爬虫00-给刚接触编程的童鞋的一点小建议

我们刚去接触一个新的领域的时候，绝大多数人都会有很多疑惑，毕竟天才只占极小的一部分。

我们遇到问题的时候，不假思索，第一惯性就是去寻求周边人的帮助，毕竟这是最省时省事的方式。

但这样只会让我们养成依赖，对于这里面蕴含的知识是否真正理解，其实应该打个问号。

python之禅中有这么一句话，“做也许好过不做，但不假思索就动手还不如不做。”

在编程的学习过程中，**重点关注的应该是推导的过程，而不是结果**。

拿到一个问题，不要着急去编写代码，首先应该尽可能地去明确问题的逻辑，之后再开始编写代码。

当这个问题的整体逻辑对你来说，有点复杂的时候，你应该尽可能去梳理自己能够明白的那部分逻辑，用代码去实现出来。

如果你发现思路是正确的，但是代码运行却报错了，该怎么办？

编程就是在不停地写bug，然后不停地解决bug，**对于这些报错，我们应该要能读懂**，因为以后你会频繁地遇见这种错误。

起初，我们肯定看不懂这些错误是什么意思，又是由什么引起的，这时候，我们可以借助百度或者翻译软件去解读这些错误。

将报错信息复制到百度上，你能发现有一连串的网页都指向这个错误。

要知道我们遇见的绝大多数问题，前人也都遇到过，网络上也基本会有相应的解答，往往还会伴随着错误的原因以及要注意的方向，这将是我们迈向自主解决问题的第一步。

通过自己搜索也好，摸索也罢，完美地解决了一个问题，带来的不仅仅是对知识理解的深入，还有满满的成就感。

对于那些不理解的，没有一点思路的部分，我们可以去寻求助教的帮助，但在这之前，**一定要保证自己认真思考过**。

而且要知道**我们发起提问的目的是什么，是要理解问题的推导思路，理解为什么要这么做，而不是只要一个结果，这样的结果并没有意义**。

既然知道了提问的目的，那就应该有针对性地去进行提问，一个优质的问题应该包含哪些方面呢。

1. 提供自己编写的代码和错误提示信息，简单说一下自己对错误的理解（也许在说的过程中你就已经明白了错误的原因）。
1. 告诉助教自己的思路是什么（这样助教能告诉你思路正不正确），哪个地方没有想明白。

那么，当我们得到了这个问题的答案之后就结束了吗？

并没有。问一问自己，当再次遇到类似的问题，自己能够解决吗，自己又是否真正理解了错误的原因？

只有深刻理解了问题的推导过程，明白这个地方为什么要这样写，那个地方为什么要那样写，才算结束。

如果你还想更进一步，要考虑的是当前问题是否有更加方便的解法？是否有更加灵活的写法？

我老师当年告诉过我一句话，至今记忆颇深。编程不应该仅仅追求功能的实现，而应该追求优质代码的编写。

同一个问题，可以有多种代码的写法，有的复杂，有的简单，有的性能高，有的性能低。我们应该追求简单且性能高的写法，这要求我们对知识深刻理解，并且能够灵活应用。

会有一些程序员工作一两年之后，觉得编程很无聊，来来去去就是编写那些代码，因为他们只是单纯地实现功能，而并没有去追求更优质的写法。

我的老师告诉我，他接触了很多门语言，其中最喜欢python，因为他总能在python中发现一些小惊喜。

总而言之，python是一门非常灵活的语言，有很多小技巧可以更好地去实现我们的需求，这些都有待我们自己去发掘。

最后，我们学习编程，目的是为了应用它去实际地解决问题，而不应该只是一味地学习。

我们用python来实现我们想实现的需求，得到的成就感是无法想象的。

愿大家都能在python之路上走得越来越远。

Python爬虫的优势

```
1  PHP：虽然是世界上最好的语言，但是天生不是干爬虫的命，php对多线程，异步支持不足，并发不足，爬虫是工具性程序，对速度和效率要求较高。
2
```

```
Java: 生态圈完善,是PYthon最大的对手,但是java本身很笨重,代码量大,重构成本比较高,任何修改都会导致大量的代码的变动.最要命的是爬虫需要经常修改部分代码
# 爬虫 -> 反爬 -> 反反爬 -> 反反反爬 ....

C/C++: 运行效率和性能几乎最强,但是学习成本非常高,代码成型较慢,能用C/C++写爬虫,说明能力很强,但不是最正确的选择.

# JS:  DOM,事件, Ajax
Python: 语法优美,代码简洁,开发效率高,三方模块多,调用其他接口也方便, 有强大的爬虫Scrapy,以及成熟高效的scrapy-redis分布策略

```

Python爬虫需要掌握什么

```
Python基础语法
HTML基础

如何抓取页面:
 HTTP请求处理,urllib处理后的请求可以模拟浏览器发送请求,获取服务器响应文件
解析服务器响应的内容:
 re,xpath,BeautifulSoup4,jsonpath,pyquery
 目的是使用某种描述性语法来提取匹配规则的数据
如何采取动态html,验证码处理:
 通用的动态页面采集, Selenium+PhantomJs(无界面浏览器),模拟真实浏览器加载js,ajax等非静态页面数据
Scrapy框架
 国内常见的框架Scrapy,Pyspider
 高定制性高性能(异步网络框架twisted),所以数据下载速度非常快,提供了数据存储,数据下载,提取规则等组件
 (异步网络框架twisted类似tornado(和Django, Flask相比的优势是高并发, 性能较强的服务器框架
```