Version 4.01; Feb. 15, 2022

编译原理预备实验

议程管理系统 *Agenda*

2-3 weeks

目 录

1 实验描述

在开始实验之前，请务必花足够时间阅读完本文档关于实验的描述与约定！

1.1 实验目的

在初步掌握 Java 语言和面向对象程序设计方法后，本实验帮助学生巩固对封装、信息隐藏、数据抽象、异常处理等机制的理解与运用，并进一步掌握继承、运行时多态性、程序包、多层体系结构、设计模式等机制，从而顺利过渡到后续的编译原理实验。

1.2 实验环境

本实验基于当前主流的面向对象开发平台，编码规范遵循主流的参考规范。

1.2.1 编程语言

Java 语言，JDK 1.5 或以上版本。

1.2.2 开发工具

学生可自由选择 Eclipse、JBuilder 等 IDE 环境，也可直接采用 UltraEdit、EditPlus 等编辑器在命令行工作。但提交的实验结果必须独立于特定的 IDE，可直接运行在 JDK 上。

1.2.3 编码规范

你需要采用面向对象风格来设计实验中的所有类，并遵循一种良好的程序设计习惯。例如，如果你将一个程序的很多功能全部放在一个长长的 `main()` 方法中实现，这样的设计与编码风格会被扣分。

学生在实验过程中应注意培养规范的编码风格。本实验要求所有源代码严格遵循 Sun 公司（现为 Oracle 公司）关于 Java 程序设计语言的编码规范（Code Conventions for the Java Programming Language，Revised April 1999），参见：https://www.oracle.com/java/technologies/javase/codeconventions-contents.html。

完成后的代码应使用 JDK 附带的文档工具 javadoc 根据源程序中的文档化注释生成相应的文档。

1.3 实验内容

采用面向对象方法和 Java 语言开发一个基于命令行交互方式的议程（agenda）管理系统。

已注册到系统的用户（user）可添加（add）、删除（delete）、查询（query）系统中记录的会议（meeting）安排；系统还提供新用户注册（register）、清除（clear）某一用户所有会议安排等管理功能。

1.4 功能需求

本实验要求完成的议程管理系统至少应实现如下功能。

1.4.1 用户注册

注册新用户时，应为该用户设置一个唯一的用户名以及一个密码。

如果注册时提供的用户名已由其他用户使用，应反馈一个适当的出错信息；成功注册后，亦应反馈一个成功注册的信息。

1.4.2 添加会议

已注册的用户可以添加一个新会议到其议程安排中。规定会议仅允许在两个已注册的用户间举行，不可创建无另一已注册用户的会议。

添加会议时提供的信息应包括：会议的起始时间（start time）和终止时间（end time）、描述本次会议的标签（label）、以及预约的另一用户名字（scheduled user）。成功添加会议后，它应既出现在发起该会议的用户的议程中，也应出现在该会议所预约的另一用户的议程中。

注意，用户不允许分身参加多个会议，即如果用户自己或被预约的用户已有一个会议安排与新会议在时间上相互重叠（overlap），该新会议将无法成功添加到议程管理系统中。

用户在添加会议后，应获得适当的反馈信息，以得知是成功地添加了新会议，还是在添加过程出现了某些错误。

1.4.3 查询会议

已注册的用户可以查询自己的议程中某一时间段（time interval）的所有会议安排。

查询会议时提供的参数应包括所关注时间段的起始时间和终止时间；查询结果返回该用户议程中在指定时间范围内找到的所有会议安排的列表，在列表中给出每一会议的起始时间、终止时间、标签、以及被预约的另一用户的名字。

注意，查询会议的结果既应包括用户作为发起人的那些会议，也应包括用户作为被预约人的那些会议。

1.4.4 删除会议

已注册的用户可以删除自己登记的某一会议安排。

删除会议时，提供的参数除执行删除功能的用户的名字及其密码外，还包括一个能惟一地标识待删除的

会议的参数（提示：可以简单地以会议标签作为一个会议的惟一标识）。

1.4.5 清除会议

已注册的用户可以清除自己创建的所有会议安排。

1.4.6 其他

本实验不要求实现程序中对象的持久性（object persistence）。

1.5 接口约定

本实验要求编写一个基于命令行交互方式的应用程序来访问议程管理系统中的信息。

系统启动后，显示输入提示符“$”（称之为 prompt，类似 MS-DOS 的提示符“>”），然后与终端用户以命令行方式进行交互。设所有命令的名字是大小写无关的，但命令的参数却是大小写敏感的。约定系统支持的交互命令格式如下：

$ register [userName] [password]

该命令用于注册一个新用户。其中，参数 userName 是用户提供的注册名字，password 是用户自己选择的密码。

$ add [userName] [password] [other] [start] [end] [title]

该命令用于添加一个新的会议安排。其中，参数 userName 是一个已注册的用户名字，password 是该用户的密码，other 是新添加会议所预约的另一已注册用户的名字，start 是会议的起始时间，end 是会议的终止时间，title 是新添加会议的标签。

$ query [userName] [password] [start] [end]

该命令用于查询某一用户在某一时间段内的会议安排。其中，参数 userName 是一个已注册的用户名字，password 是该用户的密码，start 和 end 分别是查询时间段的起始时间和终止时间，该时间段内查获的会议安排将被返回。

$ delete [userName] [password] [meetingId]

该命令用于删除某一用户的指定会议安排。其中，参数 userName 是一个已注册的用户名字，password 是该用户的密码，meetingId 指定将被从议程中删除的会议的标识。

$ clear [userName] [password]

该命令用于清除某一用户的所有会议安排。其中，参数 userName 是一个已注册的用户名字，password 是该用户的密码。

$ batch [fileName]

该命令用于批处理存储在文本文件 fileName 中的 register、add、query、delete、clear 等命令，其中每一条命令（包括其参数）单独占一行。

该命令相当于 MD-DOS 中的脚本文件（即批命令文件.bat 或.cmd）。借助该命令，你可以方便地运行多个不同的测试用例。

$ quit

该命令用于退出议程管理系统。

1.6 提交结果

如果不按指定方式组织你的实验结果，会影响到你的实验成绩！

1.6.1 提交方式

注意你的实验结果不应只包含源程序代码，还应包括一个面向对象程序的设计文档（建议命名为 design.doc 或 design.pdf）、编译与运行程序的脚本（.bat 文件）、各种测试用例（尽量支持回归测试，Regression Testing）等。

此外，至少还要包括一个纯文本的自述文件 readme.txt，其中描述你自己的姓名、学号、email、电话等基本信息，并将你在本实验中的收获体会也写在自述文件中。

你的实验结果全部存放在一个名为“学号＋姓名”（中间不要任何空格）的文件夹中，压缩成单个文件后，在指定截止时间前提交给本课程的教学助理。

1.6.2 组织实验结果

每一份实验报告提交结果应按如下方式组织：

(1) 在子目录 **src** 中存放源代码，包括：

- 议程管理系统主程序的源代码 **`AgendaService.java`**。
- 程序用到的其他源码，取决于你的设计，均存放于程序包 **agenda** 中，其下可设计其他的子程序包。

(2) 在子目录 **bin** 中存放由源代码编译出来的可执行程序，即 Java 字节码。

(3) 在子目录 **doc** 中存放根据上述源代码生成的 javadoc 文档。

(4) 在工作目录的根目录中直接存放编译、运行与调试有关文件，包括：

- 编译所有源代码的脚本文件（批命令）**`build.bat`**。
- 运行议程管理系统的脚本文件 **`agenda.bat`**。
- 其他用于调试、测试程序的脚本、配置文件、测试用例等；测试用例众多时，可安排在单独的子目录中。

(5) 在工作目录的根目录中直接存放其他文档，包括：

- 描述你的面向对象设计结果的 Word 文档 **`design.doc`**，其中包括 UML 类图及其说明。
- 描述程序运行测试用例的截图与解释的 Word 文档 **`test.doc`**。
- 关于实验完成人基本信息、提交结果描述、实验心得等的补充说明文档 **`readme.txt`**。

1.7 实验交流

如果在本实验的课堂交流时你被随机地抽点到，建议你按以下次序展示你的实验结果：

- 介绍自己对问题的理解以及程序的设计，要求使用 UML 展示你的设计结果！
- 展示你的文档化注释生成的文档（由 javadoc 生成的 HTML 文档）以及源程序清单。
- 运行多个测试用例演示程序的运行效果（屏幕截图或屏幕录像）。
- 接受老师和其他同学的提问。

2 实验评价

任课教师或教学助理将从文档、源程序、执行结果等多个角度评价你的实验结果，因而它们也是你完成该实验需要重点注意的地方。

2.1 主要评价依据

2.1.1 编程风格

- 源程序中标识符命名、程序版面、程序注释等影响程序可理解性的要素，以及是否能熟练地运用 Java 语言的文档化注释。
- 本实验要求至少遵循 Sun 公司制订的 Java 编码规范，完成后的代码必须使用 JDK 附带的文档工具 javadoc 根据源程序中的文档化注释生成相应的文档（可根据自己的英文水平自由选择使用中文或英文书写文档化注释）。

2.1.2 面向对象设计

- 软件构架及程序接口的设计是否合理：从大的方面看，软件体系结构（又称软件架构）的层次划分清晰（至少设计为 2-tier 的层次架构，即实现 Presentation Tier 和 Business Logic Tier 的分离）；从小的方面看，程序中对类或方法的抽象符合面向对象与结构化程序设计思想，程序的接口（包括对象接口和方法接口）的设计应合理。
- 程序对未来变化的适应性如何，譬如是否容易修改或不必修改就应付可能发生的变化（譬如在交互方式中增加新的命令 unregister、help 等），是否方便在不同的环境下复用程序或程序中的构件。
- 源程序中对异常的处理是否充分，可执行程序的健壮性如何（在异常情况下工作的能力）。
- 为应付未来的大型程序设计任务，在实验中是否能正确地运用 Java 语言提供的 package 机制组织你的程序。

2.1.3 其他

- 为尽可能提高程序的正确性，测试用例的选择是否完善、有无其他措施有助于提高程序的正确性。
- 作为一个完整的实验结果，文档和源程序文件的组织是合理。
- 其他可能影响程序质量的要素。

2.2 满足最低要求

如果实验结果未达到以下要求，将不会取得 70%分或以上的成绩。

- 程序编写规范（标识符命名、空白的运用、文档化注释等）。
- 实验报告的提交内容完整，组织结构清晰。
- 程序可以运行，并通过简单的测试。

2.3 如何获得高分

如果实验结果想取得 100%分的成绩，至少需在以下几方面有 3 个或以上的做得比较出色。

- 程序的表示层（Presentation Tier）与业务逻辑层（Business Logic Tier）划分清晰。
- 正确设计了对象类的抽象、对象类之间的关系。
- 在程序中运用了面向对象程序多态性（polymorphysm）的特点来解决一个实际问题，程序具有表示独立性。
- 正确运用异常处理机制分离正常与异常代码，提高程序在异常条件下的健壮性。
- 至少正确地运用了一种 GoF 设计模式（design pattern）去解决程序面临的某一特定问题，并且在设计文档 design.doc 中描述了采用该设计模式的思路。