**课程实践作业四**

将Guttag, John的《Introduction to Computation and Programming Using Python》中的：

6.2.3 When the Going Gets Tough

6.2.4 And When You Have Found “The” Bug

翻译为中文。

**提示：**

翻译重在“意达”，翻译是再创作的过程。

**要求：**

文档持续更新：

1. 推敲修饰翻译文字；
2. 结合课程学习，逐步更新加入自己编程的体验。

**提交：**

1. 文档名称：学号\_姓名\_作业四.docx；
2. 电邮：[cmh@seu.edu.cn](mailto:cmh@seu.edu.cn),

邮件主题：学号\_姓名\_作业四；

1. 推送到GITHUB。

当过程变得艰难的时候

老约瑟夫·P·肯尼迪，肯尼迪总统的父亲，据称教导自己的孩子说，“当过程变得艰难的时候，坚毅的人总会挺过去”。但是他却不曾调试好一些软件上的错误以排除故障。以下这个部分包含了一些实用的提示，告诉我们当调试变得愈加艰难的时候该怎样处理。

* 排查寻常的嫌疑，举例来说，如果你
  + 在有错误的命令行将参数传递给一个函数，
  + 拼错一个名字，比如，当你应该打一个大写字母的时候却打成了小写字母，
  + 重新初始化一个变量失败，
  + 测试认为两个浮点数的值相等（==），而不是近似相等（记住：该浮点算法与你在学校所学的对应算法是不同的），
  + 所测试值的相等（比如，通过写表达式L1==L2来比较两个列表，当你有反对相等的理由时（比如，id(L1)==id(L2)），
  + 忽略了一些内置函数的副作用，
  + 忽略“()”使一个类型对象的引用变成了一个函数的调用，
  + 创建了一个无意的别名，
  + 犯了任何对你而言具有代表性的错误。
* 停止问自己为什么程序没有如你所想的运行。而要问你自己为什么会出现这种情况。这应该是一个容易回答的问题，它将很有可能成为搞清楚怎样修正程序的第一步。
* 记住错误很有可能不在你所认为的地方。如果真在，你应该很早以前就找到了。决定找错之处的一个实用的方法是问自己哪里不会出错。正如夏洛克-福尔摩斯所言，“排除所有其他因素，依然存在的那个必定是真相”。
* 试着去解释问题给别人听。我们所有人都在开发盲区。经常发生这样的情况：我们只不过试图跟他人解释问题，却引领你想明白了你错过的事情。尝试去解释就是为什么错误不会出现在确切的地方的原因。
* 不要相信你所看到的每一件事。特别地，不要相信文件说明。代码或许不会像意见所建议的那样执行。
* 停下调试程序，开始写文件编制，这能够帮助你从不同的视角处理问题。
* 停止思考、调试，明天再试。这意味着错误稍后将被修复如果你坚持下来的话，但是很有可能你会花费大量的时间来寻找到它。也就是说，以潜伏换取效率。（对于学生而言，这是一个极好的理由来说明要尽早忙于解决程序问题）。

当你找到那个错误的时候

当你认为你已在你的代码里找到一个错误的时候，开始编译和测试修复的迫切心情几乎是不可抗拒的。这往往是好的，然而，你要放慢一些。记住我们的目标不是修复一个错误，而是让程序运行得既快速又高效，向着完美的程序迈进。

问问自己，如果这个错误解释了所有问题，或者只是解释了冰山一角。若为后者，那么考虑这个错误与别的改变的一致性或许更好。设想一下，如果你已经发现某个错误是突变的列表导致的结果，你可以局部绕过这个问题（也许通过制作一份列表的副本）。或者，你可以考虑使用元祖来代替列表（因为元祖是不变的），或者可以消除代码中其它地方与之相同的错误。

在做任何改变之前，尝试并理解所推荐的“修复”的分支。它是否会破坏别的东西？它介绍得是否过度复杂？它是否给你提供机会去整理简化别的部分的代码？

个人编程心得体会：

老实说，最初接触学习python的时候，自己的内心怀着一丝丝的畏惧，因为以前在学习C++的时候，一直处于极其吃力的状态。在上第一次课的时候就感觉云里雾里的，听不懂老师在说些什么，也不知道这个软件具体的用途是什么？从第一次作业就吃尽了苦头，装python的过程中，我自己就遇到了正常范围以外的状况，还在我坚持了下来，耗了几天的时间，终于解决。下面是我学习编程过程中感触较深的几点：

1）安装软件的过程尽管一开始是按着老师给的教程一步步进行的安装，可在半路就是出现了正常范围以外的错误，这时候，没办法只好根据错误提示去百度搜索解决方案，如果按照百度的方法却仍没能解决问题的话，你就会陷入百思不得其解的境地，整个人都快要崩溃了，一个小小的程序错误，怎么诸多尝试也那它没有办法，这时候，你就要停下来好好想想，跟其他的同学或老师进行交流，往往旁人的眼睛是雪亮的，能够很快找出错误，而且事实证明，确实如此。

所以当你因为一个小问题思考耗费了大量的时间仍然无法解决的时候，就要改变思考的思维模式，毕竟一个人的能力是有限的，而且总是泡在一个问题上太久容易走进死胡同，试着去听听别人的意见或许就会迎刃而解。

2）编程的过程是千变万化，永无止境的。初学一款软件，往往都会有一些教程，我们都喜欢跟着教程一步步的进行下去，害怕出错，但是难免却总会出现教程中没有罗列出的状况，或者没有给出你所要实现的功能的指导，这时候，我们需要毫无畏惧的去尝试、去摸索，发现教程以外的其他功能，尝试总会有所收获的，只是有关大小而已。

3）就像书中所建议的那样：“停下调试程序，开始写文件编制，这能够帮助你从不同的视角处理问题”。我也深有感触，每次完成一个作业，开始编写文档，把刚刚自己进行的过程通过文字的方式回顾一遍之后，发现自己又想通了该过程中一直在犹豫怀疑的一些问题，然后就会发觉它也不过如此，或许这就是从不同视角思考问题的好处。

总之，编程就是一个不断探索的过程，它没有最好只有更好，所以，编程作业也永远没有一个最好的答案。一个功能的实现有各种形态各异的编程代码，我们应该追求以最简的形式实现最强大稳定的功能。