最常被程序员们谎称读过的计算机书籍

马克·吐温曾经说过，所谓经典小说，就是指很多人希望读过，但很少人真正花时间去读的小说。这种说法同样适用于“经典”的计算机书籍。

在Stack Overflow(以及其它很多软件论坛)上，诸如”程序员最应该读的计算机书籍有哪些？“这样的问题会周期性的出现。这样的问题不断的被提出、被回答，只是形式不同罢了。相同的几本书总是会出现在清单的前几名内，所以，如果想知道人们谈论的都是些什么，你有必要去读一读这些书的。

大多数程序员真正读过的计算机书籍

1. 代码大全(Code Complete)——两届Software Jolt Award震撼大奖得主！
2. 程序员修炼之道（The Pragmatic Programmer）
3. C程序设计语言(C Programming Language)(第2版)
4. 重构:改善既有代码的设计（Refactoring: Improving the Design of Existing Code）
5. 人月神话（The Mythical Man-Month）
6. 编码——隐匿在计算机软硬件背后的语言（Code: The Hidden Language of Computer Hardware and Software）
7. Head First 设计模式（Head First Design Patterns）
8. 编程珠玑（Programming Pearls）
9. Effective Java中文版（Effective Java (2nd Edition)）or Effective C++（第三版）中文版
0. 测试驱动开发(Test Driven Development: By Example)

上面的这些书我自己都读过，所以我不难相信很多不是很优秀的程序员也都读过它们。如果你对编程有足够的兴趣，能够来到这里读这篇博客，你很可能读过其中的大部分，甚至还有很多不在这个清单中的，所以我就不浪费时间每本书都评论一番了。我想说的是，这个清单上的每本书都是它各自领域里的奇书。所以，很多有愿望不断提高自己的编程技术的程序员都读过这些书，这就不足为怪了。

在人们备受推崇的计算机书籍中，还有一类书受到了独特的待遇。我称下面这个清单为“最常被程序员们谎称读过的计算机书籍”。这并不是说推荐这些书的人都没有真正读过它们。我只是有相当的信心怀疑更多的人只是在口头上宣称读过下列书籍，而实际上很少人真正读过它们。下面就是这个清单。

最常被程序员们谎称读过的计算机书籍

1. **算法导论（Introduction to Algorithms）(CLRS)**这本书的名称是所有出版过的计算机书籍中最让人误解一个。它被广泛的使用在很多大学里，通常被当作毕业生必需的算法课程。于是，只要在大学里上过计算机课程的学生几乎都有一本这样的书。然而，除非你拥有计算机硕士学位(而且是算法研究领域的)，我怀疑你顶多只读过算法导论（Introduction to Algorithms）里节选的几章内容。这个书名让人误解，是因为”Introduction”这个词让人以为它很适合初级程序员。实际上不是。这本书对算法做尽可能详尽综合的介绍，就像其它一些随处可见的类似的书一样。请不要再把这本书推荐给初学者。
2. **编译原理(Compilers: Principles, Techniques, and Tools) (the Dragon Book)**.这本恐龙封面的书涵盖了开发一个编译器你所需要的全部的知识。它的内容包括词汇分析，语法分析，类型检查，代码优化，以及其它很多高深的题目。请不要把这本书推荐给初级程序员，他们需要的只是分析简单的包含数学公式或HTML的字符串。除非你真的需要实现一个能够实用的编译器(或解释器)，你根本不需要掌握这本“恐龙”书的全部强大威力。把它推荐给一个遇到简单文本分析问题的人，这证明你根本没有读

过它。

3. **计算机程序设计艺术(The Art of Computer Programming) (TAOCP)**我经常听到人们把这本书描述为“每个程序员必读”的系列计算机书籍。我认为这明显不是实情。在我说出这样大不敬的话、被你们用板砖拍死之前，请让我做解释一下。这不是一本让你一页一页翻着读的书。这是一本参考大全书。把它放在你的书架上看起来会很不错(实际上也它确实很好)，但如果想把它通读一遍，你需要几年时间，而且最后什么都没记住。这并不是说手边放这样一本书没有什么价值。它是一本参考书，当我遇到难题，走投无路时，很多次我都在这本书里找到办法。但这本书终究是被我当作参考书。它复杂难懂，很理论，里面的例子都是汇编语言的。好的一面是，如果你想在这本书里寻找针对某一问题的解决方案，如果你找不到，那就说明这个问题无解。它是一本对它所涉及到的领域做了最最详尽介绍的一本书。
4. **设计模式：可复用面向对象软件的基础(Gang of Four)**这本书是唯一一本在这个清单里我从头到尾读过的书，读的结果是，我不知道该把这本书归到哪个类别。它出现在这个清单里，并不是因为我认为只有很少人真正读过它。很多人都读过。只是因为有更多推荐过这本书的人自己却没有读过。Design Patterns这边书的问题在于，很多书里给出的信息，你在其它很多地方都能看到。这样就使得一个初学者在维基百科上读了几篇关于设计模式的内容后，就敢在面试中宣称自己看过这本书。这就是为什么Singleton成了一种新的全局变量的原因。如果有更多的人花时间读过这本也叫做Gang of Four的书的原著，那世界上就不会有这么多人会把17种设计模式硬塞到一个日志(logging)框架里了。这本书最精彩的部分是每章里描述如何正确的使用一种模式的段落。遗憾的是，这些精华却在很多其它设计模式资料里被漏掉了。
5. **C++程序设计语言(The C++ Programming Language)**这本书不像一本编程教材，更像一本编程语言参考。有很多的迹象表明有人确实读过这本书，否则我们不可能有这么多的C++ 编译器可选择。编程初学者(或者甚至其它语言的专家)，如果想学C++，不应该直接去啃C++程序设计语言(The C++

Programming Language)这本书。告诉他们去读《C++ Primer中文版》。

正如我之前说的，我知道你们当中会有一些人真正的读过这些书。那这篇文章不是针对你的，针对的是那些企图通过假装读过这些书来表现自己的民众。 **如果你自己没有读过这些计算机书籍，请不要推荐给别人。**这样做会耽误别人的时间，误人子弟，因为一些阅历更丰富的人可能会有更好的书(更针对某一领域，更容易理解，跟某种编程语言或某种编程水平更契合的书)来推荐。除此之外，你也能避免被那些真正读过计算机程序设计艺术(The Art of Computer Programming)的人用MMIX知识给拷问住造成的尴尬(如果你不知道我在说什么，那我指的就是你)。

[英文原文：Books Programmers Claim to Have Read]