

**2024年春季学期  
计算学部《软件构造》课程**

**Lab 1实验报告**

|  |  |
| --- | --- |
| 姓名 | 张浩然 |
| 学号 | 2022113077 |
| 班号 | 2237101 |
| 电子邮件 | 2291447790@qq.com |
| 手机号码 | 18293888822 |

**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3.1旨在设计并实现一个Java程序，用于检测并判断输入的矩阵是否为幻方阵（Magic Square）。幻方阵是指一个n×n的矩阵，其中所有行、列以及对角线上的元素之和均相等。

实验要求包括：

编写Java程序MagicSquare.java，实现检查矩阵的行、列和对角线值，并判断是否为魔方阵。

通过读取给定的文本文件，检查其中的矩阵是否符合魔方阵的定义，包括各种特殊情况的处理。

实现一个静态函数generateMagicSquare(int n)，用于生成一个n×n的魔方阵，并将其绘制在控制台上。

对generateMagicSquare函数进行扩展，使其能够将生成的魔方阵写入指定的文本文件中，并在输入不合法时 gracefully 退出，而不是抛出异常。

在主函数main()中调用以上函数，测试其正确性和扩展功能的实现。

通过实验，我们将深入理解魔方阵的概念与特性，并掌握Java编程技能，包括文件操作、异常处理以及函数扩展等方面的应用。

实验3.2在这个实验中，我们将使用Java语言中的简单版乌龟图形库，实现一系列与乌龟绘图相关的功能。乌龟图形是受到MIT的Logo语言启发而创建的，通过发出一系列命令，可以控制屏幕上的“乌龟”进行移动和绘制线条。

实验包括以下任务：

实现drawSquare(Turtle turtle, int sideLength)方法，使用前进和旋转两种乌龟命令，绘制正方形。

运行TurtleSoup.java中的主方法，调用你实现的drawSquare方法，在屏幕上观察绘制出的正方形。

编写commit信息并将代码提交到Git仓库。按照指导操作，添加、提交并推送代码到远程仓库，以便我们后续进行评分。

在实验的后续部分，还包括以下任务：

实现calculateRegularPolygonAngle方法，用于计算正多边形内角的大小。

实现drawRegularPolygon方法，利用之前的方法绘制正多边形。

实现calculateBearingToPoint和calculateBearings方法，用于计算乌龟从当前位置到目标位置的旋转角度。

实现convexHull方法，计算一组输入点的凸包。

实现drawPersonalArt方法，绘制个人艺术作品，可以使用乌龟的颜色方法改变笔的颜色。

完成任务后，需要再次提交并推送代码到Git仓库。通过这个实验，我们将学习如何利用乌龟图形库进行绘图，同时掌握Git版本控制的基本使用方法。

实验3.3本实验旨在设计和实现一个FriendshipGraph类，用于表示社交网络中的人际关系，并计算两个人之间的最短距离。我们采用了无向图来模拟社交网络，同时实现了一个Person类来表示个人信息。通过合理的图实现和广度优先搜索算法，我们确保了程序的可扩展性和效率。实验报告将介绍我们的实现方法、测试策略以及结果分析。

# 实验环境配置

在配置本次实验所需的开发、测试和运行环境时，我首先使用了IntelliJ IDEA作为集成开发环境。以下是我配置的步骤和相关记录：

安装IntelliJ IDEA:我从JetBrains官网下载了最新版本的IntelliJ IDEA，并按照安装向导进行了安装。

创建Java项目：在IntelliJ IDEA中，我通过选择"New Project"来创建一个新的Java项目。在创建项目时，我确保选择了合适的项目类型（Java），采用了IntelliJ构建，并配置了项目的名称和位置。

导入项目源代码：我将实验所需的源代码文件导入到IntelliJ IDEA中的项目目录中。这可以通过直接拖拽源代码文件到项目视图中实现，或者选择"File" -> "New" -> "Module from Existing Sources"导入。

配置JUnit测试：在项目中，我确保已经包含了JUnit测试框架。在IntelliJ IDEA中，我可以直接在项目设置中添加JUnit库。

编写测试用例：我在项目中编写了针对FriendshipGraph类的JUnit测试用例。这些测试用例包括对addVertex()、addEdge()和getDistance()等方法的测试。我确保测试用例覆盖了各种情况，并且可以验证代码的正确性和可靠性。

运行测试：我在IntelliJ IDEA中使用JUnit运行器来执行测试用例。我选择了要运行的测试类，并在IDEA中启动测试。在测试执行完成后，我检查了测试结果，确保所有测试用例都通过了。

解决问题和困难：

依赖问题：在一开始，可能会遇到JUnit或其他依赖项的配置问题。我通过检查项目的构建文件来解决这些问题，并确保依赖项正确配置。

路径问题：有时在导入项目或添加依赖时，可能会遇到路径错误或不匹配的问题。我仔细检查项目结构和文件路径，并在必要时手动调整路径配置。

通过以上步骤，我成功地配置了开发、测试和运行环境，并顺利完成了实验的开发和测试工作。在整个过程中，我密切关注任何可能出现的问题，并及时解决，确保了项目的顺利进行。

URL: https://github.com/ComputerScienceHIT/HIT-Lab1-2022113077

# 实验过程

## Magic Squares

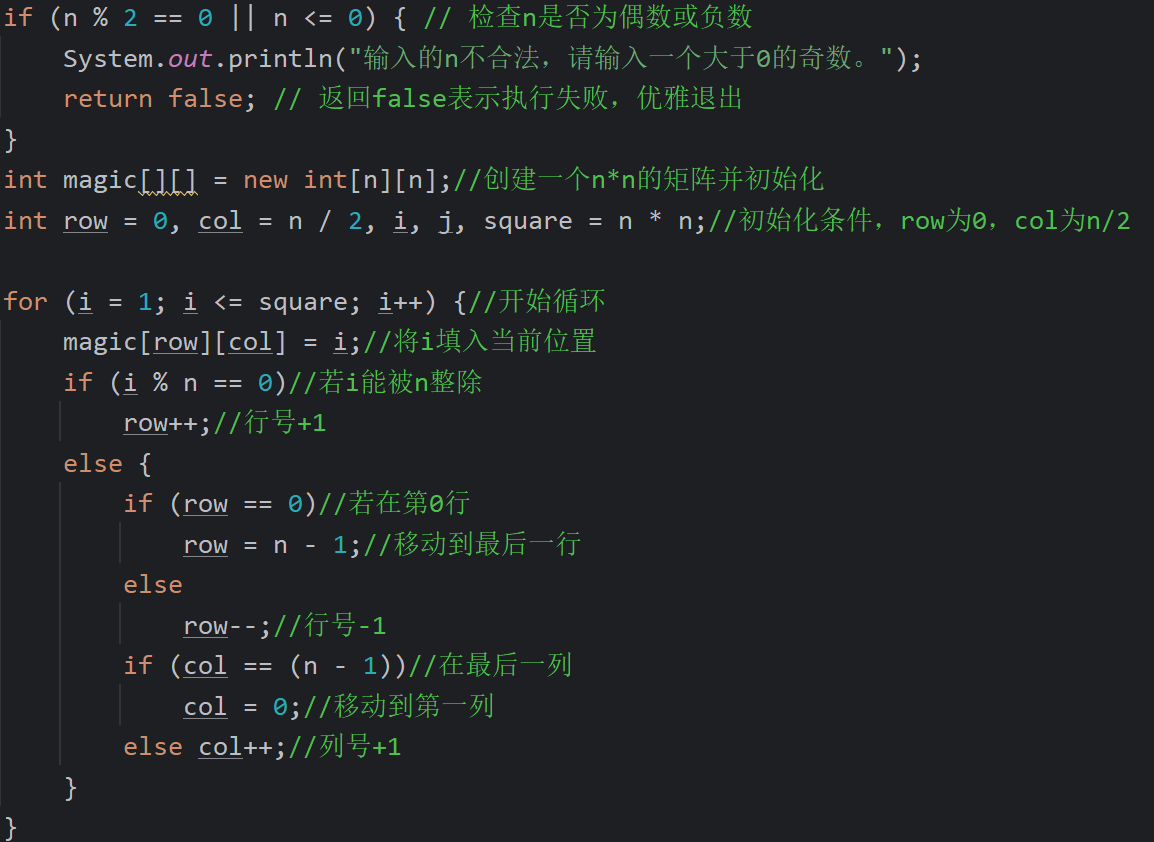
编写一个Java程序来检查矩阵的行/列/对角线和的值，并判断是否为幻方。

### isLegalMagicSquare()

设计了一个方法来检查给定文件中的矩阵是否符合幻方的定义。首先，我读取文件并解析矩阵数据。然后，我检查每行、每列和对角线的和是否相等，以验证矩阵是否为幻方。如果发现任何不符合幻方定义的情况，我将返回false并输出相应的错误提示信息。

### generateMagicSquare()

采用了经典的“奇数阶幻方生成法”，根据输入的奇数n，依次填充矩阵，并确保每行、每列和对角线的和相等。最后，我输出生成的幻方矩阵。



**程序流程图：**

开始

|

获取输入参数n

|

创建一个n×n的二维数组magic

|

初始化变量row为0，col为n / 2

|

初始化变量i和j

|

初始化变量square为n \* n

|

循环：i从1到square

| |

| 将i放入magic[row][col]

| |

| 如果i % n == 0

| | |

| | row增加1

| 否则

| | |

| | 如果row等于0

| | | |

| | | row设为n - 1

| | 否则

| | | |

| | | row减少1

| |

| | 如果col等于(n - 1)

| | | |

| | | col设为0

| | 否则

| | | |

| | | col增加1

|

循环结束

|

循环：i从0到n

| |

| 循环：j从0到n

| | |

| | 打印magic[i][j]

| |

| 打印换行

|

返回true

|

结束

**幻方的生成遵循如下规则：**

从1开始，按顺序填充1到n×n的数字。

从幻方的第一行中间列（即第0行，第n/2列）开始填充数字。

每次填充数字时，如果当前位置的数字是n的倍数，则下一个数字填充在当前位置的下一行（向下移动一行），否则向右上方填充（行数减一，列数加一），直到填满所有的位置为止。

**数学本质：**

从幻方的第一行中间列开始，将数字1放置在该位置。

从数字2开始，依次填充每个数字到幻方的下一个位置，按照特定规则选择下一个位置。

当填充的数字是奇数时，根据特定规则选择下一个位置：

首先，将起始位置设置为第一行的中间列，即第一行的中间位置。

将数字 1 放置在起始位置。

从数字 2 开始，依次填充每一个数字。

如果当前位置的数字是当前阶数的倍数（即 i%n == 0），则向下移动一行，保持列不变。

否则，将数字放置在当前位置的右上方（即行数减一，列数加一），但要注意边界条件，如果当前位置是第一行最右边的位置，则应该放置在最后一行的第一列；如果当前位置是最后一行的第一列，则应该放置在第一行的最后一列；其他情况按照常规处理。

重复步骤 3 直到填满整个幻方。

输出生成的幻方。

异常分析：这两个异常分别是 ArrayIndexOutOfBoundsException 和 NegativeArraySizeException。

ArrayIndexOutOfBoundsException 表示数组下标越界异常，即尝试访问数组中不存在的索引位置。在这个情况下，代码中出现了数组下标超出了数组范围的情况，导致程序抛出异常。在 generateMagicSquare 方法的第17行和第11行抛出了这个异常。这可能是由于输入的 n 值超出了数组的范围导致的，即超出了数组的行数或列数。

NegativeArraySizeException 表示数组大小为负异常，即尝试创建一个负数大小的数组。在这个情况下，代码尝试创建一个负数大小的二维数组，这是不允许的，因为数组的大小必须是非负数。这可能是由于输入的 n 值为负数导致的，因为幻方的阶数应该是一个正整数。

## Turtle Graphics

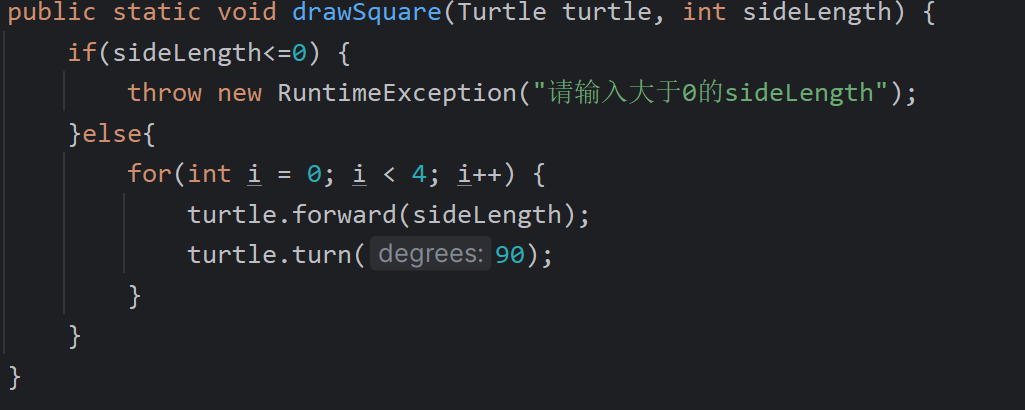
使用Turtle Graphics模拟绘制图形，并实现了drawSquare、计算多边形内角、计算旋转角度以及计算凸包等。和其他相关功能。

### Problem 1: Clone and import

从GitHub获取了任务的代码，并在本地创建了一个新的git仓库来管理代码。我使用了git clone命令克隆仓库，并导入了IntelliJ IDEA中。在克隆和导入过程中，我注意到了文件路径和依赖项的配置。

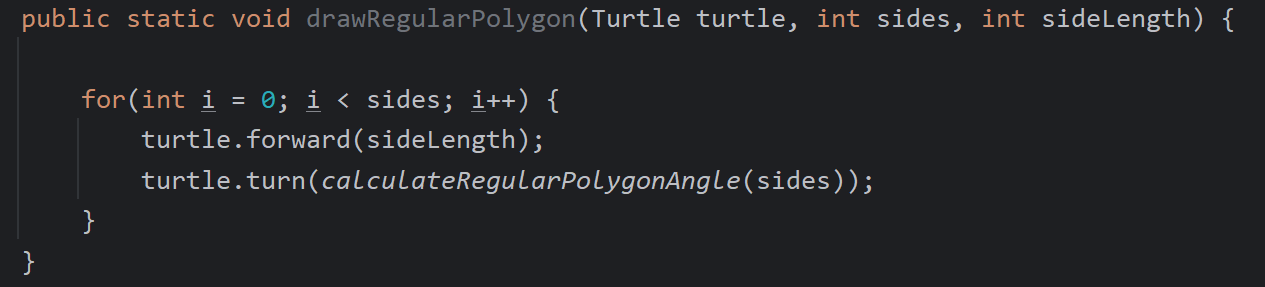
### Problem 3: Turtle graphics and drawSquare、

实现了drawSquare方法来绘制正方形。在这个方法中，我使用了Turtle对象的forward和turn方法来控制乌龟的移动和旋转，确保绘制出符合要求的正方形。我还添加了对sideLength参数的合法性检查，以确保输入的边长大于零。



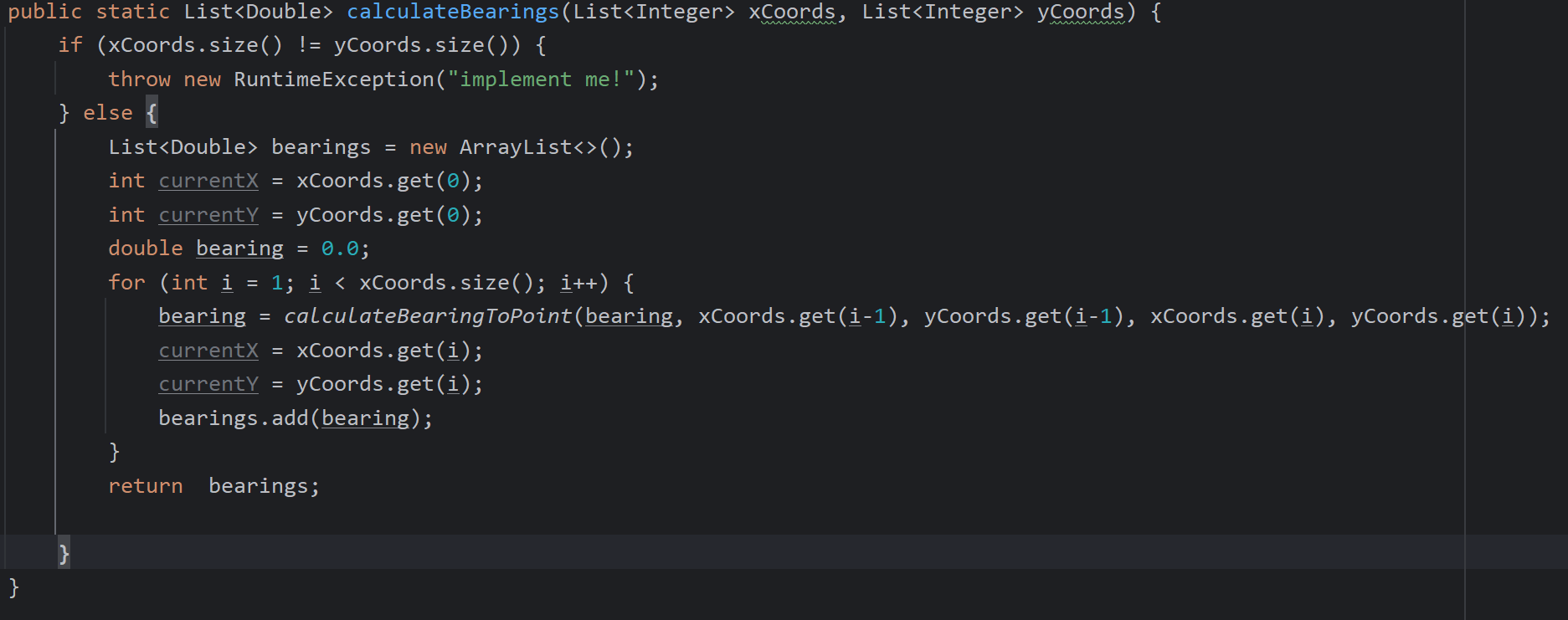
### Problem 5: Drawing polygons

设计了calculateRegularPolygonAngle方法来计算正多边形的内角，并实现了drawRegularPolygon方法来绘制正多边形。在drawRegularPolygon方法中，我循环调用了forward和turn方法来绘制多边形的边，并确保多边形的各边长度相等。



### Problem 6: Calculating Bearings

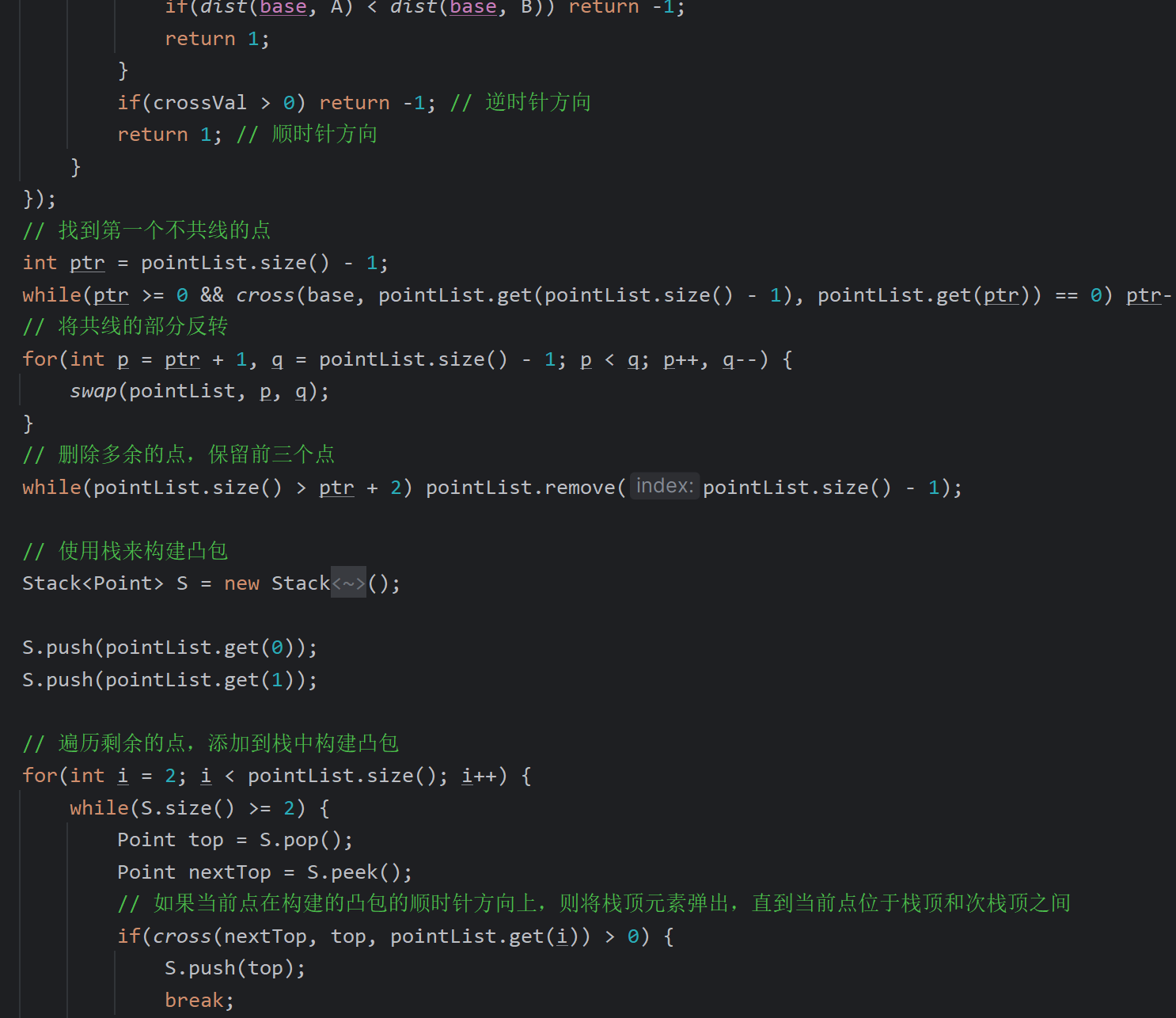
实现了calculateBearingToPoint方法来计算从当前点到目标点的旋转角度，并使用calculateBearings方法来计算一系列目标点的旋转角度。我使用了数学公式来计算角度，并确保正确处理旋转方向。



### Problem 7: Convex Hulls

实现了convexHull方法来计算一组输入点的凸包。我采用了“礼品包装算法”来计算凸包，确保输出的点集合形成了凸多边形。我还添加了辅助方法dist和cross来帮助计算凸包。

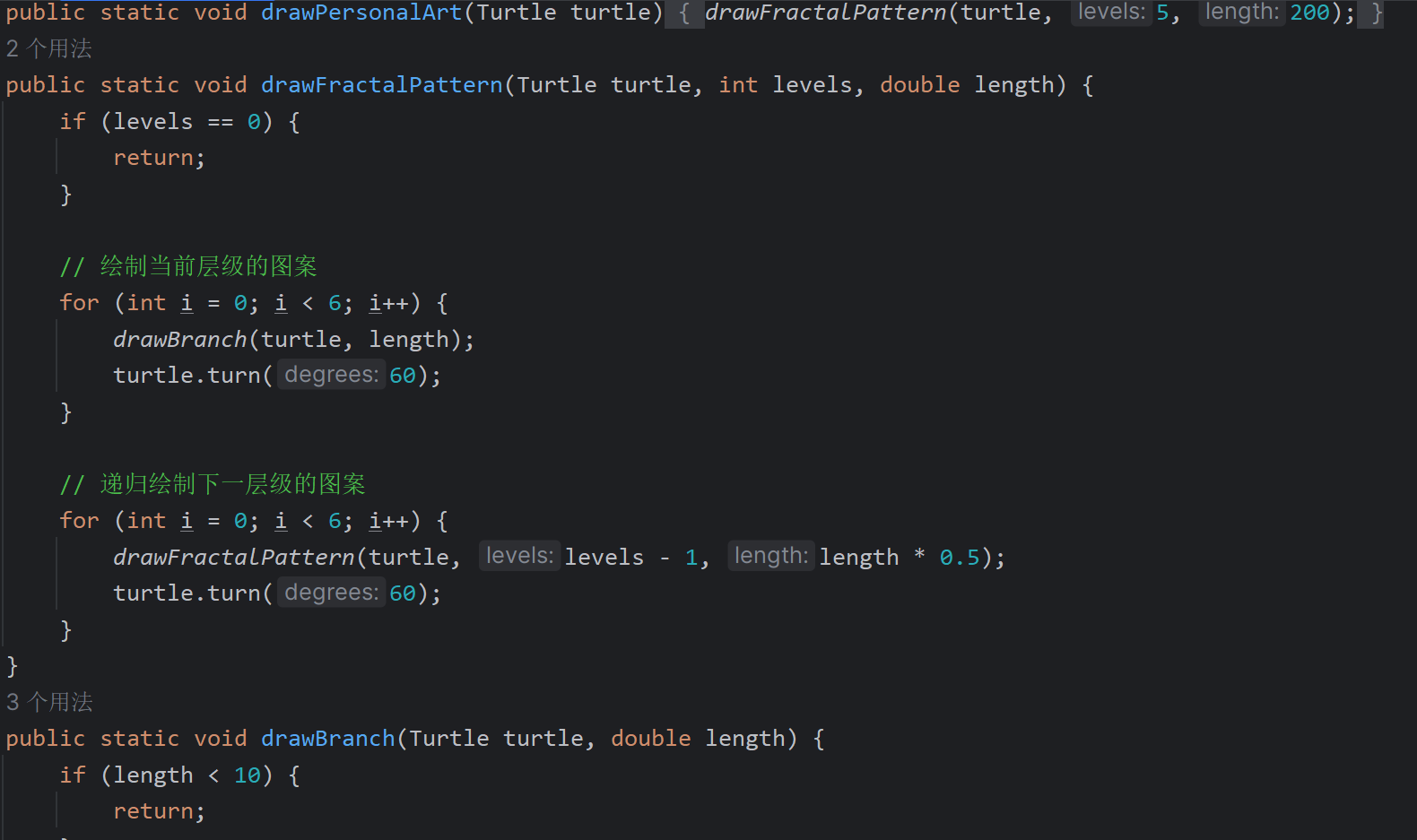


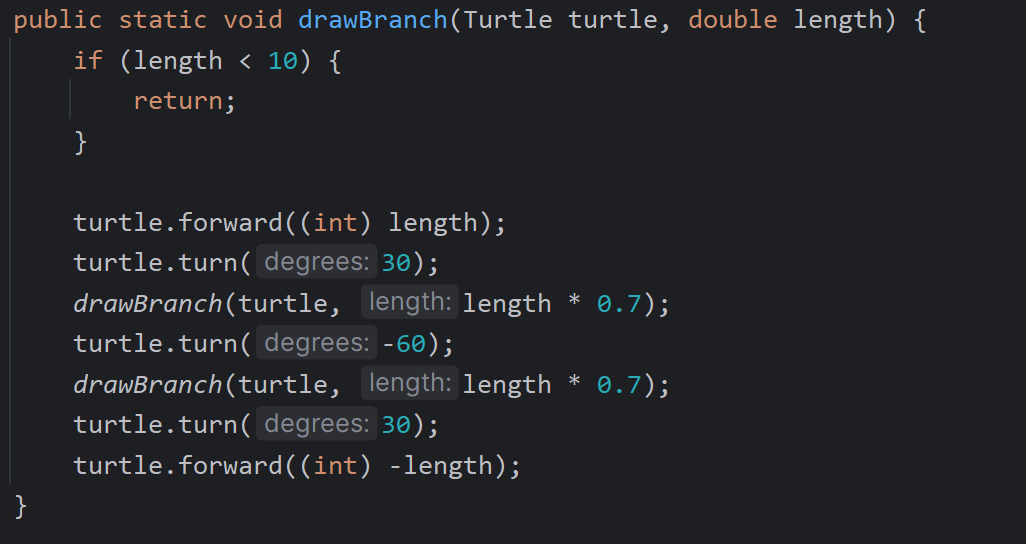




### Problem 8: Personal art

设计了drawPersonalArt方法来绘制个人艺术作品。在这个方法中，我运用了Turtle Graphics的各种功能来创作出独特的艺术效果。我也尝试了使用不同的颜色来增强作品的视觉效果。





### Submitting

我使用了Git将当前版本提交到GitHub上的Lab1仓库。我通过git add和git commit命令将修改提交到本地仓库，并使用git push命令将修改推送到远程GitHub仓库。我还查看了提交的结果，并确保代码被成功推送到了GitHub上。

## Social Network

我们需要设计一个表示社交网络中人际关系的系统。首先，我们需要实现 FriendshipGraph 类来表示社交网络的图形结构。其次，我们需要实现 Person 类来表示社交网络中的个人。最后，我们需要编写客户端代码和测试用例来验证我们的实现是否正确。

### 设计/实现FriendshipGraph类

在设计 FriendshipGraph 类时，我采用了图的邻接表表示法，使用 Map<Person, Set<Person>> 存储每个人及其朋友列表。这样可以快速查找某个人的朋友，并且添加边时也很高效。

我实现了以下方法：

addVertex(Person person)：用于向图中添加一个顶点（人）。

addEdge(Person person1, Person person2)：用于在两个人之间添加一条边（友谊关系）。

getDistance(Person source, Person destination)：使用广度优先搜索计算图中两个人之间的最短距离。

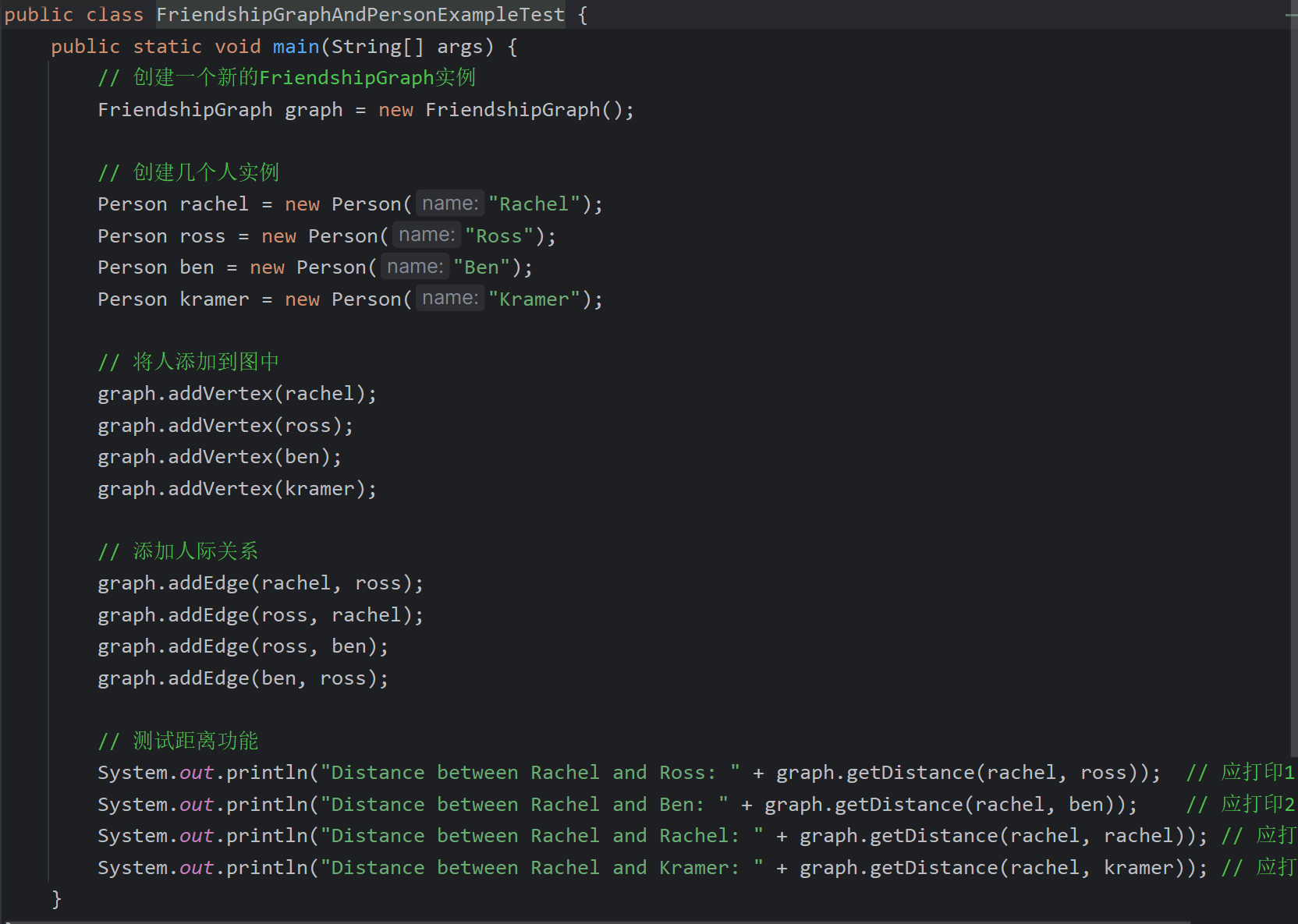
### 设计/实现Person类

在设计 Person 类时，我简单地表示一个人，只包含一个名字属性。这个类的主要目的是作为 FriendshipGraph 类中的键。



### 设计/实现客户端代码main()

在客户端代码中，我创建了一个 FriendshipGraph 对象，并测试了添加顶点、添加边以及获取距离的功能。主要思路是创建几个人对象，然后添加它们到图中，建立一些友谊关系，最后测试获取两个人之间的距离。



### 设计/实现测试用例

在设计和实现测试用例时，我遵循了单元测试的基本原则，针对每个方法编写了多个测试情况，包括正常情况、边界情况和异常情况，以确保代码的健壮性和正确性。我写了两个Java程序分别测试FriendshipGraph和Person

测试用例设计思路/过程/结果：

PersonTest 类：

testCreatePerson() 方法：测试创建 Person 对象时是否能正确设置姓名属性。

testEquals() 方法：测试 Person 对象的 equals() 方法，验证同名人、不同名人、与 null 比较以及与非 Person 对象比较的结果。

testHashCode() 方法：测试 Person 对象的 hashCode() 方法，验证同名人的哈希码是否相等。

FriendshipGraphTest 类：

testAddVertex() 方法：测试添加顶点的功能，包括添加单个顶点、添加重复顶点以及未添加的人之间的距离是否为 -1。

testAddEdge() 方法：测试添加边的功能，包括添加边后的距离计算是否正确，重复添加边的情况以及添加多条边后的距离计算是否正确。

testGetDistance() 方法：测试获取距离的功能，包括未添加边的情况、未添加顶点的情况、自己到自己的距离是否为 0，以及已添加边后两个人之间的距离是否正确计算。

通过这些测试用例，可以全面验证 FriendshipGraph 类和 Person 类的各种功能，确保它们能够正确地表示社交网络中的人际关系，并且具有正确的行为和性能。

**问题作答：**

如果将第10行注释掉，即只存在单向的社交关系 ross->rachel，那么根据广度优先搜索的原理，从 rachel 到 ross 的距离应为 -1，因为无法从 rachel 这一端到达 ross，所以期望的输出结果是 -1。实际执行程序后，结果应该与期望一致。

如果将第3行引号中的 "Ross" 替换为 "Rachel"，那么两个人的名字将相同，违反了“Each person has a unique name”的约束条件。程序执行时可能会出现异常或错误。修改FriendshipGraph 类和 Person 类，使得在发现重复名字时能够提示出错并结束程序运行。这样可以保证每个人都有唯一的名字。

# 实验进度记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 任务 | 实际完成情况 |
| 2024-03-22 | 18:30-20:30 | 编写问题1的isLegalMagicSquare函数并进行测试 | 按时完成 |
| 2024-03-22 | 20:30-22:30 | 编写问题1的  parseLine,checkMatrixDimensions，checkMatrixValues,calculateMagicSum,checkSums等函数并编写测试 | 延期1小时完成 |
| 2024-03-23 | 18:30-22:30 | 测试并研究问题一中给出的  generateMagicSquare函数并写出程序流程图 | 遇到困难，未完成 |
| 2024-03-24 | 18:30-24:30 | 将代码clone后着手编写问题2中TurtleSoup中的drawSquare、calculateRegularPolygonAngle、calculatePolygonSidesFromAngle、drawRegularPolygon、calculateBearingToPoint、calculateBearings、convexHull等函数 | 在编写  calculateBearingToPoint与convexHull遇到问题，无法推进。 |
| 2024-03-25 | 18:30-24:30 | 开始构思编写FriendshipGraph类和Person类 | 按时完成 |
| 2024-03-26 | 18:30-24:30 | 写FriendshipGraphAndPersonExampleTest、FriendshipGraphTest、PersonTest测试 | 按时完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的困难 | 解决途径 |
| 不会用Java读取文件 | 查阅JDK和Java官方文档学习了如何创建文件读取对象 |
| 不会按照\t分割字符 | 查阅JDK和Java官方文档学习了split用法 |
| 不知道convexHull函数中的礼包算法如何转化为Java语言 | 查阅CSDN进行学习和借鉴思想 |
| 不会用Java语言高效存储图这一数据结构 | 查阅JDK和Java官方文档和CSDN学习了如何用Map存储图 |

# 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. 合理规划时间：实验的任务可能比较繁重，因此需要提前规划时间，合理安排每个任务的完成时间，以确保能够按时完成。
2. 分阶段完成：将任务分解成小的阶段或子任务，逐步完成，有助于提高效率和控制进度。每完成一个阶段都可以对进度进行评估和调整。
3. 及时求助：遇到困难或不理解的地方，要及时向老师、助教或同学求助。有时候及时的解答或指导可以避免一些不必要的延误和错误。
4. 注重细节：在编写代码和测试时，要注重细节，特别是在边界条件和特殊情况下的处理。细心和严谨可以减少错误和调试的时间。
5. 反思总结：每完成一个任务或阶段，要进行反思和总结，分析自己的做法是否合理，有什么可以改进的地方，以及从中学到了什么经验和教训。
6. 持续学习：实验是一个学习的过程，要持续学习和探索新的知识和技能。不断地阅读文档、查找资料，提高自己的编程能力和解决问题的能力。
7. 自我管理能力：在没有团队协作的情况下，我需要更加自觉地管理时间和任务。这意味着要合理规划实验的时间，分配足够的时间用于学习和实践，以确保在截止日期前完成任务。
8. 独立解决问题的能力：没有团队成员一起讨论和解决问题，我需要依靠自己的能力来克服困难。这有助于提高自己的问题解决能力和独立思考能力。
9. 学会查阅文档和资源：在遇到困难或不清楚的地方，学会查阅相关的文档和资源是很重要的。这可以帮助我更快地理解问题的本质，并找到解决问题的方法。
10. 持续练习和反思：实验过程中遇到的挑战和解决方案都是宝贵的经验。通过持续练习和反思，我能够不断提升自己的编程能力和解决问题的能力，为将来的实验和项目做好准备。
11. 及时寻求帮助：如果遇到了无法解决的问题，要学会及时寻求帮助。这可以是向老师、同学或在线社区提问，及时解决困难，保证实验顺利进行。

窗体顶端

## 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？

Java编程语言对我来说口味还不错。相比其他编程语言，Java的优势在于它的跨平台性和强大的生态系统。Java可以在不同的操作系统上运行，这使得开发的应用程序具有很好的可移植性。此外，Java拥有丰富的标准库和大量的第三方库，提供了各种功能丰富的工具和框架，使得开发更加高效。然而，Java的不足之处在于相对于其他一些编程语言来说，它的语法和代码量可能稍显冗长，有时候需要编写更多的代码来完成相同的功能。

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

Eclipse和IntelliJ IDEA作为主流的Java集成开发环境（IDE），它们都有自己的优势和不足。Eclipse是一个开源的IDE，具有强大的可扩展性和丰富的插件生态系统。它适用于各种Java开发场景，并且对于初学者来说比较友好。但是，一些用户可能会认为Eclipse的性能和用户体验相对较差。相比之下，IntelliJ IDEA是一个商业化的IDE，具有更加强大的智能代码编辑功能和更好的性能。它提供了更多的高级功能和工具，但需要付费使用。

1. 关于Git和GitHub，是否感受到了它在版本控制方面的价值；

Git和GitHub在版本控制方面的价值是显而易见的。Git作为一个分布式版本控制系统，使得团队协作变得更加高效，能够轻松地管理和追踪代码的变更历史。GitHub作为一个代码托管平台，为开发者提供了一个共享和协作的平台，使得代码的交流和共享变得更加便捷。通过Git和GitHub，我们可以更好地管理项目，进行团队合作，以及参与开源社区的贡献。

1. 关于CMU和MIT的作业，你有何感受；

对于CMU和MIT的作业，我感受到了它们对于学生的要求非常严格和高效。这些作业通常涉及到丰富的知识和实践，需要学生投入大量的时间和精力来完成。通过这些作业，我学到了很多新知识和技能，并且提高了自己的问题解决能力和编程能力。

1. 关于本实验的工作量、难度、deadline；

关于本实验的工作量、难度和deadline，我觉得实验的工作量适中，难度适中。虽然实验涉及到了不少内容和任务，但是通过逐步完成每个任务，合理规划时间，还是可以比较轻松地完成的。关于deadline，我觉得在合理的时间内完成实验是非常重要的，可以通过提前规划和分配任务来避免临时抱佛脚，确保能够按时完成实验任务。

1. 关于初接触“软件构造”课程；

初接触“软件构造”课程让我感到非常兴奋和好奇。这门课程涵盖了很多软件开发的基础知识和技能，包括编程、版本控制、软件设计等方面。通过这门课程，我期待能够更深入地了解软件构造的原理和实践，提升自己的软件开发能力。