程序设计缺陷分析与实践

BUG DETECTION

◎尹　浩　于秀山　编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

程序设计缺陷分析与实践

尹 浩 于秀山 编著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内 容 简 介

代码缺陷源自程序设计，本书结合作者多年软件测试经验，重点归纳总结了 C/C++和 Java 语言在程序设计方面存在的鲜为人知的各种缺陷，以期为软件设计人员和测试人员提供有益借鉴。

全书共 5 章 2 个附录，分别介绍了程序设计缺陷静态分析方法、C/C++语言程序设计缺陷分析、Java 语言程序设计缺陷分析、软件质量静态度量以及静态测试工具使用实践。重点介绍了 C/C++语言程序在编码风格、内存管理、缓冲区使用、指针以及安全等方面存在的典型缺陷，并结合实例对每种缺陷进行了分析，同时给出了缺陷修改方法。

本书既是一本程序设计方面的高级教程，同时也是一本软件静态测试方面的教程，可作为高等院校计算机相关专业高级程序设计及软件测试课程教材，也可供软件开发工程师、测试工程师、测试经理等人员参阅。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目（CIP）数据

程序设计缺陷分析与实践 / 尹浩，于秀山编著 —北京：电子工业出版社，2011.3
ISBN 978-7-121-12969-8

Ⅰ.①程… Ⅱ.①尹… ②于… Ⅲ.①程序设计 Ⅳ.①TP311.1

中国版本图书馆 CIP 数据核字（2011）第 026335 号

责任编辑：竺南直　　特约编辑：郭　莉
印　　刷：
装　　订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：720×1 000　1/16　印张：17.25　字数：325 千字
印　　次：2011 年 3 月第 1 次印刷
印　　数：4 000 册　　定价：36.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

言及程序设计，很多读者并不陌生，并自认为是程序设计方面的高手，但当您读完此书后，除了完全外行和真正的编程高手外，您一定会感到惊讶：原来我设计的程序包含如此多的缺陷！

程序缺陷的存在使软件经常出现这样或那样的问题，影响了软件的正常使用，甚至危及人类生命和财产安全。自从软件危机爆发以来，广大软件从业人员就开始苦苦寻求如何提高软件质量的灵丹妙药，开出了许多有效的良方，然而，由于缺陷的多样性和隐蔽性，许多看似正常的程序其实隐藏着巨大的隐患，如何找出并避免这些缺陷是软件开发人员和测试人员面临的一个严峻挑战。

本书从程序设计缺陷的角度入手，结合作者多年从事软件测试的经验，将测试中发现的鲜为人知的软件问题进行了归纳整理，以期对读者能有所裨益。

本书共分 5 章，各章主要内容如下。

第 1 章从静态测试的角度，介绍了程序设计缺陷静态分析常用的文档审查、代码审查、技术评审以及代码走查方法，并对部分方法进行了比较；

第 2 章和第 3 章是本书的重点内容，详细阐述了 C/C++、Java 语言程序在编码风格、内存管理、缓冲区使用、指针、代码安全等方面存在的典型缺陷。通过实例，对每一种缺陷进行了分析，并给出了修改方法；

第 4 章介绍了软件质量静态度量方面的知识，给出了 5 种软件质量模型；详细阐述了 Halstead 度量、McCabe 度量等软件质量度量方法；

第 5 章结合测试实践，介绍了 Polyspace、Klockwork、Testbed、McCabe 代码静态分析工具的功能及使用方法。

前车之鉴，后事之师，无论是软件测试人员还是程序设计人员，只要您花点时间认真阅读本书，将会受益匪浅。

在本书的编写过程中，张旭辉、杨豹、李华莹、胡兢玉、董昕、于长钺同志参与了部分章节编写工作，在此向他们表示衷心的感谢。

鉴于作者才疏学浅，书中难免有遗漏和错误，敬请读者斧正。

作　者

2011 年春节于北京

目　录

此为试读，需要完整PDF请访问：www.ertongbook.com

第 1 章　程序设计缺陷静态分析

言及程序设计，大部分人并不陌生，也出现了很多程序设计高手，但软件产品，甚至是微软公司开发的操作系统软件，为何频繁出现问题呢？程序设计存在缺陷是一个根本原因！许多看似正确的程序，其实其中隐藏着缺陷，有很多缺陷通常情况下难以发现，一旦触发这些缺陷，软件运行将出现问题。消除程序缺陷是软件从业人员追求的目标之一，本章将从静态分析的角度，阐述常用的程序缺陷静态分析方法。

1.1　软件测试分类

随着信息技术的发展，软件所涉及的应用领域越来越广泛，因软件质量问题导致的人身伤害和财产损失事件屡有发生，软件质量越来越受到人们的关注，作为软件质量保证手段之一的软件测试，国内外学者陆续开展了相关课题研究，提出了一些有效的软件测试技术，对于这些技术，可以从不同的角度加以分类：

- 从是否需要执行被测软件的角度，可分为静态测试和动态测试；
- 从测试是否针对软件的内部结构和具体实现算法的角度，可分为白盒测试和黑盒测试；
- 从测试执行时是否需要人工干预的角度，可分为人工测试和自动测试。

其中，每一种技术还可以继续进行分类，如图 1.1 所示。

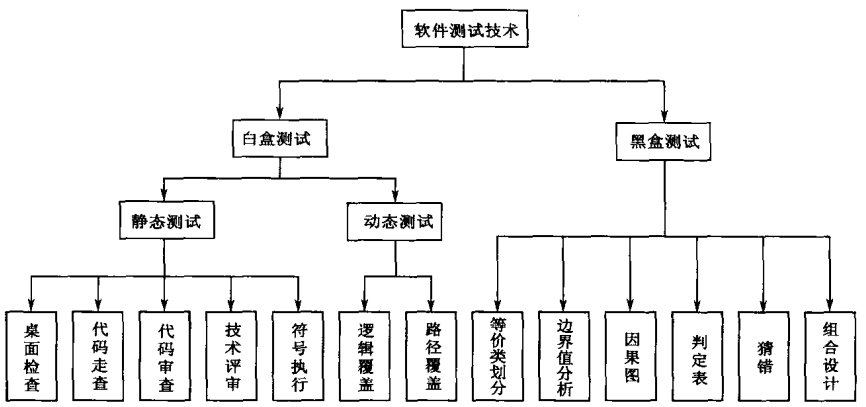

图 1.1　软件测试技术分类

需要指出的是，上述分类方法只是从测试的不同角度来划分的，对于同一个测试，既有可能是黑盒测试，也有可能是动态测试；既有可能是静态测试，也有可能是白盒测试。

黑盒测试有可能是动态测试（如功能测试），也有可能是静态测试（不运行程序，只检查软件的界面）；白盒测试有可能是动态测试（运行程序，分析代码），也有可能是静态测试（如代码审查和走查），因此，又可以派生出静态黑盒测试、动态黑盒测试、动态白盒测试、静态白盒测试等复合测试技术。

1.2　静态分析方法

静态分析（Static Analysis）方法是指通过对软件产品进行静态而非动态的分析，检查软件产品是否满足规定要求的方法。广义上的静态分析对象包括软件开发过程中产生的各种软件产品，如软件需求规格说明、软件概要设计文档、测试文档、源程序代码、评审报告等。静态分析技术包括多种形式，软件评审是其中的一种重要方式。软件评审通常包括管理评审、技术评审、审查、走查和审核。通常情况下，将对软件需求规格说明等文档的分析称为文档审查，而将对源程序代码的审查称为代码审查。

代码静态分析的对象是源程序代码，代码静态分析不需要实际执行程序，而是通过扫描源程序，从中找出违反编程规则的错误、内存管理、指针使用、以及程序结构异常、控制流异常、数据流异常等错误。

代码静态分析有人工和自动两种实现方式。人工方式主要通过测试人员或相关人员阅读源程序代码完成，代码审查和走查是两种常用的人工代码分析方法；自动方式主要借助于静态分析工具在宿主机上完成，不需要下载到目标板，不需要硬件支持。自动静态分析完成可重复、可归纳的分析活动，对于那些无法归纳和无规律的分析工作，需要人工进行分析。

静态代码自动分析原理看似与编译器类似，但它与编译器有着很大的区别。编译器只能检查程序中存在的部分词法和语法方面的错误，此外，编译器的主要目的是把源代码高效地翻译和转换成可执行的二进制代码，而不是通过代码检查来帮助你编写更干净、更健壮的代码；静态分析的主要目的是准确找到程序结构的错误和可疑代码位置，因此，要尽可能详细地记录源程序的特性，以便发现程序中存在的异常现象。

自动分析具有以下特点：

- 自动分析的综合效率一般比人工分析效率高（优秀程序分析员除外）；
- 自动分析的侧重点是可重复、可归纳的语法和语义方面的缺陷，人工分析的侧重点是高层的、面向语义的代码缺陷；
- 自动分析的平均成本通常低于人工分析的成本，因此，要尽可能多地采用自动分析方法；
- 自动分析和人工分析所发现的缺陷种类不同，两者具有良好的互补性。

1.2.1　文档审查

文档审查是对软件开发过程中产生的文档的齐套性、完整性、规范性、一致性和准确性所进行的检查。

在软件开发过程中，相关部门对文档编写颁布了相应的标准和规范，如国标、行业标准，这些标准是文档审查的依据。为了提高文档审查质量和效率，要根据这些标准制定文档审查单，根据文档审查单，对文档逐一进行审查，常用的文档审查单见附录。

1.2.2　代码审查

在 GBT11457 中，对审查做了如下定义：

审查：一种正式的评定技术。由除作者之外的某人或一小组仔细检查软件需求、设计或代码，以找出故障、违反开发标准之处和其他一些问题。

代码审查是针对代码所进行的审查，代码审查的概念是由 IBM 质量保证专家 Michael Fagan 于 1976 年提出的，AT&T 贝尔实验室从 1977 年也开始使用此审查方法。1985 年，IBM 的 Carolel、Jones 和 Robert Mays 及 Fangan 本人对审查方法做了改进，提出了通用的审查周期策略、临时分析会议、行为数据库、行为小组等概念和方法，把最初的针对错误检查扩展为针对错误预防。

代码审查方法与一般的审查方法相同，只是审查的对象不同，下面具体介绍代码审查方法。

（1）人员组成

代码审查组通常由 2 至 6 名成员组成，分别担任审查组长、讲解员、记

录员、程序员、审查员角色。所有参评人员都是审查员，程序员既不能兼任组长，也不能作为讲解员或记录员，其他角色可由审查组成员担当，个别成员可以担当一个以上的角色。为了保证公正性，审查组成员的管理者不应参加审查工作。

（2）各类人员职责

审查组长：审查组长由经过审查技术培训且公正的专业人员担任，负责制定审查计划，组织预审和审查会议。审查组长要确保审查会议有序进行并实现预定的目标，同时，还要收集审查数据，发布审查报告。

讲解员：讲解代码，并对重要的部分予以强调，引导审查组以全面、合乎逻辑的方式进行审查。

记录员：负责记录审查员发现的代码缺陷、提出的建议、应采取的措施以及决策等，此外，还负责记录用于过程分析的审查数据，记录员可由审查组长兼任。

程序员：负责提供满足进入审查条件的代码、解答问题、修改代码以保证代码满足审查出口条件。

审查员：确定并描述审查过程中发现的代码缺陷。应根据专业领域选择审查员，并兼顾其代表性（如委托方、用户、需求、设计、编码、测试项目管理、质量管理）。审查员应只提出与代码审查有关的观点。

为保证审查的充分性，应指定某些审查员负责特定的专题，例如，一个审查员可以关注于与某个或某些标准的符合性审查，另一个审查员关注整体相关性。

（3）进入条件

- 被审查的代码已完成并满足项目标准；
- 需要的支撑文档已准备完毕，包括审查程序、审查报告格式、代码缺陷单、审查单、相关的标准和规范等。

（4）审查流程

审查流程如图 1.2 所示。

其中，审查报告主要包含下列内容：

- 被审查的项目名称；
- 审查组成员；
- 审查会持续时间；

- 审查的代码；
- 审查的材料数量（如文本页数）；
- 审查的具体输入；
- 审查的目标以及这些目标是否达到；
- 包含缺陷位置、描述以及分类的缺陷表；
- 代码处置；
- 单个成员和审查组总的准备时间；
- 总的返工时间；
- 缺陷摘要（列出每类缺陷的数量）；
- 对返工工作量和返工完成时间的估计；
- 缺陷修改节省的成本估计（与后期修改比较）。

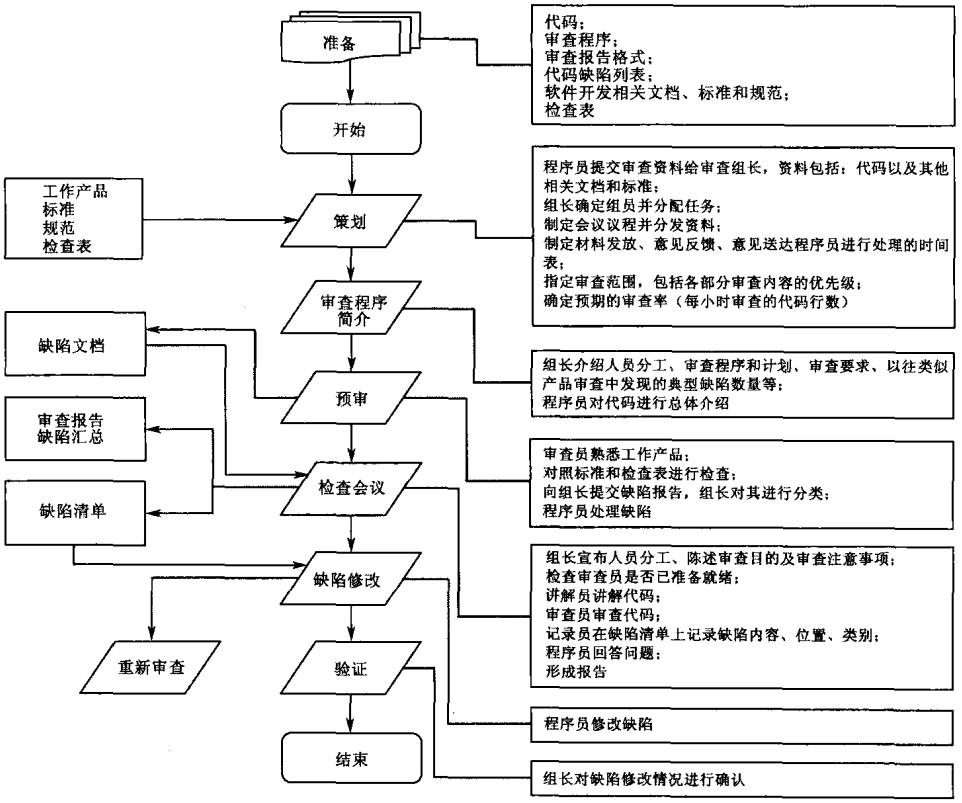

图 1.2　代码审查流程

此为试读，需要完整PDF请访问：www.ertongbook.com

（5）审查注意事项

① 不要流于形式。要挑选业界有经验的人员担当审查员，提供的审查表要可操作并有针对性，检查表中不但要规定“做什么”，而且要规定“如何做”。

② 不要偏离审查会主题。审查会重在发现问题，而非解决问题，不要把审查会变成技术攻关会。

③ 审查时对事不对人。审查时只针对审查对象发表意见，不要涉及具体的人，不要把审查和评价混淆，避免用审查结果评价个人能力。

④ 掌握好审查进度。审查会议一般应限制在 2 小时以内，每小时审查的代码量应在 100～200 行。

⑤ 注意收集、分析审查数据。在审查过程中，要注意收集审查数据，如代码规模、发现的问题数量、解决的问题数量、缺陷清除率、审查工作效率等，通过分析这些数据，为改进审查过程提供依据。

1.2.3 技术评审

技术评审是指“一组合格人员对软件产品进行的系统性评价，以检查软件产品是否满足规定的要求，是否符合相关的规范和标准”。软件产品主要包括：软件需求规格说明、软件设计文档、软件测试文档、软件用户文档、维护手册、系统构建程序、安装程序、软件发布通告等。

技术评审中的角色有：决策者、组长、记录员、技术人员、管理人员（可选）、项目组其他成员（可选）、顾客或用户代表（可选）。

技术评审流程与代码审查流程大致相同，在此不再赘述。

与技术评审相比，代码审查更正式，角色更多。代码审查必须按照项目计划实施，不能随意进行。代码审查是一个特定的测试活动，而不是一个评估软件产品适用性的活动。

1.2.4 代码走查

代码走查是一种静态分析技术或评审过程，在此过程中，设计者或程序员引导开发组成员通读已书写的设计或编码，其他成员负责提出问题并对有关技术、风格、可能的错误、是否违背开发标准等方面进行评论。

在代码走查过程中，走查组成员（通常是设计者或程序员）事先提供若干测试用例，参会人员充当计算机，程序员引导参会人员针对每个测试用例用头脑来模拟程序执行过程，并在纸上或白板上监视程序状态（变量的值），这是代码走查与代码审查最大的不同之处。

代码走查有两个目的，第一个目的是评价软件产品，包括发现缺陷、改进软件产品、提供另外的实现方案、评估软件产品与标准和规范的一致性；第二个目的是技术人员的交流和培训。通过代码走查，可以使组内成员在编程风格和产品细节、走查的目的等方面达成一致。

代码走查的另一个优点是能够准确地定位错误，从而降低调试的成本，另外，代码走查通常能够发现批量错误。

（1）人员组成

代码走查至少由两人组成（包括程序员），分别担任组长、记录员、程序员以及成员角色。每个成员可担任多个角色，组长或程序员可以作为记录员，组长也可由程序员担任。为了保证公正性，走查组成员的管理者不得参加走查。

（2）各类人员职责

组长：对走查进行指导，处理与走查有关的管理工作，如分发文档、安排会议等，制定走查目标，发布走查报告。走查组长要确保走查以有序方式进行、确保为每一个讨论项作出决议或明确应采取的措施。

记录员：负责记录走查会议期间作出的决议或确定的整改措施。此外，记录员还应记录走查期间对发现的缺陷、风格方面的疑问、遗漏、矛盾、改进建议或备选方法等方面的意见。

程序员：负责介绍代码。

成员：履行所分配角色的职责。每一个走查组成员要为走查做好充分准备并积极参加走查，识别并描述发现的软件缺陷。

代码走查最佳的人选如下：极富经验的程序员、程序设计语言专家、新程序员、程序维护人员、其他不同项目人员、该软件编程小组的程序员。

（3）进入条件

- 代码已完成并满足项目标准；
- 需要的支撑文档已准备完毕，包括缺陷分类、走查单、相关的标准和规范等。

（4）走查流程

走查流程如图 1.3 所示。

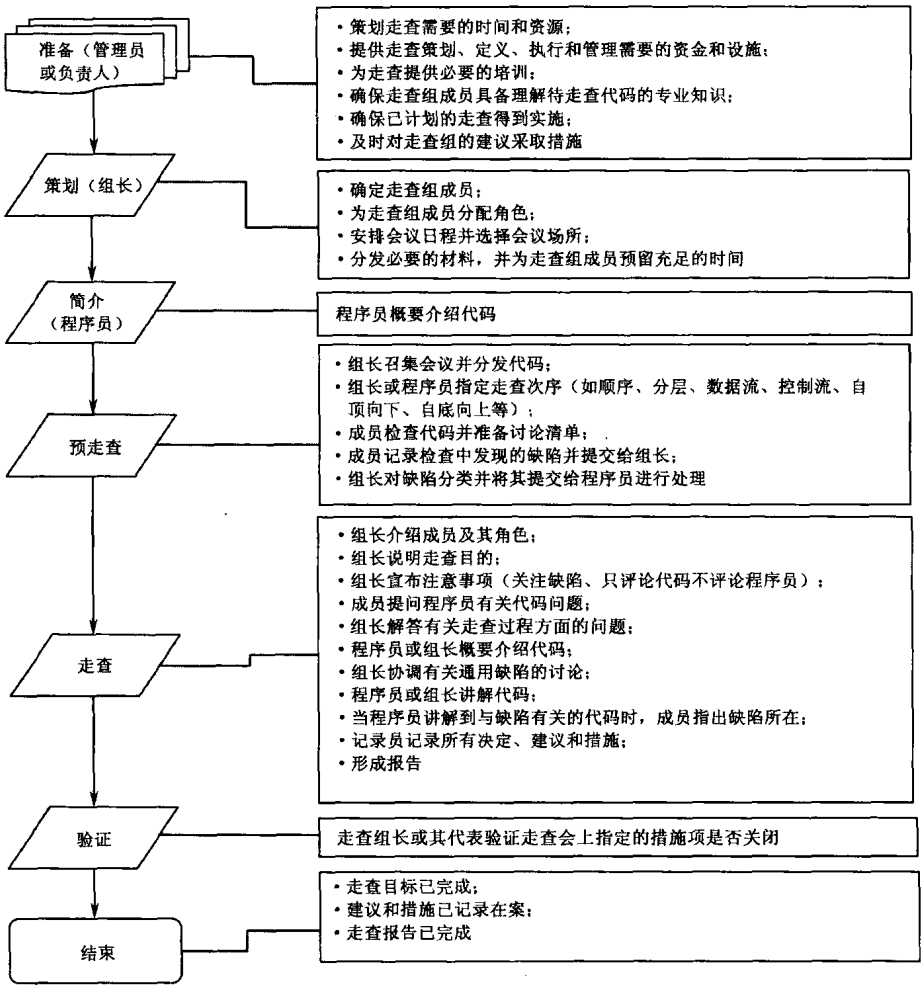

图 1.3　代码走查流程

其中，走查报告主要包含下列内容：

- 被走查的项目名称；
- 走查组成员；
- 走查的代码；
- 本次走查会所要实现的目标以及这些目标是否达到；
- 包含缺陷位置和描述的缺陷表；

- 走查组对每一个缺陷提出的建议；
- 措施、完成日期和责任人列表；
- 针对如何处理缺陷以及未解决的缺陷而提出的建议；
- 走查组对后续走查提出的建议。

（5）走查注意事项

代码走查中选用的测试用例不易太复杂，但一定要有代表性，通过这些用例，程序员能够举一反三。在大多数的代码走查中，很多问题是通过提问程序员发现的，而不是由测试用例本身直接发现的，因此，应重视其中的提问环节。其他的注意事项与代码审查基本相同。

1.2.5　评审类型比较

上面介绍了三种常用的评审方法，表 1.1 概要给出了技术评审、审查和走查三种方法的比较。

表 1.1　评审类型比较

类　别	技术评审	审　查	走　查
目的	评价工作产品与规格说明和计划的符合性；评估更改的完整性	发现缺陷；验证缺陷修改情况；验证产品质量	发现缺陷；检查备选方法；改进产品；相互学习
决议	评审组要求管理部门或技术领导落实建议	审查组选择预定的产品处置方法；修改缺陷	走查组同意将由程序员作出的更改
更改验证	组长验证措施项已关闭，其他项目控制人员验证更改情况	组长验证措施项已关闭，其他项目控制人员验证更改情况	组长验证措施项已关闭，其他项目控制人员验证更改情况
推荐的小组规模	3～6 人	2～6 人	2～7 人
成员	技术领导和同行；有资质的人员	有资质的同行	技术领导和同行；有资质的人员
组长	通常是主任工程师	经过培训的主持人	主持人或程序员
材料规模	中等或较多（取决于会议的目的）	相对较低（取决于一天审查的数量；材料较多时可拆分）	相对较低
陈述者	由评审组长确定	讲解员	程序员

续表

类　别	技术评审	审　查	走　查
数据收集	非正式的项目要求，可以局部进行	需要	推荐
输出	技术评审文档	缺陷清单、缺陷摘要、审查文档	缺陷清单、措施项、决议、后续工作建议
正式的主持人培训	有，通常只对评审组长培训	有，对所有成员培训	有，通常只对走查组长培训
指定成员角色	是	是	是
使用检查单	可选	是	可选
管理人员参与	可选	无	无
顾客或用户代表参与	可选	可选	可选

1.2.6　静态分析的优点

相对于动态测试而言，静态分析具有以下优点：

① 静态分析能够有效地发现软件中 30%到 70%的逻辑设计错误和编码错误；

② 静态分析成本较低。静态分析容易实现而且不依赖于特定的运行环境，因此，在动态测试之前进行静态测试，能够有效发现软件编码方面潜在的错误。静态分析能够很方便地检查软件中某些难以执行到的代码，如错误处理代码；

③ 静态分析可以提早进行。在软件文档和代码完成后，就可以开展静态分析，从而及早发现软件中存在的问题，同时，静态分析能够准确定位软件错误，从而减少软件修改成本；

④ 静态分析能够有效发现软件中的潜在缺陷。一个实现了所要求功能的软件产品，如果软件内部结构设计复杂、代码编写不规范，就会隐藏一些潜在缺陷，即使软件满足了用户目前的要求，这些潜在的缺陷在软件日后的维护和升级中也会带来很多困难，对于这类缺陷，难以通过动态测试发现。

静态分析不是无条件和万能的，静态分析对执行人员的要求高。执行人员除了具有相关领域知识外，还需要有较高的软件开发能力。另外，静态分析不太适合于多任务的软件以及与软件执行顺序密切相关的软件动态特性分析场合。

第 2 章　C/C++语言程序设计缺陷分析

C/C++是一种介于汇编语言和高级语言之间的中级语言，由于其容易理解，便于阅读和书写，成为很多程序员的首选编程语言。在本章中，笔者将C/C++程序典型的代码缺陷分类列出，并结合软件测试实践，对每种缺陷都给出一段实例代码，方便读者理解。

2.1　编 码 风 格

2.1.1　符号误用问题

符号是程序的一个基本组成单元，其作用相当于句子中的单词，每个符号都有其特定的含义。在程序语句中，符号的误用常常引起语句错误，导致程序异常。

1．布尔型变量被赋值

例 1：

```
1   void  f(bool flag)
2    {
3     int  a;
4     bool ok;
5     ok=true;
6     if(flag=ok) // flag 是一个 bool 型变量
7       a++;
8    }
```

例 1 中，第 6 行条件判断语句，存在布尔型变量 flag 被赋值的错误。

此为试读，需要完整PDF请访问：www.ertongbook.com