<<高质量程序设计指南>>

图书基本信息

书名：<<高质量程序设计指南>>

13位ISBN编号：9787121041143

10位ISBN编号：7121041146

出版时间：2007-5

出版时间：电子工业

作者：林锐,韩永泉

页数：394

前言

第3版　　大约在2005年年初，本书第2版（和第1版）已经售完。
至今本书仍然受到软件公司和C++程序员的关注，不断有读者询问我从何处可以买到本书、什么时候再版。
说来惭愧，我从2002年写完本书第1版后，再也没有接触过C++编程，现在对C++已经很陌生了。
2004年1月我离开上海贝尔，创办了上海漫索计算机科技有限公司，专注于IT企业的研发管理整体解决方案（包括软件产品和咨询服务）。
我自己已经从技术专家转型为企业管理者，关注商务多于软件技术。
对于出版本书第3版，我的确心有余而力不足。
幸好第2版的作者韩永泉仍然从事应用软件开发，宝刀未老，他全面操办了第3版，我只是挂名而已。
在撰写第3版的时候，为了更进一步突出本书一贯强调的“高质量程序设计”理念，对原书第2版的内容做了一些调整：　　首先是对第2版进行了全面的修订，改正了所有已经发现的错误，并对原有部分章节的内容进行了补充；　　其次，删除了第2版的第2章和第17章（名字空间和模板）。
根据我们的观察，除非是开发类库等通用程序，第17章的内容在现阶段对应用软件开发人员一般不具有实际指导价值；　　最后，增加了大约10个小节的内容，分散在各章中。
这些增加的内容是实际应用软件开发过程中经常会用到的技术，可以显著地提高编程效率，增强软件的健壮性和可移植性。
不论本书第1版和第2版是好是差，它都被过度地使用了，产生了令作者始料不及的影响。
本书的试题被国内软件公司大面积地用于C++程序员招聘考试，结果事先看过答案的应试者考了高分而被录取，还真有人向我致谢；也有不少人未看过答案而考了低分未被录取，在网上把作者骂一通。
本书的试题和答案早在2002年就公开了，不知有多少人看过，我很奇怪怎么到现在还被煞有介事地用于考试。
本书第3版即将出版，我希望读者正确地使用本书：请您学习和应用您（或公司）认为好的东西，不要把本书当做标准来看待，不要全部照搬，也不必花费很多时间去争议本书是好还是坏。
第2版　　《高质量程序设计指南—C++/C语言》第1版上市后，一度成为畅销书。
网上评论甚多，褒贬参半。
我们分析了读者的批评和建议，总结了本书第1版的主要不足：　　由于第1版原本是企业的培训教材，初衷是为了帮助程序员提高程序质量，假设读者已经熟悉C++/C语法，所以内容薄而精练、前后章节不连贯，看起来更像专题讲座。
出版社的宣传工作做得很好，本书吸引了很多C++初学者和高级程序员。
由于书中不讲解入门知识，导致很多初学者看不下去。
有一些大学生听了我的讲座后，为表敬意特地买书让我签名，翻阅之后就塞进书架当做纪念品了。
对于那些高级程序员而言，本书的大部分内容他们早已经熟悉，好不容易看到几处精彩的章节，却翻了几页就没有了，真是不过瘾啊。
我的研究专长是软件工程和企业研发管理，而非程序设计。
在C++/C编程方面自己仅仅是一名老工匠而已，我的确没有时间没有能力写出让初学者和高级程序员都喜欢的C++/C书籍。
这本书炒作得过火，让我骑虎难下了。
从2002年11月起，我就开始物色真正的C++/C高手来写本书的第2版。
恰好上海大唐移动通信设备有限公司的韩永泉正在为公司写C++/C培训教材，他也是本书第1版的读者。
韩永泉提出了很有价值的建议和批评，并把他写的培训教材发给我看，真是自投罗网啊！
韩永泉是在西安电子科技大学计算机系读的本科和硕士，内功扎实。
我和他碰头交谈了2小时，就把第2版托付给他了。
两个月后，他把第2版的书稿交给我审阅。
第2版的内容比第1版多了一倍，其广度符合我的设想，其深度完全出乎我的意料。

为了阐述清楚C++/C程序之中的许多“为什么”，本书给出了大量的“提示、建议、规则”，并从编译器实现的角度论述原理。
这种深度非一般教科书能比，我用了一个月时间才审阅并且学习完毕，删除了几十页过分深奥的内容（免得让我自己看昏倒）。
我相信第2版可以让大多数高级程序员看过瘾了。
网上有一些忌世愤俗者认为计算机领域的每个分支都已经有了世界名著，不具有世界顶尖水平的中国人再写类似书籍都是欺世盗名的行为。
这种极度自负和极度自卑的心态导致他们专爱骂国内作者。
如果中国作者的书籍中的技术错误被他们抓住，经过放大、推理、演绎之后基本上就能断定作者是卑鄙之徒，于是砖头就拍过来了（简称“拍砖”）。
拍砖者们遥相呼应，很快就能拍出江湖豪情，被拍的作者就成了倒霉蛋。
有位好心的读者怕我经受不起，特意发给我一本拍砖大法——《拍砖十二流》以增强内功。
网上自由漫骂既是网络价值的体现又是民主的体现，这是物质文明和精神文明发展到一定境界的产物。
《高质量程序设计指南——C++/C语言》第2版即将出版，作者忐忑不安地等待第二轮“拍砖”。
第1版　　软件质量是被许多程序员挂在嘴上而不是放在心上的东西！
除了完全外行和真正的编程高手外，初读本书，你最先的感受将是惊慌：“哇！
我以前捏造的C++/C程序怎么会有那么多的毛病？
”　　别难过，作者只不过比你早几年、多几次惊慌而已。
请花几小时认真阅读这本经书，你将会获益匪浅，这是前面N-1个读者的建议。
编程老手与高手的误区　　自从计算机问世以来，程序设计就成了令人羡慕的职业，程序员在受人宠爱之后容易发展成为毛病特多却常能自我臭美的群体。
如今在Internet上流传的“真正”的程序员据说是这样的：　　（1）真正的程序员没有进度表，只有讨好领导的马屁精才有进度表，真正的程序员会让领导提心吊胆。
（2）真正的程序员不写使用说明书，用户应当自己去猜想程序的功能。
（3）真正的程序员几乎不写代码的注释，如果注释很难写，它理所当然也很难读。
（4）真正的程序员不画流程图，原始人和文盲才会干这事。
（5）真正的程序员不看参考手册，新手和胆小鬼才会看。
（6）真正的程序员不写文档也不需要文档，只有看不懂程序的笨蛋才用文档。
（7）真正的程序员认为自己比用户更明白用户需要什么。
（8）真正的程序员不接受团队开发的理念，除非他自己是头头。
（9）真正的程序员的程序不会第一次就正确运行，但是他们愿意守着机器进行若干个30小时的调试改错。
（10）真正的程序员不会在上午9：00到下午5：00之间工作，如果你看到他在上午9：00工作，这表明他从昨晚一直干到现在。
……　　具备上述特征越多，越显示程序员水平高，资格老。
所以别奇怪，程序员的很多缺点竟然可以被当做优点来欣赏。
就像在武侠小说中，那些独来独往、不受约束且带点邪气的高手最令人崇拜一样。
我曾经也这样信奉，并且希望自己成为那样的“真正”的程序员，结果没有得到好下场。
我从读大学到博士毕业10年来一直勤奋好学，累计编写了数十万行C++/C代码。
有这样的苦劳和疲劳，我应该称得上是编程老手了吧？
我开发的软件都与科研相关（集成电路CAD和3D图形学领域），动辄数万行程序，技术复杂，难度颇高。
这些软件频频获奖，有一个软件获得首届中国大学生电脑大赛软件展示一等奖。
在1995年开发的一套图形软件库到2000年还有人买。
罗列出这些“业绩”，可以说明我算得上是编程高手了吧？
可惜这种个人感觉不等于事实。

读博期间我曾用一年时间开发了一个近10万行C++代码的3D图形软件产品，我内心得意表面谦虚地向一位真正的软件高手请教。
他虽然从未涉足过3D图形领域，却在几十分钟内指出该软件多处重大设计错误。
让人感觉那套软件是用纸糊的华丽衣服，扯一下掉一块，戳一下破个洞。
我目瞪口呆地意识到这套软件毫无实用价值，一年的心血白花了，并且害死了自己的软件公司。
人的顿悟通常发生在最心痛的时刻，在沮丧和心痛之后，我作了深刻反省，“面壁”半年，重新温习软件设计的基础知识。
补修“内功”之后，又觉得腰板硬了起来。
博士毕业前半年，我曾到微软中国研究院找工作，接受微软公司一位资深软件工程师的面试。
他让我写函数strcpy的代码。
太容易了吧？
错！
这么一个小不点儿的函数，他从三个方面考查：（1）编程风格；（2）出错处理；（3）算法复杂度分析（用于提高性能）。
在大学里从来没有人如此严格地考查过我的程序。
我花了半小时，修改了数次，他还不尽满意，让我回家好好琢磨。
我精神抖擞地进“考场”，大汗淋漓地出“考场”。
这“高手”当得也太窝囊了。
我又好好地反省了一次。
我把反省后的心得体会写成文章放在网上，引起了不少软件开发人员的共鸣。
我因此有幸和国内大型IT企业如华为、上海贝尔、中兴等公司的同行们广泛交流。
大家认为提高质量与生产率是软件工程要解决的核心问题。
高质量程序设计是非常重要的环节，毕竟软件是靠编程来实现的。
我们心目中的老手们和高手们能否编写出高质量的程序来？
不见得都能！
就我的经历与阅历来看，国内大学的计算机教育根本就没有灌输高质量程序设计的观念，教师们和学生们也很少自觉关心软件的质量。
勤奋好学的程序员长期在低质量的程序堆中滚爬，吃尽苦头之后才有一些心得体会，长进极慢，我就是一例。
现在国内IT企业拥有学士、硕士、博士文凭的软件开发人员比比皆是，但他们在接受大学教育时就“先天不足”，岂能一到企业就突然实现质的飞跃。
试问有多少软件开发人员对正确性、健壮性、可靠性、性能、易用性、清晰性、可扩展性、安全性、兼容性、可移植性等质量属性了如指掌？
并且能在实践中运用自如？
“高质量”可不是干活小心点就能实现的！
我们有充分的理由疑虑：（1）编程老手可能会长期用隐含错误的方式编程，习惯成自然后，被人指出发现毛病时都不愿相信那是真的！
（2）编程高手可以在某一领域写出极有水平的代码，但未必能从全局把握软件质量的方方面面。
事实证明如此。
我到上海贝尔工作后，陆续面试或测试过近百名“新”、“老”程序员的编程技能，合格率低于50%。
很少有人能够写出完全符合质量要求的if语句，很多程序员对指针、内存管理一知半解……
领导们不敢相信这是真的。
我做过现场试验：有一次部门新进14名硕士生，在开欢迎会之前对他们进行“C++/C编程技能”摸底考试。
我问大家试题难不难？

所有的人都回答不难。
结果没有一个人及格，有半数人得零分。
竞争对手如华为、中兴、大唐等公司的朋友们也做过试验，也是类似的结果。
真的不是我“心狠手辣”或者要求过高（甚至变态），而是很多软件开发人员对自己的要求不够高。
要知道这些大公司的员工素质在国内IT企业中是比较位列前茅前列的，倘若他们的编程质量都如此差的话，我们怎么敢期望中小公司拿出高质量的软件呢？
连程序都编不好，还谈什么振兴民族软件产业。
多年来，我在软件开发过程中的苦头吃得实在太多了，现在总算被折磨清醒了。
我打算定义编程老手和编程高手，请您别见笑。
定义1：能长期稳定地编写出高质量程序的程序员称为编程老手。
定义2：能长期稳定地编写出高难度、高质量程序的程序员称为编程高手。
根据上述定义，马上得到第一推论：我既不是高手也算不上是老手。
在写此书前，我阅读了不少程序设计方面的英文著作，越看越羞惭。
因为发现自己在编程基本技能方面都未能全面掌握，顶多算是二流水平，还好意思谈什么老手和高手。
希望和我一样在国内土生土长的程序员朋友们能够做到：（1）知错就改；（2）经常温故而知新；（3）坚持学习，天天向上。
林锐　2002年4月　上海贝尔阿尔卡特股份有限公司

<<高质量程序设计指南>>

内容概要

高质量程序设计是软件行业的薄弱环节，大部分企业只能大量的测试和改错来提高软件产品的质量，为此付出了高昂的代价。
因此，如何让程序员熟练地掌握编程技术和编程规范，在开发过程中内建高质量代码，是IT企业面临的主要挑战之一。

本书以轻松幽默的笔调向读者论述了高质量软件开发方法与C++／C编程规范。
它是作者多年从事软件开发工作的经验总结。
本书共17章，第1章到第4章重点介绍软件质量和基本的程序设计方法；第5章到第16章重点阐述C++／C编程风格、面向对象程序设计方法和一些技术专题；第17章阐述STL的原理和使用方法。

本书第1版和第2版部分章节曾经在Imemet上广泛流传，被国内IT企业的不少软件开发人员采用本书的附录C《大学十年》是作者在网上发表的一个短篇传记，文中所描述的充满激情的学习和生活态度，感染了大批莘莘学子。

本书的主要读者对象是IT企业的程序员和项目经理，以及大专院校的本科生和研究生。

作者简介

林锐，1973年生。
1990年至1996年，就读于西安电子科技大学，获硕士学位。
1997年至2000年，就读于浙江大学计算机系，获博士学位。
大学期间两度被评为中国百名跨世纪优秀大学生，1996年获电子工业部科技进步二等奖，1997年获首届中国大学生电脑大赛软件展示一等奖。
2000年7月加入上海贝尔有限公司，从事软件工程和CMM的研究推广工作，2003年7月当选为Alcatel集团软件工程专家。
2004年初创建上海漫索计算机科技有限公司(http：／／www．chinaspis．com)，致力于创作适合国内企业需求的软件研发管理解决方案，包括方法论和软件产品。
工作期间出版著作六部。

<<高质量程序设计指南>>

书籍目录

第1章 高质量软件开发之道 11.1 软件质量基本概念 11.1.1 如何理解软件的质量 11.1.2 提高软件质量的基本方法 31.1.3 “零缺陷”理念 41.2 细说软件质量属性 41.2.1 正确性 41.2.2 健壮性 51.2.3 靠性 51.2.4 性能 61.2.5 易用性 71.2.6 清晰性 71.2.7 安全性 71.2.8 可扩展性 81.2.9 兼容性 可移植性 81.3 人们关注的不仅仅是质量 91.3.1 质量、生产率和成本之间的关系 91.3.2 软件过程改进的基本概念 111.4 高质量软件开发的基本方法 131.4.1 建立软件过程规范 131.4.2 复用 151.4.3 分而治之 161.4.4 优化与折中 171.4.5 技术评审 181.4.6 测试 191.4.7 质量保证 211.4.8 改错 221.5 件开发的一些常识和思考 241.5.1 有最好的编程语言吗 241.5.2 编程是一门艺术吗 241.5.3 编程时应该多使用技巧吗 241.5.4 换更快的计算机还是换更快的算法 251.5.5 错误是否应该分等级 251.5.6 一些错误的观念 251.6 小结 26第2章 编程语言发展简史 272.1 编程语言大事记 272.2 Ada的故事 302.3 C/C++发展简史 312.4 Borland与Microsoft之争 322.5 Java阵营与Microsoft的较量 332.6 小结 36第3 程序的基本概念 373.1 程序设计语言 373.2 语言实现 383.3 程序库 403.4 开发环境 403.5 程序工作原理 413.6 良好的编程习惯 42第4章 C++/C程序设计入门 454.1 C++/C程序的基本概念 454.1.1 启动函数main() 454.1.2 命令行参数 474.1.3 内部名称 484.1.4 连接规范 494.1.5 变量及其初始化 514.1.6 C Runtime Library 524.1.7 编译时和运行时的不同 524.1.8 编译单元和独立编译技术 544.2 基本数据类型和内存映像 544.3 类型转换 564.3.1 隐式转换 564.3.2 强制转换 584.4 标识符 604.5 义序列 614.6 运算符 624.7 表达式 634.8 基本控制结构 654.9 选择（判断）结构 654.9.1 布尔变零值比较 664.9.2 整型变量与零值比较 674.9.3 浮点变量与零值比较 674.9.4 指针变量与零值比较 694.9.5 对if语句的补充说明 704.9.6 switch结构 704.10 循环（重复）结构 714.10.1 for语句的循控制变量 724.10.2 循环语句的效率 734.11 结构化程序设计原理 784.12 goto/continue/break语句 794. 示例 80第5章 C++/C常量 855.1 认识常量 855.1.1 字面常量 855.1.2 符号常量 865.1.3 契约性量 875.1.4 枚举常量 875.2 正确定义符号常量 875.3 const与#define的比较 885.4 类中的常量 895.5 际应用中如何定义常量 90第6章 C++/C函数设计基础 956.1 认识函数 956.2 函数原型和定义 966.3 函数调用方式 976.4 认识函数堆栈 996.5 函数调用规范 1006.6 函数连接规范 1016.7 参数传递规则 1026.8 返回值的规则 1046.9 函数内部实现的规则 1076.10 存储类型及作用域规则 1096.10.1 存类型 1096.10.2 作用域规则 1106.10.3 连接类型 1116.11 递归函数 1136.12 使用断言 1166.13 使用const提高函数的健壮性 1186.13.1 用const修饰函数的参数 1186.13.2 用const修饰函数的返回值 119 第7章 C++/C指针、数组和字符串 1217.1 指针 1217.1.1 指针的本质 1217.1.2 指针的类型及其支持的运算 1237.1.3 指针传递 1257.2 数组 1267.2.1 数组的本质 1267.2.2 二维数组 1287.2.3 数组传递 1297.2.4 动态创建、初始化和删除数组的方法 1317.3 字符数组、字符指针和字符串 1337.3.1 字符数组、字符串和\0的关系 1337.3.2 字符指针的误区 1347.3.3 字符串拷贝和比较 1347.4 函数指针 135 引用和指针的比较 137第8章 C++/C高级数据类型 1418.1 结构（Struct） 1418.1.1 关键字struct与class的困惑 1418.1.2 使用struct 1428.1.3 位域 1458.1.4 成员对齐 1478.2 联合（Union） 1598.3 （Enum） 1618.4 文件 163第9章 C++/C编译预处理 1659.1 文件包含 1659.1.1 内部包含卫哨和外部包含卫哨 1659.1.2 头文件包含的合理顺序 1669.2 宏定义 1669.3 条件编译 1699.3.1 #if、#elif和#else 1699.3.2 #ifdef 和 #ifndef 1709.4 #error 1719.5 #pragma 1719.6 #和##运算符 1719.7 预定义常量 172第10章 C++/C文件结构和程序版式 17510.1 程序文件的目录结构 17510.2 文件的结构 17610.2.1 头文件的用途和结构 17610.2.2 版权和版本信息 17710.2.3 源文件结构 17810.3 代码的式 17810.3.1 适当的空行 17810.3.2 代码行及行内空格 17910.3.3 长行拆分 18010.3.4 对齐与缩进 18110.3.5 修饰符的位置 18210.3.6 注释风格 18210.3.7 ADT/UDT版式 183第11章 C++/C应用程序命名规则 18511.1 共性规则 18511.2 简单的Windows应用程序命名 186第12章 C++面向对象程序设计方法概述 18912.1 漫谈面向对象 18912.2 对象的概念 19012.3 信息隐藏与类的封装 19112.4 类的继特性 19512.5 类的组合特性 20012.6 动态特性 20112.6.1 虚函数 20212.6.2 抽象基类 20212.6.3 动定 20512.6.4 运行时多态 20712.6.5 多态数组 20812.7 C++对象模型 21512.7.1 对象的内存映像 21512.7.2 隐含成员 22412.7.3 C++编译器如何处理成员函数 22512.7.4 C++编译器如何处理静态成员 22512.8 小结 226第13章 对象的初始化、拷贝和析构 22913.1 构造函数与析构函数的起源 22913.

<<高质量程序设计指南>>

为什么需要构造函数和析构函数 23013.3 构造函数的成员初始化列表 23213.4 对象的构造和析构次序 23413.5 构造函数和析构函数的调用时机 23513.6 构造函数和赋值函数的重载 23613.7 示例：类String的构造函数和析构函数 23813.8 何时应该定义拷贝构造函数和拷贝赋值函数 23913.9 示例：类String的拷贝构造函数和拷贝赋值函数 24013.10 用偷懒的办法处理拷贝构造函数和拷贝赋值函数 24213.11 如何实现派生类的基本函数 243第14章 C++函数的高级特性 24714.1 函数重载的概念 24714.1.1 重载的起源 24714.1.2 重载是如何实现的 24714.1.3 当心隐式类型转换导致重载函数产生二义性 24914.2 成员函数的重载、覆盖与隐藏 25014.2.1 重载与覆盖 25014.2.2 令人迷惑的隐藏规则 25114.2.3 摆脱隐藏 25314.3 参数的默认值 25414.4 运算符重载 25514.4.1 基本概念 25514.4.2 符重载的特殊性 25614.4.3 不能重载的运算符 25714.4.4 重载++和—— 25714.5 函数内联 25914.5.1 函数内联取代宏 25914.5.2 内联函数的编程风格 26014.5.3 慎用内联 26114.6 类型转换函数 26114.7 const成员函数 264第15章 C++异常处理和RTTI 26715.1 为什么要使用异常处理 26715.2 C++异常处理 26815.2.1 异常处理的原理 26815.2.2 异常类型和异常对象 26915.2.3 异常处理的语法结构 27015.2 异常的类型匹配规则 27215.2.5 异常说明及其冲突 27215.2.6 当异常抛出时局部对象如何释放 27315.2.7 对象构造和析构期间的异常 27315.2.8 如何使用好异常处理技术 27515.2.9 C++的标准异常 27815.3 虚函数面临的难题 27815.4 RTTI及其构成 28015.4.1 起源 28015.4.2 typeid运算符 2811 dynamic_cast运算符 28315.4.4 RTTI的魅力与代价 285第16章 内存管理 28716.1 内存分配方式 28716.2 常见的内存错误及其对策 28816.3 指针参数是如何传递内存的 28916.4 free和delete把指针怎么啦 29116.5 动态内存会被自动释放吗 29216.6 杜绝“野指针” 29216.7 有了malloc/free为什么还要new/delete 29316.8 malloc/free的使用要点 29516.9 new有3种使用方式 29616.9.1 plain new/delete 29616.9.2 nothrow new/delete 29716.9.3 placement new/delete 29716.10 new/delete的使用要点 30016.11 内存耗尽怎么办 30116.12 用对象模拟指针 30216.13 泛型指针auto_ptr 30516.14 带有引计数的智能指针 30616.15 智能指针作为容器元素 310第17章 学习和使用STL 32317.1 STL简介 3231 STL头文件的分布 32417.2.1 容器类 32417.2.2 泛型算法 32517.2.3 迭代器 32517.2.4 数学运算库 32517.2.5 通用工具 32517.2.6 其他头文件 32617.3 容器设计原理 32617.3.1 内存映像 32617.3.2 储方式和访问方式 32717.3.3 顺序容器和关联式容器的比较 32817.3.4 如何遍历容器 33117.3.5 存储空间重分配问题 33217.3.6 什么样的对象才能作为STL容器的元素 33317.4 迭代器 33417.4.1 迭代器的本质 33417.4.2 迭代器失效及其危险性 33817.5 存储分配器 34617.6 适配器 34717.7 泛型算法 35017 一些特殊的容器 35417.8.1 string类 35417.8.2 bitset并非set 35517.8.3 节省存储空间的vector 35717.8.4 空容器 35817.9 STL容器特征总结 36017.10 STL使用心得 362附录A C++/C试题 365附录B C++/C试题答案与评分标准 369附录C 大学十年 375附录D 《大学十年》后记 393附录E 术语与缩写解释 39 参考文献 397

<<高质量程序设计指南>>

版权说明