HZ BOOKS
华章教育

计 算 机 科 学 丛 书

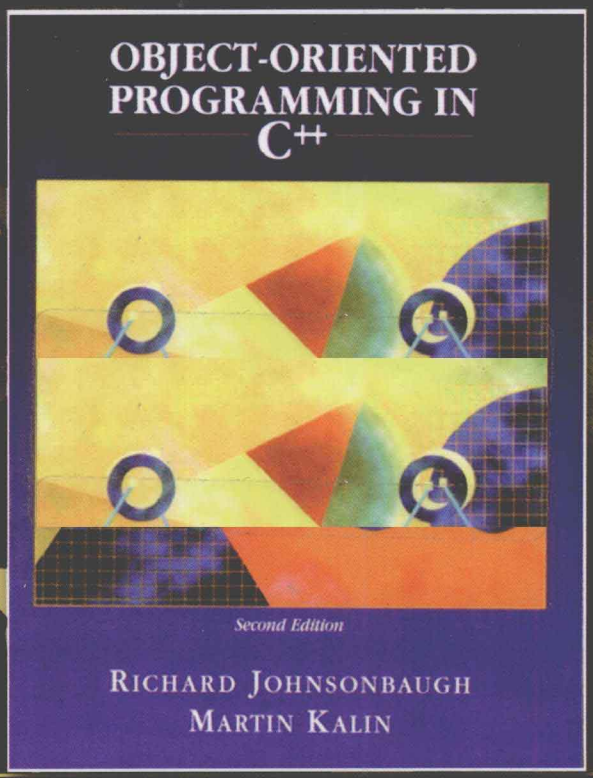
PEARSON

原书第2版

# 面向对象程序设计
## C++语言描述

(美) Richard Johnsonbaugh Martin Kalin 著 蔡宇辉 李军义 译 杨贯中 审校

## Object-Oriented Programming in C++
Second Edition




机械工业出版社
China Machine Press

计 算 机 科 学 丛 书

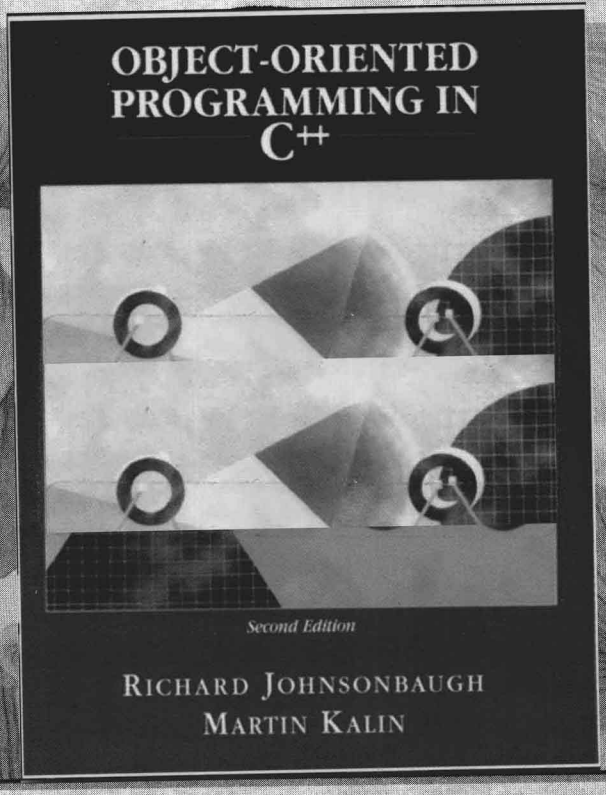
原书第2版

# 面向对象程序设计
## C++语言描述

(美) Richard Johnsonbaugh Martin Kalin 著 蔡宇辉 李军义 译 杨贯中 审校

**Object-Oriented Programming in C++**

Second Edition




机械工业出版社
China Machine Press

本书内容丰富，结构合理，写作风格严谨，深刻地论述了C++语言的面向对象编程的各种技术，主要内容包括：面向对象编程方法、C++语言的各种特性、STL、C++输入/输出流、MFC等。本书针对最新的C++规范作了全面的修订，使读者可以全面掌握C++的最新技术。为使读者学习本书时掌握重点，各章节均配备了大量的练习和编程习题。本书在各章末列举了大量易犯的编程错误及避免这些错误的方法，以帮助读者编写出更为可靠的代码。

本书以作者在大学中讲授的C++课程为基础，特别适合大学计算机专业作为面向对象编程与C++语言课程的教材，同时可供软件开发人员参考。



**本书版权登记号：图字：01-2001-3872**

**图书在版编目（CIP）数据**

面向对象程序设计：C++语言描述（原书第2版）/（美）约翰逊鲍尔（Johnsonbaugh, R.）等著；蔡宇辉，李军义译. —北京：机械工业出版社，2011.6

（计算机科学丛书）

书名原文：Object-Oriented Programming in C++, Second Edition

ISBN 978-7-111-34576-3

Ⅰ.面…　Ⅱ.①约…　②蔡…　③李…　Ⅲ.C语言-程序设计　Ⅳ.TP312

中国版本图书馆CIP数据核字（2011）第081624号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：秦　健

中国电影出版社印刷厂印刷

2011年6月第1版第1次印刷

185mm × 260mm · 29.25印张

标准书号：ISBN 978-7-111-34576-3

定价：69.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：（010）88378991；88361066

购书热线：（010）68326294；88379649；68995259

投稿热线：（010）88379604

读者信箱：hzjsj@hzbook.com

# 出版者的话

Object-Oriented Programming in C++ (Second Edition)

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自1998年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与Pearson，McGraw-Hill，Elsevier，MIT，John Wiley & Sons，Cengage等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出Andrew S. Tanenbaum，Bjarne Stroustrup，Brain W. Kernighan，Dennis Ritchie，Jim Gray，Afred V. Aho，John E. Hopcroft，Jeffrey D. Ullman，Abraham Silberschatz，William Stallings，Donald E. Knuth，John L. Hennessy，Larry L. Peterson等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：


华章网站：www.hzbook.com

电子邮件：hzjsj@hzbook.com

联系电话：（010）88379604

联系地址：北京市西城区百万庄南街1号

邮政编码：100037



华章教育

华章科技图书出版中心

# 译 者 序

Object-Oriented Programming in C++ (Second Edition)

在信息技术飞速发展的今天，计算机相关专业的学生和从事计算机相关教学的教师面临着越来越多的挑战，新技术和新名词层出不穷，种类繁多的可视化开发工具也不断涌现。许多学生在面对这些新奇的开发工具时往往忽略了基本语言的学习，其中为数不少的人误以为熟练掌握一种或几种可视化开发工具就能成为专业的软件开发人员。实际上，任何一种可视化开发工具都只是集成了若干工具的商业化产品而已，衡量一位软件开发人员水准的依据并非只是看他对开发工具的掌握程度，而是要看他是否熟练地掌握了某一门（或几门）程序开发语言，看他是否具备坚实的操作系统知识，看他是否能熟练地设计数据结构和算法等。

C++语言发展至今，已经成为许多开发人员的首选语言，国内各大高校也纷纷将C++语言设置为计算机专业必修课程。同时数据结构课程也越来越多地以C++为描述语言，特别是C++标准委员会在1994年发布了C++标准库之后，C++的应用前景和范围也就更为广阔了。

本书的主要作者Richard Johnsonbaugh是DePaul大学的教授，从事C语言和C++语言教学工作多年，具有丰富的教学经验，并组织编写过多本极受欢迎的教材。本书以他在DePaul大学讲授的C++语言为基础，并根据最新的C++标准对第1版进行了全面更新。

本书可分为两大部分。第一部分包括前面六章，主要阐述面向对象基本原理和C++语言本身的各种知识，涉及面向对象基本原理、类、继承、多态和操作符重载等。第二部分包括第7章到第9章，主要介绍C++标准库和微软基本类库（Microsoft Foundation Classes, MFC），涉及模板、标准模板库（Standard Template Library, STL）、标准输入输出库、MFC、COM等。第一部分是C++语言基本知识，第二部分是C++语言的扩展，第二部分中的STL还可作为数据结构教学的辅助材料。

C++语言的语法相当繁杂，即便是生硬地记住了这些语法而不实际运用，还是无法掌握其精髓。为此，本书作者并没有罗列语法，而是将大量实例贯穿始终，让读者在学习这些例子的同时，自然而然地理解并掌握各种C++语法，并能熟练运用，这也正是本书最大的特点之一。同时，本书在大多数章节中都给出了完整的应用程序示例，并进行了详尽的分析，告诉读者应该怎样设计，为什么要这样设计。通过这些程序，读者可以学到如何运用C++语言的各种技术进行分析、设计与编码，最终构造出精巧的应用程序。

本书翻译工作主要由蔡宇辉完成，李军义对译稿进行了修润，并测试运行了本书大部分源程序，杨贯中教授对全书进行了审校。另外，在本书翻译过程中，得到了英宇、陈治平、周昕、柳杨、廖正军、罗呈等老师和许多研究生的大力协助，在此表示衷心的感谢。

由于译者水平有限且时间仓促，书中难免存在错误，恳请读者批评指正。

蔡宇辉　李军义

2001年1月于湖南大学

本书以作者在DePaul大学讲授的C++课程为基础，可用于自学或在面向对象与C++课程中作为教材使用。本书假定读者没有任何C++知识，但已经具备C语言知识。R. Johnsonbaugh和M. Kalin合著的《*Applications Programming in ANSI C*》（3rd ed., Upper Saddle River, N.J.: Prentice Hall, 1996）可为您提供阅读本书所需的C语言知识。本书包含如下辅助材料：

- 一张光盘，其中含有Microsoft Visual C++编译器。
- 一本教师参考手册。
- 一个www站点。

上述辅助材料与本书一起，提供了一整套帮助读者掌握C++的支持系统。

与我们其他的C和C++书籍一样，我们在本书中提供大量的例子、图表、程序清单、自学练习、编程习题和示例程序，并列举了大量易犯的编程错误。我们力求本书叙述清晰并用例子说明各种各样的良好编程习惯。

本书论述面向对象原理（见第1章），强调编程实践，引导读者使用模板和标准模板库（Standard Template Library, STL，见第7章），深入讲解C++输入/输出类层次（见第8章），详细讲解大量有用的例子（如堆栈类，见3.2节和7.2节；随机存取文件类，见8.6节），介绍了用微软基本类库（Microsoft Foundation Classes, MFC）进行面向对象编程的方法（见第9章）。

本书讲述的C++依照最新的C++标准，包括很多最新的C++技术：

- 逻辑类型**bool**。
- **string**类。
- 新型头文件。
- 名字空间和名字空间**std**。
- 新的类型转换机制。
- 标准模板库。
- 例外处理。
- 运行期类型识别（RTTI）。
- **new[]**操作符。
- 模板输入/输出类。
- **stringstream**类。

## 概述

从20世纪80年代到90年代初期，C语言是许多应用程序和系统程序的首选语言。个人电脑上的很多重要软件都是用C语言编写的，如电子制表软件、字处理软件、数据库系统、通信包、统计软件、图形包等。事实上，UNIX环境中几乎所有的软件基本上是用C语言编写的，而且许多希望能够移植的大型机系统中的软件也是用C语言编写的。在20世纪80年代初期，AT&T贝尔实验室的Bjarne Stroustrup开发了C++语言，作为C语言的面向对象扩展，以用于设计大型复杂系统。现在，C++语言已经用于各种平台，从廉价的个人电脑到昂贵的大型机系统。C++将C语言作为其子集，弥补了C语言的一些不足之处，并且支持抽象数据

类型，通过模板技术还可实现类属函数。

C++语言是相当复杂的。幸运的是，大部分程序设计者不需掌握C++的所有技术细节便可从C++的强大功能中受益。我们将讲述的重点放在C++语言最有用的部分，而将其他较深奥和较特殊的部分放在补充材料部分。我们着重引导读者使用有效的设计技术来编写实际的C++程序，而不是卖弄技巧。

## 关于本书

本书主要包括以下内容：

- 大量的实例和练习。
- 与现实世界密切相关的应用程序。
- 大量编程习题，本书有100多个编程习题。
- 在每章的结尾列举出常见的编程错误。
- 涵盖STL的内容（见第7章）。
- 讨论了标准C++输入/输出类库（见第8章）。
- 涵盖MFC面向对象编程的内容（见第9章）。
- 通过小节后面的练习，读者可以检查自己对该节的掌握程度。本书包含500多个练习题，书后提供奇数编号练习的答案，偶数编号练习的答案在*教师参考手册*中。
- 提供大量图表和程序清单。
- 提供最新的C++技术。
- 主要数据结构用C++实现，如堆栈（见3.2节和7.2节）、文件（见8.6节）。
- 易于理解的代码，我们力求书中的代码清晰可读。
- 提供一套Microsoft Visual C++学生版软件。

## 对第1版所作的改动

- C++近来的改动和新增内容均在第2版得到了反映。
- 名字空间的概念提前到了2.1节，因为新型头文件需要用到名字空间**std**。
- 例外处理的概念提前到了2.8节。
- 多态成为单独的一章（第5章）。
- 继承提前到第4章，多态提前到第5章，以体现它们在面向对象编程范例中的重要性，而将操作符重载推后到第6章。
- 运行期类型识别（RTTI）放到5.5节。
- 模板和STL成为单独的一章（第7章）。
- 对第8章（第1版中为第7章）的输入/输出类层次作了重大改进，以反映这个类层次的重要修改。
- 增加第9章，讲述MFC中的面向对象编程。
- 部分深奥的内容放到补充材料部分。
- 实例数目增加到近300个。
- 练习的数目增加到500多个。
- 对示例程序进行了广泛的修改。
- 图表加边线，与文字分开。
- 为本书建立了WWW网站，提供本书的最新资料。
- 在光盘中增加了Microsoft Visual C++学生版软件。

## 本书的组织结构

第1章概述面向对象程序设计的若干重要概念，如类、抽象数据类型、对象、封装、客户/服务器模式、消息传递、继承、多态等。本章比较了面向对象方法与自顶向下的功能分解法之间的差别，并举例说明了这种差别。

第2章阐述名字空间、**string**类型、**new**和**delete**操作符、例外处理及基本C++输入/输出流。同时，还讲述了C++的改进和新增内容。学完本章后，读者就可以正确使用C++的特性了。

第3章覆盖设计类所需的基本知识，具备这些知识后，读者就可以进行类的设计了。本章讲述了类的声明、构造函数和析构函数等成员函数、静态数据成员和静态成员函数、对象指针等。第3章列举了大量实例以说明如何设计类来实现抽象数据类型，从而满足面向对象的封装要求。

第4章讲解继承（包括多继承），并通过大量的实例和示例程序（如sequence类层次）讲解基本的编程技术。

第5章讲解多态。5.1节详细讲解运行期绑定与编译期绑定的区别，5.4节与5.5节分别讲述抽象基类与运行期类型识别技术。

第6章讲解操作符重载。本章说明如何对普通操作符（如+、/等）进行重载，以及下标操作符、函数调用操作符、内存管理操作符、前置自增操作符和后置自增操作符等特殊操作符的重载。通过实例和示例程序，展示了操作符重载的强大功能。

第7章讲述模板和STL。7.2节通过堆栈模板类展示模板的使用方法，7.4节通过示例程序说明如何使用STL。

第8章讲解C++输入/输出类层次。首先，通过对C++输入/输出库进行详细讨论，使有兴趣的读者可以利用其中包含的具有强大功能的类。其次，将C++输入/输出类库作为主要的成熟的面向对象设计的实例讲解应该如何设计C++类层次。最后，将C++输入/输出类层次作为模板应用的极佳范例讲解模板的使用方法。第8章对操纵器给予特别的关注，它们是C++中实现复杂输入/输出强有力的方法。读者通过这一章的学习，可以彻底检验对C++输入/输出的掌握程度。

第9章讲解使用MFC进行面向对象程序设计的方法。本章阐述了用MFC和用Win32 API（Application Programmer Interface）进行Windows程序设计的区别，Win32 API是Microsoft访问系统服务的C库。本章还介绍了事件驱动程序设计的基本概念和结构。本章还讲述利用序列化技术处理对象持久性的方法。另外，本章简述了微软的组件对象模型（Component Object Model，COM）技术。我们在两个示例程序中概述了MFC和Visual C++。

本书有两个附录。附录A为ASCII表。附录B为部分最常用的C++函数和类成员函数，说明了这些函数的参数表和返回值，需要什么头文件，以及函数的功能简介。

本书在讲述C++的语法和语义特点时大量依赖于书中的实例、图表、程序清单。由于任何单一的方法都不可能透彻地教授好一门语言课程，所以根据多年讲述C++及其他语言类课程的经验，我们坚信通过上述不同方式的结合将达到较好的教学效果。

大多数学生同我们一样，认为学习和使用C++是令人兴奋的，本书通过生动的实例、示例程序、编程习题和代码片断，力求使读者能够体验这种感受。

## 本书的章节结构

本书章节的组织如下：

- 本章目录
- 本章概述
- 节
- 课后练习

- 节
- 课后练习

⋮

- 补充材料
- 常见编程错误
- 编程习题

除了第1章与第2章，本书其他各章都包含了一个或多个示例程序，每个示例程序都用一节的篇幅来阐述，该节又分为问题、示例程序的输入与输出、解决方案、类的实现或代码实现、代码分析等小节。多数小节都作了进一步讨论。

本书包含如下示例程序：

- 堆栈类及模板堆栈类（3.2节，7.2节）
- 影片跟踪管理（4.3节，5.2节）
- 序列的类层次（4.6节）
- 复数节（6.2节）
- 关联式数组（6.8节）
- 证券业绩报表（7.4节）
- 随机存取文件类（8.6节）
- 自动化服务器与控制器（9.5节）

补充材料部分主要讨论C++语言中不太常用的一些机制，或者对该语言的某部分提供一些额外的技术细节。

常见编程错误列举了在使用C++语言时易犯的一些错误。

本书包含了100多个编程习题，这些习题来源于各种实际的应用程序。

## 本书实例

本书包含大约300个实例，每个实例都为读者阐述了C++语言的某个知识点。每个实例的结束符号为■。

## 本书练习

本书包含500多个练习、代码片断以及少数完整的程序，答案形式一般为简短的回答。这些练习可用于家庭作业或自测验。本书最后给出了奇数编号练习的答案，偶数编号练习的答案在光盘的教师参考手册中。我们的教学经历证明课后练习是非常重要的。

每章后面都附有编程习题，有些习题是较完整的应用程序，以下列出了其中的一部分：

- 模拟（编程习题2-9）。
- 队列（编程习题3-8与7-4）。
- 进程同步（编程习题3-10）。
- 数据库（编程习题3-15）。
- 局域网（编程习题3-17与6-7）。
- 数组分层（编程习题4-4）。
- 日期服务（编程习题5-10）。
- 迭代器（编程习题7-6、7-7与7-8）。
- 课程安排（编程习题7-14）。
- 索引文件类（编程习题8-5）。
- 基于对话框的并带有图形用户界面的系统管理应用程序（编程习题9-9）。

并非所有读者都会对这些应用程序感兴趣，本书中包含这些应用程序是为了阐述用C++进行程序设计时通常会遇到的一些问题。

## 光盘内容

本书所附光盘提供了一套Microsoft Visual C++集成开发环境，该集成开发环境包含一个C++编译器、一个调试器、一个类浏览器、帮助工具、MFC、ActiveX模板库及其他应用程序开发所需的资源。本书中的所有示例程序，包括第9章所示的基于Windows的应用程序，均可使用Visual C++进行编译。

## 教师参考手册

出版社免费提供一本*教师参考手册*，其中包含偶数编号练习的答案、教学大纲示范和教学用幻灯片。

## WWW站点

我们在www站点**http://condor.depaul.edu/~mkalin**提供了本书所包含的源代码、头文件、本书中示例程序所用到的数据文件、某些较大例子的源代码、教学大纲示范、教学用幻灯片、使用Microsoft Visual C++的说明信息、有关MFC的附加技术细节、错误修正表等。

## 致谢

我们在此衷心感谢以下各位评论家：西雅图大学的Adair Dingle、独立顾问Rex Jaeschke、DePaul大学的Glenn Lancaster、南康涅狄格州立大学的Winnie Y.Yu。

我们衷心感谢Patricia Johnsonbaugh，她为本书的校订工作作出了极大的贡献。

DePaul大学计算机与通信信息系统学院及院长Helmut Epp为本书的撰写提供了大量帮助，在此表示由衷的感激。

在本书编写过程中，Prentice Hall出版公司一直为我们提供各种支持，特别要感谢Alan R.Apt、Petra Recter和Scott Disanno，在他们的帮助下，本书才得以顺利出版。

R. J.

M. K.

# 目　录

Object-Oriented Programming in C++ (Second Edition)

# 面向对象编程

C++语言是一种混合型语言，一方面我们可以把C++当做是C语言的扩展和改进，在这种意义上，C++是一种过程语言；从另一方面来看，C++又充分展示了其面向对象的特性（这些特性都是C语言和其他过程语言所不具备的），因此，C++又是一种面向对象的语言。本章我们主要讨论与面向对象的编程风格相关的基本概念及其优点，并探讨相关的程序设计技术。以后的章节将详细讨论用C++语言进行面向对象程序设计的细节。

## 1.1 面向过程的编程风格与面向对象的编程风格

### 1.1.1 面向过程的编程风格

程序由**模块**（module）构成。设计程序时，可以对这些模块分别进行设计、编码和测试，最后将这些模块有机地组合在一起形成一个完整的程序。

C++语言由C语言发展而来。我们说C语言是一种**面向过程语言**（procedural language），是因为在C语言程序中，一个模块就是一个过程。在过程语言中，由于过程是由赋值语句、测试语句、过程调用等各种命令语句组成，因此有时候我们也将过程语言称为**命令型语言**（imperative language）。和C语言一样，C++语言中的函数就是过程。可以将C++语言当做是改进的C语言来使用。也就是说，当函数构成程序中的模块时，C++语言就成了一种过程语言。

面向过程程序设计通常采用**自顶向下设计**（top-down design）方法进行设计。在这种方法中，待解问题和程序设计语言中的过程紧密相连。例如，对于制定制造汽车任务的调度表这件事情，可将待解问题标记为MainProblem。在这里我们打算用C语言、Pascal语言这类过程语言，或者是当做过程语言使用的C++语言来解决上述问题。如果采用C++来做的话，我们很快想到可以将待解问题MainProblem对应到C++的过程：**main**。但制造汽车太复杂了，如果将所有的任务都加入到**main**中，MainProblem就太复杂了。常用的办法是将待解问题分解成若干子问题：

- 生产底盘。
- 生产引擎。
- 生产动力传动系统。
- 组装。
- 检测配件和整车。

我们可以将上述子问题分别对应到**main**调用的函数，即子过程。如同MainProblem可以分解成若干子问题一样，过程**main**也可以分解为处理子问题的各种子过程。当然这些子问题可以继续分解。这种分解就反应为子过程的分解（如图1-1所示）。不断地运用这种**自顶向下**设计方法，即**函数分解法**（functional decomposition），直到每一个子问题都足够简单，使得相应的子过程很容易处理。当采用这种高度模块化的方式来设计C++程序时，C++程序的基本过程（即函数）就会足够简单和短小，甚至可以只包含一条语句（例如，只包含一条**return**语句）。

自顶向下设计的优点是既直观又有条理，有很多复杂问题是采用这一方法解决的。但这种方法也有致命的缺陷，特别是在面对**软件维护**（software maintenance）问题时，包括软件系统的测试、调试和升级等。有经验的程序员都知道，软件开发最困难的时期并不是第一次设计代码的时候，而是后面的修改阶段，因为程序出现了毛病（“受到细菌的感染”），程序的需求发生了改变，程序需要提高执行效率，等等。我们仍然用制造汽车为例来说明这个问题。假设需要对MainProblem的解决方案进行重大改变，甚至于**main**都需要