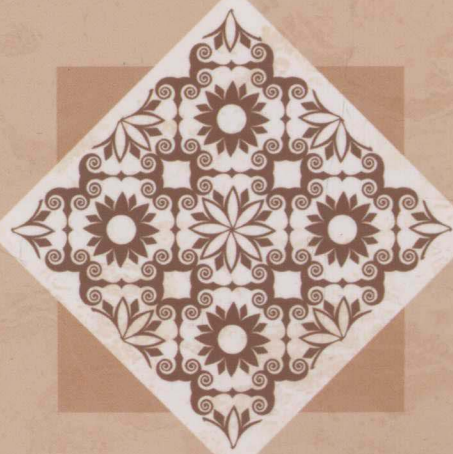


C++语言程序设计

王立柱 编著

THE C++
PROGRAMMING
LANGUAGE



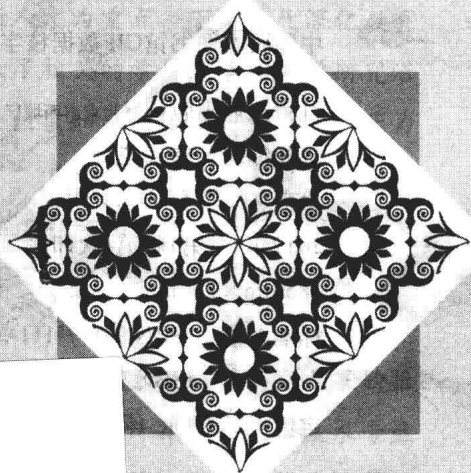
机械工业出版社
China Machine Press

高等院校计算机专业人才培养规划教材

C++语言程序设计

王立柱 编著

*T*HE C++
PROGRAMMING
LANGUAGE



机械工业出版社
China Machine Press

本书在C语言基础上介绍C++语言基础知识和程序设计方法。全书共分14章，内容包括：C语言回顾、C顺序表、C单向链表、C结构串、从C到C++的基本内容、从顺序表到顺序表类、从String结构到String类、Date类、继承和动态绑定、函数模板和类模板、链表类模板和适配器、C++的I/O流库、C++综合设计实例以及命名空间。

本书可以作为高等院校计算机及相关专业本科生的C++程序设计课程的教材，也可以作为计算机编程爱好者的自学教材和参考书。

封底无防伪标均为盗版

版权所有，侵权必究

本书法律顾问 北京市展达律师事务所

图书在版编目（CIP）数据

C++语言程序设计/王立柱编著. —北京：机械工业出版社，2012.7
（高等院校计算机专业人才能力培养规划教材）

ISBN 978-7-111-38740-4

I. C… II. 王… 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（2012）第124687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：李 荣

北京京师印务有限公司印刷

2012年7月第1版第1次印刷

185mm×260mm · 12.25印张

标准书号：ISBN 978-7-111-38740-4

ISBN 978-7-89433-492-3（光盘）

定价：29.00元（附光盘）

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



机械工业出版社华章公司多年来以“全球采集内容，服务中国教育”为己任，致力于引进国际知名大学广泛采用的计算机、电子工程和数学方面的经典教材，出版了一大批在计算机科学界享誉盛名的专家名著与名校教材，其中包括Donald E.Knuth、Alfred V. Aho、Jim Gray、Jeffery D. Ullman等名家的一批经典作品。这些作品为我国计算机教育及科研事业的发展起到了积极的推动作用。

近年来，我们一直关注国内计算机专业教育的发展和改革并大力支持、参与相关的教学研究活动。2006年，教育部高等学校计算机科学与技术专业教学指导分委员会在对我国计算机专业教育现状和社会对人才的需求进行研究的基础上，发布了《高等学校计算机科学与技术专业发展战略研究报告暨专业规范（试行）》（以下简称《规范》）。为配合《规范》的实施和推广，我们出版了“面向计算机科学与技术专业规范系列教材”。这套教材的推出，对宣传《规范》提出的“按培养规格分类”的理念、推进高校学科建设起到了一定的促进作用。

2007年，教育部下发了《关于进一步深化本科教学改革全面提高教学质量的若干意见》，强调高等教育以育人为本，以学生为主体，坚持以培养创新人才为重点，下大力气深化教育教学改革。在“质量工程”的思想指导下，各高校纷纷开展了相关的学科改革和教学研究活动。高等学校计算机科学与技术专业的教育开始从过去单纯注重知识的传授向注重学科能力的培养转型。2008年年底，教育部高等学校计算机科学与技术专业教学指导分委员会成立了“高等学校计算机科学与技术专业人才专业能力构成与培养”项目研究小组，研究小组由蒋宗礼教授（组长）、王志英教授、岳丽华教授、陈明教授和张钢教授组成，研究计算机专业人基本能力的构成和在计算机专业的主干课程中如何培养这些专业能力。

为配合“高等学校计算机科学与技术专业人才专业能力构成与培养”专项研究成果的推广，满足高校从知识传授向能力培养转型的需求，在教育部高等学校计算机科学与技术专业教学指导分委员会专家及国内众多知名高校专家的指导下，我们策划了这套“高等院校计算机专业人能力培养规划教材”。这套教材以专项研究的成果为核心，围绕计算机专业本科生应具有的能力组织教材体系。本套教材的作者长期从事教学和科研工作，他们将自己在本科生能力培养方面的经验和心得融入教材的编写中，力图通过理论教学及实践训练，达到提升本科生专业能力的目标。希望这些有益的尝试能对推动国内计算机专业学生的能力培养起到

积极的促进作用。

华章作为专业的出版团队，长久以来遵循着“分享、专业、创新”的价值观，实践着“国际视野、专业出版、教育为本、科学管理”的出版方针。这套教材的出版，是我们以教学研究指导出版的成功范例，我们将以严谨的治学态度以及全面服务的专业出版精神，与高等院校的老师们携手，为中国的高等教育事业走向国际化而努力。





编委会

主任委员：蒋宗礼（北京工业大学）

委 员：（以姓氏拼音为序）

陈道蓄（南京大学）

陈 明（中国石油大学）

胡事民（清华大学）

孙茂松（清华大学）

王 珊（中国人民大学）

王志英（国防科学技术大学）

吴功宜（南开大学）

岳丽华（中国科学技术大学）

张 钢（天津大学）

郑人杰（清华大学）

联络人：朱 劼 姚 蕾



丛书序言

作为我国规模最大的理工科专业，计算机本科专业为国家的建设培养了大批人才。2006年，教育部计算机科学与技术专业教学指导委员会发布了《高等学校计算机科学与技术专业发展战略研究报告暨专业规范（试行）》（以下简称《规范》），提出了以“按培养规格分类”为核心思想的专业发展建议，把计算机专业人才划分为研究型、工程型、应用型3个类型。在《规范》的方针指导下，培养合格的计算机本科人才。

教育包括知识、能力、素质三个方面，专业教育不仅要重视知识的传授，更应突出专业能力的培养，实施能力导向的教育。如何以知识为载体实现能力的培养和素质的提高，特别是实现专业能力和素质的提高是非常重要的。对计算机专业本科教育而言，要想实现能力导向的教育，首先要分析专业能力的构成并考虑如何将其培养落实到教学实践中。为此，教育部高等学校计算机科学与技术专业教学指导分委员会开展了计算机科学与技术专业人才培养能力（简称计算机专业能力）的培养研究。该项研究明确了计算机专业本科人才应具有的4大基本能力——计算思维能力、算法设计与分析能力、程序设计与实现能力、系统能力，并将这四大基本能力分解为82个能力点，探讨如何面对不同类型学生的教育需求，在教学活动中进行落实。

为体现研究成果在教学活动中的实现，我们根据《高等学校计算机科学与技术专业人才培养能力构成与培养》，出版了这套教材。本套教材面向高等院校从知识传授向能力培养转型的需求，在内容的选择、体系安排和教学方法上按照专业能力培养的需要进行了探索。其主要特点有：

（1）以教学研究为先导。本套教材以计算机专业能力专项研究成果为基础，体现了先进的教育理念和教学方法，内容选择、知识深度、结构安排更加符合计算机专业教育的需求。

（2）落实能力培养的思想，同时满足课程的要求。本套教材不仅关注知识点的讲授，还凸显能力培养的要求，将能力的培养分解到各门课程的各个知识点的讲授中。

（3）力求贴近教学实际。作者均长期从事实际教学工作且对专业能力培养具有一定研究，教材编写注重科学组织内容、合理安排体系、便于教学实施，更具操作性。

（4）构建立体化教材。为了方便教师的教学活动，配合主教材开发配套的实验教材、教师参考书、学生辅导书、电子课件等教辅资源。

本套丛书的出版是在配合计算机专业能力的培养和落实方面的初步尝试，我们衷心希望本套教材的出版能起到抛砖引玉的作用，也希望广大教育工作者加入到能力培养的研究和实践中来，并对相关的教材建设提出自己的宝贵意见。



平凡中的非凡

辩证逻辑和旧的纯粹的形式逻辑相反，不像后者满足于把各种思维运动形式，即各种不同的判断和推理的形式列举出来和毫无关联地排列起来。相反地，辩证逻辑由此及彼地推出这些形式，不把它们互相平列起来，而使它们互相隶属，从低级形式发展出高级形式。

——恩格斯

一、“人体解剖”的方法

C++的主要内容包括两部分，一部分是C和C++的公共子集，另一部分是C++直接支持的高级部分。公共子集主要包括基本类型、指针、数组、结构、函数和C风格字符串。高级部分则包括类、继承和动态绑定、函数模板、类模板和STL（String、Vector和List等）。从公共子集到高级部分，蕴含着程序设计风格从过程化程序设计到数据抽象，从面向对象程序设计到泛型程序设计的发展。

C++是从C扩展类型开始的，它的每一步演化和发展都是由这个扩展中的实际问题所引起的。因此，Stroustrup指出：“C++支持一种逐步推进的学习方式。你学习一个新语言的方式依赖于你已经知道些什么，还依赖于你的学习目的。”

本书的目的是让读者掌握C++的演化和发展规律，“已经知道”的是C，“逐步推进的学习方式”是从自定义类型的C描述到C++描述。本书的特点是：分析各种概念，注意概念的隶属关系，建立概念的连锁，不让其中缺少一个环节，而且使整个连锁有一组大家熟悉的典型程序设计作根据。这是一种逻辑方式，也是马克思提倡的“人体解剖”方式。我们知道，事物的本质只有在它发展到一定阶段的时候才能清晰地显露出来，于是，这一清晰的本质就是一把钥匙，借此，我们可以对整个发展过程重新做出合乎逻辑的描述，展示出其内在规律性。黑格尔形容这种方式是“密涅瓦的猫头鹰从黄昏起飞”。在马克思看来，这种逻辑方式不过是摆脱历史的形式以及起扰乱作用的偶然性，按照现实的历史过程本身的规律进行修正，因而具有逻辑上前后一贯的形式，使每一个要素可以在它完全成熟并且具有典型性的发展点上加以考察。如果把C看做不成熟的“猴体”，把C++看做成熟的“人体”，那么马克思认为“人体解剖对于猴类解剖是一把钥匙”。

Bruce Eckel在他的《C++编程思想》一书中就探索了这种逻辑方式：他先用C语言实现一个顺序表，然后再转换为C++类。顺序表便是那种“完全成熟并具有典型性的发展点”。显然，这种逻辑方式下的C已不是传统的、由不了解C++的人来讲授的、只为应试等级考试的C，而是从C++的高度来修正的、按照从C到C++的发展规律和人的认知规律来描述的C。

本书把STL的String、Vector和List作为C++发展中的成熟点，实现了从C到C++的无缝连接，其整体框架如下：

- 1) 用C描述顺序表、链表和结构串。
- 2) 具体分析用C描述顺序表和结构串时暴露出来的C局限。
- 3) 针对性地引入C++基本概念来克服上述局限性。
- 4) 把C描述顺序表和结构串转化为C++描述。
- 5) 实现String、Vector和List。

二、本书的脉络

为了阐述从C到C++的发展，并使其具有逻辑上前后一贯的形式，本书以程序设计的基本矛盾为主线，把解决实际问题作为动机，用程序设计的典型实例来贯穿。

什么是程序？程序是在某种存储模式上实现的算法，即程序=存储模式+算法。所谓存储模式是指数据和对数据的基本运算或基本操作的存在形式。所谓算法是指可以用基本运算或基本操作来表示的问题求解的有穷序列。算法好比一台机器，构造再复杂，也不过是简单机械动作的重复。最先提出这个思想的人是图灵，他认为，应该用机器保留一些最简单的操作，然后将一个复杂的计算分解为这些操作。冯·诺依曼用固化在硬件中的机器指令表示简单操作。这些机器指令的集合构成的系统成为第一个程序语言——机器语言。从此，存储模式都用程序语言来表示，程序也就成为用某种程序语言实现的算法。例如，机器语言程序、C语言程序、C++语言程序、Java语言程序等。

什么是程序设计？程序设计也是程序设计方法，它是关于存储和算法的关系和发展的科学。“纵观短暂的计算机发展史，存储和算法这两个方面一直存在，发展演化的只是它们之间的关系，这个关系就是程序设计方法”。

存储和算法是一对矛盾。存储为了算法，算法离不开存储。但是存储相对稳定，制约着算法的自由发展，而算法则越来越复杂，要求存储不断改进。矛盾推动了程序设计的持续发展。我们的理论正是对这种发展过程的阐明。而一切发展，不管其内容如何，都可以看作一系列不同的发展阶段，它们以一个否定另一个的方式彼此联系着。这里的否定不是消灭，而是扬弃，是后者克服前者中消极的东西，保留和继承积极的东西，并把它发展到新的阶段。

本书的内容安排如下（见下图）：

第1章引入指针和数组处理一组逻辑上具有前后顺序的数据，以弥补基本固有类型的不足；引入函数处理更复杂的数据，它是对运算符的推广；字符串是综合应用指针和函数而创立的第一个带有可自定义基本运算函数的存储模式即自定义类型。

第2章创建顺序表，以弥补数组的不足。

第3章创建单向链表，以弥补顺序表的不足。

第4章创建结构串，以弥补字符串的不足。

第5章引入C++基本知识，以解决顺序表和结构串中的C语言局限性。

第6章把顺序表转化为顺序表类。

第7章把结构串转化为类串。

第8章把Date结构转化为Date类。

第9章介绍类与类的继承发展的技术——继承和动态绑定。

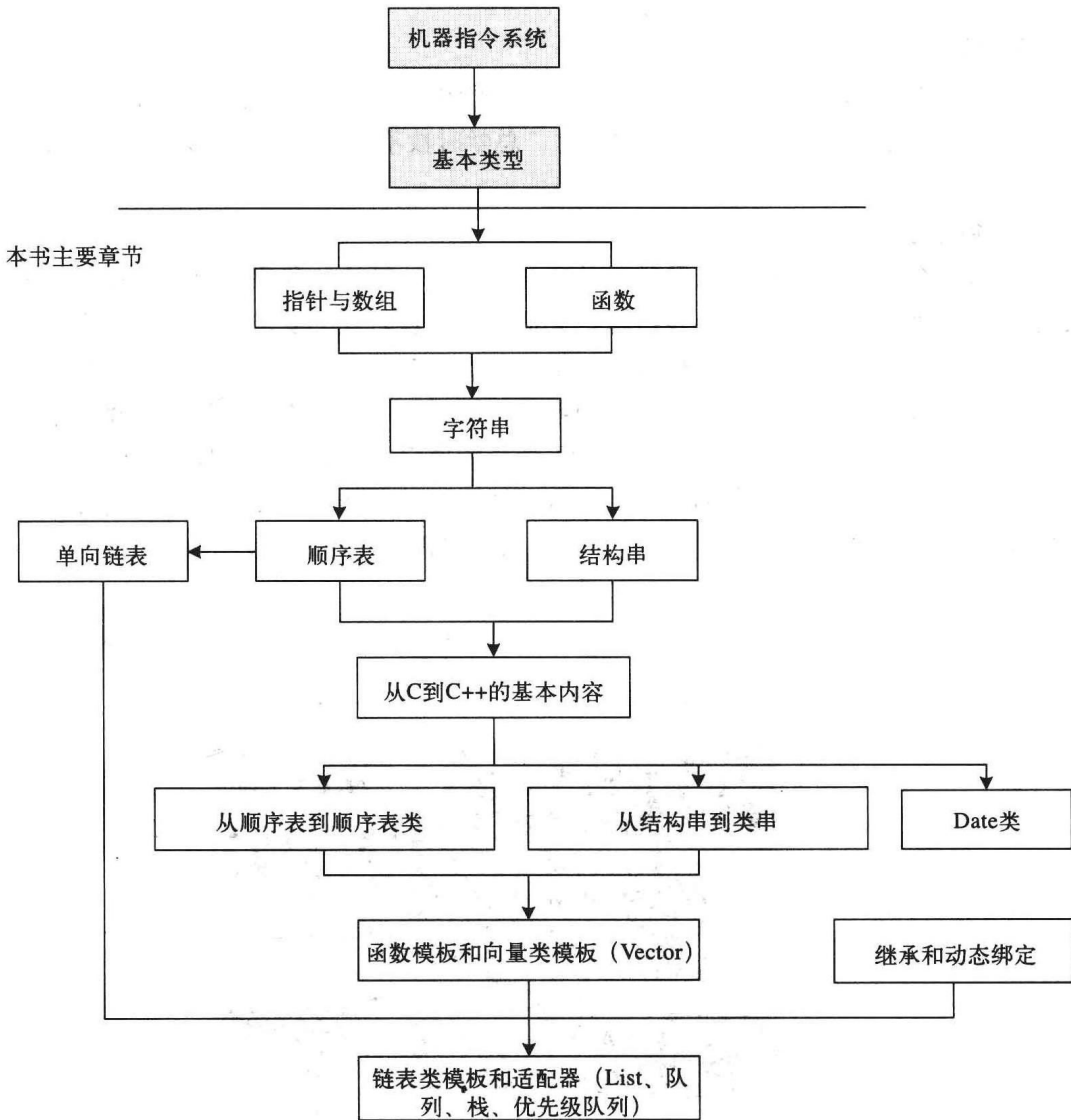
第10章创建向量类模板Vector，改进顺序表类。

第11章创建链表类模板改进单向链表。

第12章把C文件转化为C++文件。

第13章综合运用上述知识，编写中缀表达式求值和事件驱动模拟程序。

第14章介绍防止命名空间污染的方法——采用命名空间。



三、教学经验和体会——平凡中的非凡

1. 注重规律和方法

C++的出现和发展不是聪明人苦思冥想的结果，而是程序设计的发展过程使然，我们的逻辑方法正是以这个过程为研究对象。每一个学生，只要经历这个过程，都可以深切地感受到C++产生的必然性，从而认识到积累与创新的正确关系，这对培养学生的学习兴趣、树立学生的学习自信心、提高学生的科学态度和素质都是至关重要的。

这种方法在中学是学不到的。作为大学教师，不要仅仅从实用的角度或仅仅为了应试来讲授程序设计，这会使课程支离破碎，只剩雕虫小技。相反，要讲逻辑，注重整体，前后一致地讲授知识，促进学生从中学到大学的转变，使他们时刻感觉到自己是在真正地上大学，有做大学生的自豪感，这是大学教师的义务、责任和价值。

我们之所以欣赏计算机的美，不只是因为它具有令人感叹的人类智慧的美，更是因为计算机是建立在程序设计之上，而学生通过比较全面而深刻地认识程序设计的发展规律，能够挖掘出自己的潜能，最终欣赏到自身的美。

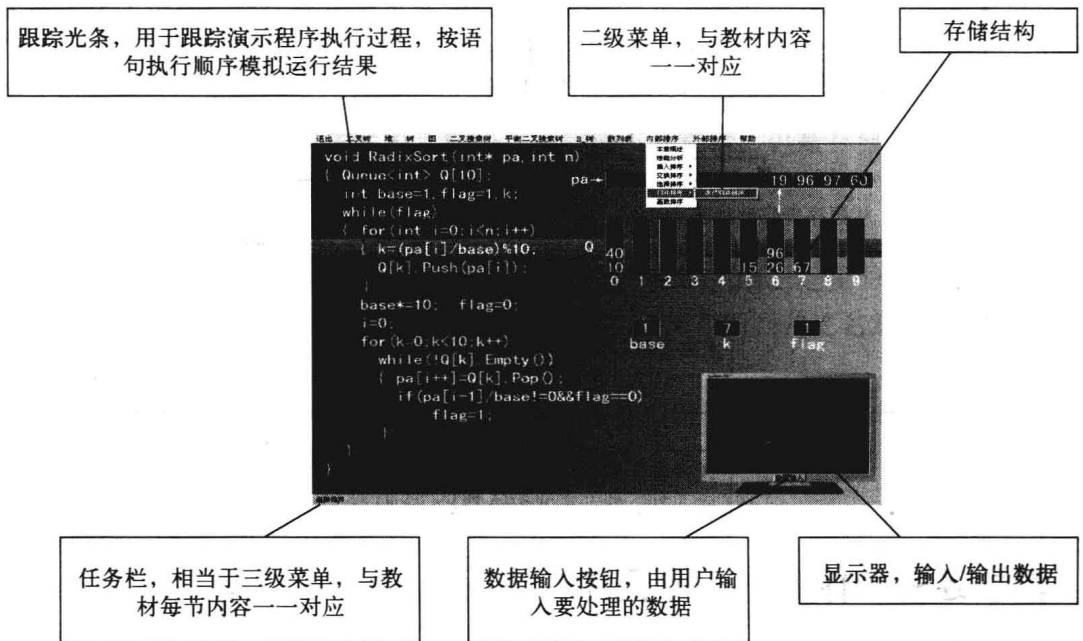
2. 因材施教

本书如果作为计算机专业教材，总学时数不少于80，那么每一章都应该讲。

本书如果作为非计算机专业的必修课教材，总学时数不少于60，那么第12章可以简单讲，第13章和第14章可以不讲，仅要求自学。

3. 多媒体教学

本书有配套的多媒体软件，既助学又助教，使结构、算法、代码、运行过程和结果同时展现、数据自由输入、运行过程步步跟踪。它适应了传媒时代的要求，扩展了学生自主学习空间，解决了算法复杂抽象、学生难以理解、教师难以讲解的难题，加强了师生的切磋交流，为教师前后横向对比、全面具体、综合系统、深刻展示思想方法地授课搭建了平台。本书配套的光盘中包括学生用助学软件、教师用教学软件以及源代码，方便学生学习及教师教学。另外，该多媒体软件是基于Authorware应用软件的编程特性开发的，本身就要求开发者有很好的程序设计基础和综合能力，它时时彰显着学习程序设计的意义（如下图所示）。



4. 考试

考试要突出整体性，树立学生的全局观和发展观，切忌偏题、编造题。本书章后的习题都是考试的参考题型。

四、致谢

感谢武警后勤学院贾冬梅老师，她参与了本书的校对，不仅指出很多文字上的错误，而且提出不少建设性的意见。

感谢天津师范大学的刘志红老师，她不仅是课程改革团队的主要成员、教学课件的制作者、课程的主讲教师，而且十年来，她的学生评教成绩一直很优秀。我们所遵循的逻辑方式不仅经过她的教学实践证明是普遍受学生欢迎的，而且也通过她的实践得到不断完善。

教学建议

教学章节	教学要求	课时 (授课课时+上机课时)
第1章 C语言回顾	了解指针和数组产生的原因 理解指针和间接引用的关系 理解指针和数组的相互依存关系 理解数组变量和数组指针的关系 理解字符串常量和字符串指针的关系	4+4
第2章 C顺序表	了解数组作为存储模式的局限性 掌握顺序表方法	2+2
第3章 C单向链表	了解顺序表作为存储模式的局限性 掌握单向链表方法	2+2
第4章 C结构串	了解C结构串的局限性 掌握结构串的实现方法	4+4
第5章 从C到C++的基本内容	理解C语言对用户类型的局限性 掌握C++的基本内容	2+2
第6章 从顺序表到顺序表类	掌握从顺序表到顺序表类的转化过程和相应的概念	2+2
第7章 从String结构到String类	掌握从结构串到类串的转化过程和相应的概念	4+4
第8章 Date类	掌握Date类的声明和实现方法	2+2
第9章 继承和动态绑定	掌握继承和动态绑定的概念和方法	2+2
第10章 函数模板和类模板	掌握向量类模板Vector	4+4
第11章 链表类模板和适配器	掌握或了解链表类模板声明和实现方法 掌握队列和栈声明及实现方法	4+4
第12章 C++的I/O流库	了解I/O流库的基本概念和方法	2+2
第13章 C++综合设计实例	熟悉中缀表达式求值和事件驱动模拟的方法	2+2
第14章 命名空间	了解命名空间的概念	2+2
总课时	第1章~第14章建议课时	38+38

目 录



出版者的话	
编委会	
丛书序言	
前言	
教学建议	
第1章 C语言回顾	1
1.1 指针和数组	1
1.1.1 指针	1
1.1.2 数组	7
1.2 函数	10
1.2.1 函数定义与调用	10
1.2.2 值传递和指针传递	13
1.2.3 返回值和指针传递	14
1.2.4 函数声明(原型)	14
1.3 const限定修饰符	15
1.3.1 const型变量	15
1.3.2 间接const型指针和参数	16
1.3.3 const型指针	17
1.3.4 全const型指针	17
1.4 字符串	18
1.4.1 字符串常量和字符串赋值	18
1.4.2 字符串基本操作函数声明	20
1.5 条件编译命令和文件包含命令	21
1.5.1 条件编译命令	21
1.5.2 文件包含命令	22
1.5.3 应用举例	23
习题	25
第2章 C顺序表	26
2.1 顺序表	26
2.2 顺序表基本运算的实现	28
2.3 删除顺序表重复数据	32
2.4 数据抽象和准封装	33
习题	35
第3章 C单向链表	36
3.1 单向链表	36
3.2 单向链表基本运算的实现	38
3.3 逆置	42
习题	44
第4章 C结构串	45
4.1 String结构	45
4.2 String结构基本运算的实现	47
4.2.1 准构造函数和准析构函数	47
4.2.2 转换赋值	48
4.2.3 复制赋值	49
4.2.4 串连接	49
4.2.5 子串插入	50
4.2.6 子串删除	52
4.2.7 求子串	54
4.2.8 其他	55
4.3 模式匹配	56
习题	58
第5章 从C到C++的基本内容	59
5.1 C语言本身的问题	59
5.2 变量和类型	61
5.3 输入/输出	63
5.4 内联函数	65
5.5 运算符重载	66
5.6 函数重载	67
5.7 引用型	69
5.7.1 概念的由来	69
5.7.2 引用型的定义和使用	71
5.8 提取符和插入符重载	74
5.9 默认参数	75
5.10 new和delete运算符	76
习题	76
第6章 从顺序表到顺序表类	78
6.1 顺序表和顺序表类	78
6.2 顺序表类基本运算的实现	82

6.3 增加的成员函数	82	9.4 虚析构函数	123
6.3.1 复制构造函数	83	9.5 纯虚函数和抽象类	124
6.3.2 复制赋值运算符重载	83	习题	126
6.3.3 下标运算符重载	85	第10章 函数模板和类模板	128
6.4 构造函数与初始化	85	10.1 函数模板	128
6.5 new和delete运算符与构造和析构	86	10.2 顺序表类模板	130
6.6 类定义	86	10.3 向量类模板Vector	132
习题	87	10.3.1 Vector定义	132
第7章 从String结构到String类	88	10.3.2 通用算法和迭代器	135
7.1 String类的声明	88	10.3.3 Vector的插入和删除函数	138
7.2 String类成员函数的实现	90	10.3.4 求素数	139
7.2.1 构造和析构	90	10.4 函数对象	140
7.2.2 成员赋值运算符	92	10.5 深入讨论——函数模板实例化中的问题	141
7.2.3 成员转换	94	习题	142
7.2.4 串连接	95	第11章 链表类模板和适配器	144
7.2.5 关系运算	98	11.1 链表类模板List	144
7.2.6 求子串	98	11.2 适配器	151
7.2.7 子串插入	99	习题	152
7.2.8 子串删除	100	第12章 C++的I/O流库	153
7.2.9 下标运算符	100	12.1 格式化输入/输出	154
7.2.10 字符查找	102	12.1.1 设置流的格式化标志	154
7.2.11 输入/输出	102	12.1.2 格式化输出函数	155
7.3 深入讨论	104	12.1.3 操作算子	156
7.3.1 转换赋值运算符函数的替代	104	12.2 文件使用方式	158
7.3.2 成员函数“类串+C串”的替代	104	12.3 字符读写函数	159
7.3.3 explicit修饰符	105	12.4 字符串读写函数	160
习题	106	12.5 无格式读写函数	160
第8章 Date类	107	12.6 随机访问	162
8.1 Date类的声明	107	12.7 文件错误处理	163
8.2 Data类基本运算的实现	110	习题	163
8.3 深入讨论	113	第13章 C++综合设计实例	164
8.3.1 转换赋值成员函数的替代	113	13.1 中缀表达式求值	164
8.3.2 静态数据成员和静态函数成员	113	13.2 事件驱动模拟	168
习题	115	习题	175
第9章 继承和动态绑定	116	第14章 命名空间	176
9.1 参数初始化表	116	14.1 命名空间的定义	176
9.2 继承和受保护成员	118	14.2 using namespace语句	177
9.2.1 继承	118	14.3 命名空间的成员	178
9.2.2 受保护成员	120	14.4 命名空间的别名	179
9.3 多态性和虚函数	120	习题	180
		参考文献	181

C语言回顾

事物是互相作用着的，并且在大多数情形下，正是忘记了这种多方面的运动和相互作用，阻碍我们的自然科学家去看清最简单的事物。

——恩格斯

1.1 指针和数组

我们常常要处理一组逻辑上具有前后顺序的数据，例如一组记录、字符串等。它们的基本运算有插入、删除、查找、交换、修改等，典型算法有排序、求和、求最大/最小值等。这些基本运算和典型算法有一个基本步骤：按照一定顺序访问所有数据（我们称之为遍历），这就要求存储时既存储数据，又存储数据的前后顺序。显然，直接引用方式解决不了这个问题。想想看，对数以万计的数据，设计数以万计的变量名称来直接引用，这本身就是不切实际的，更不用说用变量名称来表示数据的前后顺序。

因此，现在需要引入一种联系型的数据，用于表示数据的前后顺序。一个简单直观的设计方案是：让数据空间依次相邻，用空间地址的顺序表示数据的前后顺序。如果这些数据是同一类型，那么它们的空间大小便是一样的，这就很容易由第一个数据空间的地址计算出其余所有数据空间的地址，然后利用空间地址来引用空间中的数据，就可以实现遍历。实现这个设计方案的方法就是使用指针和数组。

1.1.1 指针

1. 指针和间接引用

要理解指针，就要理解间接引用。所谓间接引用，是指利用空间地址（而非空间名）来访问该空间数据的方式，这时，该空间称为间接引用空间。间接引用也称解引用。实现间接引用的方法是使用指针。指针是这样一种类型的变量，其合法值应该是0或另一个空间的地址。当指针的合法值非0时，它把两个空间紧密联系在一起，一个是指针空间，另一个是间接引用空间。指针空间存储的是间接引用空间的地址。地址是钥匙，指针空间好比是存放钥匙的空间。间接引用空间是用这把钥匙可以打开的房间。一般我们用有向线段来表示这两个空间的联系，起点是指针空间，终点是间接引用空间，如图1-1所示。



图1-1 指针空间和间接引用空间

间接引用空间也称为指针的间接引用对象。

指针的定义格式为：

类型标识符* 指针名；

其中类型标识符表示间接引用空间的类型，称为指针的**基类型**；“*”是间接引用运算符；“类型标识符*”表示指针空间的类型，即指针的类型；“*指针名”是间接引用表达式，表示间接引用空间，相当于间接引用空间的名称。例如：

```
int* np;           //定义一个基类型为整型的指针变量
char* cp;         //定义一个基类型为字符型的指针变量
double* xp;      //定义一个基类型为双浮点型的指针变量
```

指针类型是复合类型。所谓**复合类型**就是依赖其他类型而存在的类型。指针类型依赖的类型是间接引用空间的类型，即指针的基类型。这里所谓的依赖，是指一个非0指针只有在它指向的间接引用空间是编译器分配给程序的空间时才是有意义的，否则，这个指针是“野指针”。用野指针来间接引用是非法的。

一般把指针的基类型直接称为指针类型。例如，当人们说某指针是整型指针时，实际上是指这个指针的基类型是整型的。

2. 指针间接引用对象的生成

(1) 利用动态分配函数生成指针的间接引用对象

动态分配函数有malloc和calloc，下面举例说明它们的使用方法：

```
int* np=(int*)malloc(sizeof(int));           //建立整型间接引用空间，没有初值
char* cp=(char*)malloc(sizeof(char));       //建立字符型间接引用空间，没有初值
double* xp=(double*)calloc(1, sizeof(double)); //建立双浮点型间接引用空间，初值为0
```

或者

```
int* np;           np=(int*)malloc(sizeof(int));
char* cp;         cp=(char*)malloc(sizeof(char));
double* xp;      xp=(double*)calloc(1, sizeof(double));
```

函数malloc和calloc的返回值是void型指针，需要通过强制类型转换，转换为指针的基类型。

动态分配的空间没有名称，只能间接引用，如图1-2所示。

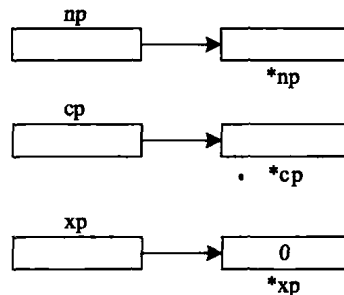


图1-2 动态分配空间示意图

这些空间占用的是被称为堆的动态存储区。如果动态分配没有成功，返回值为NULL（指针0值），这时应终止程序。