

现代软件工程专业系列教材

软件开发基础教程

RUANJIAN KAIFA JICHU

JIAOCHENG

上册

程国英 钱晓平 编著



清华大学出版社

<http://www.tup.tsinghua.edu.cn>



北京交通大学出版社

<http://press.bjtu.edu.cn>



现代软件工程专业系列教材

TP311.52
128
:1

软件开发基础教程

(上册)

程国英 钱晓平 编著

清华大学出版社
北京交通大学出版社

·北京·

内 容 简 介

本书分上、下两册，上册主要介绍面向对象（包括面向过程）的程序设计，下册以面向消息的可视化软件设计和开发为主。选择 C++ 语言进行讲述。

上册的第一部分（第 1～7 章）是基础准备及入门，主要介绍一些基本概念，展现软件开发的“平台”，使得读者大致上对计算学科及本课程的作用有所了解。第二部分（第 8～31 章）是 C++ 程序设计，结合 HIS（Hospital Information System）实例循序渐进地讲述如何进行程序设计和开发。下册将把 HIS 提高到软件的角度进行设计和开发。

本书的特点是先提出“问题”，直接面对“问题”，然后“抽象分析”问题，再如何“设计”、“解决”问题，体会面向对象和面向过程的区别与联系，展现一个“生产”软件的全貌，加强系统性和抽象分析问题的训练。如果顺利修完这门课程，应该掌握设计软件的基本知识和开发可视化应用软件的基本能力，对提高开发软件的兴趣，提高对科学、专业的觉悟，增强自信心应该有所帮助。

本书的上册适合作为计算机专业、软件工程专业及其他相关专业的 C++ 程序设计课程的教材；本书的上、下两册适合作为计算机专业、软件工程专业的软件开发基础课程的教材。由于本书贯穿实例进行并加以实现，也很适合自学者使用。

版权所有，翻印必究。举报电话：010-62782989 13501256678 13801310933

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

本书防伪标签采用特殊防伪技术，用户可通过在图案表面涂抹清水，图案消失，水干后图案复现；或将表面膜揭下，放在白纸上用彩笔涂抹，图案在白纸上再现的方法识别真伪。

图书在版编目（CIP）数据

软件开发基础教程·上册 / 程国英，钱晓平编著. —北京：清华大学出版社；北京交通大学出版社，2006.5

（现代软件工程专业系列教材）

ISBN 7-81082-753-7

I. 软… II. ①程… ②钱… III. 软件开发-教材 IV. TP311.52

中国版本图书馆 CIP 数据核字（2006）第 041316 号

责任编辑：孙秀翠 特邀编辑：刘云

出版者：清华大学出版社 邮编：100084 电话：010-62776969

北京交通大学出版社 邮编：100044 电话：010-51686414

印刷者：北京鑫海金澳胶印有限公司

发行者：新华书店总店北京发行所

开本：203×280 印张：27.25 字数：850千字

版次：2006年5月第1版 2006年5月第1次印刷

书号：ISBN 7-81082-753-7/TP·274

印数：1～4 000册 定价：42.00元

本书如有质量问题，请向北京交通大学出版社质监组反映。对您的意见和批评，我们表示欢迎和感谢。

投诉电话：010-51686043, 51686008；传真：010-62225406；E-mail: press@center.bjtu.edu.cn。

前 言

随着计算机技术和应用的飞速发展，原来对“编程”、“程序设计”的关注已上升至对“软件开发”的关注，原来的“程序=算法+数据结构”概念已扩展为“程序=算法+数据结构+程序设计方法+语言工具+开发环境”或“程序=对象+对象+…”。面向对象的程序设计如果从以往的“掌握某门语言，从而学会程序设计，并且能够编写程序”出发进行讲解或学习，得到的效果往往很不理想。虽然学生通过修完这门课程，具备了编程的基本能力，但是面对基于消息发送机制的可视化的软件开发需求，往往感到束手无策，学与用之间产生了困惑和“脱节”现象。作者从 1997 年开始执教程序设计、软件工程等课程，深有这方面的体会。

软件设计的基础是程序设计，而程序设计的关键在于“算法”和“数据结构”，除了掌握语言工具外，至少需要具有如下几部分的基础知识：软件工程、数据库、算法与数据结构、开发环境。程序设计不能不涉及这些知识，尤其是算法与数据结构，但是这些内容要揉合在一起比较困难，而程序设计课程又是在大学一年级时开设的，面对的都是刚入学的新生，他们缺少计算学科所特有的思维和分析问题方面的能力训练。因此，我建议“程序设计”是否可考虑为“软件开发基础”，一开始就为学生展现一个软件开发的“平台”，让学生一开始就站在软件的高度，学会从宏观上去观察、理解、分析问题，微观上具体着手解决问题，让学生感悟到程序设计仅仅是软件开发过程中的一部分。而且，面向对象程序设计的思想就是从宏观上分析、抽象问题，在微观上面向过程具体解决问题。小的例题能够说明面向过程的思想，但是很难说明面向对象的思想，无法感悟面向对象程序设计与面向过程程序设计的本质区别，以致学完 C++ 程序设计后的“困惑”拂之不去。整个课程的安排围绕着一个具有一定规模（具有较大的信息量、较复杂的关系）的教学案例进行，例如，HIS（Hospital Information System），从问题的提出、信息收集、需求分析、设计（包括数据库设计）、软件实现，直至测试和软件发布、维护，进行系统性的讲解，介绍软件工程基础导论、数据库的使用，侧重讲授程序设计。同时向学生提供 1~2 个实验案例作为作业，按教学进度同步进行。基本算法和数据结构的基础应用可以在本课程中进行，算法的优劣、性能分析等可在后续的“算法设计与分析”课程中专门讲解，使得学生从理论上得到升华。语言对软件设计、程序设计而言，只是实现“设计”的工具，就好比我们在思考某件事情的时候完全可以采用自己的独特的思维符号，但是一旦把思考结果向他人表述时就应该根据表述对象采用某种合适的语言进行。因此，应该注重“设计”，同时激发和培养学生对语言和应用软件的自学能力。经过几届的教学实践，效果比较理想。所幸的是，正值整理教案之际，北京交通大学出版社邀约编写软件开发类教材，为我们向学生提供正式教材而创造了机会。

教学的对象是学生，所以应该面向学生来考虑教学问题，包括教材、课程设置、教学过程。软件的技术复杂性及快速变更所造成的独特的“行业”环境，强调从业人员必须具备处理复杂问题的能力和自学的能力。我们必须加强学生理解科学的能力，提高科学素养，必须训练学生的适应性，提高学生独立解决问题和处理复杂问题的能力。本课程的任务是从“软件工程”的角度出发，培养学生具有对较大型软件的系统结构和构造技术的理解能力，具有思考软件结构问题和软件设计的初步能力；使学生养成良好的编程习惯、软件风格和具备健康的软件工程文化素养，掌握并熟练使用 C++ 程序开发工具，学会并具备熟练开发清晰易懂的、符合需求的程序及软件的能力。

本教材与钱晓平教授合作编著，从章节、具体内容、例题均不断进行讨论和斟酌，最后由钱晓平教授统稿、审校。

非常感谢我的学生，感谢我校联读班、MIT 试点班、软件学院和 ACM 试点班的学生，感谢他们在教学过程中给予我的建议、激励和帮助，感谢已毕业的学生反馈给我的宝贵信息和建议。

由于本人的学术理论水平有限,纵然全身心地投入此教材的编写之中,仍不免存在不周和不足之处,真诚地期盼得到读者的批评和建议。

程国英 2006年4月22日 于上海
gycheng@sjtu.edu.cn

目 录

第 1 部分 基础准备及入门

第 1 章 引言	(1)
1.1 程序与软件	(1)
1.2 软件与软件工程	(1)
1.2.1 软件的特点	(2)
1.2.2 软件危机	(2)
1.2.3 软件工程	(3)
1.2.4 软件工程目标	(3)
1.3 计算机科学、计算机技术、计算机工程	(3)
1.4 软件工程文化	(4)
1.5 如何学好这门课程	(4)
1.5.1 掌握学科形态和核心概念	(4)
1.5.2 实现思维方式数学化	(5)
1.5.3 理解科学、提高科学素养	(5)
1.5.4 掌握本课程的知识内容	(6)
1.5.5 上机实践	(6)
第 2 章 软件开发的一般过程	(7)
2.1 分析与限定(用户需求分析)	(7)
2.2 分析与抽象、设计	(8)
2.3 编码、求解	(8)
2.4 系统组装、系统集成测试与调试	(8)
2.5 试运行、软件验收	(9)
2.6 软件发布	(9)
第 3 章 算法与数据结构抽象	(10)
3.1 信息与数据	(10)
3.2 数据、数据类型与数据结构	(10)
3.3 问题、算法与程序	(10)
3.4 算法与数据结构	(11)
3.5 抽象的重要性	(11)
第 4 章 实例 HIS 系统的分析与限定	(12)
4.1 问题的提出	(12)
4.2 分析与限定	(12)
4.3 用户需求	(13)
第 5 章 软件工程方法学与程序设计方法学	(15)
5.1 软件工程方法学和程序设计方法学	(15)

5.2 不同的程序设计方法	(15)
5.3 结构化程序设计方法	(15)
5.3.1 经典的结构化程序设计技术	(16)
5.3.2 扩展的结构化程序设计技术	(16)
5.3.3 结构化程序设计工具: 流程图	(16)
5.3.4 实例 HIS 的结构化设计	(17)
5.4 面向对象程序设计方法	(17)
5.5 结构化程序设计与面向对象程序设计的关系	(18)
第 6 章 面向对象软件开发的基本概念	(19)
6.1 面向对象思想	(19)
6.2 面向对象的基本概念	(19)
6.2.1 对象与类	(19)
6.2.2 对象的特性	(20)
6.2.3 对象与消息	(20)
6.2.4 面向对象的三大特性	(20)
6.2.5 面向对象术语	(22)
6.3 软件工程管理和文档的重要性	(22)
6.3.1 软件工程管理	(23)
6.3.2 文档	(23)
第 7 章 C++语言工具和 BCB 开发环境	(24)
7.1 C++简介	(24)
7.2 Borland C++Builder 简介	(24)
7.3 Borland C++编辑、编译、调试环境简介	(25)
7.3.1 Borland C++ 6 应用程序窗口	(25)
7.3.2 建立 C++源文件(.cpp)	(25)
7.3.3 程序调试窗口 (Debug Windows)	(27)
思考与练习	(29)

第 2 部分 C++程序设计

第 8 章 C++程序简介	(30)
8.1 C++环境基础	(30)
8.2 C++编程简介	(30)
8.3 内存与外存	(31)
8.4 C++程序结构	(31)
8.4.1 C++程序的组成	(32)
8.4.2 C++字符集、保留字与标识符	(33)
思考与练习	(34)
第 9 章 数据与基本数据类型	(35)
9.1 数据与对象	(35)
9.2 C++的基本数据类型	(35)
9.2.1 常量及其类型	(35)
9.2.2 符号常量、const 常量	(39)
9.2.3 变量及变量定义	(39)

9.2.4 基本类型数据的数值范围	(40)
9.3 基本数据类型的输入/输出	(41)
9.3.1 printf()和 scanf()函数	(42)
9.3.2 输入、输出流对象 cin、cout	(44)
9.4 C++的数据类型	(48)
9.5 sizeof 运算符	(48)
9.6 typedef (给类型起“别名”)	(49)
思考与练习	(49)
第 10 章 运算符、表达式与语句	(50)
10.1 C++运算符及运算符的优先级和结合性	(50)
10.2 表达式	(51)
10.2.1 算术运算符与算术表达式	(51)
10.2.2 赋值运算符与赋值表达式	(52)
10.2.3 自增、自减运算符	(53)
10.2.4 关系运算符与关系表达式	(54)
10.2.5 逻辑运算符与逻辑表达式	(55)
10.2.6 条件运算符与条件表达式	(56)
10.2.7 复合赋值运算符与复合赋值表达式	(56)
10.2.8 逗号运算符与逗号表达式	(56)
10.2.9 位运算符与表达式	(57)
10.3 提高表达式的可读性	(57)
10.4 语句	(57)
10.4.1 空语句	(58)
10.4.2 复合语句	(58)
10.4.3 表达式语句	(58)
10.4.4 函数调用语句	(58)
10.4.5 定义语句	(59)
10.4.6 声明语句	(59)
10.4.7 控制语句	(59)
10.5 基本类型之间的转换	(59)
10.5.1 基本类型数据之间的运算	(59)
10.5.2 隐式类型转换	(60)
10.5.3 显式类型转换(强制类型转换)	(60)
思考与练习	(61)
第 11 章 控制结构	(63)
11.1 选择结构	(63)
11.1.1 if 语句	(63)
11.1.2 switch 语句	(66)
11.2 循环结构	(68)
11.2.1 for 语句	(68)
11.2.2 while 语句	(69)
11.2.3 do-while 语句	(71)
11.3 goto 语句	(72)
11.4 break 语句	(73)

11.5 continue 语句.....	(74)
思考与练习	(74)
第 12 章 程序功能与函数	(77)
12.1 C++程序组件与函数.....	(77)
12.2 库函数.....	(77)
12.2.1 常用库函数.....	(78)
12.2.2 rand 函数.....	(79)
12.3 函数定义.....	(80)
12.3.1 函数定义与形参.....	(80)
12.3.2 无参函数定义.....	(80)
12.3.3 有参函数定义.....	(81)
12.4 函数原型.....	(81)
12.5 函数声明.....	(81)
12.6 函数重载.....	(82)
12.7 函数调用.....	(84)
12.7.1 函数调用与实参.....	(84)
12.7.2 带参数缺省值的函数.....	(84)
12.7.3 函数调用机制.....	(86)
12.7.4 重载函数的调用.....	(90)
12.8 全局对象与局部对象.....	(92)
12.8.1 全局对象.....	(92)
12.8.2 局部对象.....	(93)
12.9 递归.....	(94)
12.9.1 函数的递归调用.....	(94)
12.9.2 递归实例.....	(94)
12.10 内联函数.....	(96)
12.11 实例 HIS 的功能组成.....	(97)
12.11.1 HIS 的“角色”及其操作.....	(97)
12.11.2 HIS 的功能列表.....	(97)
思考与练习	(98)
第 13 章 存储类型与作用域	(101)
13.1 函数的存储类型.....	(101)
13.1.1 外部函数.....	(101)
13.1.2 静态函数.....	(102)
13.2 数据对象的存储类型.....	(102)
13.2.1 自动存储类型 (auto).....	(102)
13.2.2 寄存器存储类型 (register).....	(102)
13.2.3 外部存储类型 (extern).....	(103)
13.2.4 静态存储类型 (static).....	(104)
13.3 对象标识符的作用域.....	(104)
13.3.1 块 (局部) 作用域.....	(105)
13.3.2 函数原型作用域.....	(105)
13.3.3 函数作用域.....	(106)
13.3.4 文件作用域.....	(107)

13.4 对象的生存期	(108)
13.4.1 静态生存期	(108)
13.4.2 动态生存期	(108)
13.5 标识符的可见性	(109)
13.5.1 可见性	(109)
13.5.2 作用域运算符	(109)
思考与练习	(109)
第 14 章 数组	(111)
14.1 数组基本概念及定义	(111)
14.1.1 数组的数据抽象	(111)
14.1.2 数组的定义	(112)
14.1.3 数组元素	(112)
14.1.4 数组长度	(114)
14.2 数组初始化	(114)
14.2.1 一维数组的初始化	(114)
14.2.2 二维数组的初始化	(115)
14.2.3 关于数组初始化的说明	(115)
14.3 字符串和字符数组	(116)
14.3.1 字符串结束标志	(116)
14.3.2 字符数组的输入、输出	(116)
14.4 数组应用举例	(117)
14.4.1 递归与迭代	(117)
14.4.2 一维数组应用举例	(117)
14.4.3 二维数组应用举例	(121)
思考与练习	(122)
第 15 章 指针	(124)
15.1 指针对象的定义与初始化	(124)
15.1.1 指针变量的定义	(124)
15.1.2 指针变量的初始化	(124)
15.2 取址与间接访问	(125)
15.2.1 取址运算符与取址表达式	(125)
15.2.2 间接访问运算符与间接访问表达式	(125)
15.2.3 定义指针和使用指针的区别	(125)
15.3 指针概念	(126)
15.3.1 内存与内存地址	(126)
15.3.2 指针与内存地址	(127)
15.3.3 指针的类型	(128)
15.3.4 值调用与引用调用	(130)
15.3.5 指针的破坏性	(131)
15.3.6 NULL 指针	(132)
15.3.7 为什么要用指针	(132)
15.4 const 限定符与指针	(133)
15.4.1 指向变量的指针变量	(133)
15.4.2 指向常量的指针变量	(133)

15.4.3 指向变量的指针常量	(134)
15.4.4 指向常量的指针常量	(134)
15.5 指针运算	(134)
15.6 指针与数组	(135)
15.6.1 数组名与指向数组的指针变量	(135)
15.6.2 一维数组与指针	(135)
15.6.3 二维数组与指针	(136)
15.6.4 指针数组	(137)
15.7 指针与函数	(138)
15.7.1 指针参数	(138)
15.7.2 返回指针的函数	(139)
15.8 函数指针	(140)
15.8.1 函数指针的定义	(140)
15.8.2 函数指针的应用	(141)
15.9 堆内存分配与 new、delete 运算符	(142)
15.9.1 new、delete 运算符	(143)
15.9.2 关于动态分配的说明	(143)
15.10 例题：“数数”游戏	(144)
思考与练习	(145)
第 16 章 引用	(147)
16.1 “引用”与“引用”声明	(147)
16.1.1 声明“引用”	(147)
16.1.2 “引用”与“堆”对象	(149)
16.2 引用与函数	(149)
16.2.1 用“引用”参数按引用调用	(149)
16.2.2 用 const 限定形参	(150)
16.2.3 函数返回“引用”	(151)
16.2.4 函数调用作为左值	(152)
思考与练习	(153)
第 17 章 结构	(154)
17.1 构造新类型	(154)
17.1.1 声明结构类型	(154)
17.1.2 定义结构对象	(155)
17.1.3 结构对象的初始化	(155)
17.2 访问结构对象	(155)
17.2.1 同一结构类型的对象之间的赋值运算	(156)
17.2.2 访问结构成员	(156)
17.2.3 结构与数组	(156)
17.2.4 结构与函数	(157)
17.3 共用与枚举	(158)
17.3.1 共用	(158)
17.3.2 枚举	(159)
思考与练习	(160)
第 18 章 类与封装	(161)

18.1	软件方法发展的必然	(161)
18.1.1	“盲人摸象”与“面向对象”	(161)
18.1.2	数据抽象与信息隐藏	(162)
18.1.3	复合	(162)
18.1.4	接口与实现方法的分离	(162)
18.1.5	面向对象与面向过程	(162)
18.2	从结构到类	(162)
18.3	类的构成与声明	(162)
18.3.1	声明类	(162)
18.3.2	定义类的成员函数	(163)
18.3.3	类作用域	(164)
18.3.4	类的成员及其访问控制	(165)
18.4	定义对象与访问成员	(166)
18.4.1	类与对象	(166)
18.4.2	定义对象	(166)
18.4.3	访问对象成员	(166)
18.4.4	this 指针	(167)
18.4.5	指向类成员的指针	(167)
18.5	构造函数与析构函数	(169)
18.5.1	构造函数与初始化对象	(169)
18.5.2	析构函数	(174)
18.5.3	构造函数与析构函数的说明	(176)
18.6	对象的赋值运算与“赋值”成员函数	(176)
18.7	拷贝构造函数	(177)
18.7.1	拷贝构造函数的声明与定义	(177)
18.7.2	缺省的拷贝构造函数	(177)
18.7.3	何时自动调用拷贝构造函数	(179)
18.8	构造函数用于类型转换	(181)
18.9	类的 static 成员	(182)
18.9.1	静态数据成员、静态成员函数	(182)
18.9.2	静态成员的必要性	(183)
18.10	const 对象与 const 成员函数	(183)
18.10.1	const 对象	(183)
18.10.2	const 成员函数	(183)
18.11	实例分析	(184)
18.11.1	const 对象与 const 成员函数: 判断两直线是否平行	(184)
18.11.2	返回 private 成员的引用: 破坏类的封装性	(186)
18.12	嵌套类与局部类	(187)
18.12.1	嵌套类	(187)
18.12.2	局部类	(187)
18.13	实例 HIS 的对象与类	(188)
	思考与练习	(188)
第 19 章	继承	(191)
19.1	继承的概念	(191)

19.2	派生类的声明、定义	(191)
19.3	继承类别及其访问控制	(194)
19.3.1	公有继承	(194)
19.3.2	保护继承	(195)
19.3.3	私有继承	(195)
19.3.4	访问控制	(196)
19.4	派生类对象的构造与析构	(198)
19.5	继承关系中的指针	(200)
19.5.1	指针的指向	(200)
19.5.2	强制类型转换	(201)
19.6	在派生类中重定义基类成员函数	(201)
19.7	对象之间的关系	(202)
19.7.1	“是”关系	(202)
19.7.2	“有”关系	(202)
19.7.3	“知道”和“使用”关系	(202)
19.8	多重继承	(203)
19.9	虚拟继承	(204)
19.9.1	多继承的模糊性	(204)
19.9.2	虚基类与虚拟继承	(204)
19.10	复杂继承关系中派生类对象的构造顺序	(206)
19.11	实例 HIS 的类之间的关系	(208)
19.11.1	“是”(继承)关系	(209)
19.11.2	“有”关系	(209)
19.11.3	“知道”和“使用”关系	(209)
	思考与练习	(209)
第 20 章	多态	(212)
20.1	多态性的工作方式	(212)
20.1.1	静态多态性	(212)
20.1.2	动态多态性	(213)
20.2	虚函数与多态性	(213)
20.2.1	虚函数的引入	(213)
20.2.2	虚函数的声明和定义	(214)
20.2.3	多态编程	(214)
20.2.4	正确使用虚函数	(218)
20.3	虚析构函数	(221)
20.4	纯虚函数与抽象类	(222)
20.4.1	纯虚函数	(223)
20.4.2	抽象类	(223)
20.4.3	关于抽象类的说明	(223)
20.5	实例 HIS 的多态性	(224)
	思考与练习	(224)
第 21 章	友元	(225)
21.1	友元的声明和使用	(225)
21.1.1	友元函数	(225)

21.1.2 友元类.....	(226)
21.2 实例 HIS 中的友元.....	(227)
思考与练习.....	(228)
第 22 章 运算符重载	(229)
22.1 运算符重载的需要性.....	(229)
22.2 如何重载运算符.....	(229)
22.2.1 一元操作符重载的一般形式.....	(229)
22.2.2 二元操作符重载的一般形式.....	(231)
22.2.3 重载自增、自减运算符.....	(232)
22.2.4 重载赋值运算符.....	(234)
22.2.5 重载流插入运算符.....	(235)
22.3 运算符重载函数的返回.....	(236)
22.4 类型转换运算符.....	(236)
22.4.1 基本数据类型间的转换.....	(236)
22.4.2 构造类型与基本类型之间的转换.....	(236)
22.5 运算符重载的说明.....	(237)
22.6 例题:超长位数的整数的加、减运算.....	(238)
22.7 实例 HIS 中的操作符重载.....	(244)
思考与练习.....	(245)
第 23 章 输入/输出流与文件处理	(247)
23.1 I/O 流类.....	(247)
23.1.1 iostream 类库的头文件.....	(247)
23.1.2 输入/输出流类和对象.....	(247)
23.1.3 输入流.....	(248)
23.1.4 输出流.....	(249)
23.1.5 无格式输入/输出.....	(249)
23.2 格式控制.....	(250)
23.2.1 流操纵算子.....	(250)
23.2.2 用户自定义的流操纵算子.....	(251)
23.2.3 流格式状态及其设置.....	(251)
23.3 文件流.....	(252)
23.3.1 文件的组织.....	(253)
23.3.2 文件操作.....	(253)
23.4 串流类.....	(255)
23.5 实例 HIS 中的数据文件.....	(256)
思考与练习.....	(256)
第 24 章 异常处理	(257)
24.1 异常的基本概念.....	(257)
24.1.1 异常处理机制.....	(257)
24.1.2 异常指定.....	(260)
24.2 标准头文件<exception>.....	(260)
24.3 构造函数、析构函数的异常处理.....	(263)
24.4 new 异常处理.....	(264)
24.5 auto_ptr 模板类与动态内存分配.....	(265)

24.6	具有继承关系的类的异常处理	(266)
24.7	实例 HIS 中的异常处理	(267)
	思考与练习	(268)
第 25 章	模板技术	(269)
25.1	模板简介	(269)
25.2	函数模板和模板函数	(269)
25.2.1	函数模板的声明和定义	(270)
25.2.2	函数模板的使用举例	(270)
25.2.3	函数模板的重载	(272)
25.3	类模板和模板类	(273)
25.3.1	类模板的声明	(274)
25.3.2	类模板的使用举例	(274)
25.4	类模板与继承	(275)
25.5	类模板的友元	(275)
25.6	类模板的 static 成员	(276)
25.7	实例 HIS 中的模板应用	(276)
	思考与练习	(276)
第 26 章	常用数据结构	(278)
26.1	自引用类	(278)
26.2	动态数组	(278)
26.3	链	(283)
26.3.1	链的基本概念	(283)
26.3.2	链的基本操作	(285)
26.3.3	链应用举例	(287)
26.3.4	“链节点”类模板	(295)
26.3.5	“链”类模板	(296)
26.3.6	链模板使用举例	(305)
26.4	堆栈	(306)
26.4.1	栈的基本操作	(306)
26.4.2	从链模板派生栈模板	(306)
26.4.3	利用数组实现栈结构	(308)
26.5	队列	(310)
26.5.1	队列的基本操作	(310)
26.5.2	从链模板派生队列模板	(310)
26.5.3	利用数组实现环形队列	(311)
26.6	树	(314)
26.6.1	树的基本概念	(314)
26.6.2	二叉树	(315)
26.6.3	任何非二叉树转换为二叉树	(315)
26.6.4	二叉树的遍历	(316)
26.6.5	树节点和“树”类模板	(317)
26.6.6	计算算术表达式	(321)
26.7	图	(321)
26.7.1	图的基本概念	(322)

26.7.2 图结构的实现.....	(322)
26.8 实例 HIS 中的数据结构.....	(323)
思考与练习.....	(324)
第 27 章 排序与查找基本算法.....	(325)
27.1 排序.....	(325)
27.1.1 插入排序.....	(325)
27.1.2 选择排序.....	(326)
27.1.3 交换排序.....	(327)
27.2 查找.....	(330)
27.2.1 线性查找.....	(330)
27.2.2 二分查找.....	(330)
27.3 实例 HIS 中的排序与查找.....	(331)
思考与练习.....	(332)
第 28 章 标准模板库 STL.....	(333)
28.1 容器简介.....	(333)
28.1.1 顺序容器.....	(334)
28.1.2 关联容器.....	(338)
28.1.3 容器适配器.....	(340)
28.2 迭代器简介.....	(342)
28.3 算法简介.....	(344)
第 29 章 预处理器、ANSI/ISO C++ 补充.....	(346)
29.1 预处理器与预处理指令.....	(346)
29.1.1 宏定义.....	(346)
29.1.2 文件包含.....	(347)
29.1.3 条件编译.....	(347)
29.1.4 #error 和#pragma.....	(348)
29.1.5 行号.....	(348)
29.1.6 预定义的符号常量.....	(348)
29.1.7 断言 (宏 assert).....	(349)
29.2 ANSI/ISO C++ 补充.....	(349)
29.2.1 类型转换运算符.....	(349)
29.2.2 mutable 存储类型.....	(351)
29.2.3 名空间.....	(351)
29.2.4 explicit 构造函数.....	(352)
29.2.5 运算符关键字.....	(352)
29.2.6 程序运行时的类型信息 (RTTI).....	(353)
第 30 章 C++ 编码标准.....	(354)
30.1 注释.....	(354)
30.1.1 注释的原则.....	(354)
30.1.2 函数注释.....	(355)
30.1.3 “战略性”注释、“战术性”注释.....	(355)
30.2 命名原则.....	(356)
30.3 函数.....	(357)
30.3.1 函数声明、定义.....	(357)

30.3.2	函数参数	(358)
30.3.3	函数返回值	(358)
30.3.4	函数重载	(358)
30.4	表达式和流程控制	(359)
30.4.1	表达式使用原则	(359)
30.4.2	goto 语句	(360)
30.4.3	循环控制	(360)
30.5	宏	(360)
30.6	代码格式	(360)
30.7	初始化和清除	(362)
30.7.1	初始化基本原则	(362)
30.7.2	构造函数、拷贝构造函数	(362)
30.7.3	赋值成员函数	(362)
30.7.4	析构函数	(362)
30.7.5	内存分配和释放	(362)
30.8	类型的使用、转换	(363)
30.8.1	类型使用原则	(363)
30.8.2	类型转换原则	(363)
30.8.3	类的设计和使用	(363)
30.9	继承	(364)
30.9.1	继承的基本原则	(364)
30.9.2	多重继承	(365)
30.9.3	继承和模板的区别原则	(365)
30.10	操作符重载	(365)
30.11	友元	(366)
30.12	模板	(366)
30.12.1	模板使用的基本原则	(366)
30.12.2	模板编译的特殊性	(366)
30.13	异常处理	(366)
30.13.1	异常处理的基本原则	(366)
30.13.2	模板类型的异常	(367)
30.14	文件和目录	(367)
30.14.1	文件名	(367)
30.14.2	头文件	(368)
30.14.3	文件、目录的组织	(368)
30.15	编译	(368)
30.15.1	条件编译	(368)
30.15.2	编译原则	(369)
30.16	兼容性和性能	(369)
30.16.1	考虑兼容性	(369)
30.16.2	考虑性能要求	(370)
第 31 章	实例 HIS 的实现	(371)
31.1	数据文件	(371)
31.1.1	用户密码文件	(371)