**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。

要求：

1. 翻译重在“意达”，翻译中能体现自己理解的“意”为佳；
2. 翻译以后根据自己编程体验，写体会。
3. 翻译文档内容：英文原文、翻译和编程体验

一、翻译

6.2.3举步维艰之时

据说肯尼迪总统的父亲教育他的孩子的时候曾经说过；“意志坚定的人会一如既往地前进，纵使前路荆棘遍布，举步维艰。”但是他从来没有调试过程序，当程序调试遇到瓶颈无法进行下去时，下面有几条实用的应对方法可以帮助你找到问题所在。

从常见的错误着手，比如说，你是否遇到过以下情况：

1.将参数传递给函数时顺序错误

2.拼写错误，例如：大小写错误

3.变量初始化失败

4.检查两个浮点类型变量是否相等而不是约等于（浮点数运算和你在学校里学习的运算是不一样的）

5.当你想要两个对象相等时，检查他们的数值是否相等（检验语句L1==L2）

6.忽略了一些内置函数的副作用

7.忽略（）将函数类型对象的引用转换成函数调用的作用

8.使用了一个未定义的函数或变量的别名

9.其他常犯错误

不要一昧地去想你的程序为什么没有按照你所设想的运行，而应该从另一方面着手，去思考程序为什么会这样运行，这相对于前者更加容易，也同样会启发你如何去解决程序中出现的问题。

错误可以出现在任何你所想不到的地方，你可以尝试去检查你所认为错误不可能在的地方，如S.H.所说：排除所有的可能性因素，剩下的即真相。

我们在考虑问题时都会有思维盲点，因而我们可以尝试请教其他人，当你将自己所遇到的问题阐述给他人时你也会发现你所遗漏的一些细节，比如说你可以将为什么错误没有发生在你所预定的位置这一问题同他人交流。

不要轻信你所读的任何资料，程序或许不会像那些参考资料中描述的那样运行。

停止调试程序，记录下你的编程过程，这也许有助于你从其他角度发现问题所在。

暂时放下调试工作，第二天再继续，这或许意味着拖延了你的进度，但是相对来说你可能会花更少的时间就解决了这一问题，也就是说提高了你的工作效率（同学们，这是一个很好的早一些而不是更晚去开始课程练习题程序设计的理由）

6.2.4 当你找到错误后

当你发现错误之后，不要急于着手解决问题，你的目的并不只在于解决程序中的一个错误，而是为了以后编写程序不出现类似的错误。

首先，思考这个错误是否是整个程序出错的根源所在或只是众多错误中的冰山一角，若是后者，调试时要注意此处错误与其他变化的一致性，例如当你发现了一个错误的原因是改变了一个列表，你可以局部地避开这一问题，或者可以用元组代替列表，这样做或许会消除程序中其他部位的类似错误。

更改程序前，尝试去思考各种不同的修改方法会不会带来其他的问题，它会不会使程序的其他地方出错或使程序过于冗长？是否能够使程序的其他部分变得简洁？

确保你可以使你的程序恢复到修改以前，以防你费尽心思修改你的程序却越改越错并且还不能还原到修改前，磁盘空间充足，记得存储你的老版本程序。

最后，如果你的程序出现了很多解释不了的错误，你应该考虑一下是否有更好的方法去编写程序或使用一些更加简洁的算法，这样或许可以正确地实现你的程序。

1. 体会

对于一个初学者来说，编程难，调试程序更难，我们总是会遇到这种状况，当自己费劲九牛二虎之力码出程序代码之后却不能运行或者不是按照所设想的运行，这种情况在所难免，然而更糟糕的是调试程序的时候，从头到尾字斟句酌却不能发现错误所在，而本文则给出很好的解决方法：

1. 首先不要有思维定势，不要拘泥于以往的编程经验，程序的所有位置都有可能出现错误，如果调试程序遇到瓶颈可以从你所以为最不可能出现错误的位置着手。
2. 我们也可以与他人交流我们所遇到的问题，这样有助于我们突破思维盲点。
3. 磨刀不误砍柴功，当我们实在想不出问题所在时可以稍适休息，这样也有助于提高我们的工作效率

发现问题就要解决问题，更改代码的时候要全方位考虑该修改对于整个程序的影响，并养成每一次修改前都保存代码的习惯。我们也要放宽眼界，多思考，不要仅仅局限于解决程序中的一处错误，而是要争取下次不出现同样的错误，提高我们的编写程序的能力。

学习一门计算机语言固然困难重重，这期间我们会犯各种各样的错误，但是功夫不负有心人，只要我们能够认真思考，从每一次出现的错误中吸取经验教训，多加练习，就一定能量变达到质变，进而熟能生巧，正如文中所说：“When the going gets tough, the tough get going”