|  |  |
| --- | --- |
| **学 号：** | 0122010870126 |

****

**软件工程实践（一）**

|  |  |
| --- | --- |
| **学 院** | **计算机与人工智能学院** |
| **专 业** | **软件工程** |
| **班 级** | **软件2002** |
| **姓 名** | **陈元硕** |
| **编 号** | **0122010870126** |
| **指导教师** | **唐祖锴** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2022 | 年 | 12 | 月 | 31 | 日 |

**目录**

[1 任务概述 1](#_Toc91495513)

[2 任务分析 1](#_Toc91495514)

[3 开发计划 1](#_Toc91495515)

[4 软件配置计划 1](#_Toc91495516)

[5 测试计划 1](#_Toc91495517)

[6 实施情况 1](#_Toc91495518)

[7 实施过程问题记录与分析 2](#_Toc91495519)

[8 任务总结 2](#_Toc91495520)

[9 参考文献 2](#_Toc91495521)

# 1 任务概述

**任务目的：**

理解软件代码规范的重要性

理解代码变化对软件质量带来的影响

掌握基于Git的个人代码版本维护方法

掌握MarkDown文件编写方法

**任务内容：**

阅读、理解和标注样例代码  
分析和学习代码质量特征、设计方法和编程风格  
运用所学方法，对开源代码进行标注  
对样例工程进行简单功能扩充和维护

**任务要求：**

**1.** 阅读和描述样例工程  
  **\*** fork样例工程，并clone到本地仓库；  
  **\*** 在本地开发环境上运行样例工程，理解样例工程的代码逻辑；  
  **\*** 精读样例工程软件代码，描述代码结构及部件组成；  
  **\*** 以UML图描述样例工程的组成及结构图（类及类之间的关系）  
  **\*** 可结合markdown语法和mermaid插件绘制所需图形  
  
**2.** 标注样例工程中的代码  
  **\*** 基于javadoc规范标注代码，对包、类、方法、代码片段、参数和语句等代码层次进行注释（可参考Game类的标注样例）；  
  **\*** 注释后的代码提交到本地代码库后，同步推送到远程代码仓库；  
  **\*** 可参考ESLint、github/super-linter等开发插件了解关于代码规范的相关知识；  
  
**3.** 扩充和维护样例工程  
  **\*** 对样例代码中的功能设计进行分析，找出若干设计缺陷和改进点，并进行修正或扩充，并集成到工程代码中；  
  **\*** 可借助代码质量分析工具或代码规范检查工具对代码质量进行分析，发现潜在问题；  
 **>** 提示：样例工程的代码结构存在一些可以改进的功能点，可参考下列说明进行改进：  
 在Game类的processCommand()方法中，当用户输入的命令被辨认出来以后，有一系列的if语句用来分派程序到不同的地方去执行。从面向对象的设计原则来看，这种解决方案不太好，因为每当要加入一个新的命令时，就得在这一堆if语句中再加入一个if分支，最终会导致这个方法的代码膨胀得极其臃肿。如何改进程序中的这个设计，使得命令的处理更模块化，且新命令的加入能更轻松？请描述你的解决思路，并对你的解决方案进行实现和测试。  
  
**4.** 功能扩充点  
  **\*** 样例工程“world-of-zuul”具备最基本的程序功能，该项目具有极大的扩展空间，各位同学可选择或自行设计系统结构优化或功能扩充需求，完成3项左右的功能扩充实现；  
  
 **>** 可供参考的结构优化或功能扩充项包括但不限于以下内容：  
 **1.** 扩展游戏，使得一个房间里可以存放任意数量的物件，每个物件可以有一个描述和一个重量值，玩家进入一个房间后，可以通过“look”命令查看当前房间的信息以及房间内的所有物品信息；  
 **2.** 在游戏中实现一个“back”命令，玩家输入该命令后会把玩家带回上一个房间；  
 **3.** 在游戏中实现一个更高级的“back”命令，重复使用它就可以逐层回退几个房间，直到把玩家带回到游戏的起点；  
 **4.** 在游戏中增加具有传输功能的房间，每当玩家进入这个房间，就会被随机地传输到另一个房间；  
 **5.** 在游戏中新建一个独立的Player类用来表示玩家，并实现下列功能需求：  
 **\*** 一个玩家对象应该保存玩家的姓名等基本信息，也应该保存玩家当前所在的房间；  
  **\*** 玩家可以随身携带任意数量的物件，但随身物品的总重量不能操过某个上限值；  
  **\*** 在游戏中增加两个新的命令“take”和“drop”，使得玩家可以拾取房间内的指定物品或丢弃身上携带的某件或全部物品，当拾取新的物件时超过了玩家可携带的重量上限，系统应给出提示；  
  **\*** 在游戏中增加一个新的命令“items”, 可以打印出当前房间内所有的物件及总重量，以及玩家随身携带的所有物件及总重量；  
  **\*** 在某个或某些房间中随机增加一个magic cookie（魔法饼干）物件，并增加一个“eat cookie”命令，如果玩家找到并吃掉魔法饼干，就可以增长玩家的负重能力；  
 **6.** 扩充游戏基本架构，使其支持网络多人游戏模式，具备玩家登陆等功能；  
 **7.** 为单机或网络版游戏增加图形化用户界面，用过可以通过图形化界面执行游戏功能；  
 **8.** 可以为游戏增加数据库功能，用于保存游戏状态和用户设置；  
 **9.** ......  
  
**5.** 编写测试用例  
  **\*** 针对功能改进和扩充，在项目结构中编写单元测试用例，对代码执行单元测试；

# 2 任务分析

本次任务重点在于，基于“巨洞冒险”游戏展开软件开发。为了实现这一目的，需要对程序的内容进行详细阅读，理解游戏的运行过程，针对游戏的部分缺点进行补充完善，并扩充游戏内容。

在对程序的分析过程中，使用了UML建模技术，并绘制了相关的类图，软件开发过程使用JAVA语言的IDEA软件进行开发，并通过GITHUB实现代码管理。本次任务的重点在于开发者对程序的详细理解，以及立足于自身理解对程序进行改进。

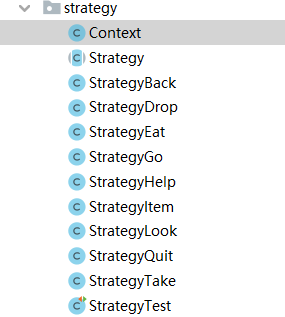
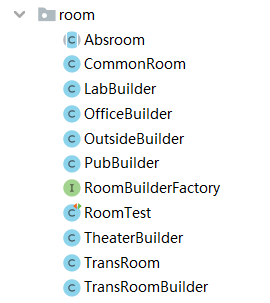
# 3 开发计划

1. 阅读代码，通过Javaboc对代码中实现的方法添加注释，并利用软件亿图图示画出类图建模
2. 创建markdown文件REPORT，利用markdown语法记录软件开发过程
3. 使用IntelliJ IDEA Community Edition 2022.1.2来进行软件开发
4. 在对代码更改、添加之后使用Github进行版本管理
5. 开发完成后利用Junit进行代码测试

# 4 软件配置计划

**版本的编码规范：**使用Ver.\*.\*进行版本编码，比如Ver.2.1意为第二个版本的第一次更新

**命名规范：**功能的实现采用上下文模式，而房间的创建使用工厂模式，命名结构如下：



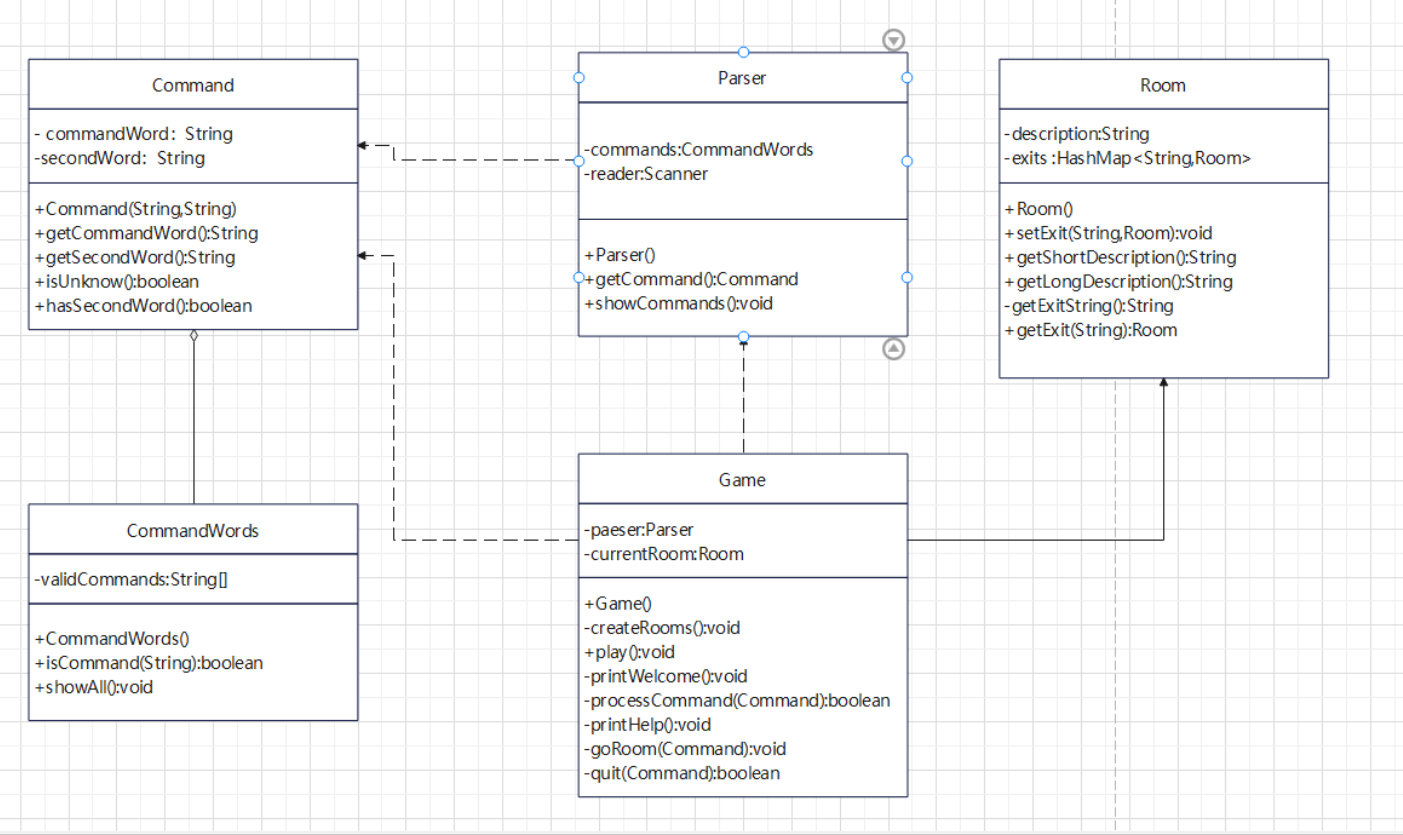
**分支管理规范：**开发始终在master分支下进行合并

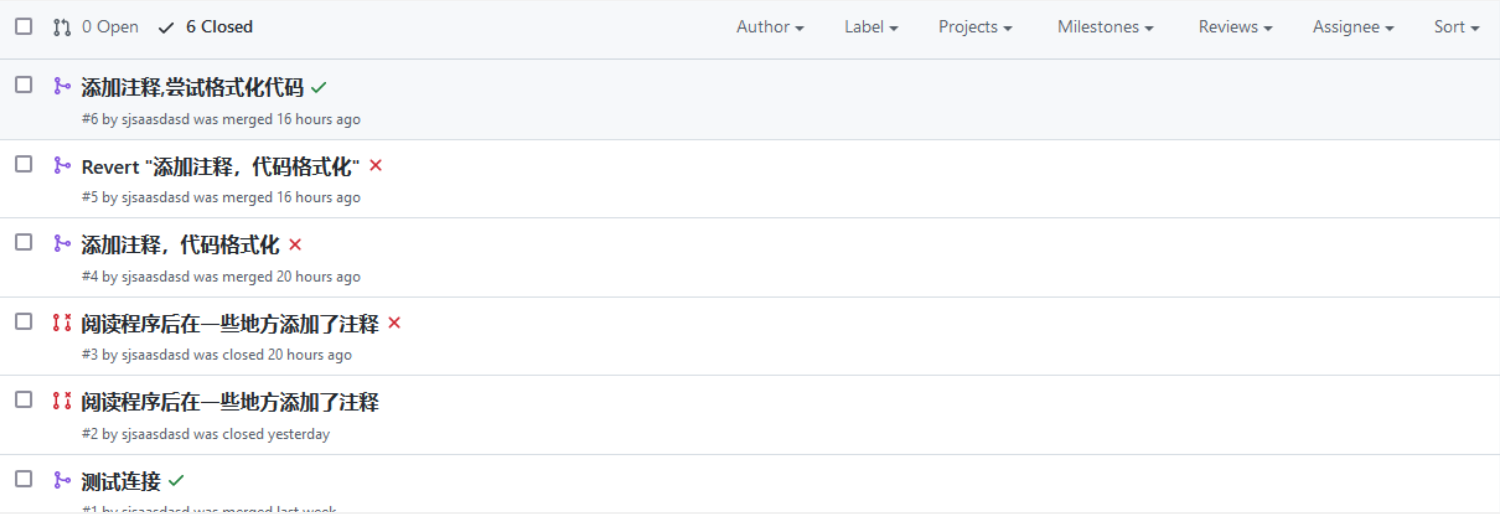
**提交规范：**版本的编码＋更新说明

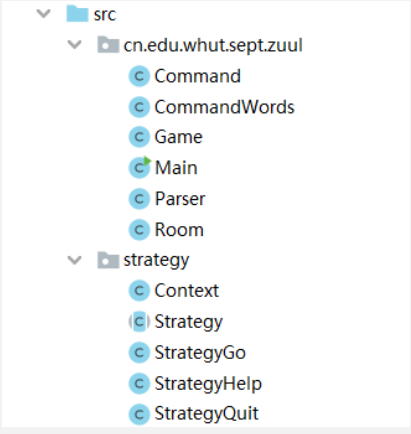
# 5 测试计划

在实验过程中使用Junit对代码进行单元测试，主要的测试对象为strategy类中实现的方法，和Romm类抽象之后的部分方法，两个测试类分别命名为RoomTest和StrategyTest。

# 6 实施情况 1.阅读和描述样例工程 / 标注样例工程中的代码 在网上申请了班级任务后，把代码fork到了仓库sept1-sjsaasdasd-1，详细阅读了代码之后在javadoc 中添加了注释，并建立了以下类图：

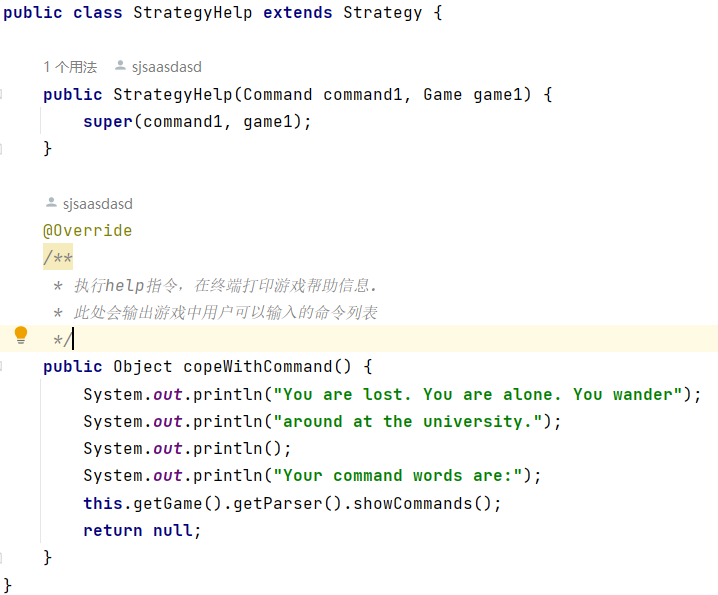
  
 之后尝试将新的代码加入仓库首先要将它提交到本地仓库，然后从网站上create new pull request前几次因为注释的位置导致代码格式化审核出现问题，最后一次才成功

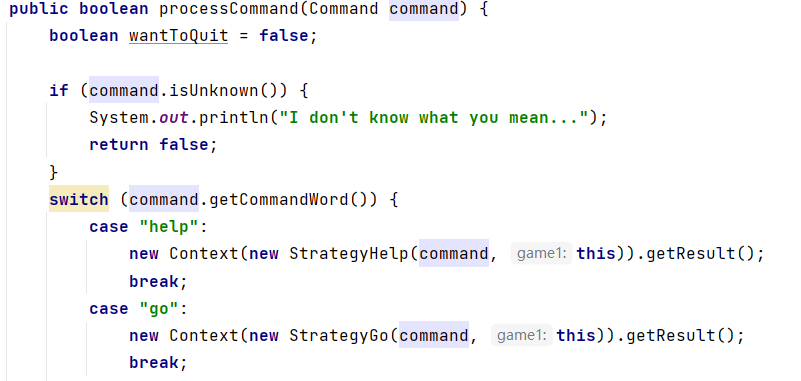
  
 到此为止对GitHub的使用和代码的基本框架有了较为详细地了解  
**2.扩充和维护样例工程**  
问题说明：  
在Game类的processCommand()方法中，当用户输入的命令被辨认出来以后，有一系列的if语句用来分派程序到不同的地方去执行。从面向对象的设计原则来看，这种解决方案不太好，因为每当要加入一个新的命令时，就得在这一堆if语句中再加入一个if分支，最终会导致这个方法的代码膨胀得极其臃肿。如何改进程序中的这个设计，使得命令的处理更模块化，且新命令的加入能更轻松？请描述你的解决思路，并对你的解决方案进行实现和测试。  
 经过查找资料之后发现可以采用“策略模式”来减少代码中的if——else语句，实现算法的上下文分离。策略模式需要定义一个上下文对象Context、抽象的策略角色Strategy，并在  
 功能的实现中将策略角色Strategy具象化，通过继承来实现不同操作。

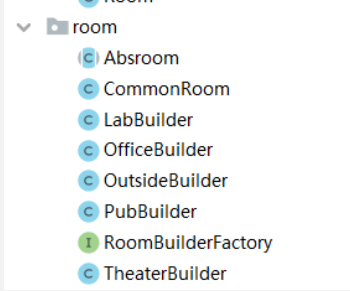


这里我们通过继承将动作“go”，“help”，“quit”等进行分离，具体操作通过重写Strategy类的copeWithCommand()方法实现最终我们得到的。下边展示一下样例类的构成：  
Context类  
  
Strategy类  


最终在每个Strategy的子类中copeWithCommand()方法的实现结果和原本在Game类中的实现方法差别不大，但返回值设置为Object令方法显得更为灵活，例如StrategyHelp类：

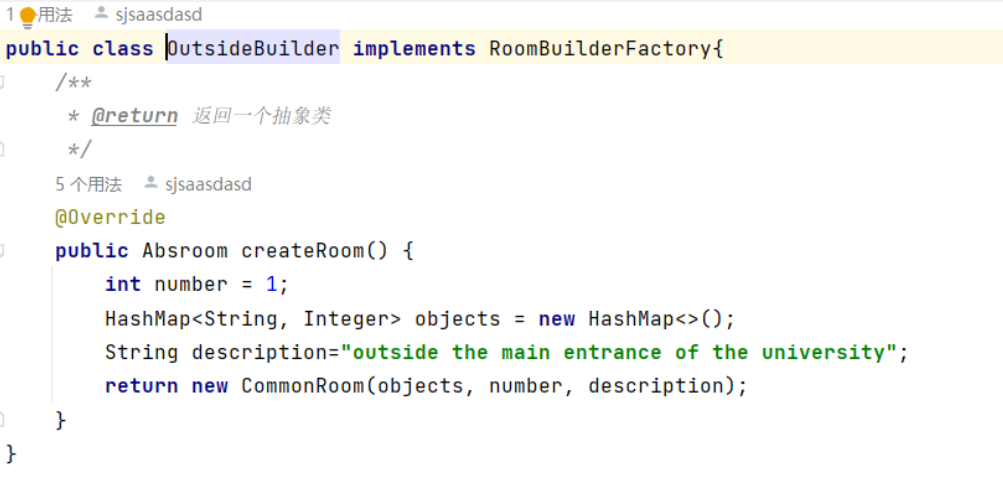


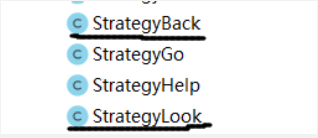
优化后的processCommand()如下：  
  
 优化后我们只需要根据新的操作定义新的策略，然后传入Command，Game对象即可。

**3. 功能扩充点**  
**扩充点①**：将Room类修改为抽象的Absroom类，利用工厂模式来创建Room，并在房间中添加物品  
  
 这次修改的重点在于Absroom，RoomBuilderFactory两个类。其中Absroom本质上是将Room类抽象并在其中添加保存房间内物品的Map型变量

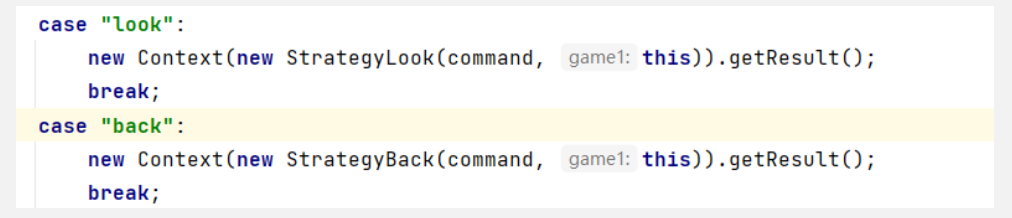
  
 RoomBuilderFactory实际上是一个接口，包含了生成一个房间的抽象方法createRoom()

  
 在这个接口的方法实现时，实际上是将初始化的内容在createRoom方法中补全，利用CommonRoom的构造方法返回一个“房间”例如：

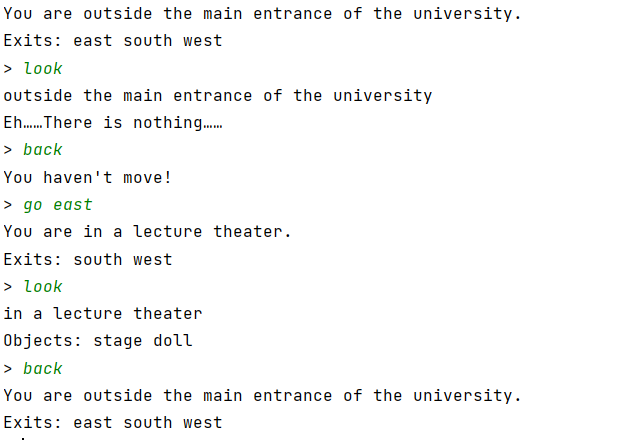
  
**修改点②**：增加look和back功能  
 利用策略模式将look和back的策略实现，然后在CommandWords类和Game类中添加相应的字符串和方法调用即可。







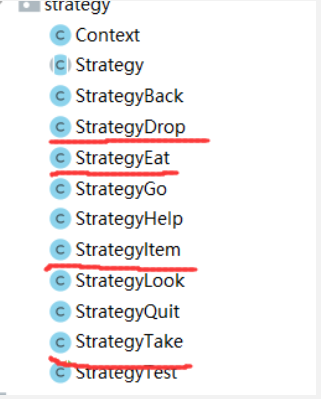
**功能演示：**



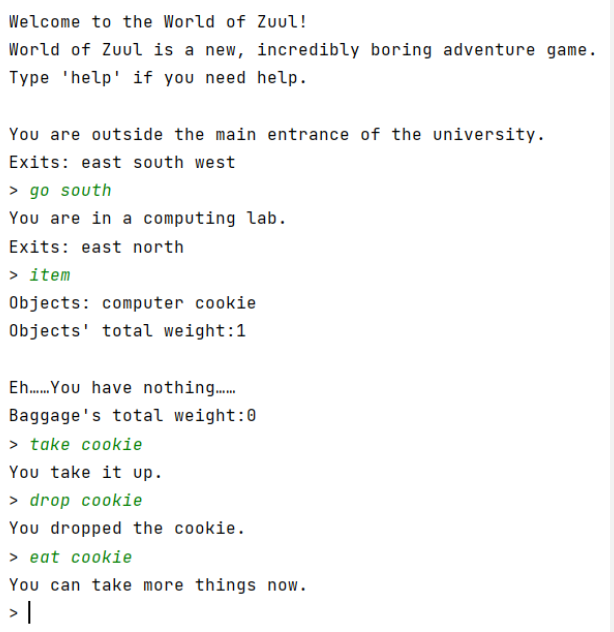
提交结果：



**扩充点③：**增加魔法饼干和eat操作，添加take，drop物品的功能和item显示房间内物品与玩家的行李

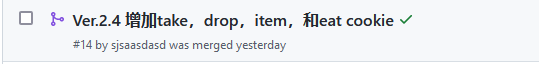


**功能演示：**



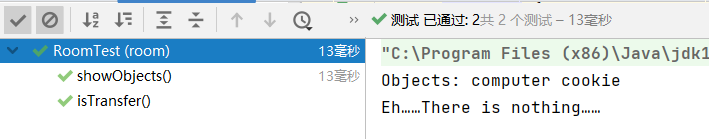
到此为止完成功能的扩充。

提交结果：

  
**4. 测试设计**  
 关于测试主要针对Strategy和Room两个大类进行目的在于测试关键的方法运行： **RoomTest**



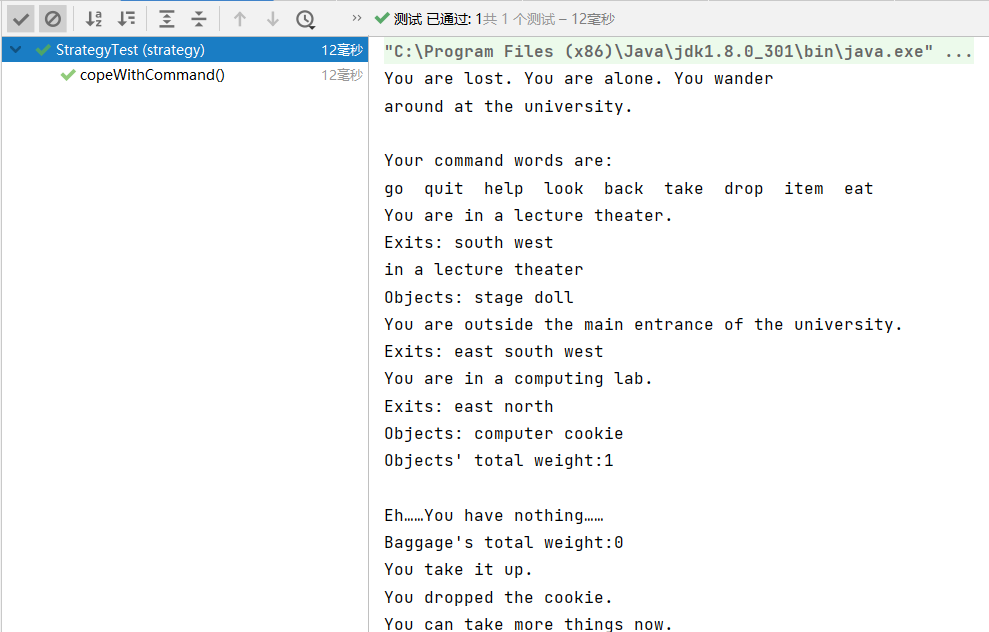
测试结果：



**StrategyTest**

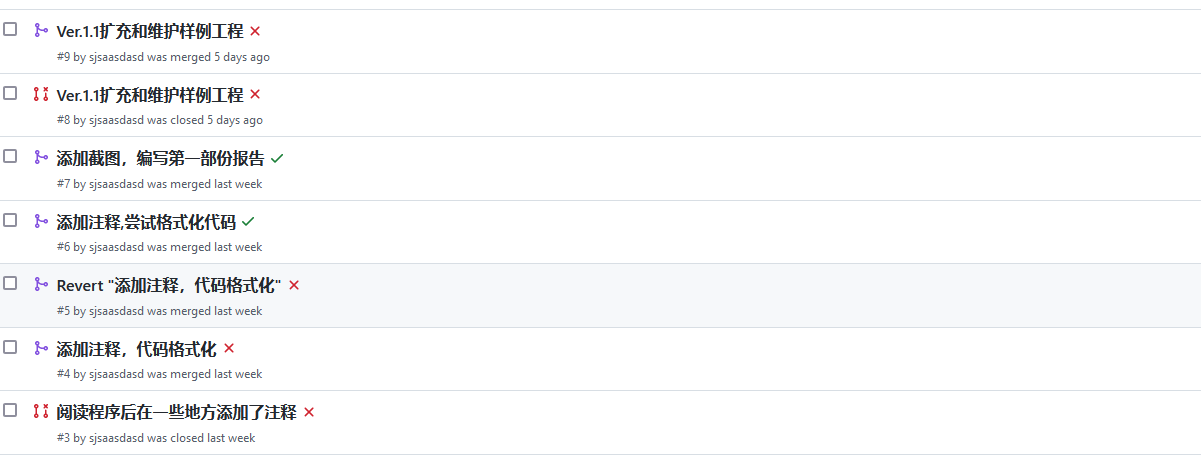


测试结果：

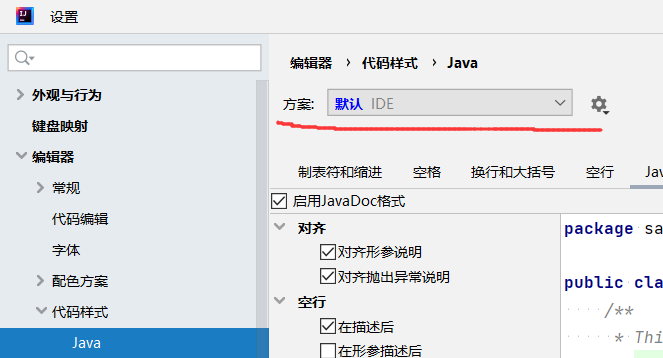


# 7 实施过程问题记录与分析

在开发过程中遇到的主要问题是代码格式的问题，在前几次提交时由于代码格式的问题均没有成功：



最后选择从外部导入代码规范：



最终代码格式的问题得到了解决。

# 8 任务总结

本次实验中我个人的收获是十分大的，起初我是不太清楚Github的使用方法的，在接受任务之后便对相关的操作进行查询，最终算是可以较为熟练的使用Git进行代码管理，并简单地学习了markdown语法对REPORT文档进行了补充，让整个开发过程更有条理。

另外在进行代码分析和程序优化扩展的途中也在网络上查询了部分资料，依次来完成对游戏结构的优化，如果没有这些资料修改起来是比较麻烦的。总之在本次实验中我认识到了代码管理的重要性，通过版本迭代来有节奏地进行软件开发，同时也要灵活运用手边的资料，可以让自己的工作事半功倍。

# 9 参考文献

**[1]汪成亮, 张聪, 张勤. UML技术在实际工程中的应用[J]. 计算机工程与应用, 2002, 38(1):4.**

**《软件工程实践（一）》成绩评定表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓 名** | | **陈元硕** | **学 号** | **0122010870126** | |
| **专业、班级** | | **软件2002班** | | | |
| **成绩评定：** | | | | | |
| 评价内容 | | | 满分 | 实得分 | |
| 得分 | 小计 |
| 实践任务  完成情况 | 软件项目设计、改进与扩充 | | 20 |  |  |
| 个人软件过程与项目管理 | | 15 |  |
| 代码版本管理 | | 25 |  |
| 代码注释与编码规范 | | 25 |  |
| 单元测试 | | 15 |  |
| 实践报告  总评情况 | 学习态度与考勤 | | 10 |  |  |
| 报告格式的规范性 | | 10 |  |
| 报告的逻辑结构与语言表达 | | 15 |  |
| 实践内容的正确性与合理性 | | 60 |  |
| 文献引用及标注 | | 5 |  |
| 总分 | | | 100 |  |  |
| **最终评定成绩（以优、良、中、及格、不及格评定）** | | | | | |

**指导教师签字：**

**年 月 日**