行，此时row++

，下次复制数组越界访问；对于第二个异常NegativeArraySizeException，官方说明：如果应用程序尝试创建负大小的数组，则抛出该异常。

3.1.2.2结果：生成幻方矩阵



## Turtle Graphics

在这里简要概述你对该任务的理解。

该任务主要关于turtle绘图方法的实现，同时也涉及到一些git使用方法的问题。该任务主要根据几何知识，坐标系的知识，通过javaOO编程实现相关的绘制。

### 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。

3.2.1.1获取任务代码：根据git官网指导，通过git clone url 指令将远程仓库中的代码克隆到你git仓库中

3.2.1.2本地创建仓库：

cd +想要设为仓库的目录；

git init创建仓库 随后可见目录中生成.git文件

3.2.1.3管理本地开发：git add +相关文集 进行暂存；git commit -m “commit messege”进行commit操作，git push origin master 将本地已经暂存的文件push到远程仓库的master分支中，其中我设置了origin对应的url为我的实验仓库url。

### Problem 3: Turtle graphics and drawSquare

设计思路：

这个任务实际上非常简单，我觉得任务的目的是为了帮助我们先去熟悉绘图基本操作，通过4次循环调用两个对象方法，绘制出一个简单的正方形。

实现：**for**(**int** i=0;i<4;i++)

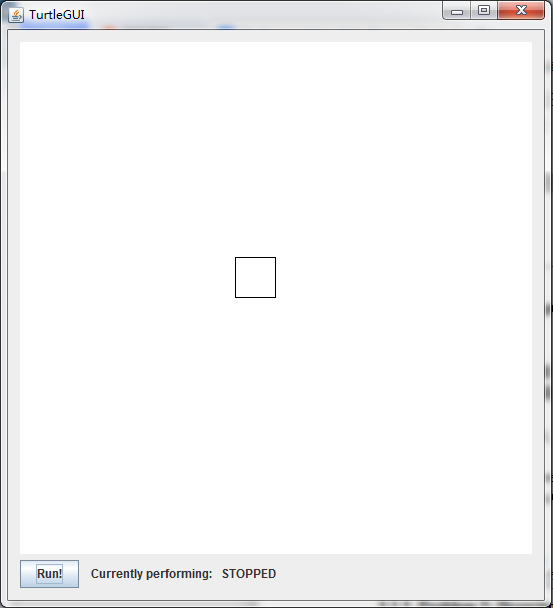
{

turtle.forward(sideLength);//sideLength为边长

turtle.turn(-90);

}

结果：

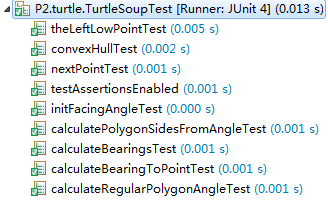


### Problem 5: Drawing polygons

3.2.3.1设计与实现思路：本任务在逻辑层面，只需要应用几何知识套用相关公式即可，主要是熟悉JUnit的测试方法。最后再drawregularpolygon函数循环调用上一函数。

3.2.3.2实现：在calculateRegularPolygonAngle函数中应用多边形内角和公式，返回。在drawregularpolygon中调用sides次上一函数，进行绘图。随后用对应test文件右键run as选择JUnit test即可查看测试结果，对应单元为绿色成功，若为蓝色或者红色，则说明有错误，单击框体下方出现错误相关trace可用于查找原因。

3.2.3.4结果：测试成功



### Problem 6: Calculating Bearings

3.2.4.1设计与求解思路：

1. 设计calculateBearingToPoint，该函数要求给定当前点及其面向角度，以及需要朝向的点，返回需要顺时针旋转的角度。我的想法是，求出两点的相对坐标，以当前点为原点，求出后者的相对坐标，随后根据Hints，利用atan2函数，求出后者的相对角度，但该角度为零的轴（x正向）与前面给的面向角度为零的轴(y正向）不同，且单位不同，因此需要转换坐标系并乘一个系数，使二者统一。随后根据大小关系，返回相应转动幅度。
2. 设计calculateBearings，输入为一组点的序列，要求返回一组对应的转向角度。思路较为简单，只需要循环调用上一函数即可，只需要处理一些传入空序列等的细节即可。

3.2.4.2实现:

1. 如上先求相对坐标：

**int** relativePositionX = targetX - currentX;

**int** relativePositionY = targetY - currentY;

随后调用atan2函数，并再次转换坐标系，转换单位：

**double** relatAngle = Math.*atan2*(relativePositionY, relativePositionX);

**double** traverRelatAngle =((relatAngle>=-(Math.***PI***))&&relatAngle<=((Math.***PI***)/2))?((Math.***PI***)/2-relatAngle):((5.0/2\*Math.***PI***)-relatAngle);

traverRelatAngle = traverRelatAngle\*(180.0/Math.***PI***);

随后判断大小关系，返回不同值：

**if**(traverRelatAngle >= currentBearing) //如果相对角度大于等于面向角度，直接相减即可

{

**return** traverRelatAngle-currentBearing;

}

**else** //如果小于 则相减后为负 加360即为转向角

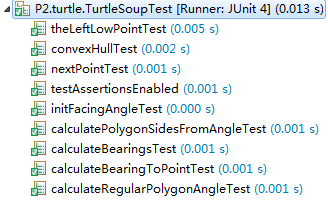
{

**return** 360.0+traverRelatAngle-currentBearing;

}

1. 对x与y序列不等长或者为空进行相应处理，正常情况下，循环调用n-1次上一函数，add入对应list返回。

3.2.4.3结果：测试成功



### Problem 7: Convex Hulls

3.2.5.1设计求解思路（连同对遇到的问题的思考与解决一同叙述）：该任务输入一组点，要求返回一组包含其所有点的最小子集。根据函数specification 的提示，我先去Wikipedia上搜索礼品包装算法，了解了大概思路，也就是先确定一个起始点（实现了对应方法），随后不断寻找下一个最外围的点（实现对应方法），直到下一个点为起始点。其中还需处理一个问题，对于0,1,2个点可直接返回点集，对于一条线上可能产生多个点的情况，我使用了全局变量来记录到下一个点是否有角度偏转，（虽然全局变量可能带来副作用，但我确实没想出好的方法）如果没有偏转，则有可能不应该存入输出点集中。

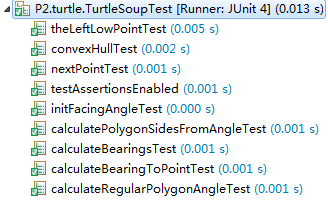
3.2.5.2实现：

首先，实现theLeftLowPoint方法，该方法输入原函数中输入的点集，找出输入坐标中x和y最小的点并返回。

随后，实现nextPoint方法，该方法通过遍历输入点集中除其以外的点，循环调用calculateBearingToPoint函数并记录相应的偏转角，找出有最小的偏转角的点，就是nextpoint返回，其中偏转角由两个点的坐标和我设置的全局变量facingAngle求出，确定下一点后更新facingAngle。

最后，主函数中处理0、1、2点集合，正常情况下调用theLeftLowPoint函数，并循环调用nextPoint函数，将符合条件的点加入返回点集中。

3.2.5.3结果：测试成功！



### Problem 8: Personal art

3.2.6.1设计与实现：通过循环调用*drawRegularPolygon函数，并设置不同的边长和边数，改变每个图形的颜色，画出对称的组合图形。*

3.2.6.2过程：两层循环、switch根据循环遍数变换画笔颜色：

**for**(**int** j=1;j<=2;j++)

{

**for** (**int** i =3;i<8;i++)

{

**switch**(i)

{

**case** 3:

turtle.color(PenColor.***BLUE***);

**break**;

**case** 4:

turtle.color(PenColor.***YELLOW***);

**break**;

**case** 5:

turtle.color(PenColor.***RED***);

**break**;

**case** 6:

turtle.color(PenColor.***MAGENTA***);

**break**;

**case** 7:

turtle.color(PenColor.***CYAN***);

**break**;

**default**:**break**;

}

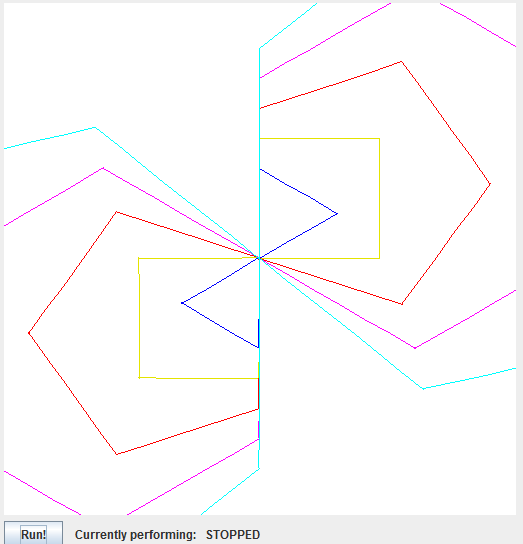
*drawRegularPolygon*(turtle, i, 30\*i);

}

turtle.turn(180);

}

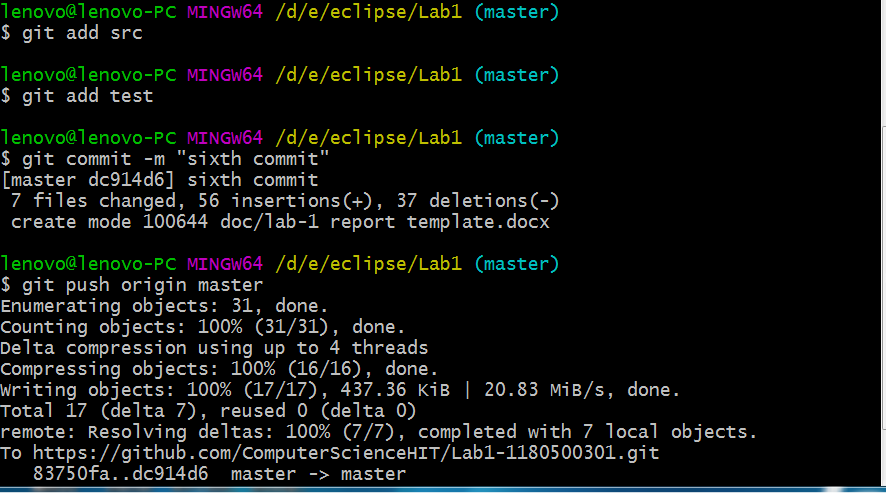
3.2.6.3结果：



### Submitting

如何通过Git提交当前版本到GitHub上你的Lab1仓库。

如3.2.1.3所讲，git add +相关文集 进行暂存；git commit -m “commit messege”进行commit操作，git push origin master 将本地已经暂存的文件push到远程仓库的master分支中，其中我设置了origin对应的url为我的实验仓库url。



## Social Network

在这里简要概述你对该任务的理解。

通过实现Person和FriendShip 两个类，进一步帮助我们理解Java面向对象编程和抽象编程的特点，更加熟练地应用Set、List、Map等容器。

### 设计/实现FriendshipGraph类

给出你的设计和实现思路/过程/结果。

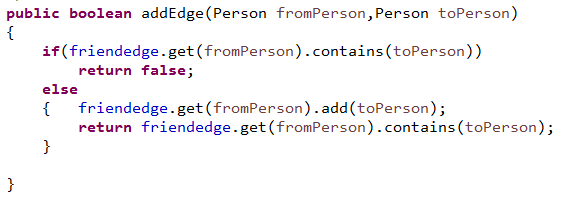
3.3.1.1设计和实现思路：根据实验要求，设计实现社交关系网，我认为该类可以分别应用集合以及HashMap成员变量来保存成员和成员的对应关系。addVertex方法向类的set中加入“点”（成员），addEdge方法向HashMap中以对应成员为键值get出的集合中加入另一成员；



关于getDistance函数：①该函数输入两个成员（可认为是图中的两个点），要求返回从前者到后者的最短路径长度。由于该图边上无权值，根据所学知识和提示，应用图的先广搜索，搜索后者位于第几层即可。当然也需要处理两者为同一点或者前后不连通的情况。②关于如何记录搜索到的点位于 “第几层”，我思考了很久，后来我想出并应用了一个实现方法，该方法有些复杂，因为为了实现它，我又在Person类中（下文会提到）加入了两个private成员变量及对应的添加、获取函数，这使两个类中的耦合性大了一些，但总归是实现了记录边数的功能。（后来，我在查询如何降低耦合性的过程中，看到一位17级学长的实验报告，他通过应用一个HashMap来记录对应点的边数，同时省去了耦合性增大的麻烦，我为此上交两个最终版本的实验作业，一个是完全是我自己设计解决思路的实验作业，另一份是在该任务中借鉴该学长思路的作业）我另外实现了DFS方法，用于图中的先广搜索，途中记录每个点是从哪个点找到的，若找到对应点返回true。在getDistance方法中调用该函数，若得到true则进行回溯，一直找到最初点，回溯次数就是最短边数。

3.3.1.2过程：

addEdge：为了方便测试的编写，我将该方法由原来的void型修改为boolean，便于返回一个真值判断是否执行成功。



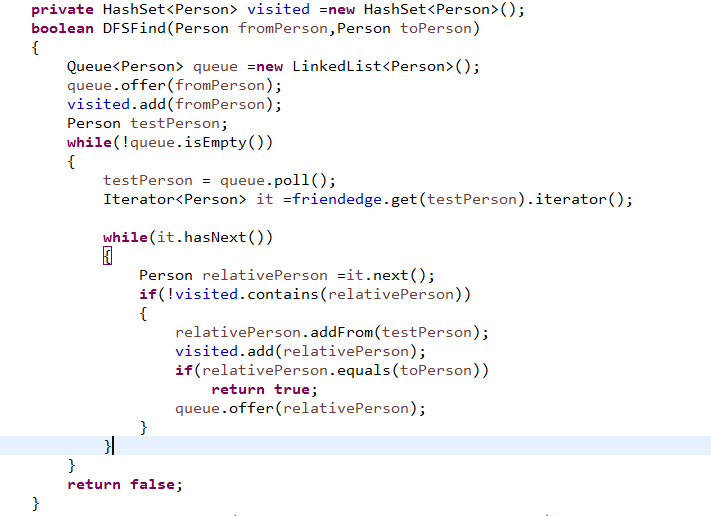
addVertex：主要也是类中集合内容的添加，以及相关初始化。最后返回仍是判断是否执行成功，这里不再展示。



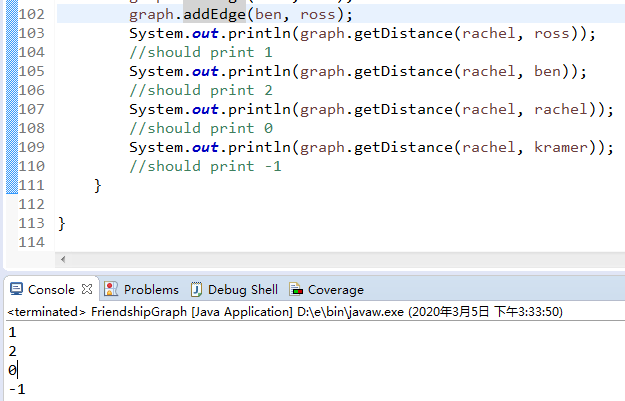
getDistance：调用了DFSFind函数进行先广搜索，使原函数更为简洁。若出现异常情况，返回0或-1。



DFSFind：套用上学期学习的先广搜索方法即可，只是应用了Iterator以及增加了从哪个成员来的记录。



3.2.1.3结果：成功



### 设计/实现Person类

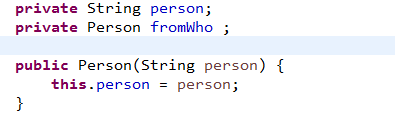
给出你的设计和实现思路/过程/结果。

3.3.2.1设计实现思路：

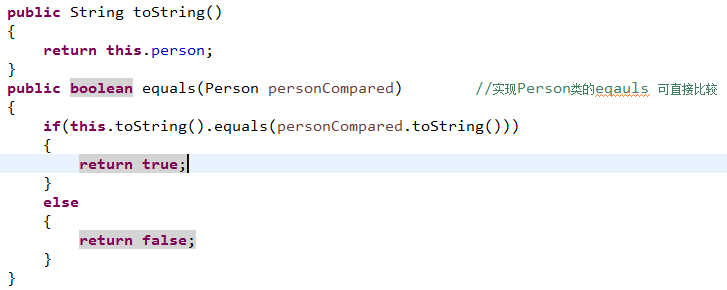
Person类应该为每个人设计属性，其中应含有成员变量保存其名字。同时为了方便操作，应该实现toString和equals函数。

3.3.2.2过程：

相应成员变量以及构造器：（fromWho为上文提到的记录来自的人的变量）



实现equals和toString：

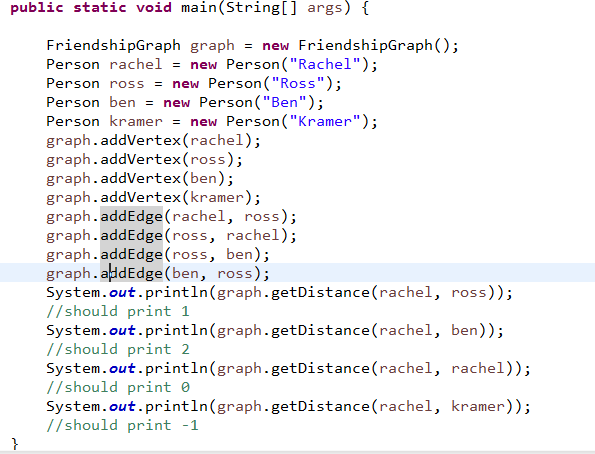


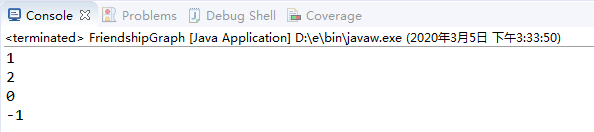
结果:成功应用。

### 设计/实现客户端代码main()

3.3.3.1设计思路：只需要声明一张关系图，声明几个人，在图中添加相应的点和边，并调用getDistance进行测试即可。

3.3.3.2过程及结果：





### 设计/实现测试用例

给出你的设计和实现思路/过程/结果。

3.3.4.1设计实现思路：

1. FriendShip类：由前面对方法的实现可知，addEdge和addVertex方法在操作成功时，都会返回true，因此对这两个方法的实现只需要加入几个样例判断并调用相关函数，判断返回值是否为true即可，getDistance函数和DFSFind方法测试比较相似，都是建立一个图，判断返回值。

②．Person类：由于Person主要实现人物属性的功能，类中的方法都简单而功能单一，因此测试用例普遍较少。

3.3.4.2过程及结果：

①FriendShip类：

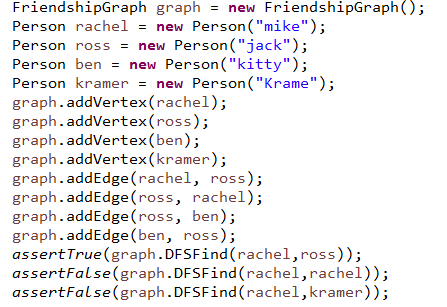
addEdge方法：



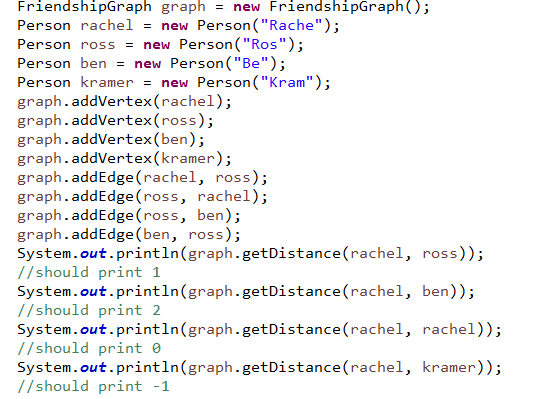
addVertex方法：



DFSFind方法：找到返回True否则False

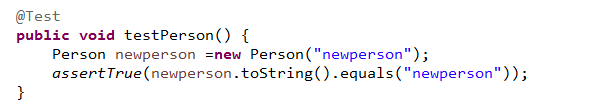


getDistance方法：

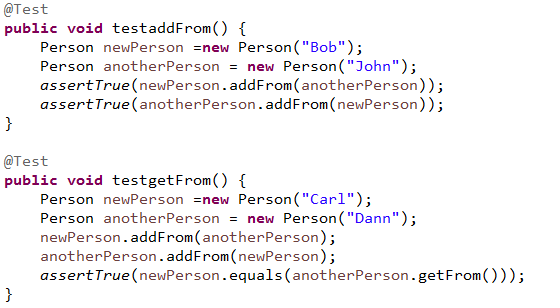


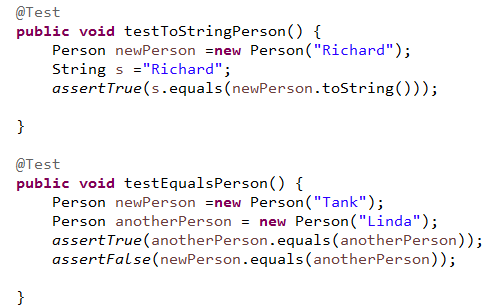
②Person类：

构造器的测试：故意申请同名，应该exit退出，因此测试会为蓝色而非绿色。

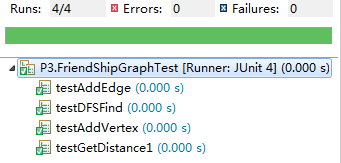


其他测试：

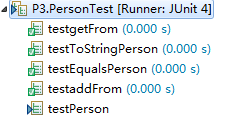




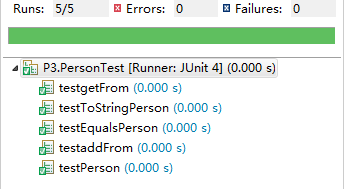
3.3.4.3结果：

①

②构造器testPerson故意触发exit因此为蓝色：



（若不故意触发，只测试能否新建用例则全绿：

）

# 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 |
| 2020-02-26 | 20:40-21:50 | 编写问题1的isLegalMagicSquare函数并进行测试 | 按计划完成 |
| 2020-02-27 | 9：00-10:00 | 分析并为generateMagicSquare花流程图 | 延时半小时 |
| 2020-02-27 | 10:30-12:00 | 完成3.2任务1 | 花了很久才掌握git的使用 |
| 2020-02-27 | 2:30-5:30 | 完成3.2任务2、4、5 | 延时一小多 |
| 2020-02-28 | 晚8:00-10:30 | 完成任务6、7 | 未按时完成，任务7较难 |
| 2020-02-29 | 上午 | 完成任务7、8 | 完成 |
| 2020-02-29 | 下午 | 完成3.3 | 未按时完成，任务难度比我想的大 |
| 2020-03-01 | 上午 | 完成3.3 | 按时完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的难点 | 解决途径 |
| 从文件中读入与写出到文件 | 学习java相关基础知识 |
| 不会遍历集合 | Java官网、博客学习Iterator用法 |
| DFS搜索记忆不清 | 复习数据结构课程，设计DFS方法 |
| 不会处理异常 | 对相应内容进行学习 |
| 不知道礼品包装算法 | Wikipedia搜索并学习 |

# 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

本次java实验，通过对一些方法、类的实现，我学习了java中文件读写、处理异常、遍历以及使用集合等技巧，同时也增强了对面向对象编程的理解。另一方面，我基本学会了Git、github、JUnit等工具的使用，这也极大方便我在日后对开发的练习。

当然，本次实验也收获了一些教训，比如对基础知识要掌握的牢固一些，以免使用时发憷，嘿嘿。

## 针对以下方面的感受

1. Java编程语言是否对你的口味？

是的，面向对象编程，相对于c方便了很多。

1. 关于Eclipse IDE

功能很多，方便上手使用。

1. 关于Git和GitHub

第一次学习使用Git，以前只是听说很牛，如今学会使用很有成就感！

1. 关于CMU和MIT的作业

二者的作业都难度适中，虽然模块多，但每个模块都不是难得思考不出来的程度，每完成一个模块，就会很有成就感！真的很喜欢。

1. 关于本实验的工作量、难度、deadline

工作量以及难度都可以接受，后面的几个任务难度有所提升但都可以通过思考来解决。Deadline合适。

1. 关于初接触“软件构造”课程

目前仅仅是熟练了一些java编程和测试方法，对课程有信心也很有兴趣，期待学习更多酷而且容易理解的知识。