

**2018年春季学期  
计算机学院大二软件构造课程**

**Lab 2实验报告**

|  |  |
| --- | --- |
| 姓名 | 郭潇 |
| 学号 | 1160300521 |
| 班号 | 03005 |
| 电子邮件 | wqsnzszw@163.com |
| 手机号码 | 18523735573 |

**目录**

[1 实验目标概述 1](#_Toc507927440)

[2 实验环境配置 1](#_Toc507927441)

[3 实验过程 1](#_Toc507927442)

[3.1 Poetic Walks 1](#_Toc507927443)

[3.1.1 Get the code and prepare Git repository 1](#_Toc507927444)

[3.1.2 Problem 1: Test Graph <String> 1](#_Toc507927445)

[3.1.3 Problem 2: Implement Graph <String> 1](#_Toc507927446)

[3.1.3.1 Implement ConcreteEdgesGraph 2](#_Toc507927447)

[3.1.3.2 Implement ConcreteVerticesGraph 2](#_Toc507927448)

[3.1.4 Problem 3: Implement generic Graph<L> 2](#_Toc507927449)

[3.1.4.1 Make the implementations generic 2](#_Toc507927450)

[3.1.4.2 Implement Graph.empty() 2](#_Toc507927451)

[3.1.5 Problem 4: Poetic walks 2](#_Toc507927452)

[3.1.5.1 Test GraphPoet 2](#_Toc507927453)

[3.1.5.2 Implement GraphPoet 2](#_Toc507927454)

[3.1.5.3 Graph poetry slam 2](#_Toc507927455)

[3.1.6 Before you’re done 2](#_Toc507927456)

[3.2 Re-implement the Social Network in Lab1 2](#_Toc507927457)

[3.2.1 FriendshipGraph类 2](#_Toc507927458)

[3.2.2 Person类 3](#_Toc507927459)

[3.2.3 客户端main() 3](#_Toc507927460)

[3.2.4 测试用例 3](#_Toc507927461)

[3.2.5 提交至Git仓库 3](#_Toc507927462)

[3.3 The Transit Route Planner（选做，额外给分） 3](#_Toc507927463)

[4 实验进度记录 3](#_Toc507927464)

[5 实验过程中遇到的困难与解决途径 3](#_Toc507927465)

[6 实验过程中收获的经验、教训、感想 4](#_Toc507927466)

# 实验目标概述

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。

# 实验环境配置

简要陈述你配置本次实验所需环境的过程，必要时可以给出屏幕截图。

特别是要记录配置过程中遇到的问题和困难，以及如何解决的。

利用和实验1相似的方法从github上利用clone操作获取实验项目

在这里给出你的GitHub Lab2仓库的URL地址（Lab2-学号）。

https://github.com/ComputerScienceHIT/Lab2-1160300521

# 实验过程

请仔细对照实验手册，针对三个问题中的每一项任务，在下面各节中记录你的实验过程、阐述你的设计思路和问题求解思路，可辅之以示意图或关键源代码加以说明（但千万不要把你的源代码全部粘贴过来！）。

## Poetic Walks

1. 利用给定的Graph接口实现ConcreteVerticesGraph和ConcreteVerticesGraph两种实现
2. 将完成的实现扩展为泛型
3. 使用已经完成的Graph实现poet
4. 在实验过程总逐渐掌握测试策略、IR、AF、Safety from Rep Expouser的编写

### Get the code and prepare Git repository

如何从GitHub获取该任务的代码、在本地创建git仓库、使用git管理本地开发。

1. 使用git clone 从提供网址获取实验代码
2. 在eclipse中import实验项目
3. 利用git添加实验给定的仓库地址
4. 在使用git push 指令时应注意选择新添加的仓库

### Problem 1: Test Graph <String>

以下各部分，请按照MIT页面上相应部分的要求，逐项列出你的设计和实现思路/过程/结果。

测试策略：

1. add操作按照graph的大小为0、1、n；vertex是否在图中划分
2. set操作按照graph的大小为0、1、n；weight的大小是否为0；target，source是否在graph中；加入的edge是否是new edge划分
3. remove操作按照graph的大小为0、1、n；vertex是否在图中;删除的vertex是edge 的source、target、source和target划分
4. vertices操作按照graph的大小为0、1、n划分
5. sources操作按照graph的大小为0、1、n；edge的数量为0、1、n划分
6. targets操作按照graph的大小为0、1、n；edge的数量为0、1、n划分

### Problem 2: Implement Graph <String>

以下各部分，请按照MIT页面上相应部分的要求，逐项列出你的设计和实现思路/过程/结果。

#### Implement ConcreteEdgesGraph

Edge的设计：

1. RI为满足边的起始和终点不为null且权值大于0
2. AF为符合上述要求的一条边
3. Safety from Rep Expouser为所有变量为private；所有的变量通过get方法观察

ConcreteEdgesGraph的设计：

1. RI为满足顶点不为null且所有边权值大于0
2. AF为符合上述要求的所有边和顶点
3. Safety from Rep Expouser为所有变量为private；所有类型是mutable的返回值，均创建新的变量返回防止泄露

实现：

1. 用一个set存放所有的顶点，一个list存放所有边
2. checkrep检查顶点和边是否符合RI的要求
3. add操作依靠set的add方法实现，若加入的顶点已存在则返回false
4. set操作需检查要加入的边是否存在，若存在且权值为0则删除已有的边，权值大于0则修改权值并返回以前的权值；若加入的边不存在其权值大于0则加入新边
5. remove操作使用迭代器时使用迭代器的remove方法
6. vertices、sources、targets方法均创建新的变量返回

#### Implement ConcreteVerticesGraph

Vertex的设计：

1. RI为满足lable不为null且入度和出度均大于0
2. AF为符合上述要求的所有点
3. Safety from Rep Expouser为所有变量为private；所有的变量通过get方法观察；remove方法只返回true或false；add方法返回不变量或数字0

ConcreteVerticesGraph的设计：

1.RI为满足顶点入读和出度均大于0

2.AF为符合上述要求的所有顶点

3.Safety from Rep Expouser为所有变量为private；所有类型是mutable的返回值，均创建新的变量返回防止泄露

实现：

1.用一个list存放所有顶点

2.checkrep检查顶点是否符合RI的要求

3.add操作依靠list的add方法实现，判断顶点是否存在依靠自己写的方法且若加入的顶点已存在则返回false

4.set操作需检查要加入的边是否存在，若存在且权值为0则删除已有的边，权值大于0则修改权值并返回以前的权值；若加入的边不存在其权值大于0则加入新边

5.vertices、sources、targets方法均创建新的变量返回

### Problem 3: Implement generic Graph<L>

#### Make the implementations generic

设计：

对ConcreteEdgesGraph、Edge、ConcreteVerticesGrap、Vertex均用泛型L代替之前的String

#### Implement Graph.empty()

**设计：**

**用**ConcreteEdgesGraph实现Graph.empty()，即返回一个新的ConcreteEdgesGraph

### Problem 4: Poetic walks

#### Test GraphPoet

测试策略：

1. 文本的行数为单行还是多行
2. bridge的数量为0、1、n
3. bridge的权值为相同还是不同

#### Implement GraphPoet

**设计：**

1.RI为满足vertex不含有空白字符也不为空白字符

2.AF为符合上述要求的所有顶点

3.Safety from Rep Expouser为所有变量为private；所有类型是mutable的返回值，均创建新的变量返回防止泄露

实现：

1. 利用GraphPoet函数读入相应文本，利用已经实现的graph，把每个单词看做顶点，把当前单词和相邻的单词作为两个顶点加入相应的边，权值为边出现次数
2. poem函数寻找权值不为0的最大权值的边作为相应的bridge加入到生成的新诗句中

#### Graph poetry slam

### Before you’re done

请按照<http://web.mit.edu/6.031/www/sp17/psets/ps2/#before_youre_done>的说明，检查你的程序。

如何通过Git提交当前版本到GitHub上你的Lab2仓库。

在这里给出你的项目的目录结构树状示意图。

## Re-implement the Social Network in Lab1

1. 利用P1中的graph实现P2中的friendshipgraph，使用用person作为点的类型
2. 利用graph的方法实现friendshipgraph的各种操作
3. 在实验过程总逐渐掌握测试策略、IR、AF、Safety from Rep Expouser的编写

### FriendshipGraph类

给出你的设计和实现思路/过程/结果。

设计：

1．RI为null

2．AF为符合具有名字的person

3．Safety from Rep Expouser为所有变量为private；所有的变量通过get方法观察

### Person类

设计：

1．RI为顶点不重复，边的source和target均不为null

2．AF为符合具有上述条件的有向图

3．Safety from Rep Expouser为所有变量为private；所有的变量通过get方法观察；所有类型是mutable的返回值，均创建新的变量返回防止泄露

实现：

1. addvertex操作通过graph的add方法实现，检查加入的person是否为null和重复的情况
2. addedge操作通过graph的set方法实现，权值均设为1；检查加入的person是否有null，已存在，source和target是否相同或是否在图中的情况
3. getdistance操作通过广度优先搜索实现，从起始person开始搜索，最先搜索到目标person所花距离则为最短距离

结果：

1. 若加入点和边时存在person为null、重复和已存在的情况，则抛出异常
2. getdistance确定两个perosn间距离是否合法，不合法则抛出异常，合法则调用solve函数返回最短距离
3. isgraph和isin分别判断名字重复和是否已被访问，若重复、已被访问则返回true；否则返回false

### 客户端main()

设计：按照实验要求，将无向图扩展为有向图，并添加题目中的相应操作。

结果：与题目中预期结果一致

### 测试用例

测试策略：

1. addvertex按照vertex是否在graph中；graph的大小为0、1、n划分
2. addEege按照target，source是否在graph中；是否相同；是否为null；加入的边是否为新边划分
3. getDistance按照vertex个数为0、1、n；target，source是否相同；target，source是否存在划分

### 提交至Git仓库

如何通过Git提交当前版本到GitHub上你的Lab3仓库。

在这里给出你的项目的目录结构树状示意图。

## The Transit Route Planner（选做，额外给分）

请自行组织本节内的目录结构。

# 实验进度记录

请尽可能详细的记录你的进度情况。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 计划任务 | 实际完成情况 |
| 2018-03-12 | 13:45-15:30 | 熟悉P1的任务要求并完成problem1 | 计划基本完成 |
| 2018-03-12 | 18:00-21:00 | 完成ConcreteEdgesGraph | 计划完成 |
| 2018-03-13 | 18:30-21:00 | 完成ConcreteVerticesGraph | 计划完成 |
| 2018-03-14 | 16:00-17:30 | 完成P1的problem3 | 计划完成 |
| 2018-03-15 | 14:00-17:30 | 完成P1的problem4 | 计划完成 |
| 2018-03-17 | 13:30-17:00 | 完成P2 | 只做完了代码部分，测试部分为完成 |
| 2018-03-18 | 15:00-16:30 | 完成P2的测试部分 | 计划完成 |
| 2018-03-19 | 13:45-15:30 | 完善之前的测试部分并阅读P3的要求 | 计划完成 |
| 2018-03-21 | 18:30-21:00 | 完成P3部分bus、stop、ltinerary的设计 | 计划完成 |
| 2018-03-22 | 14:00-16:00 | 完成P3部分builder的设计 | 对读写文件操作不熟悉，未完成 |
| 2018-03-22 | 19;00-21:00 | 完成P3部分builder的设计 | 计划完成 |
| 2018-03-24 | 13:30-16:30 | 完成P3部分planner的设计 | 编写的程序出错，计划延后修改 |
| 2018-03-25 | 18:00-20:30 | 完成P3的测试部分 | 计划完成 |
| 2018-03-26 | 13:45-15:30 | 修改P3部分planner的设计 | 编写的程序依然存在问题 |
| 2018-03-27 | 14:00-17:30 | 完成P1,P2的IR、AF、Safety from Rep Expouser的编写 | 计划完成 |
| 2018-04-05 | 14:00-17:30 | 完成实验报告 | 大部分已完成 |
| 2018-04-07 | 18:30-20:00 | 完成实验报告 | 计划完成 |

# 实验过程中遇到的困难与解决途径

1. 对P1中接口、重写、泛型、抽象类以及checkrep的不理解导致未能读懂实验具体要求

自己查找资料加上做实验中课程的不断学习，逐渐理解实验要求并不断改进

1. 对测试策略、IR、AF、Safety from Rep Expouser的编写不熟悉

复习课上的ppt，仿照ppt上的示例进行上述内容的编写

1. 在P2中实现getdistance时，应用队列实现广度搜索时距离出错

加入一个map记录搜索过程中的距离，每次进队时更新，直到找到目标点即为最短距离

1. 在P3中复用P1的graph导致实现困难

充分阅读实验要求，发现应用graph的set操作添加边时，难以将公交车的路数添加，于是在输出路线时放弃该项内容。

# 实验过程中收获的经验、教训、感想

本节除了总结你在实验过程中收获的经验和教训，也可就以下方面谈谈你的感受（非必须）：

1. 对接口、重写、泛型、抽象类有了基本的了解，并意识到这样编程的明显优点
2. 学会了如何编写基本的测试策略、IR、AF、Safety from Rep Expouser，意识到这些内容对程序的实现和安全上有重要意义。
3. 我认为面向ADT的编程和直接面向应用场景编程的优劣取决于ADT的好坏和适用度，比如利用P1实现P2由于两者的方法比较吻合，所以实现容易；而利用P1实现P3时，主要因为set方法加入边时难以满足P3的要求，加大了实现难度和工作量。
4. 使用泛型和不使用泛型的编程，我更倾向于使用泛型，因为将特定类型改为泛型花费的工作量很小，但却可以更为广泛使用。
5. 在给出ADT的规约后就开始编写测试用例，能够更清楚的了解实验的要求，实现更多细节；但如果需要自己想出规约的话，很难做到完善
6. 对于本实验的工作量，我认为P1、P2工作量偏少；而P3的工作量因为难以复用，使用工作量偏大