



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

Lab Manuals for Software Construction

Lab-2

Abstract Data Type (ADT) and Object-Oriented Programming (OOP)



School of Computer Science and Technology

Harbin Institute of Technology

Spring 2018

目录

1	实验目标.....	1
2	实验环境.....	1
3	实验要求.....	2
3.1	Poetic Walks	2
3.2	Re-implement the Social Network in Lab1	2
3.3	The Transit Route Planner (选做, 额外给分)	3
4	实验报告.....	4
5	提交方式.....	4
6	评分方式.....	5

1 实验目标

本次实验训练抽象数据类型（ADT）的设计、规约、测试，并使用面向对象编程（OOP）技术实现 ADT。具体来说：

- 针对给定的应用问题，从问题描述中识别所需的 ADT；
- 设计 ADT 规约（pre-condition、post-condition）并评估规约的质量；
- 根据 ADT 的规约设计测试用例；
- ADT 的泛型化；
- 根据规约设计 ADT 的多种不同的实现；针对每种实现，设计其表示（representation）、表示不变性（rep invariant）、抽象过程（abstraction function）
- 使用 OOP 实现 ADT，并判定表示不变性是否违反、各实现是否存在表示外泄（rep exposure）；
- 测试 ADT 的实现并评估测试的覆盖度；
- 使用 ADT 及其实现，为应用问题开发程序；
- 在测试代码中，能够写出 testing strategy 并据此设计测试用例。

2 实验环境

实验环境设置请参见 Lab-0 实验指南。

除此之外，本次实验需要你在 Eclipse IDE 中安装配置 Eclemma（一个用于统计 JUnit 测试用例的代码覆盖度的 plugin）。请访问 <http://www.eclemma.org>，了解 Eclemma 并学习其安装、配置和使用。

本次实验在 GitHub Classroom 中的 URL 地址为：

<https://classroom.github.com/a/z9utaaos>

请访问该 URL，按照提示建立自己的 Lab2 仓库并关联至自己的学号。

本地开发时，本次实验只需建立一个项目，统一向 GitHub 仓库提交。实验包含的 2(+1)个任务分别在不同的目录内开发，具体目录组织方式参见各任务最后一部分的说明。请务必遵循目录结构，以便于教师/TA 进行测试。

3 实验要求

3.1 Poetic Walks

请阅读 <http://web.mit.edu/6.031/www/sp17/psets/ps2/>，遵循该页面内的要求完成编程任务。

- 在 Get the code 步骤中，你无法连接 MIT 的 Athena 服务器，请使用 Git 指令从获取初始代码：
https://github.com/rainywang/Spring2018_HITCS_SC_Lab2/tree/master/P1
- 在作业描述中若遇到“commit and push”的要求，请将你的代码 push 到你的 GitHub Lab2 仓库中。
- MIT 作业页面提及的文件路径，请按照下表的目录结构进行调整。例如“test/poet”应为“test/P1/poet”，“src/poet”应为“src/P1/poet”。
- 其他步骤请遵循 MIT 作业页面的要求。

项目的目录结构：

```
项目名称: Lab2_学号
    src
        P1
            graph
            ...java
            poet
            ...java
            ...txt
        test
            P1
                graph
                ...Test.java
                poet
                ...Test.java
                ...txt
```

请使用 git 指令将符合上述结构的代码 push 到你的 GitHub Lab2 仓库中。

3.2 Re-implement the Social Network in Lab1

回顾 Lab1 实验手册中的 3.2 节 Social Network，你针对所提供的客户端代码实现了 FriendshipGraph 类和 Person 类。

在本次实验中，请基于你在 3.1 节 Poetic Walks 中实现的 Graph<L>类，重新

实现 Lab1 中 3.1 节的 `FriendshipGraph` 类。

注 1: 可以忽略你在 Lab1 中实现的代码, 无需其基础上实现本次作业;

注 2: 3.1 节实现的 `ConcreteEdgesGraph` 和 `ConcreteVerticesGraph` 都不再是泛型, `L` 替换为了 `String`。但在 P2 中, 图中的节点仍需为 `Person` 类型。故你的新 `FriendshipGraph` 类需要重新从 `Graph<L>` 中派生, 以支持 `Person` 类型的节点。

注 3: 尽可能不变动 Lab1 的 3.1 节中写出的客户端代码 (例如 `main()` 中的代码), 即同样的客户端代码仍可运行。如果不能运行, 请重新撰写客户端代码和 `FriendshipGraphTest.java` 中的测试用例, 以对新的 `FriendshipGraph` 代码进行充分测试。

注 4: 该任务中撰写的所有 ADT、测试用例, 请遵循 3.1 节的要求撰写 AF、RI、safety from rep exposure、testing strategy。

项目的目录结构:

```
项目名称: Lab2_学号
    src
        P2
            FriendshipGraph.java
            Person.java
            ...
    test
        P2
            FriendshipGraphTest.java
            ...
```

(无需将 3.1 节实现的 `Graph<L>` 的程序源文件复制到 P2 目录下)

请使用 `git` 指令将符合上述结构的代码 `push` 到你的 GitHub Lab2 仓库中。

3.3 The Transit Route Planner (选做, 额外给分)

该问题来自于 CMU 15-214 软件构造课的作业 2。

请从 https://github.com/rainywang/Spring2018_HITCS_SC_Lab2/tree/master/P3 下载该问题的具体要求 (hw2.pdf)。忽略该文档第 1 页的“Using polymorphism for representational abstraction”, 意即你无需再从 0 开始构造一个 `Graph` 接口。为实现该文档后续部分要求的 Transit route planner, 你需要使用本次实验 2.1 节中已经实现的 `Graph<L>`。

项目的目录结构:

项目名称: Lab2_学号

src

P3

....java

test

P3

...Test.java

(P3 目录下不限定子目录和 java 文件的名称, 请自由设计)

请使用 git 指令将符合上述结构的代码 push 到你的 GitHub Lab2 仓库中。

4 实验报告

针对上述三个编程题目, 请遵循 CMS 上 Lab2 页面给出的**报告模板**, 撰写简明扼要的实验报告。

实验报告的目的是记录你的实验过程, 尤其是遇到的困难与解决的途径。不需要长篇累牍, 记录关键点即可, 但需确保报告覆盖了本次实验的所有开发任务 (3 个问题, 每个问题下有一系列任务)。

注意:

- 实验报告不需要包含所有源代码, 请根据上述目的有选择的加入关键源代码, 作为辅助说明。
- 请确保报告格式清晰、一致, 故请遵循目前模板里设置的字体、字号、行间距、缩进;
- 实验报告提交前, 请“目录”上右击, 然后选择“更新域”, 以确保你的目录标题/页码与正文相对应。

5 提交方式

截止日期: 第 6 周周日 (2018 年 4 月 8 日) 夜间 23:55。截止时间之后通过 Email 等其他渠道提交实验报告和代码, 均无效, 教师和 TA 不接收, 学生本次实验无资格。

源代码: 从本地 Git 仓库推送至个人 GitHub 的 Lab2 仓库内。

实验报告: 除了随代码仓库 (doc) 目录提交至 GitHub 之外, 还需手工提交至 CMS 实验 2 页面下。

6 评分方式

TA 在第 3-5 周实验课上现场验收：学生做完实验之后，向 TA 提出验收申请，TA 根据实验要求考核学生的程序运行结果并打分。现场验收并非必需，由学生主动向 TA 提出申请。

Deadline 之后，教师使用持续集成工具对学生在 GitHub 上的代码进行测试。教师和 TA 阅读实验报告，做出相应评分。