**编程一点都不神秘，一点都不难**

编程和其他技能没有本质差别。就像语言学习里有语法和单词；就像数学里有不同的步骤和不同的题目；就像所有的技艺和手艺，编程里也有前人总结的针对不同任务的技巧，工具和好的习惯。这些东西，你都可以自由使用，修改。

有个人曾这样断定，程序大牛和编程届的码农之间有一个明显的区别——后者往往缺乏足够的智慧在编程界获得真正的成功。在这个人看来，这种智慧包含对指针和递归的理解。我在学校曾学过指针和递归。在学生时代，能理解指针和递归的感觉真心超级爽。这种快感激发我踏上了计算机学习之路。但在课堂练习之外，我极少需要碰到这两个概念。并且，之后，我一次又一次的发现，不用这两个概念同样可以写出很多有趣的程序。

所以，在每次迷惘和对前途，对是否有能力去学习的时候，保持冷静，保持初心，当你开始编程了，就不要害怕，不要怀疑学的是否有用，不要去想自己是否足够聪明。想这些都没意义。确实，编程任务越复杂，越难懂，你就需要越多的工具和技巧，时间才能完成。但哪一个领域不是如此呢？

**程序，没人能一次搞定**

当你第一次学习编程时，你会很快撞上这么个问题。你觉得你已经配置好所有的开发环境了，你编写的代码没有问题了，你查了一遍又一遍，但你的代码就是**有！问！题！**你对如何排错毫无头绪，你想放弃，你想怎么这么难，你觉得你不可能搞定，自己不是一个编程的料。但是，你要明白，这种经历对任何一个水准的程序员而言，都是再正常不过了，因此每一个ide才有debug这个工具。**有这种经历，并不代表你的智商，你的悟性，或你和coding的适合度有任何问题。**不管你是编程大牛还是一个新手，都会有这样的经历，区别在于如何对待这种经历：这是一个更好的学习的机会，有错误，才能更深化理解，才能进步。

新手和大牛的一大区别就是信念。什么是信念？就是深信出错的原因是符合逻辑的，并可以找到的，深信问题肯定是可以被解决的，深信总有办法可以实现自己的目标，可以完成项目。从0到1之路也许并不明显，但只要有耐心和坚持总是可以找到的。

**总有人说你做的不对**

大括号{}应该怎么放？该不该用tab来缩进？该不该对代码添加注释？对于这些问题，大家的做法各有不同。没有谁有标准答案。很多程序员热衷推销自己偏好的处理方式，但这不意味这答案只有一个。事实上，与那些说我做的部队的人面对面，然后在努力去搞懂他们说的ide是不是对的，这种来来回回，是我职业生涯的压力源之一。如果你与一个团队的其他成员一起code，总会有人不认同你的某些行为。有事他们是对的，但事实上你到底是对是错，永远值得你亲自去推敲。有时他们纯粹就是物理去闹，你别理他们就好了。

**坚持比方法重要**

关于“正确编程学习法”和“最佳编程学习法”的文章有很多。的确，学习编程的方法有很多。你可以看书来学，你可以做练习来学，你可以给别人的程序找bug来学。当然，也有很多中编程语言，你可以挑选来做为你的第一门语言。自学型的鞭策和那个课程或者讲座系列常常有个问题：一开始你总会学的很爽，但难度会陡然上升。print命令总是很简单，但要整整搞定一个实用程序的编写，往往让人抓狂。你很有可能觉得跟着教程走并没有搞懂，然后你就开始抱怨教程有问题。当你撞上“编程玻璃顶”时，哪些教程和线上资源对你的意义已经不大了，因为他们默认你已经是一个编程好手了。

更让整个编程学习进阶之路变难的因素是，你根本不知道自己缺什么。甚至搞清楚自己接下来需要做什么都成了难题。不管上什么编程课，你都会有一段“撞墙期”，唯一的解决办法就是坚持到底。这以为这你要不断的尝试新事务，学习新知识，不断地，一步步地，去解决问题，去编程你要的程序。如果你这时认真回看自己的编程初心，你就更有可能获得成功。

坚持到底，就会胜利。关键是你没有坚持。你改变的决心还不够大。

、

初心：

我喜欢coding时候的键盘声。

我认为编程可以赚钱，并且可以給我带来安全感。

**在决定做一件事情之前，先不要考虑它的困难有多难，要先想想收益明显不明显，如果明显那就克服一切困难把它搞定**