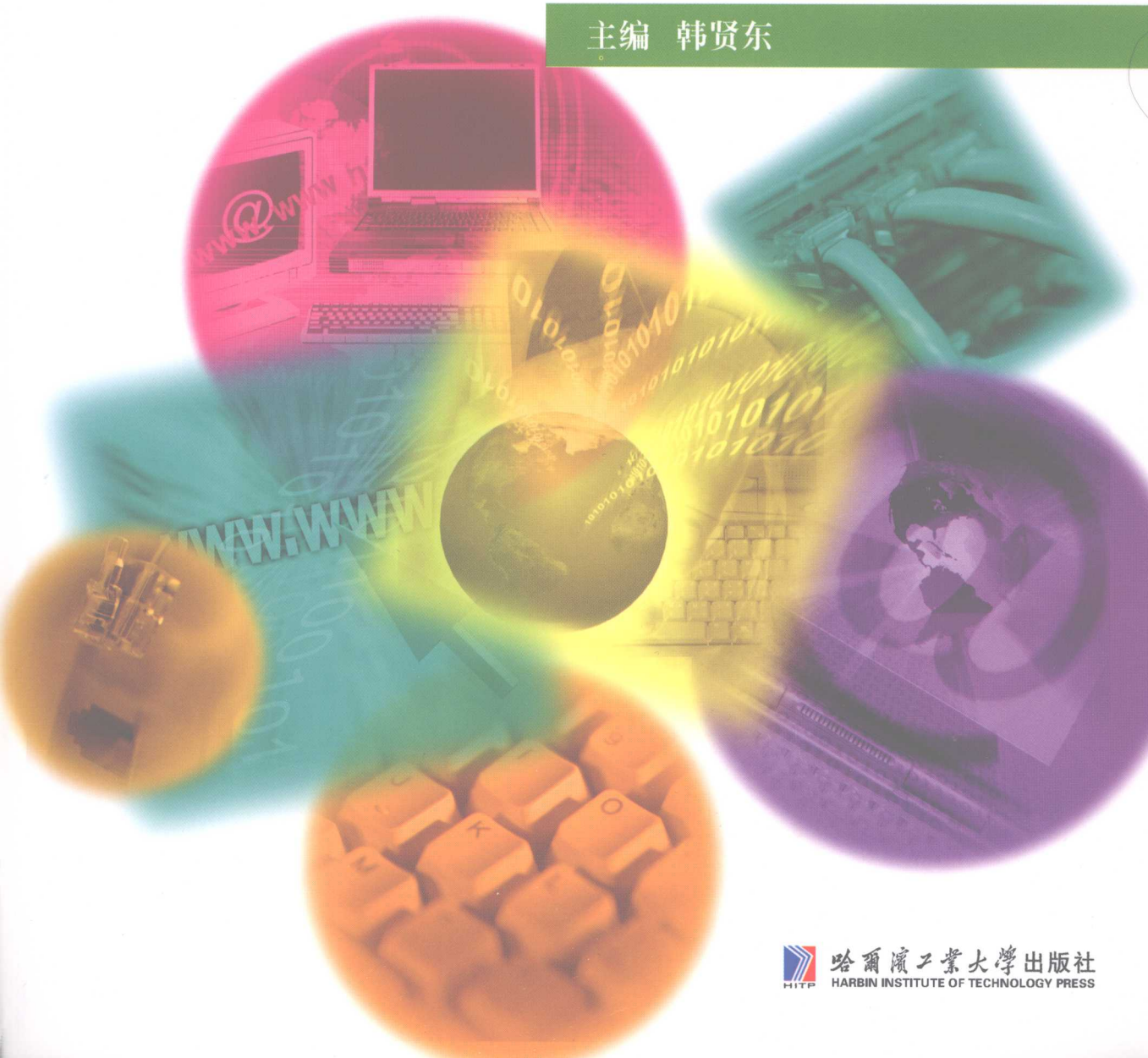
高等学校“十一五”规划教材·计算机系列

# C++程序设计语言

主编 韩贤东



哈尔滨工业大学出版社
HARBIN INSTITUTE OF TECHNOLOGY PRESS

高等学校“十一五”规划教材·计算机系列

# C++程序设计语言

主　编　韩贤东
副主编　刘兴丽　付　伟

哈尔滨工业大学出版社

## 内容简介

本教材包含C++面向对象编程(OOP)部分和泛型编程部分(GP)。面向对象编程部分主要介绍封装、继承、多态等面向对象编程的概念和方法;泛型编程部分主要介绍容器、迭代器、函数对象、算法、适配器等泛型编程的概念和方法。

本教材适合于本科阶段的学生在已经完成C语言学习的基础上,用大约48学时理论课及28学时的实验课掌握C++的面向对象编程和泛型编程。本教材适合作为高等学校计算机专业和非计算机专业程序设计课程教材,也可作为组件、游戏、面向对象数据库、ACM/ICPC竞赛编程的参考书。

**图书在版编目(CIP)数据**

C++程序设计语言/韩贤东主编.—哈尔滨:哈尔滨工业大学出版社,2008.8

高等学校"十一五"规划教材·计算机系列

ISBN 978-7-5603-2767-9

Ⅰ.C… Ⅱ.韩… Ⅲ.C语言-程序设计-高等学校-教材 Ⅳ.TP312

中国版本图书馆CIP数据核字(2008)第128617号

责任编辑 贾学斌 王桂芝

出版发行 哈尔滨工业大学出版社

社 址 哈尔滨市南岗区复华四道街10号 邮编150006

传 真 0451-86414749

网 址 http://hitpress.hit.edu.cn

印 刷 哈尔滨工业大学印刷厂

开 本 787mm×1092mm 1/16 印张12.25 字数306千字

版 次 2008年11月第1版 2008年11月第1次印刷

书 号 ISBN 978-7-5603-2767-9

定 价 25.00元

# 高等学校“十一五”规划教材·计算机系列

# 编 委 会

# 序

当今社会已进入前所未有的信息时代，以计算机为基础的信息技术对科学的发展、社会的进步，乃至一个国家的现代化建设起着巨大的推进作用。可以说，计算机科学与技术已不以人的意志为转移地对其他学科的发展产生了深刻影响。需要指出的是，学科专业的发展都离不开人才的培养，而高校正是培养既有专业知识、又掌握高层次计算机科学与技术的研究型人才和应用型人才最直接、最重要的阵地。

随着计算机新技术的普及和高等教育质量工程的实施，如何提高教学质量，尤其是培养学生的计算机实际动手操作能力和应用创新能力是一个需要值得深入研究的课题。

虽然提高教学质量是一个系统工程，需要进行学科建设、专业建设、课程建设、师资队伍建设、教材建设和教学方法研究，但其中教材建设是基础，因为教材是教学的重要依据。在计算机科学与技术的教材建设方面，国内许多高校都做了卓有成效的工作，但由于我国高等教育多模式和多层次的特点，计算机科学与技术日新月异的发展，以及社会需求的多变性，教材建设已不再是一蹴而就的事情，而是一个长期的任务。正是基于这样的认识和考虑，哈尔滨工业大学出版社组织哈尔滨工业大学、东北林业大学、大庆石油学院、哈尔滨师范大学、哈尔滨商业大学等多所高校编写了这套“高等学校计算机类系列教材”。此系列教材依据教育部计算机教学指导委员会对相关课程教学的基本要求，在基本体现系统性和完整性的前提下，以必须和够用为度，避免贪大求全、包罗万象，重在**突出特色**，体现**实用性和可操作性**。

(1)在体现科学性、系统性的同时，突出实用性，以适应当前IT技术的发展，满足IT业的需求。

(2)教材内容简明扼要、通俗易懂，融入大量具有启发性的综合性应用实例，加强了实践部分。

本系列教材的编者大都是长期工作在教学第一线的优秀教师。他们具有丰富的教学经验，了解学生的基础和需要，指导过学生的实验和毕业设计，参加过计算机应用项目的开发，所编教材适应性好、实用性强。

这是一套能够反映我国计算机发展水平，并可与世界计算机发展接轨，且适合我国高等学校计算机教学需要的系列教材。因此，我们相信，这套教材会以适用于提高广大学生的计算机应用水平为特色而获得成功！

2008 年 1 月

# 前　言

笔者对当前 C++ 中文教本的一些了解，奠定了本教材编写的结构：Stanley Lippman 的《C++ Primer》第三版（潘爱民等译），结合了 Stanley 多年的教学和实践经验及 C++ 标准委员会前语言组负责人 Josée Lajoie 对 ISO/IEC14882 的深刻理解，是一部经典之作。名称虽然说是初学者的 C++，实际上它是一部 1 000 多页的 C++ 大全，但初学者想在短时间内通读全书达到掌握 C++ 的目的，实非易事。Stanley Lippman 也认识到这一点，后来他出了一本专门针对 C++ 初学者的书，叫《Essential C++》（侯捷译），笔者以此为教本进行 C++ 教学时发现，作为一本泛型编程（GP）的入门篇，它缺乏面向对象编程（OOP）这一环节的正规介绍，学生理解掌握 GP 时显现出许多障碍。

基于上述考虑，本教材在章节编排上，默认学生先前已学过 C 语言编程（CP）、初步具备数据结构方面的知识，在介绍 OOP 时注重它在 CP 和 GP 之间的桥梁作用。另外笔者认为，语言教学应注重语言特性的讲授，算法是数据结构等算法类课程的教学内容，所以本书 C++ 的代码举例避免介绍和实现某个算法，而以简短的代码来介绍语言和类库的使用，且代码举例所依托的背景知识是学生所熟悉和易于理解且紧扣章节内容的。整书章节的具体安排如下：

（1）本教材共分 14 章，隐含地分两部分："第一部分 面向对象编程部分（1～8 章）"和"第二部分 泛型编程部分（9～14 章）"，除第 1 章之外，每章还设有实验题目，供实验课中使用。

（2）使学生在已经完成 C 语言学习的基础上，用大约 48 学时理论课及 28 学时的实验课掌握 OOP 和 GP。

（3）通过学习"第 2 章 从 C 向 C++ 过渡"，首先使读者由 C 语言平滑过渡到 C++ 语言，继而学习构造、继承、多态等 OOP 概念。

（4）第 5、6、8 章，首先是作为 OOP 的补充，也是 GP 部分的基础。

（5）第 7 章 既是流类的知识性章节，也是 OOP 类层次体系结构实现的实例。

（6）GP 部分主要以介绍标准模板库（STL）为主，不涉及 Boost 部分，因为目前 Boost 还没有成为国际标准，但学好 STL 是继而掌握 Boost 的必由之路。

（7）通过学习"第 9 章 向泛型编程过渡"，使学生在 C/C++ 指针、指针函数与 GP 部分的迭代器、函数对象之间建立联系。

(8)“第 10 章”集中介绍 GP 中容器、迭代器、函数对象、算法、适配器等基本概念。

(9)从“第 11 章”开始,是有关 STL 基本概念的实例和应用。

本书第 2、3、4 章由黑龙江科技学院刘兴丽执笔,第 5、8、9 章由哈尔滨师范大学付伟执笔,第 1、6、7 及 10 ~ 14 章由哈尔滨师范大学韩贤东执笔,全书由韩贤东统稿。考虑到 ACM/ICPC,本书强化了 OOP 的格式输入输出部分(第 7 章),重点放在 GP 部分,并且用竞赛环境中使用的 GNU C++ 编译、调试通过所有书中举例代码。书中举例代码、课件读者如有需要,可通过邮箱 hitcpp 2008@126.com 与作者联系索取。

由于作者水平有限,书中疏漏及不妥之处在所难免,敬请读者批评指正。

编　者

2008 年 7 月

# 目　录

# 第1章 概　述

本章重点：Bjarne Stroutrup 对 C++ 的三个基本定义；OOP 和 GP 的历史。

本章难点：OOP 和 GP 代码的举例，但不要求读者在学习的这一阶段完全理解这一部分，在教师的讲解之下，对 GP 与 CP、OOP 的区别有个初步的印象即可。

在这一章中，通过介绍与 C++ 相关语言的演进历程，使读者对 C++ 的过去、目前现状和今后发展方向有个概要性的了解；通过几段实例代码，比较 CP(C Programming)、OOP(Object-Oriented Programming)、GP(Generic Programming)的不同点，阐述 GP 与 CP、OOP 的区别。

## 1.1 C++ 程序设计语言

Bjarne Stroutrup 给 C++ 下的基本定义是：

- 一种经过改进更为优化高效的 C 编程(CP)语言。
- 支持面向对象编程(OOP)。
- 支持泛型编程(GP)。

C++ 之父把 C++ 与 C 的兼容放到了首要位置，所以 C++ 的历史首先应该对其之前的 C 历史作简要的回顾，之后对 C++ 的历史作介绍。

在回顾历史之前，有必要对 C、C++ 面向对象、C++ 泛型编程建立感性认识。

### 1.1.1 C 磁盘数据读写程序

下面是用 C 编程(CP)的方法实现的磁盘文件读写程序，用到文件指针和一系列与磁盘操作有关的 C 语言 IO 标准库中的函数。

【例 1.1】 C 磁盘数据读写程序如下。

| 行号 | EX_01_01.cpp 文件代码 | 说　明 |
|---|---|---|
| 1 | # include < cstdio > | C++ 中的 C 头文件 |
| 2 | | |
| 3 | int main() | |
| 4 | { | |
| 5 | int nT; | |
| 6 | FILE * fin; | 输入文件指针 |
| 7 | if((fin = fopen("InputFile.txt", "r")) == NULL) | 只读方式打开文件 |
| 8 | return -1; | |

| 行号 | EX_01_01.cpp 文件代码 | 说　明 |
|---|---|---|
| 9 | FILE * fout; | |
| 11 | if((fout = fopen("OutputFile.txt", "w")) == NULL) | 只写方式打开文件 |
| 12 | return -1; | |
| 13 | while(fscanf(fin,"%d", &nT) == 1) | 读取文件中的整型值 |
| 14 | { | |
| 15 | fprintf(fout,"%d ",nT); | 写入输出文件 |
| 16 | } | |
| 17 | fflush(fout); | 清空缓存 |
| 18 | fclose(fout); | 关闭输出文件 |
| 19 | fclose(fin); | 关闭输入文件 |
| 20 | return 0; | 调试断点 |
| 21 | } | |
| 22 | | |

| 行号 | InputFile.txt 文件内容 | 说　明 |
|---|---|---|
| 1 | 5 7 8 6 1 2 9 | 输入文件已有的内容 |
| 2 | | |

| 行号 | OutputFile.txt 文件内容 | 说　明 |
|---|---|---|
| 1 | 5 7 8 6 1 2 9 | 由代码 13 ~ 18 行输出到文件的内容 |
| 2 | | |

上述代码的流程是,先在 7、11 行获取文件指针,13 ~ 16 行利用获取的文件指针完成文件数据的拷贝,17 ~ 18 行完成文件缓存的清空及指针相关资源的释放操作。

### 1.1.2 C++面向对象磁盘数据读写程序

下面是用面向对象编程(OOP)的方法实现的读写磁盘文件的例子,其中用到 IO 文件流对象,读写文件不是使用函数,而是通过系统提供的重载的左移、右移运算符函数,代码书写相对于 C 更加清晰、简洁,易于排错、理解和维护。

【例 1.2】 C++面向对象磁盘数据读写程序如下。

| 行号 | EX_01_02.cpp 文件代码 | 说　明 |
|---|---|---|
| 1 | #include <iostream> | IO 流头文件 |
| 2 | #include <fstream> | 文件流头文件 |
| 3 | using namespace std; | 使用 std 名称空间 |
| 4 | | |
| 5 | int main() | |
| 6 | { | |
| 7 | int nT; | |
| 8 | ofstream fout("OutputFile.txt"); | 实例化输出文件对象 |
| 9 | if(! fout) return -1; | |
| 11 | | |
| 12 | ifstream fin("InputFile.txt"); | 实例化输入文件对象 |
| 13 | if(! fin) return -1; | |

| 行号 | EX_01_02.cpp 文件代码 | 说 明 |
|---|---|---|
| 14 | | |
| 15 | while(fin >> nT) | 读取文件中的整型值 |
| 16 | { | |
| 17 | fout << nT << ends; | 写入输出文件 |
| 18 | } | |
| 19 | | |
| 20 | fout.close(); | 关闭输出文件 |
| 21 | fin.close(); | 关闭输入文件 |
| 22 | return 0; | 调试断点 |
| 23 | } | |

| 行号 | InputFile.txt 文件内容 | 说 明 |
|---|---|---|
| 1 | 5 7 8 6 1 2 9 | 输入文件已有的内容 |
| 2 | | |

| 行号 | OutputFile.txt 文件内容 | 说明 |
|---|---|---|
| 1 | 5 7 8 6 1 2 9 | 由代码 15 ~ 20 行输出到文件的内容 |
| 2 | | |

上述代码的流程是,先在 8、12 行获取输入输出文件对象,15 ~ 18 行利用获取的文件对象 fin、fout 完成文件的拷贝,20、21 行关闭文件对象来释放资源。

### 1.1.3 C++ 泛型方法磁盘数据读写程序

下面是用泛型编程(GP)的方法实现的磁盘文件读写程序,这里涉及到容器、迭代器、算法的概念,读写文件采用算法,这是与以往的 C/C++ 方法不同的地方,以往我们多少要作一些代码编写工作,如例 1.1 和例 1.2 中的文件读写 while 循环部分的代码,在泛型编程中运用算法就可以避免这种过程代码的书写。因为是在 OOP 基础上发展出来的 GP,所以除具有 OOP 的优点之外,GP 的代码编写更加高效、易于扩充。

【例 1.3】 C++ 泛型方法磁盘数据读写程序如下。

| 行号 | EX_01_03.cpp 文件代码 | 说 明 |
|---|---|---|
| 1 | #include <iostream> | |
| 2 | #include <iterator> | GP 相关头文件 |
| 3 | #include <fstream> | |
| 4 | #include <algorithm> | GP 相关头文件 |
| 5 | using namespace std; | |
| 6 | | |
| 7 | int main() | |
| 8 | { | |
| 9 | ofstream fout("OutpuFile.txt"); | 实例化输出文件对象 |
| 11 | if(! fout) return -1; | |
| 12 | | |
| 13 | ifstream fin("InputFile.txt"); | 实例化输入文件对象 |
| 14 | if(! fin) return -1; | |