

**2022年春季学期  
计算学部《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 吕文杰 |
| 学号 | 120L021110 |
| 班号 | 2003010 |
| 电子邮件 | 960836322@qq.com |
| 手机号码 | 15732921102 |

**目录**

[1 实验目标概述 1](#_Toc97648154)

[2 实验环境配置 1](#_Toc97648155)

[3 实验过程 1](#_Toc97648156)

[3.1 Magic Squares 1](#_Toc97648157)

[3.1.1 isLegalMagicSquare() 1](#_Toc97648158)

[3.1.2 generateMagicSquare() 1](#_Toc97648159)

[3.2 Turtle Graphics 1](#_Toc97648160)

[3.2.1 Problem 1: Clone and import 2](#_Toc97648161)

[3.2.2 Problem 3: Turtle graphics and drawSquare 2](#_Toc97648162)

[3.2.3 Problem 5: Drawing polygons 2](#_Toc97648163)

[3.2.4 Problem 6: Calculating Bearings 2](#_Toc97648164)

[3.2.5 Problem 7: Convex Hulls 2](#_Toc97648165)

[3.2.6 Problem 8: Personal art 2](#_Toc97648166)

[3.2.7 Submitting 2](#_Toc97648167)

[3.3 Social Network 2](#_Toc97648168)

[3.3.1 设计/实现FriendshipGraph类 2](#_Toc97648169)

[3.3.2 设计/实现Person类 2](#_Toc97648170)

[3.3.3 设计/实现客户端代码main() 2](#_Toc97648171)

[3.3.4 设计/实现测试用例 3](#_Toc97648172)

[4 实验进度记录 3](#_Toc97648173)

[5 实验过程中遇到的困难与解决途径 3](#_Toc97648174)

[6 实验过程中收获的经验、教训、感想 3](#_Toc97648175)

[6.1 实验过程中收获的经验和教训（必答） 3](#_Toc97648176)

[6.2 针对以下方面的感受（必答） 3](#_Toc97648177)

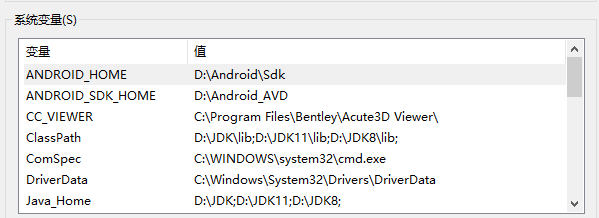
# 实验目标概述

实验的主要目标是通过解决三个问题，掌握基本的Java编程技能，包括能够利用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。此外，还要掌握Git的基本使用方法。

# 实验环境配置

1. 从各个官网下载JDK、Eclipse、Git的安装包并成功安装。
2. 配置JDK环境。

编辑计算机中DK所需系统变量。之前已经加入过其他版本的路径，只需加入新的路径即可。



1. Eclipse学习与配置

使用Eclipse创建Java项目的过程与之前使用的编译器大抵相同。Java的基本操作在实验前已经有所掌握。而对于Junit，只需在Eclipse中导入包即可。Junit的练习使用过程不再赘述。

1. Git初步使用

查看Git的学习手册和其他来源的教程，熟悉Git的基本操作。在向自建远程仓库中push本地内容时时常报错，如下图：



上传到实验仓库时也会有这样的报错。在CSDN上寻求帮助后得知，这可能是因为网络问题没能成功连接到 GitHub 的仓库。间隔一段时间后多次重试即可正常连接。其他基本操作练习不再赘述。

1. 我的GitHub Lab1仓库的URL地址：

https://github.com/ComputerScienceHIT/HIT-Lab1-120L021110.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

首先，编写函数检测从文件中读入的数据是否满足Magic Square的要求。其次，阅读已有的产生Magic Square的函数并补全，再使用先前的检测函数对输出的文件再次操作。

### isLegalMagicSquare()

思路：

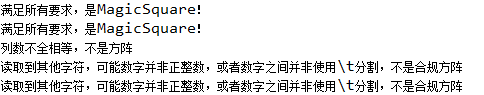
通过FileReader类打开并依次读入文件中内容，先默认文件内容满足条件并录入到一个二位数组中，在此过程中检测基本格式是否确实满足。存入完毕后，再验证具体的数据是否满足其他性质以进一步判断是否是Magic Square。

过程：

1. 实例化一个FileReader对象，调用read函数依次读取字符并处理。Read返回的是字符的ASCII码值，读取完毕则返回-1，因此可以通过值的大小判断字符是什么，文件是否读取完毕。是数字则处理后存入数组，是\t则表示该数字读入完毕，是\n则表示本行结束。读取过程中记录行数和列数，在第二行之后的读取中判断行数是否相等，在读取结束后判断行数与列数是否相等。
2. 使用多个循环和条件判断语句对已存入数组中的数据做处理，判断是否满足行、列、对角线之和相等的性质。满足条件则返回true，反之返回false，每次返回之前输出本次判定结束的原因和结果。

结果：

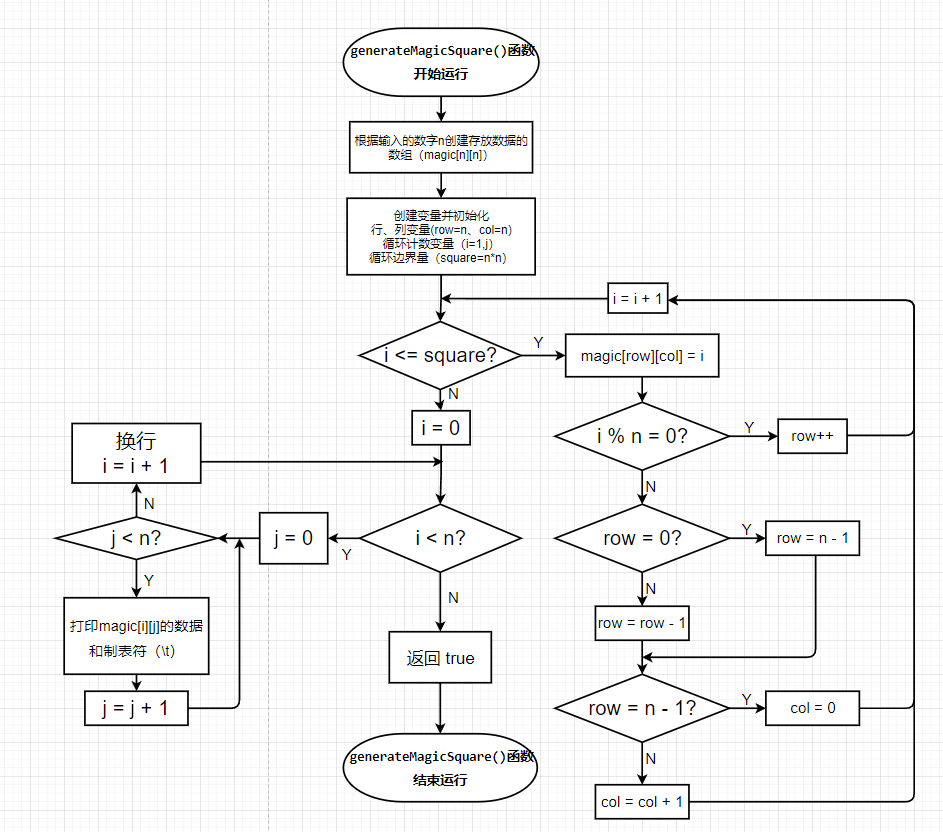
函数能够正常读入5个文件的数据并返回正确结果。输出的结果如下：



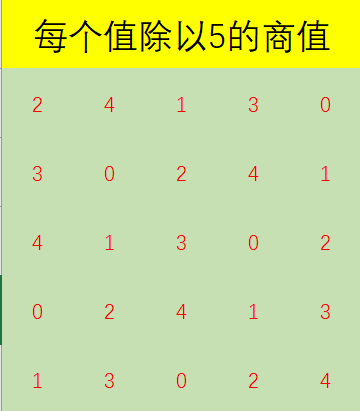
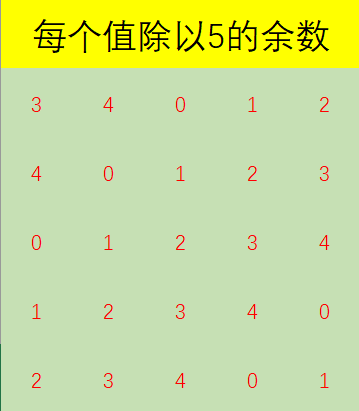
### generateMagicSquare()

思路与过程：

已经给出的函数的程序流程图如下：



能够生成nxn的Magic Square的原因：函数实际上以i=n分组，填满这个nxn的方阵。根据给出的规则，可以让每一行/列的数字来自不同的组。这使得每一行/列的数字除以五得到的商各自不同（即0到n-1都有）。同时，每一组达到某一行/列填充的数的余数之间存在一个递增/递减的规律，这使得每一行/列的数字除以五得到的余数各自不同（同上，0到n-1都有）。满足了这俩点，也就能够保证每一行/列的和是相等的。以一个5x5的方阵为例：



这样每一行/列的和就是：

（5\*0+5\*1+5\*2+5\*3+5\*3+5\*4）+（1+2+3+4+5）= 65。

对应为：

第i组对应n\*（i-1）（商部分）+ 第i组对应i（余数部分） = (n^2+1)\*n/2

接下来只需要满足对角线和是相等的就可以了。满足这个条件就需要把中值放在中间（5x5的中值就是13），对应的第一个数字就在第一行的中间。

这样，一个合格的幻方就完成了。

异常原因：

当n为偶数时，弹出的异常表示数组访问越界。经分析后可知，会访问到magic[n][n/2]，而数组是没有magic[n]这一行的。

当n为负数时，弹出的异常表示访问了负下标的位置，而数组是没有负数下标的

扩展过程：

1. 写入文件需要实例化File和FileWriter对象 ，前者用于存放这个路径，后者用于进行写入操作。写入的格式参考上文中要求的格式，具体操作并不复杂，此处不再赘述。要注意的是写完之后应该关闭文件，否则会写入失败。
2. 只需要在函数之前加入对n的判断就可以避免n的错误输入。

## Turtle Graphics

在从GitHub上得到的文件中，只需要对TurtleSoup.java进行处理。而需要处理的函数下面都有处理要。根据要求编写对应的代码并成功运行即可。

### Problem 1: Clone and import

1. 初始化本地仓库：git init
2. 打开远程仓库的网页：

<https://github.com/rainywang/Spring2022_HITCS_SC_Lab1/tree/master/P2>

3）获得远程仓库的地址：

<https://github.com/rainywang/Spring2022_HITCS_SC_Lab1.git>

4） 得到这个仓库里的所有内容：

git clone <https://github.com/rainywang/Spring2022_HITCS_SC_Lab1.git>

5) 可以直接在本地文件管理器中处理得到的所有文件。这里只需要P2文件夹部分即可。根据仓库建立的位置也可以直接在仓库处理得到的文件进行代码的编写工作。

6）如果在当前仓库进行开发，可以在每次开发结束后直接将代码push到实验的远程仓库。注意要修改远程仓库的地址。不在当前仓库则要修改文件位置或者本地仓库位置。

### Problem 3: Turtle graphics and drawSquare

forward(unit)函数是沿着当前方向移动unit像素，unit为整型变量。初始的方向是朝上。

turn(degree)函数是顺时针旋转degree度，角度值，double为double类型。

这一问的要求是画一个正方形，简单不做赘述。

### Problem 5: Drawing polygons

这一步要求实现两个函数。函数calculateRegularPolygonAngle是计算出正多边形的内角。输入是边数sides，类型是整型。只需要使用正多边形的内角和公式计算即可。函数drawRegularPolygon是画出边数为sides、边长是sidelength的正多边形。

之后要在Junit中测试。函数功能正常就会显示绿色，表示通过。过程中也可以设置断点来帮助测试。简单不做赘述。

### Problem 6: Calculating Bearings

这一步要求实现两个函数。函数calculateBearingToPoint 是以当前点为原点，计算下一个点的方位角。计算出的是右转角，即顺时针旋转过去需要的角度。根据代码中的提示，这里可以调用函数atan2帮助计算。函数calculateBearing是计算一系列点的方位角。点的横纵坐标分别储存在xCoords和yCoords里，每次计算是以上一个点为原点。根据提示，这里调用实现的上一个函数calculateBearingToPoint依次计算即可。

继续进行测试。这两个函数的实现同样简单，不做赘述。

### Problem 7: Convex Hulls

这一步要求实现一个凸包算法。凸包即从点集中选择最少的点连线，组成的图形能够包括点集中的所有点。下面阐述我选择的凸包算法。

1. 如果点数少于等于三个，可以直接返回这些点。
2. 如果点数多于三，则对这些点进行处理。首先取纵坐标最小的点，然后以这个点为原点计算出新的坐标。
3. 以新的坐标计算每个角的幅角，并按照幅角由小到大的顺序对点进行排序。如果幅角相等，则将与原点距离更近的排在前面。
4. 对处理完的点，易知第一个和第二个点是在凸包中的。连接这两个点，计算第三个点与这个位置的相对位置。如果点在直线的左边，跳转到5），否则跳转到6）
5. 这个点是凸包中的点。连接这个点和上一个点，产生接下来作为基准的线，然后继续处理。
6. 这个点的上一个点不是凸包中的点。把这个点从凸包中剔除，同时连接上上个点和上上上个点，产生接下来作为基准的线，然后继续处理
7. 当处理完最后一个点后，剩下来的点就是凸包中的点。

具体实现过程中，要处理给到的点集。由于给的是Set类，这个类不能简单遍历所有元素，要调用迭代器或使用另一种for循环。我选择将Set的点集修改为List类，这样就有了固定的顺序，方便后续操作处理。

在判断凸包的点时，可以使用栈，从而记录之前处理后的数据。这样能在回退到之前的点时直接从栈中取数据，不需要再额外记录。

其他的代码实现没有困难，问题主要在于代码量较长，容易出现遗漏，需要更加仔细，不做赘述。

### Problem 8: Personal art

画一个自己的作品。没有需要阐述的内容。

### Submitting

提交过程大致如下：

1. 初始化仓库：git init
2. 跟踪当前目录下的文件或文件夹，也就是把这些文件写入到暂存区：

git add 文件名（或文件夹名）

1. 也可以使用“git add . ”来跟踪当前目录下的所有文件和文件夹
2. 将暂存区的内容添加到本地仓库：git commit -m “备注”
3. 添加远程仓库
4. 将本地仓库中的内容发送到远程仓库，前一个是远程仓库，后一个是本地仓库：

git push origin master

## Social Network

建立一个关系网。这个关系网是无向图，但是要支持有向图的一些操作。所以此处选择直接建立有向图。这个过程要实现5个函数，分别为FriendshipGraph、Person、addVertex、addEdge、getDistance，依次实现图的创建、人的创建、在关系图中添加某个人、在关系网中添加两人的关系以及输出两个人之间的社交距离。

### 设计/实现FriendshipGraph类

思路与过程：

选择使用链表来实现主要的内容，如下图：



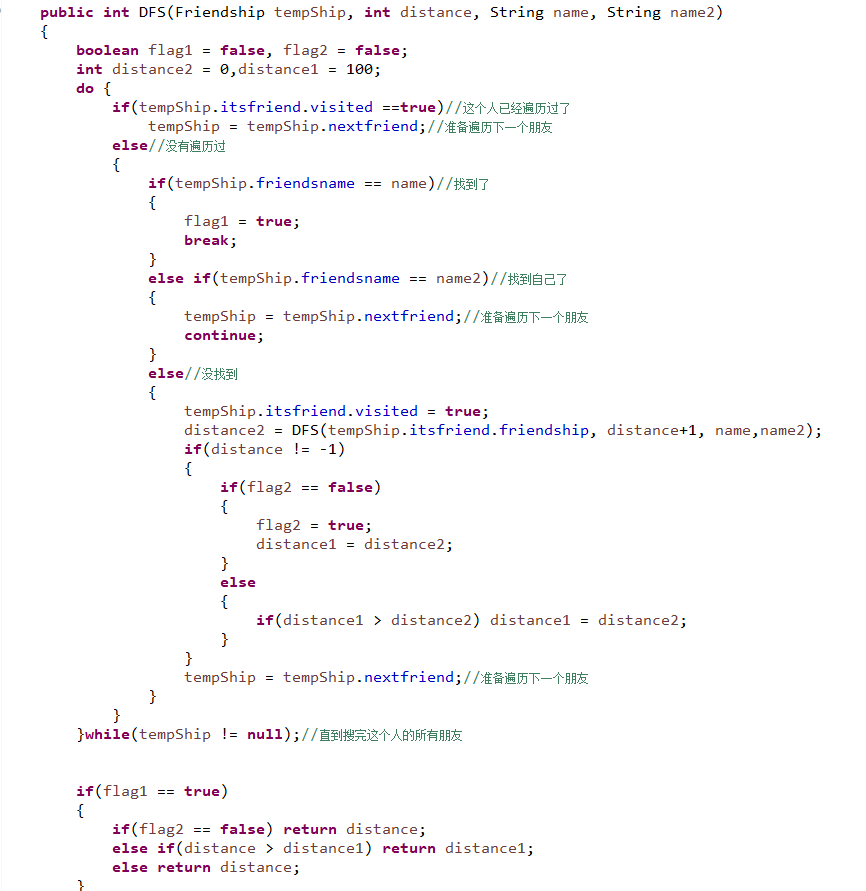
1. firstperson 用于后续的查找，lastperson能够方便向关系网中添加的步骤。
2. Vertex即每个人的结点，存放这个人在关系网中的信息，如名字、这个人的关系网、这个人的朋友数量。visited表示这个人是否被访问过，用于之后getdistance中的搜索算法。
3. Friendship即每个人的关系网，存有他所有的朋友。每个结点存有朋友的名字friendsname、指向这个朋友在关系网中的位置itsfriend。Itsfriend可以方便之后getdistance中的搜索算法。

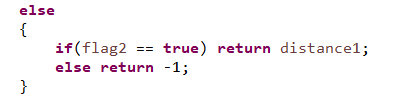
之后是实现的几个函数。FriendshipGraph用于实例化一个关系网，设置最初的关系网的firstperson、lastperson为空。addVertex、addEdge都需要从firstperson开始遍历以查找对应的人，不同的是addEdge在找到这个人之后会对他的关系网进行处理，而addVertex仅仅涉及能否找到这个人。这三个函数没有需要阐述的内容。

对于getDistance函数，首先需要遍历关系网，找到这两个人。找不到则返回-1，找到了再进行进一步找两人的社交关系。这里使用了深度搜索算法，如下图。

搜索过程从name这个人开始。主体部分内容如下：

1. 首先确认这个人是否遍历过，即查看关系网中这个人结点的visited值。已经遍历过则沿着name的关系网遍历下一个朋友.
2. 如果没有遍历过，要分三种情况：
   1. 如果这个人就是要找的name2，那么就将flag1设置为true，表示在当前距离找到了这个朋友。flag1会用于下面的判断
   2. 如果这个人是name，则跳过自己，遍历下一个朋友
   3. 若不是上面两种情况，那么就表示当前没有找到。这是将这个人标记为已遍历，然后调用这个函数搜索这个朋友的关系网。不同的是调用函数时distance值要加一。这个调用后的结果存储在distance2中。然后对两个距离进行处理。如果能够在调用函数的时候找到朋友，则令flag2设为true，同时与现存在distance1中的距离进行对比。如果distance2更近，则使用distance2的值并保存在distance1中，否则不修改distance1的数据。
3. 重复这个过程，直到这个人的关系网遍历完毕
4. 将distance1和distance做对比，即比较当前距离和查找其他人关系网得到的距离。先经过查看flag1和flag2确定有效数据。如果其中有一个数据无效，就返回有用的数据。如果两个数据都有效，返回更小的数据。如果两个都无效，则返回-1，表示没有找到这个人。





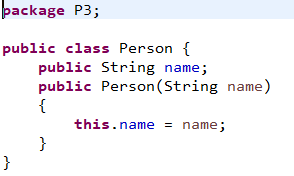
结果：

编写的代码没有问题，在后面的使用中也表现正常。

### 设计/实现Person类

思路与过程：

Person类只需要存有这个人的名字即可。因为这个任务的所有操作都是对关系网进行的，对Person只用到了姓名。

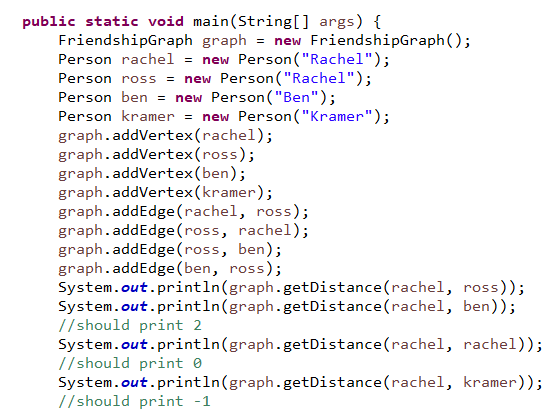


结果：

没有发现异常。

### 设计/实现客户端代码main()

Main中使用了实验给出的代码。



结果：

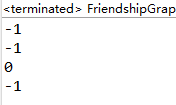
测试的结果是没有问题，没有需要说明的内容。

对任务中特殊要求的回答：

1. 注释掉第10行，即删去Rachel到Ross的关系

初步判断会依次输出-1、-1、0、-1。因为去掉了Ross，Rachel和Ben就不能通过Ross这个共同朋友建立关系，他们之间以及他们和Ross之间就没有了朋友关系。Kramer本来就不在朋友关系内，结果不变。Rachel和自己的关系为0，这是提前要求的，跟剔除Ross没有关系。

测试得到的结果也是如此。值得注意的是如果调换getDistance中rachel和ross的位置，是会输出1的。因为关系网的设计是基于有向图设计的，Ross到Rachel的关系建立还是成功的。



1. 替换Ross的名字

程序会报告错误。这里我已经添加了名字的检验，也就是说，在关系网中添加这个人时，需要检验的不是他的Person类名，而是这个类中的人名name。后续的要求也在后续被实现。



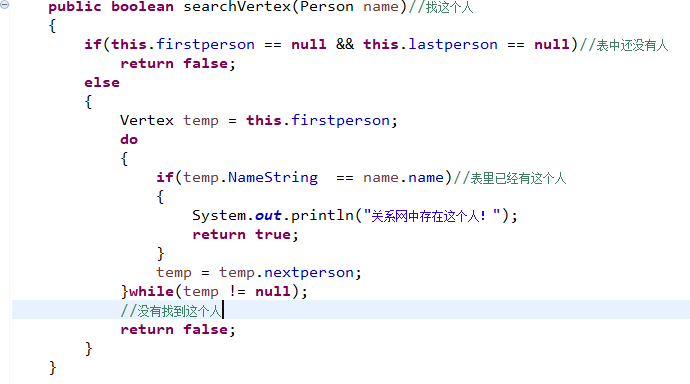
### 设计/实现测试用例

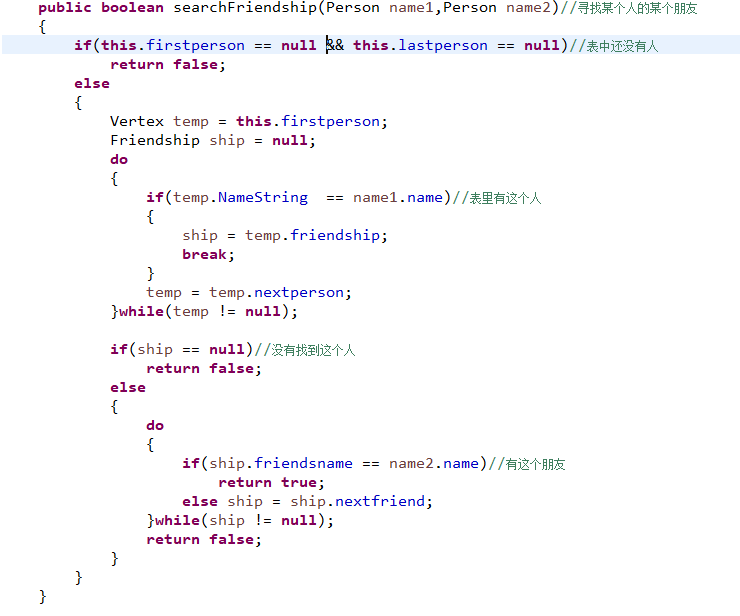
思路：

选择设计三个测试来分别验证addVertex、addEdge、getDistance三个函数的正确性。

过程：

依次添加代码即可。需要注意的是，在原本设计的函数中，缺少仅寻找某一人的函数和仅在一个人的关系网中寻找某人的函数，因此在测试前增加了新的函数searchVertex和searchFriendship，如下图：





结果：

编写的所有测试内容全部通过。

# 实验进度记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 任务 | 实际完成情况 |
| 2022-05-05 | 19:58-23:15 | 配置需要的实验环境  学习使用Git  编写任务1的代码和任务2的部分代码 | 按计划完成 |
| 2022-05-06 | 20:48-23:45 | 编写任务2的代码 | 遇到困难，未完成 |
| 2022-05-07 | 9:32-11:25 | 修改任务2代码的错误，编写任务3的部分代码 | 按计划完成 |
| 2022-05-07 | 13:25-15:27 | 完成任务3的全部代码，提交所有代码 | 按计划完成 |
| 2022-05-08 | 16:45-18:30 | 编写实验报告 | 未完成 |
| 2022-05-08 | 20:10-21:57 | 编写实验报告 | 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 |
| Git中获取P2文件出现问题 | 在CSDN中寻求帮助，发现是指令有误，修改后能够正确获取文件 |
| 任务2中Debug和Run模式下测试结果不同 | 请教他人，经指点后自行修改了代码中的错误 |
|  |  |

# 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（必答）

1. 编写量大的代码时要提前做好构思，写下自己的思路。否则可能会出现某些处理步骤的遗漏，导致产生错误而浪费大量时间。
2. 学习向他人请教，不要固执己见。学会在遇到错误时及时转变思路。

## 针对以下方面的感受（必答）

1. Java编程语言是否对你的口味？与你熟悉的其他编程语言相比，Java有何优势和不足？

由于在大创项目中接触过以Java为基础的kotlin语言，现在再次使用Java并没有遇到很大的困难。而相比与C语言来说，Java中提供了更过更方便的函数，在大多数情况下能够简化操作。

但是以我的角度而言，Java中的某些操作可能有些过于“自由”，对同一种操作会有很多种编写代码的方式。这时往往要了解多个方式的优缺点，实际上增加了一部分的工作量。

总体而言，Java的使用过程还是很流畅的。

1. 关于Eclipse或IntelliJ IDEA，它们作为IDE的优势和不足；

本实验中我主要使用的是Eclipse。它能够详细的列出所有的信息，但是能够处理的情况太多，有些不太适应。

1. 关于Git和GitHub，是否感受到了它在版本控制方面的价值；

目前没有，因为我还没有每次编写完毕后上传的习惯，每次都是只保存在本地。实际使用时也是直接调用本地文件，不需要进行其他操作。

1. 关于CMU和MIT的作业，你有何感受；

作业内容十分有趣，而且有较高的自由度。实际上很多部分都可以融入作者的个人思想。我认为这样的作业是很有吸引力的

1. 关于本实验的工作量、难度、deadline；

工作量、难度处于可以接受的成度，但是希望能延后时间提交。这样能够有更多时间学习和习惯Java的使用。

1. 关于初接触“软件构造”课程；

仍处于很模糊的状态。实际上课程所讲的内容在自己的脑海中不知道应该如何归类。可能需要继续学习一段时间才能给这个可能有一个更好的定位。