零基础小白如何入门 Python 编程

背景

1. 本文主要根据作者2个月以来对于 300 人的零基础python教学经验总结而出，适合零基础、负基础学习 python 编程语言的同学阅读。

2. 本文所述仅仅是方法，看完本文并不会让你学会任何一门编程语言，仅仅能让你少走一些弯路，少被毒害一些。

引子

我们来探讨的问题是 —— 如何从零开始学习python。

相信你能看到这篇文章，一定心里有一个**学习编程**或**教编程**的想法，本文分享的知识可以为这两个目标提供一些参考价值。

为什么是我来讲这个事情呢？在过去的两个月时间内，作者在自己开发的教育平台——uband友班上面开办了一个python班级，报名的用户 75%是零基础用户，还有25%是之前学过，这25%中有一些负基础用户。

如何教？是不是我发一教材，然后让大家顺着开始学？是不是我们先进行一些理论学习，视频授课？

我分析了一下情况，发现以下几个比较有意思的点：

1. 来学习的人之中有 70%是女生。

2. 学生党和工作党的比例是 3:2。

3. 绝大部分人一天只能抽出1 ~ 2个小时学习（非脱产）。

4. 基础几乎都没有。

无疑这是一个极大的挑战，下面我就来讲一下如何教学。

学编程从理论还是从训练开始？

这个问题让我回想到了自己的科班的学习过程，有这样一个故事给大家分享：

> “2012年，过年回家的火车上，我提着行李走上了拥挤的车厢，对面的小哥顶着蓬松的头发，我知道，这也许是我的同行。24小时的上海到贵州的旅程就没那么孤单，因为我们都在讨论架构、算法、编程学习…这个工作了5 年的前辈给我的建议是，你现在看这个架构、编程理论等书籍，不求甚解即可，等你有了大把经验，回来很快就能秒懂。

时间过去了2年，两年中我也没有看任务关于python的书籍，不过用它写了七七八八不下几万行脚本解决各种问题。有一些我在图书馆等人，随手翻看了一本《Python cookbook》，是一本500+页的大部头书，我就那么一页页翻看，等的人到了，我也看完了。

请问，我等了多久？

答案是：两个小时”。

这个故事让我思考了学习的两种不同的模式。

学习的模式 — 创造还是模仿

先给大家看两个学习者：

1. 想要通过学习python编程完成一个自己的个人网站。
2. 想要通过学习python，对比python和其他脚本编程语言的设计优劣，然后改进之。

显然这两个人的学习方式势必是不一样的。

第一个人的学习重点在于 —— 模仿，这将是大多数程序员所做的工作，在这个阶段，不要说编程是创造性的工作，顶多能说编程是一个有逻辑性的工作，模仿就可以了。

第二个人，显然是一个编程科学家，北美的高校有一个研究方向叫做 programming language，我有几个过去的同学都读了这个专业的博士，还有我们著名的编程届网红“王垠”同学也是读的这个专业，这个专业的使命是什么—— “创造一门编程语言” or “改进现有的编程语言”，他们的工作，模仿是远远不够的，需要了解整个理论体系、设计原则、处理细节等等…他们不能靠模仿来学习，因为没有什么可以模仿。

而学习编程最大的悲剧 —— 就是明明自己的目的是模仿逐步形成自己的技能，而误以为自己要去创造新事物。

所以学习编程之前需要搞懂 —— 做创造性的工作；还是做模仿性的工作。

创造性的工作举例：诗歌创作、生物学研究、新车设计、设计一门编程语言、设计一个新的web容器。

模仿性的工作举例：朗读英文诗歌、播音工作、编程一个APP、驾车、游泳、吉他弹唱、跳交际舞、雕刻小人、书法…

可以看到我们很大一部分事情，都是模仿类型的工作，模仿类型的工作时不需要从理论开始的，而是从练习开始的。比如学习游泳，没有一个人是从看了一个月游泳的书籍开始的，而是教师在水下给你讲，你去模仿他的动作，然后给你纠正。比如学习英语，就不说了，这么多的人学习了十几年还不能用，就是缺乏练习。

而学习编程，还是教编程，大部分也是模仿式的学习，模仿式的学习就要从练习开始，或者以练习为主的训练开始，不然很可能浪费了大量的时间而没有看到成果而放弃。（我们接触到的编程学习者这样的情况不在少数）

了解了这个道理，我们开展教学就容易多了，那就是本着一个原则 —— 一开始就模仿写代码。

开始学习

本文后面的部分是讲 —— 如何进行模仿性的python学习，如果上面你的目的是创造性的，可以不看了。

那么我们就要开始做真正的学习了，以下我以我自己设计的教学流程作为参考，指导一个零基础开始的教学过程。

从一个故事开始学习变量和判断

首先我们从一个老妈买菜的故事开始，第一周学习的内容是学习变量，可以看下面一张图，我们是如何教变量的，记住这个是第一天，第一天学习编程，就要开始写代码，这是我的教学过程要坚持的。不管写的多，写的少，都要开始模仿开始写。

需求是下图这样的：

what is variable?

变量，是你编程生涯中最好的伙伴～

Sample1 : 老妈去买菜

价格多少	白菜的描述	便宜与否
数字	字符	判断

我妈去菜市场，看到了有小贩卖菜
小贩牌子上写了“西双版纳大白菜”
我老妈询问了价格，是 6 元一斤，在我老妈的印象里
小于 5 元的是便宜的，如果便宜，就买个 2斤
否则扬长而去

请写代码～开始你的表演～

我们来看代码


```
#!/usr/bin/python
# -*- coding: utf-8 -*-
# @author: Guoshushu

# For beginner
# 1. variable - num,str,boolean
# 2. if
# 3. > < >= <= ==
# 4. print
def main():
  who = 'xiao的老妈 '
  good_price = 6  #小贩的价格
  good_description = "西双版纳大白菜" #小贩的招牌

  is_cheap = False #是否便宜
  reasonable_price = 5 #老妈能接受的最高价格
  buy_amount = 2 #准备买 2 斤

  # 开始你的表演
  # go 我们来走一组
  print "%s上街看到了%s，卖 %d 元/斤" % (who, good_description,
good_price)

  if good_price <= reasonable_price:
    print '她认为便宜'
```

```
        is_cheap = True
        print '她买了 %d 斤' % (buy_amount)
      else:
        print '她认为贵了 '
        is_cheap = False
        print '她并没有买，扬长而去'

      #homework
      #1. 看 day1-homework.py

# run function
if __name__ == '__main__':
    main()
```

看到了么，这一段代码实现了我们的需求，而且完全是 自然语言编写 ，普通人都可以看懂，我们教会了:

```
# For beginner
# 1. variable - num,str,boolean
# 2. if
# 3. > < >= <= ==
# 4. print
```


可以注意到一点，我没有教大家学习 `if __name__ = “__main__”`：这些事什么意思，而是让大家开始模仿，大家都是成年人，都有悟性，不需要花费太多的时间去低效地学习理论，之后自然会明白。

上面这一段代码，看懂就可以了么？

不，我会要求我的学员自己在电脑上打一遍，真正他们打了过后，就会发现其实看懂是没有用的，因为他们会发生以下的问题。

1. 跑代码前没有保存成.py的文件；
2. if之后的符号用了中文的：（冒号）。
3. 缩进写错了报 indent error。
4. True和False写成了 true 和 false。
5. **name** 写成了 *name*
6. 64位的系统装成了32位python，虽然已经提醒过。
7. 变量名字之间用了空格，报错。

   ....

这些问题，不去实际写代码是发现不了的，但是好消息是，发现了一次，之后就基本不会再犯错了，所以我们说编程模仿练习的必要，以及编程中犯错报bug的必要。

如何学习列表

这里再举一个学习列表的例子，我们来看下面的代码：

```
# -*- coding: utf-8 -*-
# @author: Guoshushu

def main():
  good1 = '大白菜'
  good2 = '空心菜'
  good3 = '花菜'
  good4 = '生姜'
  good5 = '小龙虾'

  # ....... 省略掉 100 个
  good100 = '蚌壳'


  print '老妈看到了 %s '% (good1)
  print '老妈看到了 %s '% (good2)
  print '老妈看到了 %s '% (good3)
  print '老妈看到了 %s '% (good4)
  print '老妈看到了 %s '% (good5)

def main2():
  goods = '大白菜，空心菜，花菜，生姜，小龙虾'
  print '老妈看到了 %s' %(goods)

def main3():
  print '-------'
  lst = ['大白菜', '空心菜', '花菜', '生姜', '小龙虾'] #列表
  for lst_item in lst:  #遍历
    print '老妈看到了 %s '% (lst_item)

if __name__ == '__main__':
  main()
  main2()
  main3()
```


相信大家已经看到了，这一段是说明列表为何要使用，学习的过程需要了解为什么要用列表，那就是 变量不够用了嘛。

我们看上述3个函数就能够明白，列表这个数据结构的演变过程，其实就是变量不够用了，要用一个更牛逼的结构来一次放很多个数据，这样理解起来会很自然。

相似的方法我用到了教授元祖、数组等等方面，你要如何学？ 我给你的建议就是，用起来，用起来。

所有的 3 周的学习的代码在我的github上，都可以自取，当然我不可能在一篇文章里面全部讲完，你可以去 github 下载我们的代码记录，模仿。

本文以下附件内的github地址是我们 28 天的教学记录和作业记录，这个chat的同学可以试着开始写，不过不好意思不能提供视频教程。

一个月的碎片时间学习后，我们的学员可以完成以下的作业：

你可以在这一周用github的代码自学下，在chat里我们讨论这个。

教学的体会

最好的心态是不断犯错、不怕犯错

我们的教育告诉我们，不要犯错，犯错是不好的，但是编程中，我们写出了一个bug，编译器给我们报错了，那么就是一个学习的机会。

看到报错的时候心态就是 —— 我又有了一次学习的机会。而不是 —— 妈的如何又报错了。这是编程时候面对报错的太多。

道理我都讲完了，同时你可以跟着我的github代码进行学习模仿，最重要的当然是坚持，但是我倾向于写成 —— 用正确的方法坚持学习。就能很轻松学会。如果你是想要教编程老师，请一点要用模仿的方式教别人，不然会误人误己，切记。有任何问题，可以在我的chat中讨论。

碎碎念

我前面提到过，其中有大部分的编程学员都是女同学，我发现她们的学习能力也是超强的，一个月的时间，而且几乎都是每天 1 小时左右的时间，就可以在引导下基本掌握编程的基础知识，很了不起。

社群的力量还是非常之大的，一开始学习，配置环境、写第一行代码的时候，群里的同学也是鸡飞狗跳的，我真是感慨自己有勇气开一个在线的教学生动手的编程课程，要知道，一般为了省事，大家都是采用录视频讲一个看似非常详细的课程，来规避这个麻烦。

不过实践下来，这个效果的确最好，我们可以在chat里面聊一聊，我也会请我的学员来分享自己的学习经历。

资源

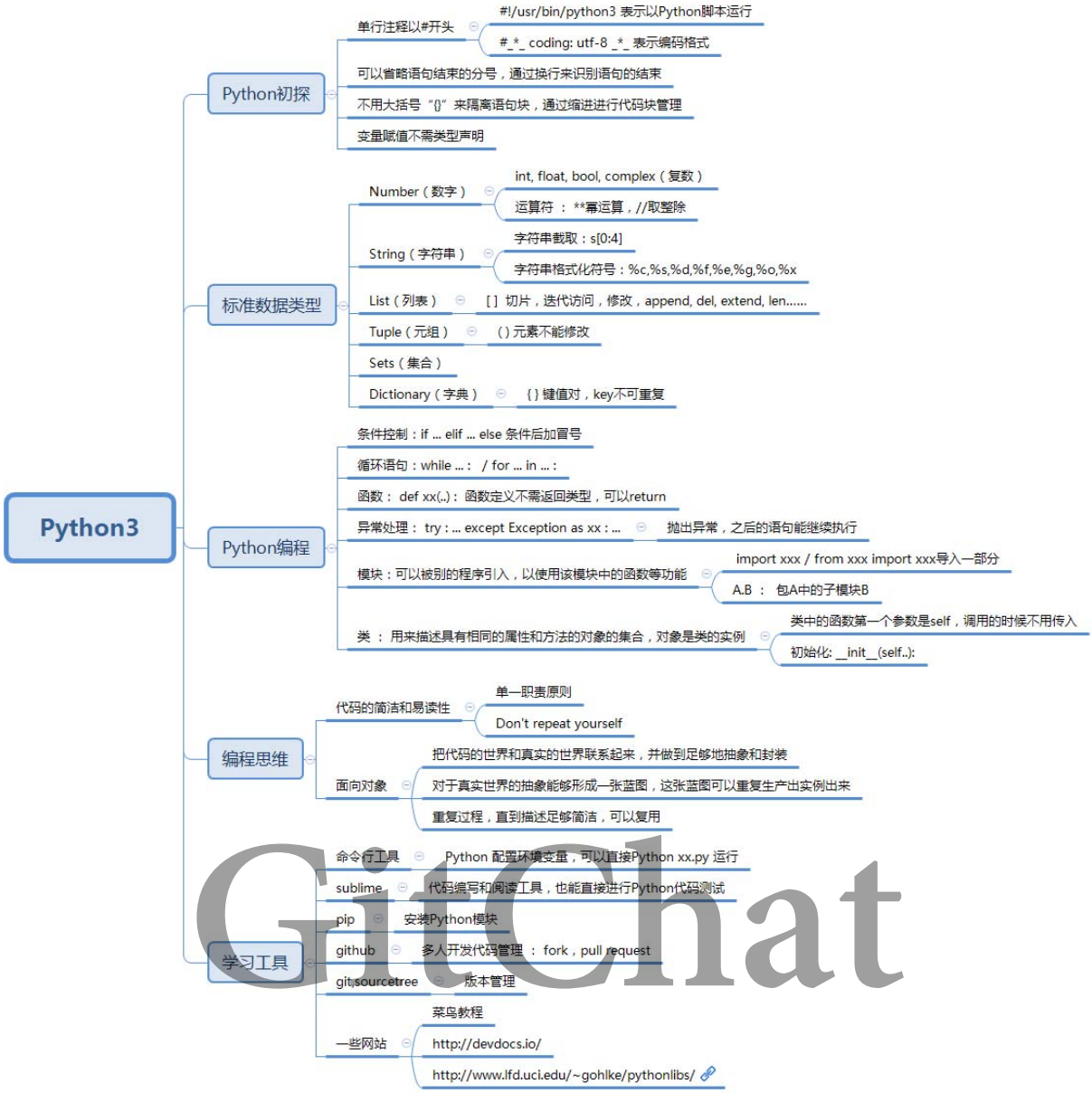
- 编程班github
- 菜鸟教程python

其中 github 里包含了所有 4 周的作业和同学提交的作业，在homework文件夹里面，大家可以下载下来模仿学习。菜鸟教程在这一个月的过程中，基本已经教授完毕，而且是潜移默化的过程。

反馈和总结

以下是几篇学员的总结，我贴出来给大家查看。

一个月的知识汇总：



学员写来的部分邮件：

> I really appreciate ShuShu’s coding class. It’s like he’s bringing a torch of enlightment to all of us “coding cavemen”.
>
> My girlfriend who is the organizer of “R Girl” in Taiwan is doing almost the samething. We all belive girls can do coding as well as boys. Just, who is the first person to devote in girls’ education?
>
> Anyway, thanks a lot. Hope I can join in this class next time.

开始报名的时候想的是反正这一个月是毕业季也不用上课，只要把论文答辩搞好就没事了，还想着可以扩展一下自己的知识领域，我一个英专生脑子一热就报了编程班。结果真的是忙成狗，没打过n本论文的人都不好意思说自己要毕业，要准备答辩，还要帮着老师忙毕业季各种事情，更可怕的是我9月份还要考司法考试，真是在这个期间无数次想要放弃编程班的学习。。。但是我一直相信一句话，自己选的路，跪着也要走下去，所以呀，自己挖的坑不管怎样都要自己填平。好在不是每天都要交作业，只要有时间我就会

听蜀黍的视频课，打个代码练习一下，想不明白就放在一边过一段时间再想想，在练习一下，实在跑不通就在群里问问小伙伴，于是，这一个月就这样跌跌撞撞走过来了，真的觉得坚持下来自己已经成功了！同时，感谢群里小伙伴和我的队友对我的帮助，耐心解答我的问题；也真的真的很感谢笃师的细致讲解，让抽象的编程知识变得通俗易懂；也觉得笃师真的很辛苦，每天那么忙还要深夜录制视频，保证第二天的推送，老师这么认真，作为学生也应该对自己负责，好好学习每天的知识。。。

个人感受

在大学期间，我只学习过简单的计算机入门，后来自己考了个office，基本就是考前刷了点题就通过了，但是这真的和编程有很大的不同。随着基础知识的不断深入，学习了更多的知识，我真的觉得编程是一个全新的世界。开始每次跑代码，发现错误就很慌张，为什么又bug了呢，觉得变成好麻烦，自己是不是学不会了呢，但是蜀黍一直强调学好编程就是一个不断debug的过程，于是我就每天接受自己各种花式报错的袭击，慢慢的自己的心态发生了转变，学会分析自己的错误，一点一点去改正，最后跑成功的喜悦真的难以形容。还记得画小乌龟的那节课，看着屏幕上的效果图，自己成就感十足；接触类的概念，我觉得编程不只是要一个解决问题的方法，更重要的是要一条通向成功的捷径，于是有了高度抽象化的概念；字典、元组、列表等等都使僵硬的数据变得有条理。一个月的编程课下来，我觉得自己具备了分析解决简单问题的能力，一步一步规划要实现的内容，而不再只是看到需求背景就很慌张不知道怎么解决，这就是最大的进步。

课程建议

建议在基础知识讲完之后，s2班级可以有不同笃师交替进行授课讲解，这样也可以分担蜀黍的教学任
务，同时学生也能够体会不同程序员的编程思路。

不足反思

在这一个月的学习当中，我发现自己举一反三的能力还有待提升加强，可能也是练习不够，同样的函数换了个小背景，就要想好久才能写出代码；同时，我的复盘只是自己回顾了一下每周学到的知识，但是如果能够讲给另一个人听我觉得效果会更好，让别人能够听懂自己才是真的明白了学到的知识；没有系统总结自己的错误，有的时候犯过的错误会在下一次又不经意的出现，今后的学习应当多思考不要再重复自己的错误。

学习感悟

历时一个月左右的编程班很快就结束了，在这一阶段基本把python的基础内容学习了一遍，虽不能说完全精通，但也至少在我的脑海中留有印象，至少在蜀黍的指导下能够输出一些代码，做些任务。从“一窍不通”到“入门”（算入门了吧），学习了不少，起初自己也想学点编程，后来总是不了了之，这次跟着蜀黍和大家坚持下来，还是蛮开心的。

但是，在日常的练习中，自己思考还是比较少，有所欠缺，常常跟着蜀黍的思路走，所以自己独立思考解决问题还是比较少，可能自己练得太少，所以在写大作业的时候基本没什么思路，需要借助蜀黍的指点，才能完成作业（唉= =#看来学习能力不够强，还要多多锻炼，多多敲代码）。

在最近的一段学习时间中，由于一段时间处于考试周，学习断断续续的，投入的时间并不是很多，经常以完成任务就结束了（不应该啊），而且群里讨论参与度也比较少，平时阅读也比较少（敲脑袋，要反省）。

课程体验

1. 蜀黍备课详细，解释地道，通俗易懂。
2. 任务合理，基本能够按要求完成。
3. 很适合这种非专业、零基础的学员。
4. 群里的成员积极讨论，能够很好帮助大家解决问题，同时还有问题集锦，有助于复习。

下期计划

继续跟着蜀黍学习，哈哈

1. 反复学习蜀黍的学习资料。
2. 认真完成每天的作业。
3. 认真做总结，和同伴交流。
4. 学有余力，多阅读相关材料、网站等。
5. 多敲代码，多敲代码，多敲代码。

无论如何，基于自己的兴趣报了编程班（应该不会产生厌学情绪吧），希望在编程班中和大家一起学习，掌握一门技术（技多不压身，哈哈），学以致用嘛！

更多的内容将在chat里面提供。

后话

Talk is cheap, show me your code.

练习在编程入门的过程中，的确是最重要的，希望看了这一篇文章的同学，如果有对于编程学习的想法，一定不要止于看书、止于假学习，这个技能，是一行一行敲出来的。

共勉。