哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

2019 年春季学期

计算机学院《软件构造》课程

Lab 4 实验报告

姓名	肖行文
学号	1170300316
班号	1703003
电子邮件	xiaoxingwen@stu.hit.edu.cn
手机号码	13266573776

目录

1 实验目标概述

利用以下技术提高程序健壮性和正确性：

- 错误处理
- 异常处理
- Assertion 和防御式编程
- 日志
- 调试技术
- 黑盒测试及代码覆盖度

程序在出错后可对异常进行处理，提示用户并优雅退出或更换方法继续执行程序。

2 实验环境配置

本实验要求安装 SpotBugs，可以从 Eclipse MarketPlace 中直接下载安装。

仓库地址：https://github.com/ComputerScienceHIT/Lab4-1170300316

3 实验过程

请仔细对照实验手册，针对每一项任务，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！）。

3.1 Error and Exception Handling

创建 3 个新类继承 Exception 用于处理特定问题。

3.1.1 IllegalNumberException

这个 Exception 用于处理关于数字输入的错误，并且包含返回出错数字的方法，方便定位。

Stellar System 中，如角度必须在范围$[0, 360]$之间；数字大于 10000 时要使用科学计数法，小于 10000 时要用直接表示法；行星速度大于 0；行星半径应该小于轨道半径。

Atomic Structure 中，如总轨道数应该大于 0 且是一个整数；电子所处轨道应该小于轨道总数；每层电子数量大于等于 0 且是一个整数；轨道数都是整数；不应该存在没有电子的轨道。

SocialNetworkSystem 中，任何人的年龄应该大于 0 且为整数；社交亲密度范围为[0, 1]；亲密度最多有三位小数等。

3.1.2 SameLabelException

此 Exception 用于检测新 Object 是否合法，并且包含返回出错标签名字的方法，方便定位。

Stellar System 中不存在两个同名星球，SocialNetworkSystem 中不存在同一个人指向他自己的关系。

3.1.3 FalseDependencyException

此 Exception 用于检测依赖关系是否合法，并且包含返回出错依赖关系涉及到对象名称的方法。

Atomic Structure 中不允许向不存在的边上添加电子。

3.1.4 MissingObjectException

此 Exception 用于检测添加的 Object 是否合法，并返回 Object 名字。

如 SocialNetworkSystem 中添加关系时，应保证关系双方都在朋友列表中，如果任何一方不在，应该 throw Exception。

3.2 Assertion and Defensive Programming

3.2.1 checkRep()检查 invariants

每个 ADT 都有 checkRep()，列举如下：

Position()存储了极坐标，要保证角度在[0, 360]之间。

CentralObject()所表示的中心物体都需要保证坐标处于(0, 0)，每次动画刷新都需要检查。

PhysicalObject()中如 Friend 需要检查年龄是否大于 0 等等，这些 PhysicalObject()数字上的问题在 IllegalNumberException()中进行了处理。

SocialTile 需要保证亲密度在(0, 1)之间，在读入文件的时候进行检查。

3.2.2 Assertion 保障 pre-/post-condition

Track()中，对 Atom 的电子所属的轨道进行保护，电子不能被放在比电子层数还大的轨道上，这点利用上文提到的 IllegalNumberException 进行了处理。IllegalNumberException 对程序进行重新启动，并且提供用户一个重新选择的机会。在每个电子类中，用 Assertion 对这个条件进行了再次保护。

Track()中也对 StellarSystem 添加两个半径相同（考虑到轨道半径很大且存储有误差，设定为相近轨道也会提示）的轨道，在添加下一个轨道前，与已有的轨道进行比对，阻止相同的情况。

SocialNetworkSystem 中在每添加一个新 Friend，会检查是否是重复添加，保证不会添加一个与存在朋友一样的朋友。

StellarSystem 中要对轨道大小进行排序再显示出来，所以应该对轨道排序函数进行 post condition 检查，确保排序正确。

每个场景的类中，需要保证文件读入的每一行满足输入条件个数与 Lab3 要求相符，例如 Planet 就必须输入 8 条属性，如果检测到不是 8 个，就应该停止。

3.3 Logging

3.3.1 写日志

将日志记录的内容分为 severe 和 Info 两类，severe 记录各种 Exception，Info 记录普通的操作，如轨道的增加和删除等。

为了方便使用，将 Java logging 和一些功能写进一个类中（LogFile.java）。由于 logging 的一部分功能不方便用户直接阅读（比如 logging 更新记录时是向前插入的，一般来说向后插入更直观；时间显示的是格林尼治时间，而用户更想要本地时间），所以对 logging 进行了改写。

为了显示正确的时间，直接调用系统日历：

```
public static String get_nowDate() {
    Calendar D = Calendar.getInstance();
    int year = 0;
    int moth = 0;
    int day = 0;
    year = D.get(Calendar.YEAR);
    moth = D.get(Calendar.MONTH) + 1;
    day = D.get(Calendar.DAY_OF_MONTH);
```

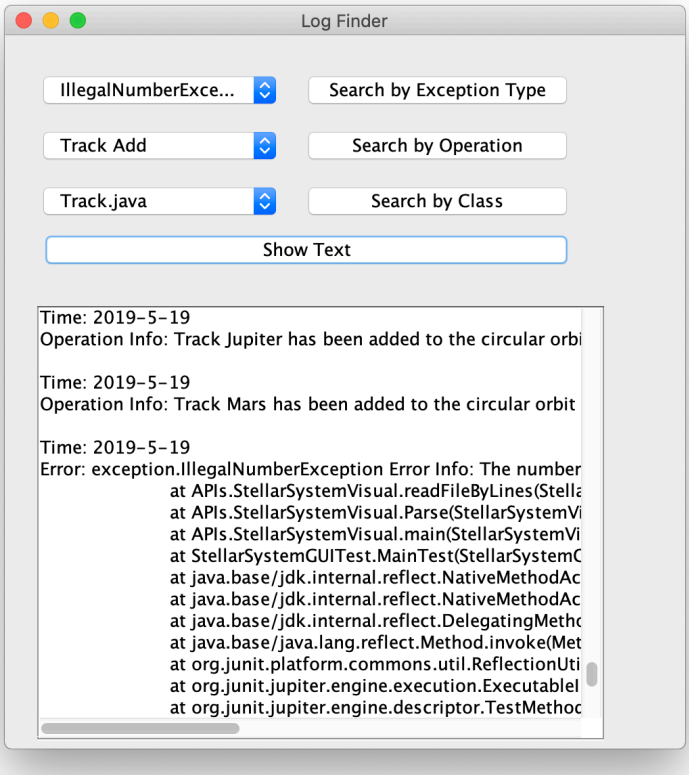
为了读起来更直观，用 GUI 显示日志，并用 JTextArea.append()方法实现向后追加日志内容而非加在头部：

```
while (str1 != null) {
    textArea.append("\n" + str1);
    str1 = in.readLine();
}
```

为增加可读性，对输出格式进行了重新整理，所有 Exception 的 log，均将 stacktrack 内容写进 log，方便用户定位：

```
StringWriter sw = new StringWriter();
e.printStackTrace(new PrintWriter(sw));
String exceptionAsString = sw.toString();
```

3.3.2 日志查询



用户以 GUI 的形式对查询内容进行操作，查询的方式其实就是字符串匹配。比如要查询有关 IllegalNumberException 有关的内容，就找出它在 log 文件中的位置并把整段内容输出在 JTextArea 中。

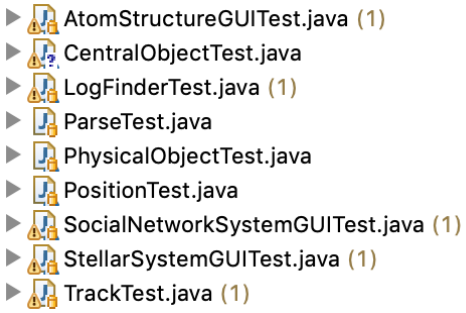
3.4 Testing for Robustness and Correctness

3.4.1 Testing strategy

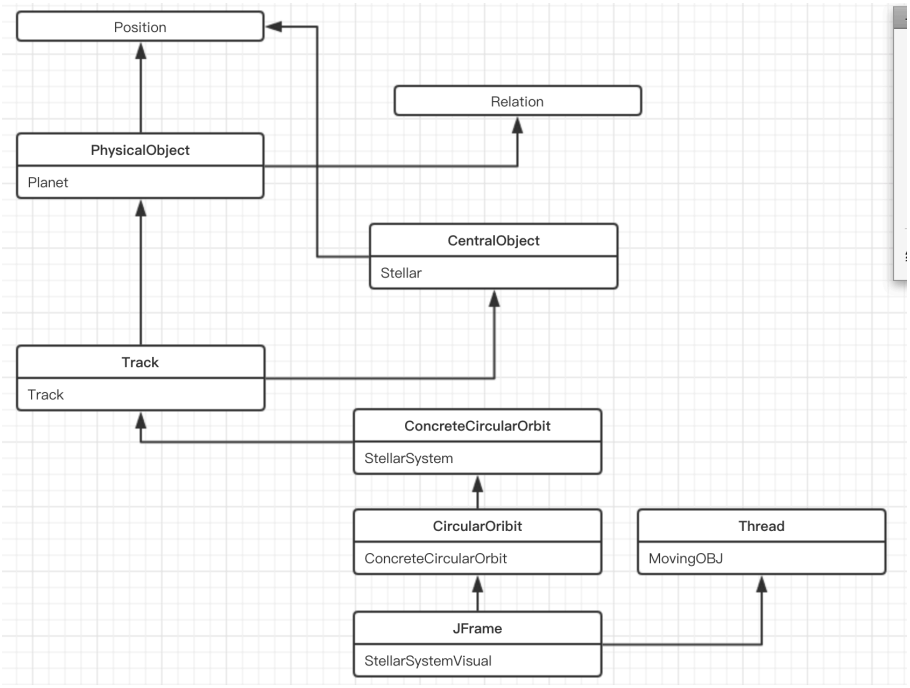
测试主要针对各个 ADT，GUI 部分无法涉及，尽量覆盖每个分支和循环。针对每个函数的 Pre Condition 和 Post Condition 撰写测试。

3.4.2 测试用例设计

测试策略为划分等价类进行正常输入范围测试和边界条件测试。



一部分 Test 用来测试基础 ADT 的正确性，一部分用于检测 GUI 的启动（测试 GUI 内部控制非常困难）。



根据程序的 UML 图从基础部件测试到高级部件。

3.4.3 测试运行结果与 EclEmma 覆盖度报告

总体：

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
Lab4-1170300316	42.1 %	3,238	4,450	7,688
src	47.0 %	3,238	3,649	6,887
APIs	53.2 %	2,113	1,861	3,974
centralObject	0.0 %	0	150	150
circularOrbit	66.6 %	339	170	509
debug	0.0 %	0	667	667
exception	42.6 %	46	62	108
extensions	28.2 %	101	257	358
Log	71.4 %	192	77	269
physicalObject	47.3 %	210	234	444
track	58.1 %	237	171	408

注：extensions 和 centralObject 的 Package 是为 3.12change 所写，基本不参与测试，GUI 部分代码无法测试。

生成的测试文件位于 doc 文件夹内。

3.5 SpotBugs tool

Check for oddness that won't work for negative numbers in debug.FindMedianSor...	FindMedianSo...	/Lab4-117030031...	line 60	SpotBugs Problem (...
physicalObject.PhysicalObject.equals(Object) does not check for null argument [T...	PhysicalObjec...	/Lab4-117030031...	line 71	SpotBugs Problem (...
Dead store to degree rather than field with same name in extensions.Position.Rota...	Position.java	/Lab4-117030031...	line 42	SpotBugs Problem (...
Dead store to radius rather than field with same name in track.Track.SetRadius(do...	Track.java	/Lab4-117030031...	line 112	SpotBugs Problem (...
Null passed for non-null parameter of APIs.AtomStructureVisual.main(String[]) in...	AtomStructur...	/Lab4-117030031...	line 13	SpotBugs Problem (...
Null passed for non-null parameter of APIs.LogFinder.main(String[]) in LogFinderT...	LogFinderTest...	/Lab4-117030031...	line 11	SpotBugs Problem (...
Null passed for non-null parameter of APIs.SocialNetworkCircleVisual.main(String[...	SocialNetwork...	/Lab4-117030031...	line 13	SpotBugs Problem (...
Null passed for non-null parameter of APIs.StellarSystemVisual.main(String[]) in S...	StellarSystem...	/Lab4-117030031...	line 13	SpotBugs Problem (...
Null passed for non-null parameter of AtomStructureVisual.main(String[]) in APIs....	CircularOrbitG...	/Lab4-117030031...	line 69	SpotBugs Problem (...
Null passed for non-null parameter of AtomStructureVisual.main(String[]) in APIs....	CircularOrbitH...	/Lab4-117030031...	line 69	SpotBugs Problem (...
Null passed for non-null parameter of new java.io.PrintWriter(Writer) in APIs.Social...	SocialNetwork...	/Lab4-117030031...	line 178	SpotBugs Problem (...
Null passed for non-null parameter of SocialNetworkCircleVisual.main(String[]) in...	CircularOrbitG...	/Lab4-117030031...	line 77	SpotBugs Problem (...
Null passed for non-null parameter of SocialNetworkCircleVisual.main(String[]) in...	CircularOrbitH...	/Lab4-117030031...	line 77	SpotBugs Problem (...
Null passed for non-null parameter of SocialNetworkCircleVisual.main(String[]) in...	SocialNetwork...	/Lab4-117030031...	line 190	SpotBugs Problem (...
Null passed for non-null parameter of StellarSystemVisual.main(String[]) in APIs.Ci...	CircularOrbitG...	/Lab4-117030031...	line 61	SpotBugs Problem (...
Null passed for non-null parameter of StellarSystemVisual.main(String[]) in APIs.Ci...	CircularOrbitH...	/Lab4-117030031...	line 61	SpotBugs Problem (...
Dead store to le in Log.LogFile.main(String[]) [Of Concern(15), High confidence]	LogFile.java	/Lab4-117030031...	line 116	SpotBugs Problem (...
Dead store to trackTrack in TrackTest.NewPlanetTrack() [Of Concern(15), High co...	TrackTest.java	/Lab4-117030031...	line 31	SpotBugs Problem (...
Test for floating point equality in extensions.Position.isEqual(Position) [Of Concer...	Position.java	/Lab4-117030031...	line 83	SpotBugs Problem (...

以上内容是 SpotBug 发现的潜在威胁。

出现次数最多的是以 null 作为参数，但这是由一个 GUI 调用另一个 GUI 界面的 main()函数的过程中使用的，不是 bug。

```
AtomStructureVisual.main(String[]) in AP
AtomStructureVisual.main(String[]) in AP
new java.io.PrintWriter(Writer) in APIs.So
SocialNetworkCircleVisual.main(String[])
SocialNetworkCircleVisual.main(String[])
SocialNetworkCircleVisual.main(String[])
StellarSystemVisual.main(String[]) in APIs
StellarSystemVisual.main(String[]) in APIs
```

另外一类是逻辑上的死锁：

```
public boolean SetRadius(double radius) {
    if (radius == this.radius) {
        return false;
    } else {
        radius = this.radius;
        return true;
    }
}
```

还有一类是对数据类型范围内的某些数不适用，如对负 int 求模的问题。

已经对这些 bug 进行修改，生成报告文件已经保存在项目的 doc 文件夹内。

3.6 Debugging

3.6.1 理解待调试程序的代码思想

（1） FindMedianSortedArrays

此题利用了一种求两个数组的并集的中位数的算法。

如果有两个数组长度为 m 的 A 与长度为 n 的 B，找出中位数的过程中要划分 A 和 B，A 和 B 进而会成为 4 个部分，比中位数小的 A 数组部分、比中位数大的 A 数组部分、比中位数小的 B 数组部分、比中位数大的 B 数组部分。很显然小的部分的数字个数要等于大的部分的数字个数。

假定 A 数组在游标 i 处分开，B 数组在游标 j 处分开，那么有 i + j = m – i + n – j。如果 n >= m，那么 j = (m + n + 1) / 2 – i。同时也要满足 B 的 j – 1 处小于 A 的 i 处，A 的 i – 1 处小于 B 的 j 处。

（2） RemoveComments

为了除去注释部分，给定代码中有个 inBlock 来判断一部分代码是否处于注释中。这个注释包括 /* 开头和 */ 结尾的部分（理论是应该还有 // 后面的部分）。

（3） TopVotedCandidate

这个程序用一个两层 ArrayList 存储投票信息，外层 ArrayList 的 Index 代表被投票了多少，内层 ArrayList 表示在这个次数下得 Vote(person, time)信息。

为了查找得票数最多的参选人，使用两次二分查找 ArrayList，外层找到次数，内层找到具体的参选人。

3.6.2 发现并定位错误的过程

（1） FindMedianSortedArrays

阅读代码，发现有一行要找出中间数写错了，进行修正：

```
// if ((m + n + 1) % 2 == 1) {
// Choose the mid one when total array is even
if ((m + n) % 2 == 1) {
    System.out.println("Return " + maxLeft);
    return maxLeft;
}
```

修正之后仍然出错，便向上寻找，输出 if 判断的每个分支，找到第一部分存在输出错误，继续往上寻找。

```
if (i == 0) {
    //BUG here
    maxLeft = B[j - 1];
    System.out.println("B " + B[j - 1]);
} else if (j == 0) {
    maxLeft = A[i - 1];
    System.out.println("A" + A[i - 1]);
} else {
    maxLeft = Math.max(A[i - 1], B[j - 1]);
    System.out.println(A[i - 1] + B[j - 1]);
}
```

发现应对 j 的取最小点和最大点时做限制：

```
if (i < iMax && B[j - 1] > A[i] && j > 0) {
    iMin = i + 1;
    //revised
} else if (i > iMin && A[i - 1] > B[j] && j < n) {
    iMax = i - 1;
} else {
```

之前已经对后面的取中点进行修改了，前面的取中点保持一致：

```
// int i = (iMin + iMax + 1) / 2;
int i = (iMin + iMax) / 2;
```

至此程序已经运行正确。

（2） RemoveComments

跑测试样例的时候遇到的第一个错误是数组越界，应该修改循环条件：

```
// Array index out of bounds exception
while (i < line.length() - 1) {
    if (!inBlock && i+1 <= line.length() && chars[
        inBlock = true;
```

现在还存在问题：1. 不去除 // 开头的注释 2. 一行只有一个字符时不会被读入。

对于前者加入判断：

```
if (chars[i] == '/' && chars[i+1] == '/') {
    i = line.length() - 1;
    continue;
}
```

int java.lang.String.length()

后者产生的原因是在修改数组越界的时候，数组长度为 1 时根本不会

进入 while 循环，只需在 while 循环前加入对数组长度为 1 的单独处理即可：

```
// In case of line of 1 char, which w
if (line.length() == 1) {
    newline.append(chars[0]);
}
```

（3） TopVotedCandidate

首先根据各个函数的名字粗略浏览代码，发现 count()这个函数的功能（如果是计数的话）并没有实现：

```
int c = count.getOrDefault(p, 0);
count.put(p, c + 1);
```

理论上记了一个数之后应该计数 + 1，原本代码却仍然是 put(p, c)，便修改。为了配合这个修改，部分其它代码也对 c 加上了 1.

下面有两个 while 循环，可知是二分查找。从第二个 while 循环就能看出第一个 while 循环错误的位置：lo = mi，修改为：

```
if (A.get(mi).get(0).time <= t)
    // lo = mi;
    lo = mi + 1;
```

此时运行程序仍然有错，发现结果与候选人总有一个人的偏差，发现第二个 while 循环时间比较写错了，将 < 改成 <=，同时修改：

```
// int j = Math.max(lo, 0);
int j = Math.max(lo - 1, 0);
```

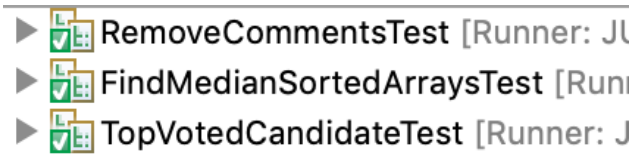
至此程序运行正确。

3.6.3 如何修正错误

修正是在发现错误的过程中同时进行的，在 3.6.2 中已经给出修正的方法和过程。

3.6.4 结果

通过了这几个程序原本的样例和自己写的几个样例。



4 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
5.13	18:00 - 20:00	完成 3.1	按时完成
5.14	20:00 - 22:00	完成 3.2	按时完成
5.15	19:00 - 22:30	完成 3.3 算法	按时完成
5.16	18:00 - 22:30	完成 3.3 GUI	按时完成
5.17	18:00 - 23:30	完成 3.6	按时完成
5.18	12:00 - 16:00	完成 3.4、3.5	按时完成

5 实验过程中遇到的困难与解决途径

遇到的难点	解决途径
Java Logging 无法满足很多用户需求，而且呈现方式不美观。	重写 Logging 的很多方法，能让其输出到.txt 文件并插入 txt 尾部更新，对提示信息进行修改让用户能看懂。
Debug 过程 Eclipse 的功能过于薄弱	用 VSCode 进行 Debug，使用一些 Debug 插件辅助我进行判断。

6 实验过程中收获的经验、教训、感想

6.1 实验过程中收获的经验和教训

(1) 每个函数应该有清晰的 Pre Condition 和 Post Condition，方便加入 checkRep 等检查代码。
(2) 每写完一个 ADT 就检查一个 ADT。
(3) 借助 Debug 工具进行 Debug 速度更快，VSCode 有很多高效的插件。

6.2 针对以下方面的感受

(4) 健壮性和正确性，二者对编程中程序员的思路有什么不同的影响？
要保证健壮性，程序员需要比保持正确性思考更多。保证健壮性，程序员在编程中应该为其它用户和其它程序员规避错误，这是一种更全面的思考，要考虑应对不可预期的使用情况的措施。

(5) 为了应对 1%可能出现的错误或异常，需要增加很多行的代码，这是否划算？（考虑这个反例：民航飞机上为何不安装降落伞？）
是否划算应该看程序的价值。如果程序应用在重要领域，或者对用户是直观重要的，那么增加很多的处理代码是必要的。但如果是程序的价值小于增加代码的人工成本，可以考虑简化这部分工作。

(6) “让自己的程序能应对更多的异常情况”和“让客户端/程序的用户承担确保正确性的职责”，二者有什么差异？你在哪些编程场景下会考虑遵循前者、在哪些场景下考虑遵循后者？
二者的差异在于程序面向的用户和使用场景不同。前者面向专业用户或者需求高度稳定的领域，后者面向非专业用户和日常领域。比如现在的服务器操作系统，比个人电脑操作系统要稳定很多，个人电脑操作系统崩溃概率大，甚至很多情况下会造成用户的工作内容丢失。而应用在专业领域的软件价格高昂，后期维护成本高，就是要保证程序的极度稳定，能应付各种错误。

(7) 过分谨慎的“防御”（excessively defensive）真的有必要吗？如果你在完成 Lab5 的时候发现 Lab5 追求的是 I/O 大文件时的性能（时间/空间），你是否会回过头来修改你在 Lab3 和本实验里所做的各类 defensive 措施？如何在二者之间取得平衡？
会，不能单一地注重时间空间效率或者安全性。要舍弃一些消耗很高资源但仅带来很小安全性提升的 defensive 措施，同时要舍弃一些应对较小出错概率的 defensive 措施。

(8) 通过调试发现并定位错误，你自己的编程经历中有总结出一些有效的方法吗？请分享之。Assertion 和 log 技术是否会帮助你更有效的定位错误？

从发现 bug 的位置一层一层向前找错误，每一层打印 log，在没有错误和有错误的 log 之间就存在 bug。

Assertion 和 log 非常有效。

(9) 怎么才是“充分的测试”？代码覆盖度 100%是否就意味着 100%充分的测试？

充分的测试意味着考虑到几乎所有的等价类和边界情况，可以应对诸如模糊测试之类的形式化方法。

代码覆盖度 100%并不意味着测试完备，这仅仅意味着所有代码都被跑过了。而至于测试有没有覆盖到边界情况（极端情况）、所有等价类就不一定了。

(10) Debug 一个错误的程序，有乐趣吗？

Debug 自己的错误程序会有乐趣，Debug 别人的错误程序没有乐趣。

一是因为很多程序没有完备的 post condition 和 pre condition，二是没有足够清楚的注释解释每个变量和函数的意义。

(11) 关于本实验的工作量、难度、deadline。

工作量和 deadline 合适，难度在 debug 部分大。

(12) 到目前为止你对《软件构造》课程的评价和建议。

希望每个 Lab 内容有条理，阐述清晰，就像 MIT 的实验内容一样。

(13) 期末考试临近，你对占成绩 60%的闭卷考试有什么期望或建议？//请严肃的提出，杜绝开玩笑，教师会认真考虑你们的建议。

希望考试知识性或者定义性的内容可以简单一些，考试内容与 lab 有一定的联系。