

**2019年春季学期  
计算机学院《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 郭子阳 |
| 学号 | 1170300520 |
| 班号 | 1703005 |
| 电子邮件 | guoziyang0033@gmail.com |
| 手机号码 | 18800420598 |

**目 录**

[1 实验目标概述 1](#_Toc1392798)

[2 实验环境配置 1](#_Toc1392799)

[3 实验过程 3](#_Toc1392800)

[3.1 Magic Squares 3](#_Toc1392801)

[3.1.1 isLegalMagicSquare() 3](#_Toc1392802)

[3.1.2 generateMagicSquare() 4](#_Toc1392803)

[3.2 Turtle Graphics 6](#_Toc1392804)

[3.2.1 Problem 1: Clone and import 6](#_Toc1392805)

[3.2.2 Problem 3: Turtle graphics and drawSquare 6](#_Toc1392806)

[3.2.3 Problem 5: Drawing polygons 6](#_Toc1392807)

[3.2.4 Problem 6: Calculating Bearings 7](#_Toc1392808)

[3.2.5 Problem 7: Convex Hulls 7](#_Toc1392809)

[3.2.6 Problem 8: Personal art 8](#_Toc1392810)

[3.2.7 Submitting 8](#_Toc1392811)

[3.3 Social Network 8](#_Toc1392812)

[3.3.1 设计/实现FriendshipGraph类 8](#_Toc1392813)

[3.3.2 设计/实现Person类 9](#_Toc1392814)

[3.3.3 设计/实现客户端代码main() 9](#_Toc1392815)

[3.3.4 设计/实现测试用例 9](#_Toc1392816)

[3.4 Tweet Tweet 9](#_Toc1392817)

[3.4.1 Problem 1: Extracting data from tweets 10](#_Toc1392818)

[3.4.2 Problem 2: Filtering lists of tweets 10](#_Toc1392819)

[3.4.3 Problem 3: Inferring a social network 10](#_Toc1392820)

[3.4.4 Problem 4: Get smarter 11](#_Toc1392821)

[4 实验进度记录 11](#_Toc1392822)

[5 实验过程中遇到的困难与解决途径 11](#_Toc1392823)

[6 实验过程中收获的经验、教训、感想 11](#_Toc1392824)

[6.1 实验过程中收获的经验和教训 11](#_Toc1392825)

[6.2 针对以下方面的感受 12](#_Toc1392826)

# 实验目标概述

本次实验通过求解四个问题，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

实验环境：

系统：macOS Mojave 10.14.3，JDK11.0.2

ide：Eclipse IDE for Eclipse Committers 2018-12 (4.10.0)

编辑器：Visual Studio Code 1.31.1

构建工具：Apache Ant，Travis CI

版本管理：git

代码托管：Github

配置过程：

从OracleJDK官网下载JDK并解压到系统目录，并在~/.bash\_profile中配置PATH、CLASSPATH和JAVA\_HOME。

在使用Eclipse进行Junit单元测试时会出现org.junit不存在的情况，主要是由于junit没有正确导入，只需要右键项目->Build Path->Add Library并选择junit，导入junit4即可。

P4的Main.java中的main方法在运行前会检查是否启用assertion。只需要在run->Run Configurations中的arguments选项卡下的VM arguments中添加JVM参数“-ea”即可。

有时P1的MagicSquare.java中的main方法读取txt包下的矩阵文件时可能会读取不到，这是因为运行时运行的是class字节码文件，而编译时txt文件没有被复制到编译文件的位置导致无法读取。这时读取目录即可采用相对项目根目录读取，**new** File("").getCanonicalPath()获取的就是项目的根目录，若要获取src下P1包中 的txt文件夹下的1.txt的路径即可写成**new** File("").getCanonicalPath() + “/src/P1/txt/1.txt”。

P2的TurtleSoup.java中有一个convexHull()方法，求一组给定点点最小凸包，可采用Wikipedia上的gift wrapping算法。P3的FriendshipGraph.java中有getDistance()方法，求给定两个点在有向图中的最短距离，可用邻接矩阵（List）存储邻接关系，并使用广度优先搜索，记录遍历的层数即可。P4的getMentionedUsers()方法，找出一组给定的tweet的text中提到（@-mentioned）的用户。可使用正则表达式匹配，按照给定的规则（1-140字符，有效字符为大小写字母、数字、下划线和-，且前后不得紧邻有效的字符），正则表达式为“(?<![A-Za-z0-9\_-])@[A-Za-z0-9\_-]{1,140}(?![A-Za-z0-9\_-])”，其中使用了零宽断言来规定前后紧邻的字符。P4的guessFollowsGraph()方法猜测一组tweets中的关注关系，除了给定的@-mentioned可以判断以外，我还使用如果提到相同的话题（#-mentioned），那么两人互相关注的判断。

由于Lab0中要求脱离Eclipse独立构建，而Maven和Gradle都使用了配置式依赖导入，而且默认的目录结构与要求结构不符（要求src与test平级），故选用Apache Ant构建。

安装Apache Ant需要将下载下来的压缩包解压到系统目录，并在~/.bash\_profile中配置环境变量ANT\_HOME和PATH。

使用Ant构建需要在项目根目录下新建配置文件build.xml。

这里有一个问题是，由于在Eclipse中test测试类与其他文件都在同一包下，而项目中需要将测试文件与其它文件分开，单独放在test文件夹中。就需要编译成class时将test和src下的文件编译到同一个目录下，而且要保证src和test目录的包结构相同。

用Ant运行P2时可能会报“Cannot load com.apple.laf.AquaLookAndFeel”的错误，这是因为Ant使用默认的JVM运行时，无法加载系统的界面库，需要转换为系统的JVM运行。在build.xml中运行P2的java标签下加入属性fork=”true”即可。

使用Ant运行P4的Main和其它测试类同样需要开启断言。在对应的java标签和junit标签下加上” <jvmarg value="-ea"/>”标签即可。

由于Lab0还提到了Travis-CI自动构建，于是就尝试了一下。在项目根目录下新建.travis.xml文件，指定language、jdk、install和script。jdk需要指定为oraclejdk11，否则可能由于版本问题无法构建。Travis CI默认支持Maven、Gradle和Ant，取决于根目录下的配置文件。将Travis CI与Github关联，将项目push到Github后，Travis CI就会自动开启一个虚拟机，并按照配置文件构建项目并实时输出日志。

Github Lab1仓库URL地址：

https://github.com/ComputerScienceHIT/Lab1-1170300520

# 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

要求验证某个读入的矩阵是否是幻方，读入的文件要求数字间以\t分割且都是正整数，幻方要求为方阵且行列对角线之和相等。

### isLegalMagicSquare()

设计与实现：

首先初始化一个List<List<Integer>>二维List以存储矩阵，使用文件输入缓冲流BufferedReader从目标文件中读取一个矩阵，并将按行读入的String分割转换为List<Integer>，存入二维List中。该步骤可能由于文件不存在而造成IOException，或者存在负数、小数或者不是用“\t”分割而造成NumberFormatException。读取完成后验证该矩阵是否为方阵，如果不是，输出错误并返回false，最后再分别计算各行、各列以及对角线之和，如果相等，则为Magic Square，否则不是，输出错误并返回false。

结果：

以下为在main方法中调用五次isLegalMagicSquare()验证txt文本的结果：

以下检测矩阵1.txt

true

以下检测矩阵2.txt

true

以下检测矩阵3.txt

该矩阵不是方阵！

false

以下检测矩阵4.txt

文件数字不规范！请检查文件！

java.lang.NumberFormatException

at P1.MagicSquare.convertTointList(MagicSquare.java:116)

at P1.MagicSquare.readFromFile(MagicSquare.java:79)

at P1.MagicSquare.isLegalMagicSquare(MagicSquare.java:53)

at P1.MagicSquare.main(MagicSquare.java:24)

false

以下检测矩阵5.txt

文件数字不规范！请检查文件！

java.lang.NumberFormatException: For input string: "12673 12796"

at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)

at java.base/java.lang.Integer.parseInt(Integer.java:652)

at java.base/java.lang.Integer.parseInt(Integer.java:770)

at P1.MagicSquare.convertTointList(MagicSquare.java:115)

at P1.MagicSquare.readFromFile(MagicSquare.java:79)

at P1.MagicSquare.isLegalMagicSquare(MagicSquare.java:53)

at P1.MagicSquare.main(MagicSquare.java:24)

false

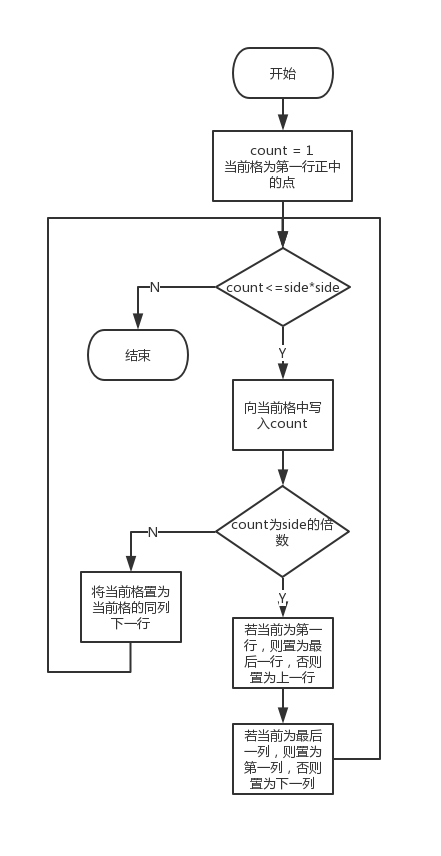
### generateMagicSquare()

设计与实现：

给出的代码使用Merzirac法生成奇阶幻方：

1. 首先向第一行正中的方格内填写1
2. 以下依次向右上角的方格内填写2、3、4……
3. 若右上角的方格内已经有数字，则向下移动一格继续填写
4. 若右上角的方格超出矩阵的行，则移到矩阵下一列的最下端继续填写
5. 若右上角的方格超出矩阵的列，则移到矩阵下一行的最左端继续填写

给出的代码没有对2.1步骤进行判断，而是使用当前填入的数字是否是行列数的倍数来决定，如果是行列数的倍数，则右上角必然有数字，需要下移。

流程图如下：

结果：

以下为main方法中调用generateMagicSquare()方法并使用3.1.1中的方法检验的结果：

调用参数为5时：

要求2：以下测试generateMagicSquare()方法：

17 24 1 8 15

23 5 7 14 16

4 6 13 20 22

10 12 19 21 3

11 18 25 2 9

以下检测矩阵6.txt

true

调用参数为6时：

要求2：以下测试generateMagicSquare()方法：

请输入偶数值！

java.lang.ArrayIndexOutOfBoundsException: Index 6 out of bounds for length 6

at P1.MagicSquare.generateMagicSquare(MagicSquare.java:199)

at P1.MagicSquare.main(MagicSquare.java:37)

false

调用参数为-7时：

要求2：以下测试generateMagicSquare()方法：

请输入正整数！

java.lang.NegativeArraySizeException: -7

at P1.MagicSquare.generateMagicSquare(MagicSquare.java:196)

at P1.MagicSquare.main(MagicSquare.java:37)

false

## Turtle Graphics

该题估计起源于Python的turtle库，控制一个小乌龟在屏幕上爬行，画出各种图案。MIT用Java改写，提供了基础接口的实现（如forward和turn等），需要我们自己实现更加高级一点的接口。

### Problem 1: Clone and import

在本地新建一个文件夹，使用终端cd到该文件夹中，git clone <https://github.com/rainywang/Spring2019_HITCS_SC_Lab1.git>，将整个项目clone下来（git不允许下载单文件）。并将其中的P2文件夹复制到自己的项目目录中，在src下与P1平级。接着在Eclipse中继续开发即可。

### Problem 3: Turtle graphics and drawSquare

该Problem要求使用现有的接口（forward和turn）实现drawSquare方法，要求该方法调用时，根据给定的变长画出一个正方形。使用一个简单的for循环即可解决。

### Problem 5: Drawing polygons

该Problem需要实现两个方法：calculateRegularPolygonAngle()和drawRegularPolygon()方法，并使用Junit测试。calculateRegularPolygonAngle()方法要求根据给定的边数作为参数，计算出对应的正多边形的内角。只需要使用公式即可，由于insideAngle为double类型，而参与计算的数字都是int整型，如果直接计算可能会发生截断，需要在写成2d或180d即可转为浮点运算。drawRegularPolygon()方法要求以变长与边数作为参数，控制turtle画出正多边形。使用循环即可，计算角度时可调用calculateRegularPolygonAngle()方法计算，唯一要注意的就是turtle旋转的角是外角，而不是计算出来的内角。

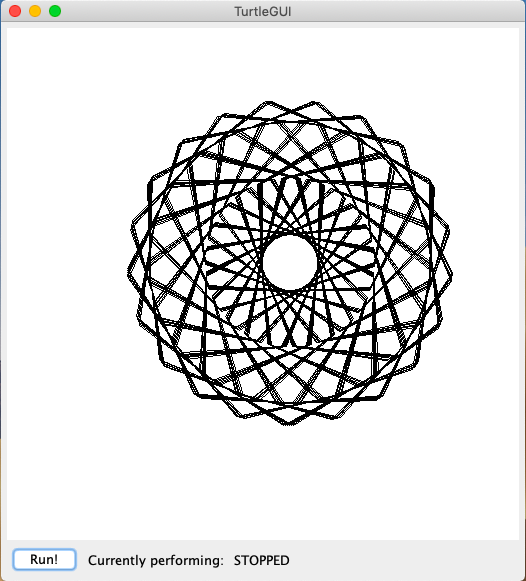
### Problem 6: Calculating Bearings

该Problem需要实现两个方法：calculateBearingToPoint()和calculateBearings()方法，并进行Junit测试。calculateBearingToPoint()方法要求计算在当前点、当前朝向的情况下，顺时针转向目标点所需要转动的角度。给定的参数有当前朝向的度数、当前点的X与Y值和目标点的X与Y值，当前朝向的度数以向上方为0度。基本是个数学问题，比较繁琐，分多种情况讨论即可，需要用到反正切函数Math.atan2方法。calculateBearings()方法的参数为两个List，第一个List中为X的值，第二个为Y的值，需要返回一个List，包含所有需要转动的角度。只需要遍历两个集合，调用calculateBearingToPoint()方法即可，可设置一个变量里路当前朝向的角度。

### Problem 7: Convex Hulls

该Problem涉及一个著名的图形学概念：凸包。凸包可以理解为给定的点集的一个子集，该子集中点点连线可以将全集都囊括在内。要求实现的convexHull()方法就是在给定的Point的Set中寻找一个最小的凸包。由于是最小的凸包，如果同一条直线上的多个点都是凸包边界上的点，只取两端的点以保证最小。可使用gift wrapping（礼物包装）算法。从最左上（左下）角的点开始，逐个扫描点集中的点，选取与当前朝向偏角最小的点加入结果集，并将其作为下一个当前点继续循环。如果有多个点都是最小转角的点，则选择距离最远的点即可。

### Problem 8: Personal art

实现drawPersonalArt()，可以自己画一些有趣的东西，可作出一些简单的图形之后偏移一个较小的角度，并循环，即可作出较美观的花纹。

### Submitting

在项目的根目录打开终端，使用git add .命令将所有文件的改变存储在暂存去，再使用git commit -m “提交内容”将暂存区的内容提交到本地的版本库，创建一个新的版本。最后使用git push origin master将本地版本库的master分支推送到远程服务器（origin）的master分支上，这里的远程服务器就是GitHub的服务器。

## Social Network

要求实现一个模拟的社交网络，形成一个图结构，可以向图中添加点（人）和边（关系），并且可以计算各个点之间最短的距离。

### 设计/实现FriendshipGraph类

FriendshipGraph类是社交网络的抽象，描述了一个社交网络关系图。描述图有两种方法：邻接表和邻接矩阵，为了计算距离方便使用邻接矩阵。由于图中的点数不是固定的，故使用List嵌套，描述邻接矩阵的属性类型为List<List<Integer>>，并添加一个List<Person>来存储当前的图中的点。所有属性私有，并添加相应的setter/getter方法。根据示例代码，该类需要实现addVertex()方法、addEdge()方法和getDistance()方法。addVertex()方法用于向图中添加一个点，由于不可重复添加，添加之前需要首先验证该点是否已存在，是则跑出异常。添加人还需要扩充邻接矩阵。addEdge()方法是在两点之间添加一条边，由于要求中说明添加社交关系需要调用addEdge()两次，所以addEdge()方法仅添加一条有向边。getDistance()方法用于计算两个点之间点最短距离。由于有向路的权值都为1，仅需要采用广度优先遍历，记录遍历的层次即可，遍历到目标点时到层次即为最短的距离。

### 设计/实现Person类

该类为对社交网络中的点（人）点抽象。根据示例代码，仅需添加属性name并加入getter/setter即可。

### 设计/实现客户端代码main()

客户端代码直接使用说明中给出的示例代码，输出如下：

1

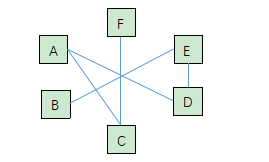
2

0

-1

和预期输出相同

### 设计/实现测试用例

测试用例使用如下的图：

共测试两个方法：getDistance()方法和addVertex()方法。getDistance()测试是否可以得到合理的输出。addVertex()主要测试当添加相同的人时抛出异常。

## Tweet Tweet

该试验对一些Twitter进行基础的分析，通过特征分析，猜测出各个用户之间的关注关系，并分析出“最受欢迎”的人。

### Problem 1: Extracting data from tweets

Problem1主要是对给定的tweets进行信息提取。要求实现两个方法：getTimespan()和getMentionedUsers()并进行相应的测试。getTimespan()传入一个List的tweets，找出这些tweets的最小时间跨度并返回。Tweet中使用的描述时间点的类为Instant类，是jdk8引入的一个替代Data的类。Instant对象可使用isBefore()和isAfter()接口进行比较。只需要遍历tweets，找出最早和最晚的tweets，使用其Instant构造出Timespan返回即可。getMentionedUsers()用于找出tweets中被提到（@）的其它用户。判断用户名的规则是：由数字（0-9）、大小写字母（a-z，A-Z）、下划线（\_）和中划线（-）组成，并且在tweets中@前与用户名后不得紧跟用户名有效的字符。由此可使用正则表达式进行判断。描述以上规则的正则表达式为：”(?<![A-Za-z0-9\_-])@[A-Za-z0-9\_-]{1,140}(?![A-Za-z0-9\_-])”。其中使用了零宽断言式以判断待匹配字符串前后的状况。遍历tweets，使用该正则表达式进行匹配，并将结果存入List，返回即可。

### Problem 2: Filtering lists of tweets

Problem2是对tweets进行按要求过滤。要求实现writtenBy()、inTimespan()和containing()方法。writtenBy()方法在给定的tweets中过滤出某个用户写的tweets。只需要简单地遍历tweets并调用getAuthor()获取每个tweet的作者，并与要求的username对比即可。inTimespan()方法在tweets中过滤出固定的时间段内创建的tweets。可使用Timespan类的getStart()和getEnd()方法获取到开始和结束时间点到Instant，和给定的tweet通过getTimestamp()方法获得的Instant比较即可。containing()方法要求过滤出所有包含给定单词的tweets。包含给定的单词要求在文本中也是单独成词。可使用正则表达式匹配，如包含talk可使用” (?<![A-Za-z0-9\_-])talk(?![A-Za-z0-9\_-])”作为正则表达式匹配。遍历tweets匹配即可。

### Problem 3: Inferring a social network

Problem3则是对一组tweets进行分析，进行影响力排名之类的。要求实现guessFollowsGraph()和influencers()方法。guessFollowsGraph()方法返回一个Map，key为username，value为该用户可能关注的用户的Set。依据是如果A@了B那么就可以认为A关注了B。于是可以遍历tweets并调用Filter.getMentionedUsers()找出提到的username即可。influencers()方法要求对影响力进行降序排序。可遍历Map，以建立一个key为用户名、value为被follow的次数的Map。使用Collections.sort接口对该Map进行排序，使用匿名内部类实现Comparator接口，保证排序的依据为value的大小。之后即可将所有的Key提取出来并组成一个List，由于默认排序为升序，使用Collections.reverse方法将List倒置后返回即可。

### Problem 4: Get smarter

Problem4增强Problem3的判断，增加一些判断的依据，使系统更加“聪明”。可以认为，当两个人在tweet中都提到了共同的话题（#topic），则可以认为两人互相follow于是可以用正则表达式将所有的topic筛选出来，在根据topics将所有关注相同话题的用户组成一个List，让这个List中的用户相互关注即可。

# 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 |
| 2019-02-25 | 21:45-23:50 | 编写完成问题1，并添加了Ant自动化构建 | 按计划完成 |
| 2019-02-26 | 12:32-15:21 | 编写完成问题2，并完成了Travis CI在线 | 按计划完成 |
| 2019-02-26 | 20:50-22:49 | 编写完成问题3，将junit整合进ant | 按计划完成 |
| 2019-02-27 | 18:10-23:20 | 编写完成问题4 | 按计划完成 |
| 2019-02-28 | 17:30-20:12 | 审查全部实验，补上测试用例 | 按计划完成 |
| 2019-03-01 | 18:00-19:05 | 修改了凸包问题的算法 | 按计划完成 |
| 2019-03-03 | 14:00-19:00 | 完成实验报告 | 按计划完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的难点 | 解决途径 |
| 不懂convexHull()的意义与实现算法 | Google搜索，Wikipedia搜索 |
| 不会使用正则表达式匹配字符串 | 菜鸟教程速成 |

# 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

深入理解了工程的构建方式与在线持续集成（CI）的方式，代码版本管理方式，单元测试的编写，更接近现代工程的管理。了解了Git的运作方式，提交和版本的概念与运用等。

## 针对以下方面的感受

Java语言被设计为一门完全的面向对象的语言，故思维模式比较接近正常人的思维过程，易于理解和掌握。Java的IDE使用的Eclipse是一个开源的项目，由Eclipse基金会负责维护，由于是开源项目，自我感觉没有商业软件Intellij IDEA用起来顺手。Git是版本管理软件，而Github是Git项目的托管平台，Github Classroom确实方便了计算机专业的教学。CMU和MIT的题目出的比较有水平，简单的作业融合了比较前沿的知识和背景，简单而有深度。实验的工作量较大，但不是很难，而且时间充裕，足够完成并优化。软件构造确确实实是一门教授写出更精彩代码的课程，实验都这么有水平。