

**2019年春季学期  
计算机学院《软件构造》课程**

**Lab 3实验报告**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**目录**

[1 实验目标概述 1](#_Toc3922069)

[2 实验环境配置 1](#_Toc3922070)

[3 实验过程 1](#_Toc3922071)

[3.1 待开发的三个应用场景 1](#_Toc3922072)

[3.2 基于语法的图数据输入 1](#_Toc3922073)

[3.3 面向复用的设计：CircularOrbit<L,E> 1](#_Toc3922074)

[3.4 面向复用的设计：Track 2](#_Toc3922075)

[3.5 面向复用的设计：L 2](#_Toc3922076)

[3.6 面向复用的设计：PhysicalObject 2](#_Toc3922077)

[3.7 可复用API设计 2](#_Toc3922078)

[3.8 图的可视化：第三方API的复用 2](#_Toc3922079)

[3.9 设计模式应用 2](#_Toc3922080)

[3.10 应用设计与开发 2](#_Toc3922081)

[3.10.1 TrackGame 2](#_Toc3922082)

[3.10.2 StellarSystem 2](#_Toc3922083)

[3.10.3 AtomStructure 2](#_Toc3922084)

[3.10.4 PersonalAppEcosystem 2](#_Toc3922085)

[3.10.5 SocialNetworkCircle 2](#_Toc3922086)

[3.11 应对应用面临的新变化 2](#_Toc3922087)

[3.11.1 TrackGame 3](#_Toc3922088)

[3.11.2 StellarSystem 3](#_Toc3922089)

[3.11.3 AtomStructure 3](#_Toc3922090)

[3.11.4 PersonalAppEcosystem 3](#_Toc3922091)

[3.11.5 SocialNetworkCircle 3](#_Toc3922092)

[3.12 Git仓库结构 3](#_Toc3922093)

[4 实验进度记录 3](#_Toc3922094)

[5 实验过程中遇到的困难与解决途径 3](#_Toc3922095)

[6 实验过程中收获的经验、教训、感想 4](#_Toc3922096)

[6.1 实验过程中收获的经验和教训 4](#_Toc3922097)

[6.2 针对以下方面的感受 4](#_Toc3922098)

# 实验目标概述

目标是编写具有可复用性和可维护 性的软件，主要使用以下软件构造技术：

⚫ 子类型、泛型、多态、重写、重载

⚫ 继承、代理、组合

⚫ 常见的 OO 设计模式

⚫ 语法驱动的编程、正则表达式

⚫ 基于状态的编程

⚫ API 设计、API 复用

本次实验给定了五个具体应用（径赛方案编排、太阳系行星模拟、原子结构 可视化、个人移动 App 生态系统、个人社交系统），学生不是直接针对五个应用 分别编程实现，而是通过 ADT 和泛型等抽象技术，开发一套可复用的 ADT 及其 实现，充分考虑这些应用之间的相似性和差异性，使 ADT 有更大程度的复用（可 复用性）和更容易面向各种变化（可维护性）。。

# 实验环境配置

|  |  |
| --- | --- |
|  | 配置 |
| URL | <https://github.com/ComputerScienceHIT/Lab3-1170300614> |
| 系统 | Win10 |
| JDK版本 | 1.8.0\_16 |
| JAVA\_PATH | C:\Program Files (x86)\Common Files\Oracle\Java\javapath |

在这里给出你的GitHub Lab3仓库的URL地址（Lab3-学号）: <https://github.com/ComputerScienceHIT/Lab3-1170300614>

# 实验过程

## 待开发的三个应用场景

首先请列出你要完成的具体应用场景（至少3个，1和2中选一，3必选，4和5中选一，鼓励完成更多的应用场景）。

* 2. StellarSystem
* 3. AtomStructure
* 4 PersonalAppEcosystem

分析你所选定的多个应用场景的异同，理解需求：它们在哪些方面有共性、哪些方面有差异。

## 基于语法的图数据输入

使用正则表达式进行读取，考虑到各种变量的差异。我们下面分别分析

label：即([\\w+)word：([a-zA-Z](file:///\\w+)word：([a-zA-Z)]+)

sentence：([\\w\\s]+)

number：由科学记数法表示为(\\d{0,4}(.\d+)?|[1-9](.\\d+)?(e\\d+)?)。

angel:：(360(.0+)?|3[05]\\d(.\\d+)?|[12]\\d{2}(.\\d+)?|[19]\\d(.\\d+)?|\\d(.\\d+)?)

还有建立各个system时，通过使用静态变量的方式储存所有划分出的system

PS:StellarSystem中的Planet 匹配为([\\w+)\\s?,\\s?(\\w+)\\s?,\\s?(\\w+)\\s?,\\s?(\\d{0,4}(.\\d+)?|[1-9](.\\d+)?(e\\d+)?)\\s?,\\s?(\\d{0,4}(.\\d+)?|[1-9]0?(.\\d+)?(e\\d+)?)\\s?,\\s?(\\d{0,4}(.\\d+)?|[1-9](.\\d+)?(e\\d+)?)\\s?,\\s?(CC?W)\\s?,\\s?(360(.0+)?|3[0-5]\\d(.\\d+)?|[12]\\d{2}(.\\d+)?|[1-9]\\d(.\\d+)?|\\d(.\\d+)](file:///\\w+)\\s%3f,\\s%3f(\\w+)\\s%3f,\\s%3f(\\w+)\\s%3f,\\s%3f(\\d%7b0,4%7d(.\\d+)%3f|%5b1-9%5d(.\\d+)%3f(e\\d+)%3f)\\s%3f,\\s%3f(\\d%7b0,4%7d(.\\d+)%3f|%5b1-9%5d0%3f(.\\d+)%3f(e\\d+)%3f)\\s%3f,\\s%3f(\\d%7b0,4%7d(.\\d+)%3f|%5b1-9%5d(.\\d+)%3f(e\\d+)%3f)\\s%3f,\\s%3f(CC%3fW)\\s%3f,\\s%3f(360(.0+)%3f|3%5b0-5%5d\\d(.\\d+)%3f|%5b12%5d\\d%7b2%7d(.\\d+)%3f|%5b1-9%5d\\d(.\\d+)%3f|\\d(.\\d+))?)

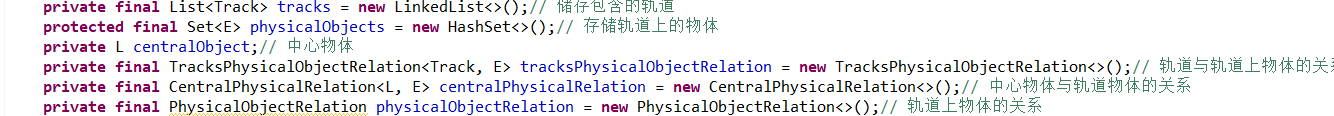
PersonalAppEcosystem的匹配([\\w+)\\s?,\\s?(\\w+)\\s?,\\s?([vV\\d.\\\_]+|(ver)?[vV\\d.\\-\_]+(ver)?[vV\\d.\\-\_]+)"\\s?,\\s?\"([\\w\\s]+)\"\\s?,\\s?\"([\\w\\s]+)\](file:///\\w+)\\s%3f,\\s%3f(\\w+)\\s%3f,\\s%3f(%5bvV\\d.\\_%5d+|(ver)%3f%5bvV\\d.\\-_%5d+(ver)%3f%5bvV\\d.\\-_%5d+)%22\\s%3f,\\s%3f\%22(%5b\\w\\s%5d+)\%22\\s%3f,\\s%3f\%22(%5b\\w\\s%5d+)\)"

其他不做赘述。

## 面向复用的设计：CircularOrbit<L,E>

设计思路我们沿用上个lab2的方法。首先设计初始化方法empty()返回一个实例。再使用ConcreteCirculatOrbit类来实现接口，当中由实验参考书知道L代表中心物体CentralObject，E代表轨道物体PhysicalObject。

**介绍一下变量还有使用的容器**



**分析一下方法的话：**就是根据参数列表的不同方法无法在底层实现，需要分别实现。最后就是各个应用的具体实现类，根据不同的应用实现独有的性质储存在这个具体类中。

## 面向复用的设计：Track

Track类是immutable的

**首先分析一下变量属性：**C:\Users\Don YAN\AppData\Roaming\Tencent\Users\1502833622\TIM\WinTemp\RichOle\4JF%)KKXWS2@SR$)@Y$KO18.png

**再分析一下方法：**

有get还有equals，hashcode.

## 面向复用的设计：L

由实验参考书要求知道，L即为CentralOrbit且CentralObject是immutable的类型。CentralObject：轨道系统的中心物体，设为一个接口。

## 所以L的属性有label;方法有get,hashcode还有equals.

由于StellarSystem, Atom Structure, Personal App EcosystemD的要求区别。注意其中的equals方法构造。

## 面向复用的设计：PhysicalObject

由实验参考书要求知：PhysicalObject也是immutable的类型。且PhysicalObject表示轨道物体，故我们设计为一个接口。

介绍一下属性：label，位置还有角度

分析方法知：有equals,hashcode还有获取角度，

PS:分析StellarSystem, Atom Structure, Personal App Ecosystem时根据要求具体实现个部分physicalObject。

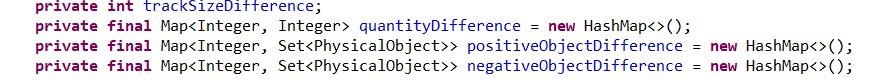
## 可复用API设计

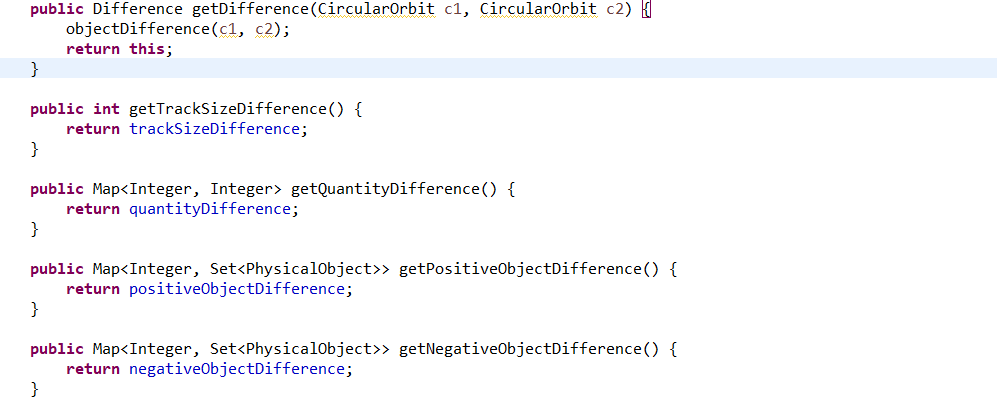
计算多轨道系统中各轨道上物体分布的熵值时：使用信息熵的公式

计算任意两个物体之间的最短逻辑距离。这里的逻辑距离是指：e1 和 e2 之 间通过最少多少条边（relation）即可连接在一起。两个物体之间若无关系， 则距离无穷大。首先判断两个物体是否相同，若相同则返回0，如不是相同的物体则将关系即距离都可以看作1，只需要对各个系统的physicalObject构成的图进行广度优先搜索即可。

计算任意两个物体之间的物理距离。若物体有具体位置，则可在直角坐标系 里计算出它们之间的物理距离。参数为1个轨道系统和2个轨道系统上的轨道物体，通过两个物体各自的角度和半径用余弦定理求出几何距离即可。

计算两个多轨道系统之间的差异：即difference类的构建过程。属性



方法有

## 图的可视化：第三方API的复用

使用Java原生的可视化库swing并实现动态可视化，观察轨道物体的运动情况。

## 设计模式应用

#### StellarSystem

注意使用 decorator 设计模式。首先继承PhysicalObject类，添加行星，颜色，半径，公转速度，公转方向等构造器介绍省略。再继承CentralObject类，并添加相关的变量，从文件读入轨道。最终建造行星系统

#### Atom Structure

类似于行星系统进行操作。使用 state 和 memento 设计模式管理电子跃迁 的状态，并可进行状态的恢复

**Personal App Ecosystem**

较为复杂。首先我们先继承了PhysicalObject还有CentralObject类。然后实现了正则表达式的文件读入。再完成各种操作。

## 应用设计与开发

利用上述设计和实现的ADT，实现手册里要求的各项功能。

以下各小节，只需保留和完成你所选定的应用即可。

### TrackGame

### StellarSystem

从文件中创建行星系统并实现动态可视化

添加轨道，轨道行星；删除轨道，行星，计算任意时间的行星位置

计算行星系统的熵，计算恒星与行星的逻辑距离，计算行星之间的物理距离

### AtomStructure

从文件中创建原子核系统并实现可视化

添加电子轨道，添加电子；删除电子轨道，电子；

计算原子核的熵

实现电子跃迁

### PersonalAppEcosystem

从文件中创建个人app生态系统并实现可视化当前系统

添加轨道,app;删除app,轨道

计算系统的熵,不同时间段轨道的差异,计算App之间的逻辑距离

### SocialNetworkCircle

## 应对应用面临的新变化

以下各小节，只需保留和完成你所选定的应用即可。

### TrackGame

### StellarSystem

行星轨道变成符合实际的椭圆形轨道（意即轨道的 属性从“半径”变化成描述椭圆的多个参数）

### AtomStructure

原子核需要表达为多个质子和多个中子，即处于中 心点的物体可以是多个物体构成的集合

### PersonalAppEcosystem

为轨道系统增加“扇区”特性，根据其中包含的所 有 App 的“业务领域”属性进行聚类，隶属于同一 业务领域的所有 App 放在同一个扇区里

### SocialNetworkCircle

## Git仓库结构

请在完成全部实验要求之后，利用Git log指令或Git图形化客户端或GitHub上项目仓库的Insight页面，给出你的仓库到目前为止的Object Graph，尤其是区分清楚312change分支和master分支所指向的位置。

# 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 | 计划任务 | 实际完成情况 |
| 2019.3．20 | All day | 了解实验参考书 | 基本完成 |
| 3.21-3.29 | Freeday | 完成基本框架的构建 | 没有实现 |
| 3.30-3.31 | All day | 实现框架的实现 | 实现失败 |
| 4.01-4.06 | Free day | 实现底层的框架重构 | 基本实现 |
| 4.09-4.15 | Freeday | 实现主要方法的构造 | 基本完成 |
| 4.16-4.22 | All day | 实现StellarSystem,atom,以及personapp的创建 | 没有完成 |
| 4.23-4.29 | Free day | 实现以上系统的建立 | 基本完成 |
| 5.01-5.04 | Free day | 实现新加入功能 | 基本完成 |
| 5.05 | Freeday | 完成代码完善 | 基本完成 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的难点 | 解决途径 |
| 无法很好理解实验参考书的逻辑结构 | 与同学谈论梳理实验结构 |
| 建造circularOrbit出现问题 | 与同学一起谈论 |
| 进行的GUI操作失误 | 上网进行swing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

通过本次实验了解了如何进行可复用性和可维护性软件的编程过程。同时也了解到自己对于Java使用的不当之处。

## 针对以下方面的感受

（1）重新思考Lab2中的问题：面向ADT的编程和直接面向应用场景编程，你体会到二者有何差异？本实验设计的ADT在五个不同的应用场景下使用，你是否体会到复用的好处？

答：面向ADT编程极大提高了代码的服用率并扩大了适用范围，无需重新编程即可将原有代码带入到新的使用场景，减少了不必要的麻烦。同时也解放了大量的生产力。

1. 重新思考Lab2中的问题：为ADT撰写复杂的specification, invariants, RI, AF，时刻注意ADT是否有rep exposure，这些工作的意义是什么？你是否愿意在以后的编程中坚持这么做？

答：通过使用rep exposure可以减少运行过程中的错误，增加了数据的安全性，增强了代码的健壮性。非常愿意继续使用。

1. 之前你将别人提供的API用于自己的程序开发中，本次实验你尝试着开发给别人使用的API，是否能够体会到其中的难处和乐趣？答：是的
2. 在编程中使用设计模式，增加了很多类，但在复用和可维护性方面带来了收益。你如何看待设计模式？

答很有必要：

1. 你之前在使用其他软件时，应该体会过输入各种命令向系统发出指令。本次实验你开发了一个解析器，使用语法和正则表达式去解析输入文件并据此构造对象。你对语法驱动编程有何感受？

答：使用起来比较陌生，但是应该尝试

1. Lab1和Lab2的大部分工作都不是从0开始，而是基于他人给出的设计方案和初始代码。本次实验是你完全从0开始进行ADT的设计并用OOP实现，经过三周之后，你感觉“设计ADT”的难度主要体现在哪些地方？你是如何克服的？

答：自主设计的框架可能会有很多的问题，自己慢慢摸索。

1. 你在完成本实验时，是否有参考Lab4和Lab5的实验手册？若有，你如何在本次实验中同时去考虑后续两个实验的要求的？

答：没有。

1. 关于本实验的工作量、难度、deadline。

答：难度较大

1. 到目前为止你对《软件构造》课程的评价。

答：难度有点大。