

**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 | 李大鑫 |
| 学号 | 1170300825 |
| 班号 | 11703008 |
| 电子邮件 | hahalidaxin@163.com |
| 手机号码 | 13075361573 |

**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 1](#_Toc1392800)

[3.1 Magic Squares 1](#_Toc1392801)

[3.1.1 isLegalMagicSquare() 1](#_Toc1392802)

[3.1.2 generateMagicSquare() 1](#_Toc1392803)

[3.2 Turtle Graphics 1](#_Toc1392804)

[3.2.1 Problem 1: Clone and import 2](#_Toc1392805)

[3.2.2 Problem 3: Turtle graphics and drawSquare 2](#_Toc1392806)

[3.2.3 Problem 5: Drawing polygons 2](#_Toc1392807)

[3.2.4 Problem 6: Calculating Bearings 2](#_Toc1392808)

[3.2.5 Problem 7: Convex Hulls 2](#_Toc1392809)

[3.2.6 Problem 8: Personal art 2](#_Toc1392810)

[3.2.7 Submitting 2](#_Toc1392811)

[3.3 Social Network 2](#_Toc1392812)

[3.3.1 设计/实现FriendshipGraph类 2](#_Toc1392813)

[3.3.2 设计/实现Person类 2](#_Toc1392814)

[3.3.3 设计/实现客户端代码main() 2](#_Toc1392815)

[3.3.4 设计/实现测试用例 3](#_Toc1392816)

[3.4 Tweet Tweet 3](#_Toc1392817)

[3.4.1 Problem 1: Extracting data from tweets 3](#_Toc1392818)

[3.4.2 Problem 2: Filtering lists of tweets 3](#_Toc1392819)

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

[3.4.4 Problem 4: Get smarter 3](#_Toc1392821)

[4 实验进度记录 3](#_Toc1392822)

[5 实验过程中遇到的困难与解决途径 3](#_Toc1392823)

[6 实验过程中收获的经验、教训、感想 4](#_Toc1392824)

[6.1 实验过程中收获的经验和教训 4](#_Toc1392825)

[6.2 针对以下方面的感受 4](#_Toc1392826)

# 实验目标概述

本次实验通过求解四个问题（其中一个可选），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

Java 环境：JDK11

IDE 软件：Eclipse

VCS：git—repo: <https://github.com/ComputerScienceHIT/Lab1-1170300825>

硬件环境：Intel Core i7-6700HQ x64CPU,16G RAM,256G SSD +1T HDD.

# 实验过程

## Magic Squares

问题理解：本问题要求实现判断一个nxn矩阵是否是一个幻方，并且理解一个针对n为奇数的幻方的生成算法。主要考察点：Java基本语法，文件读写，判断输入的合法性。

### isLegalMagicSquare()

实现思路：

1. 使用java输入流读入按行读入文件

对于每行尝试使用\t进行分割，对于分割之后的每一项，如果不能使用Integer.valueOf成功转化（发生Exception），这有两种可能原因，一种是该矩阵不满足使用\t分隔，一种是该项包含非数字。

1. 将第一行的数字个数作为基准数目n，在读入的过程中判断该矩阵是否是nxn的矩阵。
2. 如何判断得到的nxn矩阵是否是一个幻方？使用三次for循环分别检查行、列、对角线数字的和是否相等。

### generateMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

思路：使用罗伯法构造一个n为奇数的nxn幻方。罗伯法的算法为

把1（或最小的数）放在第一行正中；按以下规律排列剩下的(n×n－1)个数：

（1）每一个数放在前一个数的右上一格；

（2）如果这个数所要放的格已经超出了顶行那么就把它放在底行，仍然要放在右一列；

（3）如果这个数所要放的格已经超出了最右列那么就把它放在最左列，仍然要放在上一行；

（4）如果这个数所要放的格已经超出了顶行且超出了最右列，那么就把它放在底行且最左列；

（5）如果这个数所要放的格已经有数填入，那么就把它放在前一个数的下一行同一列的格内。

## Turtle Graphics

任务理解：该任务要根据代码注释提示补全代码从而实现一个完整的绘制工具Trurtle Graphics。主要考点：计算几何基础知识，正多边形的内角外角，java函数调用绘制多边形，计算向量之间的夹角，计算凸包，使用Junit进行单元测试。

### Problem 1: Clone and import

从github获取代码：网页download

从本地创建git仓库：1.git init初始化本地仓库 2.git add remote origin 添加远程库源 3.在远程仓库创建master分支 4.git pull origin master将远程仓库同步到本地 5.git add->git commit->git push 将本地文件加入本地仓库，将本地仓库同步到远程仓库。

此时git仓库本地与远程可以同步。

### Problem 3: Turtle graphics and drawSquare

问题思路：

1. forward (steps) 首先根据当前坐标和朝向计算走steps步之后的坐标，根据起始点生成线段类lineSeg，根据朝向以及lineSeg构造Action添加到ActionList中等待执行。
2. turn(degrees) 更改当前朝向，向ActionList中添加turn的Action。
3. drawSquare，调用*drawRegularPolygon*(turtle,4,sideLength)

### Problem 5: Drawing polygons

问题思路：

1. calculateRegularPolygonAngle 计算正多边形的内角，（边数-2）/sides。
2. drawRegularPolygon ：调用*calculateRegularPolygonAngle（sides）*计算正多边形的内角为rotation，调用turtle.forward和turtle.turn进行前进和转向。

### Problem 6: Calculating Bearings

问题思路：

1. calculateBearingToPoint ：计算（当前点，当前朝向）构成的向量与（当前点，目标点）构成的向量之间的夹角，首先计算（当前点，目标点）构成的向量与y轴之间的tan，使用Math.atan2计算与y轴夹角，与当前朝向夹角（与y轴之间）作差计算两向量之间的夹角，需要注意负角变正角。
2. calculateBearings：对列表中的每两个相邻节点调用calculateBearingToPoint计算夹角，列表保存。

### 3.2.5 Problem 7: Convex Hulls

问题思路：使用gift-wrapping算法实现convexHull

1. 首先根据坐标选出最左下角的点作为点B。添加一个点A，A位于B点稍下的位置。将(A,B)作为向量Vec1。
2. 扫描所有的不处于凸包中的点，计算A点与该点之间的夹角，夹角通过两向量点击与向量的模计算cos然后通过acos函数获得，找到形成夹角最小的点，如果有多个选择距离当前**直线距离最远**的那个点（其实可以选择距离前一个点最远的点）。将点B作为新的A，点C作为新的B，再次循环。
3. 当新找到的节点是最左边的初始节点时结束循环。

### Problem 8: Personal art

使用下面代码实现drawPersonalArt：

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

**for(**int i **=** 0**;** i **<** 15**;** i**++){** // i < 15 to limit size of shapes

drawRegularPolygon**(**turtle**,** i**,** i**\***10**);**

drawRegularPolygon**(**turtle**,** i**\***10**,** i**);**

**}**

### Submitting

Git add .

Git commit -m “P2 is DONE”

Git push origin master

## Social Network

实现Person和FriendshipGraph两个类，模拟社交网络，提供添加节点（人），节点之间添加边（社交关系），通过BFS计算两节点之间最短路的功能。主要考点：java类的实现，java基本数据结构，BFS求最短路。

### 设计/实现FriendshipGraph类

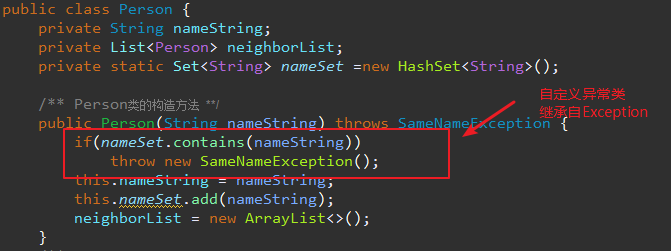
设计思路：

1. 存储结构：使用List<Person> persons存储所有节点。Map<String,Integer> namePool 用于记录所有的用户名字，从而提示重名错误。
2. addVertex：添加节点，向persons List中添加一个person对象
3. addEdge：调用pa的addNeighbor方法添加一条由pa指向pb的有向边。
4. getDistance：特殊判断stPerson是否与edPerson相等，相等则返回0。使用Queue与Map实现BFS求最短路。Map用来存放距离源点的最短路同时作为当前点是否已经访问过的标志。

### 设计/实现Person类

设计思路：

1. 存储结构：String nameString存放当前Person的名字，List<Person> neighborList存放当前节点为起始点的所有出边的另一端的Person。
2. addNeighbor：向neighborList添加新的Person。
3. getName()：获得Person的name的getter
4. getNeighList()：获得Person的相邻节点列表的getter
5. 避免重名情况的解决方法：



### 设计/实现客户端代码main()

Main函数代码如下：



### 设计/实现测试用例

测试思路：

1. 测试单边联通的特殊情况
2. 测试重名的情况
3. 测试不能到达的情况
4. 测试到达路径有多条的情况

## Tweet Tweet

### Problem 1: Extracting data from tweets

实现思路：

1. getTimespan：如果tweets为空则将当前时间作为始末时间点构造Timespan，否则调用getEarliestTime和getLatestTime分别获得所有tweets中的最早发表时间和最晚发表时间构造Timespan。时间比较调用Time.isBefore和Time.isAfter函数。
2. getMentionedUsers：对于每条tweet，使用正则表达式匹配满足username-metion条件的字符串，正则表达式匹配字符串为"@([\\w,-]+)"，这样匹配到的项满足的条件是后面不包含username合法字符，为了保证前面也满足这一条件，还需要对于匹配项的前一个字符进行特殊判断。
3. getMetionedHashTags：自实现函数，同样利用正则表达式匹配一条tweet中的所有hash tag，正则表达式字符串为："#([\\w,-]+)"。

### Problem 2: Filtering lists of tweets

实现思路：

1. writtenBy：for循环检查所有tweet，比较author和username在lowercase下是否相同。
2. inTimespan：利用Time.isBefore和Time.isAfter进行比较。
3. containing：首先spec中对于word的定义为被空格字符包围或者处于string的结束，由定义出发这里不考虑实例中有标点符号位于word尾部的情况。对于一条text，直接调用String.split(“ ”)进行分割，然后检查word的lowercase是否存在于lowercase的words之中。

### Problem 3: Inferring a social network

实现思路：

1. guessFollowGraph：第一种证据思路是如果一名author在tweet中@过其他人，那么该author就会follow这些被@的人。扫描每条tweet，调用Extract.getMentionedUsers提取出该条tweet中所有被@到的人，将这些人加入到Map[author]->Set<Stirng>中。
2. influencers：按照社交网络的影响力（follow数目）对所有user进行排名。首先根据followsGraph统计每个人的follow数目，InfluenceList存放所有username String，Map<String,Integer> influencers存放每个人对应的follow数目，然后调用Array.sort对influnceList进行排序，重写comparator为：

**new** Comparator<Object>() {

**public** **int** compare(Object o1,Object o2) {

**int** fl1=influencers.get((String)o1);

**int** fl2=influencers.get((String)o2);

**if**(fl1>fl2) {

**return** -1;

} **else** **if**(fl1<fl2) {

**return** 1;

}

**return** 0;

}

}

### Problem 4: Get smarter

实现思路

1. guessFollowGraph：第二种证据思路是如果两个人有很多共有的hash tag或者共享有一个热度很低的hash tag那么任务两个人会互相影响即互相follow。首先调用Extract.getMetionedHashTags(tweets)提取出每个人发表过的hash tag，然后统计每个hash tag的出现此处作为其热度。对每两人求其hash tag的交集，如果共享tag超过SHARETAGS\_THRESHOLD或者共享有一个热度低于UNINFULENCETAG\_THRESHOLD则认为两者互相影响。

# 实验进度记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 任务 | 实际完成情况 |
| 2019-02-25 | 14:01-17:09 | 编写问题1并进行测试 | 按计划完成 |
| 2019-02-25 | 17:09-23:03 | 编写问题2并进行测试 | 按计划完成 |
| 2019-02-26 | 18:00-19:49 | 编写问题3并进行测试 | 按计划完成 |
| 2019-02-27 | 18:00-21:10 | 编写问题4并进行测试 | 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 |
| 计算几何知识 | 上网搜索 |
| 凸包的gift-wrapping算法 | 网上搜索算法，画图实现，多次测试 |
| SociaNetwork的get smartter实现Map<String,Set<String>>使用不太熟练 | 仔细实现，多次测试 |

# 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. 编写完程序之后并不是结束，最好使用Junit进行单元测试
2. 程序测试一定要综合考虑多种边界情况，输入的标准与否是程序首先应该要确定的。
3. Java中文件输入输出流需要最后close否则会造成内存泄漏。
4. 学会看代码的spec

## 针对以下方面的感受

1. Java编程语言是否对你的口味？

答：Java很好用，各种基本数据结构都能在Java中找到。但写代码方面可能不想Python那样有强大而多样的缩写方法。

1. 关于Eclipse IDE

答：一般。一个比较老的IDE了，代码提示用起来比较难受，输入等号的时候默认会引用接近词语；设计项目目录（package,folder）刚开始用起来比较难受，知道了很简单；界面比较过时，如果安装插件之后还勉强能看；快捷键还没学，所以用起来不是那么顺心应手。目前的感觉是没有IDEA好用。

1. 关于Git和GitHub

答：git操作入门其实很简单，github作为一个远程仓库应用广泛，同时可以作为一个展示个人劳动成果的网站。

1. 关于CMU和MIT的作业

答：因为是第一次试验，所以比较简单。

1. 关于本实验的工作量、难度、deadline

答：实验工作量始终，因为是第一次试验，所以难度比较简单，deadline很宽松。

1. 关于初接触“软件构造”课程

答：根据这两堂课的感受来说，讲义有点停留于概念，例子太少所以学生很难感受到其中真意，希望以后的课能够有更多的实例供大家了解。