



普通高等教育规划教材

程序设计(C++版)

廖 雷 主编



机械工业出版社
CHINA MACHINE PRESS



普通高等教育规划教材

程序设计（C++版）

主 编 廖 雷

副主编 杨泽凡

参 编 杜 青 胡文瑜

主 审 刘友华



机械工业出版社

本书是计算机科学与技术专业的程序设计课程教材,以 C++作为讲解的语言,内容包括 C++概念、数据类型、运算符、表达式、分支和循环语句、函数、数组、指针、引用、结构、类和对象、继承和派生、友元函数、虚函数、运算符重载、I/O 流、模板等。教材以 Visual C++作为上机环境。

编者力求体现概念准确、编排合理、循序渐进、讲解通俗的特点,妥善处理好过程化程序设计和面向对象程序设计的关系,读者可以不具备其他高级语言和程序设计的基础知识。针对应用型本科人才的培养特点,在注重基础性、学科完整性的基础上,强调实践性、应用性,因此本书有专门章节介绍上机步骤,提供了较多的习题和上机实习题。

本书可作为本专科计算机专业的教材,也可作为其他相关专业和计算机培训的教学用书,还可供程序开发人员和自学者参考。

图书在版编目(CIP)数据

程序设计: C++版/廖雷主编. —北京:机械工业出版社,

2004. 7

普通高等教育规划教材

ISBN 7-111-14599-2

I. 程... II. 廖... 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(2004)第 052294 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

责任编辑:王小东

封面设计:饶 薇 责任印制:李 妍

北京蓝海印刷有限公司印刷·新华书店北京发行所发行

2004 年 8 月第 1 版第 1 次印刷

787mm×1092mm 1/16·20.5 印张·494 千字

定价:28.00 元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

本社购书热线电话(010)68993821、88379646

封面无防伪标均为盗版

普通高等教育应用型人才培养规划教材编审委员会名单

主 任：刘国荣 湖南工程学院

副主任：左健民 南京工程学院

陈力华 上海工程技术大学

鲍 泓 北京联合大学

王文斌 机械工业出版社

委 员：（按姓氏笔画排序）

刘向东 华北航天工业学院

任淑淳 上海应用技术学院

何一鸣 常州工学院

陈文哲 福建工程学院

陈 峻 扬州大学

苏 群 黑龙江工程学院

娄炳林 湖南工程学院

梁景凯 哈尔滨工业大学(威海)

童幸生 江汉大学

计算机科学与技术专业分委员会名单

主 任：黄陈蓉 南京工程学院

副主任：吴伟昶 上海应用技术学院

委 员：（按姓氏笔画排序）

汤 惟 江汉大学

沈 洁 扬州大学

陈文强 福建工程学院

肖建华 湖南工程学院

邵祖华 浙江科技学院

靳 敏 黑龙江工程学院

序

工程科学技术在推动人类文明的进步中一直起着发动机的作用。随着知识经济时代的到来,科学技术突飞猛进,国际竞争日趋激烈。特别是随着经济全球化发展和我国加入WTO,世界制造业将逐步向我国转移。有人认为,我国将成为世界的“制造中心”。有鉴于此,工程教育的发展也因此面临着新的机遇和挑战。

迄今为止,我国高等工程教育已为经济战线培养了数百万专门人才,为经济的发展作出了巨大的贡献。但据IMD 1998年的调查,我国“人才市场上是否有充足的合格工程师”指标排名世界第36位,与我国科技人员总数排名世界第一形成很大的反差。这说明符合企业需要的工程技术人员特别是工程应用型技术人才市场供给不足。在此形势下,国家教育部近年来批准组建了一批以培养工程应用型本科人才为主的高等院校,并于2001年、2002年两次举办了“应用型本科人才培养模式研讨会”,对工程应用型本科教育的办学思想和发展定位作了初步探讨。本系列教材就是在这种形势下组织编写的,以适应经济、社会发展对工程教育的新要求,满足高素质、强能力的工程应用型本科人才培养的需要。

航天工程的先驱、美国加州理工学院的冯·卡门教授有句名言:“科学家研究已有的世界,工程师创造未有的世界。”科学在于探索客观世界中存在的客观规律,所以科学强调分析,强调结论的惟一性。工程是人们综合应用科学(包括自然科学、技术科学和社会科学)理论和技术手段去改造客观世界的实践活动,所以它强调综合,强调方案优缺点的比较并做出论证和判断。这就是科学与工程的主要不同之处。这也就要求我们对工程应用型人才的培养和对科学研究型人才的培养应实施不同的培养方案,采用不同的培养模式,采用具有不同特点的教材。然而,我国目前的工程教育没有注意到这一点,而是:① 过分侧重工程科学(分析)方面,轻视了工程实际训练方面,重理论,轻实践,没有足够的工程实践训练,工程教育的“学术化”倾向形成了“课题训练”的偏软现象,导致学生动手能力差。② 人才培养模式、规格比较单一,课程结构不合理,知识面过窄,导致知识结构单一,所学知识中有一些内容已陈旧,交叉学科、信息学科的内容知之甚少,人文社会科学知识薄弱,学生创新能力不强。③ 教材单一,注重工程的科学分析,轻视工程实践能力的培养;注重理论知识的传授,轻视学生个性特别是创新精神的培养;注重教材的系统性和完整性,造成课程方面的相互重复、脱节等现象;缺乏工程应用背景,存在内容陈旧的现象。④ 老师缺乏工程实践经验,自身缺乏“工程训练”。⑤ 工程教育在实践中与经济、产业的联系不密切。要使我国工程教育适应经济、社会的发展,培养更多优秀的工程技术人才,我们必须努力改革。

组织编写本套系列教材,目的在于改革传统的高等工程教育教材,建设一套富有特色、有利于应用型人才培养的本科教材,满足工程应用型人才培养的要求。

本套系列教材的建设原则是:

1. 保证基础,确保后劲

科技的发展,要求工程技术人员必须具备终生学习的能力。为此,从内容安排上,保证学生有较厚实的基础,满足本科教学的基本要求,使学生日后具有较强的发展后劲。

2. 突出特色,强化应用

围绕培养目标,以工程应用为背景,通过理论与工程实际相结合,构建工程应用型本科教育系列教材特色。本套系列教材的内容、结构遵循如下9字方针:知识新、结构新、重应用。教材内容的要求概括为:“精”、“新”、“广”、“用”。“精”指在融会贯通教学内容的基础上,挑选出最基本的内容、方法及典型应用;“新”指将本学科前沿的新进展和有关的技术进步新成果、新应用等纳入教学内容,以适应科学技术发展的需要。妥善处理好传统内容的继承与现代内容的引进。用现代的思想、观点和方法重新认识基础内容和引入现代科技的新内容,并将这些按新的教学系统重新组织;“广”指在保持本学科基本体系下,处理好与相邻以及交叉学科的关系;“用”指注重理论与实际融会贯通,特别是注入工程意识,包括经济、质量、环境等诸多因素对工程的影响。

3. 抓住重点,合理配套

工程应用型本科教育系列教材的重点是专业课(专业基础课、专业课)教材的建设,并做好与理论课教材建设同步的实践教材的建设,力争做好与之配套的电子教材的建设。

4. 精选编者,确保质量

遴选一批既具有丰富的工程实践经验,又具有丰富的教学实践经验的教师担任编写任务,以确保教材质量。

我们相信,本套系列教材的出版,对我国工程应用型人才培养质量的提高,必将产生积极作用,为我国经济建设和社会发展作出一定的贡献。

机械工业出版社颇具魄力和眼光,高瞻远瞩,及时提出并组织编写这套系列教材,他们为编好这套系列教材做了认真细致的工作,并为该套系列教材的出版提供了许多有利的条件,在此深表衷心感谢!

编委会主任

湖南工程学院院长 刘国荣教授

前 言

程序设计是计算机科学与技术专业的一门重要的专业基础课程,是数据结构、操作系统、数据库原理、编译原理和软件工程等后继课程的基础。在程序设计语言的选择上,已经出现了从 C 语言向 C++语言过渡的趋势。实际上,C++是一门混合语言,它既支持面向过程的程序设计,也支持面向对象的程序设计,掌握了 C++语言,再学习其他程序设计语言,就是一件轻而易举的事情。因此,本书选择了 C++作为讲解的程序设计语言。

本书共分 17 章,各章介绍内容如下:第 1 章介绍 C++概念,第 2 章介绍数据类型、运算符、表达式、赋值语句、输入输出,第 3 章介绍 Visual C++ 6.0 集成环境,第 4 章介绍分支和循环结构,第 5 章介绍函数和预处理,第 6 章介绍数组,第 7 章介绍指针,第 8 章介绍引用,第 9 章介绍结构体,第 10 章介绍类和对象,第 11 章介绍继承和派生,第 12 章介绍友元、虚函数、静态成员等,第 13 章介绍运算符重载,第 14 章介绍 I/O 流,第 15 章介绍异常处理,第 16 章介绍模板,第 17 章是上机实习题。

本书力图体现以下特点:

- 1) 定位准确。本书的读者定位是应用型本科学生,针对应用型本科人才的培养特点,在注重基础性、学科完整性的基础上,强调实践性、应用性。
- 2) 妥善处理好过程化程序设计和面向对象程序设计的关系。读者可以不具备程序设计的背景知识,本书既有充足的篇幅介绍分支、循环、函数等程序设计语言共性的成分,又在后面章节中努力帮助读者树立面向对象程序设计的理念。
- 3) 概念准确、编排合理。在内容编排上,注意分散难点,便于读者循序渐进。
- 4) 深入浅出,讲解通俗。尽量通过一些实例来说明有关概念。
- 5) 两个并重,强化实践,重视应用。即程序设计语言和程序设计技巧并重,经典实例和实用程序并重。书中有专门的章节介绍 C++上机环境,并提供了较多的习题和上机实习题。

本书由廖雷任主编。各章具体分工如下:廖雷编写第 1、3、10 和第 16 章的主要内容,杨泽凡编写第 6、7、8、9 章,杜青编写第 12、13、14、15 章,胡文瑜编写第 2、4、5 章和第 16 章的部分内容,第 17 章由 4 位编者共同完成。南京大学副教授刘友华博士在百忙之中抽出时间认真地审阅了全书,并提出了宝贵意见,编者深表感谢。

本书编写过程中,得到了机械工业出版社和教材编委会的指导,学校各级领导和同事的关心和帮助。袁璟老师参与了第 11 章的编写工作,周业举、彭仕瑞、倪恩伟在文稿校对、程序测试方面付出了辛勤劳动,在此一并致谢。

由于编者水平有限,恐有不正确和不完善之处,欢迎读者指正。编者的电子信箱是:廖雷 liaolei@sina.com,杨泽凡 yangzefan@163.com,杜青 dqingmail@163.net,胡文瑜 turbo_hwy@hotmail.com。

编 者

2004 年 6 月

目 录

序

前言

第1章 C++概念	1
1.1 计算机语言和程序	1
1.1.1 计算机语言	1
1.1.2 程序	1
1.2 结构化程序设计	2
1.3 面向对象程序设计	3
1.3.1 面向对象的基本特征	3
1.3.2 面向对象的三类基本机制	3
1.4 C++概念	4
1.4.1 C++的发展	4
1.4.2 C++与面向对象的关系	4
1.5 C++的一个简单实例	4
1.5.1 一个简单的示范程序	4
1.5.2 C++程序的组成部分	5
1.6 编辑、编译、连接和运行一个 C++程序	6
习题	7

第2章 数据类型、运算符、表达式、 赋值语句、输入输出	8
2.1 C++语言的词法单位	8
2.1.1 关键字	8
2.1.2 标识符	8
2.1.3 分隔符	9
2.2 基本数据类型	10
2.3 常量和变量	11
2.3.1 常量	11
2.3.2 变量	13
2.3.3 符号常量	15
2.4 运算符和表达式	15
2.4.1 算术运算符	16
2.4.2 关系运算符	18

2.4.3 逻辑运算符	18
2.4.4 位运算符	19
2.4.5 赋值运算符、赋值表达式与赋值 语句	20
2.4.6 条件运算符	22
2.4.7 逗号运算符	22
2.4.8 sizeof 运算符	22
2.4.9 混合运算与强制类型转换	22
2.4.10 关于表达式运算顺序的讨论	24
2.5 数据的输入与输出	24
2.5.1 I/O 流	24
2.5.2 预定义提取符 ">>" 和 数据输入	25
2.5.3 预定义插入符 "<<" 和 数据输出	25
2.5.4 C++的输入输出格式控制	26
2.5.5 C风格的格式化输入输 出函数	27
2.6 自定义数据类型	33
2.6.1 用 typedef 定义类型	34
2.6.2 枚举类型	34
2.7 程序实例	37
习题	40

第3章 Visual C++ 6.0 集成环境简介	43
3.1 Visual C++ 6.0 概述	43
3.2 Visual C++ 6.0 集成开发环境	43
3.2.1 进入 Visual C++ 6.0	43
3.2.2 Visual C++ 6.0 的菜单栏	44
3.2.3 常用的热键	44
3.3 一个简单的例子	45
3.4 常用调试手段	47
习题	50

第4章 分支和循环结构	51	5.8.2 宏定义指令 (#define 和 #undef) ...	92
4.1 语句概述	51	5.8.3 条件编译指令	92
4.2 if 语句	52	习题	94
4.2.1 if 语句的基本形式	53	第6章 数组	96
4.2.2 if 语句的嵌套形式	55	6.1 一维数组	96
4.3 switch 语句	57	6.1.1 一维数组的声明	96
4.4 while 语句	59	6.1.2 一维数组元素的使用	97
4.5 do-while 语句	60	6.1.3 一维数组的初始化	98
4.6 for 语句	60	6.1.4 将数组传递给函数	99
4.7 循环的嵌套	62	6.1.5 一维数组程序举例	100
4.8 break、continue、goto 语句	64	6.2 二维数组	105
4.8.1 break 语句	64	6.2.1 二维数组的定义	105
4.8.2 continue 语句	64	6.2.2 二维数组元素的使用	105
4.8.3 goto 语句	65	6.2.3 二维数组的初始化	106
4.9 程序实例	65	6.3 字符数组	109
习题	69	6.3.1 字符数组与字符串概念	109
第5章 函数和预处理	71	6.3.2 字符数组赋初值	109
5.1 函数的定义、说明和调用	71	6.3.3 字符串处理函数	110
5.1.1 函数的定义	71	6.3.4 程序实例	111
5.1.2 函数的说明	72	6.4 数组的高级应用	113
5.1.3 函数的调用	73	习题	114
5.1.4 函数的分开编译	76	第7章 指针	116
5.1.5 函数的参数传递	77	7.1 变量的地址	116
5.2 函数的嵌套和递归	78	7.2 指针变量	118
5.2.1 函数的嵌套	78	7.2.1 声明指针变量	118
5.2.2 函数的递归	79	7.2.2 指针变量初始化	118
5.3 内联函数	82	7.2.3 指针变量的引用	119
5.4 带默认形式参数值的函数	83	7.2.4 指针变量的地址	121
5.5 函数的重载	84	7.2.5 使用指针应注意的问题	121
5.6 作用域和可见性	86	7.3 指针运算	122
5.6.1 作用域	86	7.4 指针与数组	123
5.6.2 可见性	87	7.5 堆内存分配	125
5.6.3 作用域和可见性的关系	88	7.5.1 堆内存分配	125
5.7 生命期	88	7.6 指针与函数	128
5.7.1 生命期	88	7.6.1 数组作为函数参数	128
5.7.2 局部变量与全局变量	89	7.6.2 变量地址作为函数参数 (传地址)	129
5.8 预处理	91	7.6.3 指针函数	130
5.8.1 文件包含指令 (#include)	91		

7.7 字符指针	131	10.2.1 对象定义的形式	175
7.7.1 字符指针的声明	131	10.2.2 对象成员的表示方法	175
7.7.2 字符串的赋值	131	10.3 构造函数	177
7.7.3 字符指针与字符数组的区别	131	10.3.1 定义构造函数	177
7.8 指针数组	133	10.3.2 缺省构造函数	178
7.9 程序实例	135	10.3.3 重载构造函数	178
习题	139	10.3.4 拷贝初始化构造函数	179
第8章 引用	141	10.3.5 构造函数和 new 运算符	180
8.1 引用的概念	141	10.4 析构函数	181
8.2 引用的操作	142	10.5 this 指针	182
8.2.1 用引用传递函数参数	143	10.6 程序实例	183
8.2.2 利用引用返回多个值	145	习题	189
8.2.3 用引用返回值	146	第11章 继承与派生类	192
8.3 程序实例	148	11.1 继承	192
习题	149	11.1.1 基本概念	192
第9章 结构体	151	11.1.2 单一继承	193
9.1 结构体	151	11.1.3 多重继承	198
9.1.1 结构体类型	151	11.2 初始化基类成员	201
9.1.2 结构体类型变量	152	11.3 冲突、支配规则和赋值兼容性	203
9.1.3 结构体数组与结构体指针	153	11.3.1 冲突	203
9.1.4 结构体变量的引用	155	11.3.2 支配原则	204
9.2 链表	158	11.3.3 赋值兼容性	205
9.2.1 链表结构	158	11.4 虚基类	206
9.2.2 创建链表	159	11.5 程序实例	208
9.2.3 单链表上的查找运算	160	习题	215
9.2.4 单链表上的插入运算	161	第12章 类的其他特性	217
9.2.5 单链表上的删除运算	161	12.1 友元	217
9.2.6 输出单链表	162	12.1.1 友元函数	217
9.3 程序实例	163	12.1.2 友元类	218
习题	168	12.2 虚函数	219
第10章 类和对象	170	12.3 静态成员	222
10.1 类	170	12.3.1 静态数据成员	222
10.1.1 什么是类	170	12.3.2 静态成员函数	226
10.1.2 定义类	170	12.4 const 对象和成员函数	229
10.1.3 定义类时需注意事项	173	12.4.1 const 对象	229
10.1.4 类和结构的关系	174	12.4.2 常成员函数	230
10.2 对象	175	12.5 指向类对象和类成员的指针	232

12.5.1 指向类对象的指针	232	习题	285
12.5.2 指向类的数据成员的指针	233	第 15 章 异常处理	287
12.5.3 指向类的成员函数的指针	233	15.1 异常的概念	287
12.6 程序实例	234	15.2 异常处理的实现	287
习题	241	15.3 异常处理的规则	289
第 13 章 运算符重载	244	15.4 异常的捕获和处理	292
13.1 运算符重载概念	244	15.5 程序实例	293
13.2 几个特殊运算符的重载	245	习题	296
13.2.1 加减运算符的重载	245	第 16 章 模板	297
13.2.2 增量运算符的重载	248	16.1 函数模板	297
13.2.3 强制类型转换运算符 的重载	252	16.1.1 函数模板说明	297
13.3 字符串类	253	16.1.2 使用函数模板	297
13.3.1 string 字符串类简介	253	16.1.3 重载模板函数	299
13.3.2 string 类对象的赋值 与连接	255	16.2 类模板	300
13.3.3 string 类对象的比较	257	16.2.1 类模板说明	300
13.3.4 string 类的特性	258	16.2.2 使用类模板	301
13.4 程序实例	259	习题	305
习题	262	第 17 章 上机实习题	306
第 14 章 I/O 流	265	上机实习题 1 Visual C++ 6.0 集成环境 ...	306
14.1 C++ 的基本流类体系	265	上机实习题 2 分支和循环结构	306
14.1.1 流的概念	265	上机实习题 3 函数和预处理	308
14.1.2 C++ 的基本流类体系	265	上机实习题 4 数组	311
14.2 I/O 标准流类	266	上机实习题 5 指针	311
14.3 文件流类	268	上机实习题 6 引用	311
14.4 串流类	271	上机实习题 7 结构	312
14.5 格式控制	272	上机实习题 8 类和对象	312
14.5.1 格式控制符	272	上机实习题 9 继承和派生	312
14.5.2 流格式控制成员函数	276	上机实习题 10 类的其他特性	313
14.6 使用 I/O 流成员函数	278	上机实习题 11 运算符重载	313
14.6.1 open 和 close 函数	278	上机实习题 12 I/O 流	314
14.6.2 getline 函数	279	上机实习题 13 异常处理	314
14.6.3 read 和 write 函数	279	上机实习题 14 模板	314
14.7 程序实例	281	附录 ASCII 代码表	315
		参考文献	316

第 1 章 C++ 概念

1.1 计算机语言和程序

1.1.1 计算机语言

计算机语言是指程序设计人员和计算机都可以识别的程序代码规则，它是进行程序设计的工具，又称为程序设计语言。计算机语言一般可分为三类：

1. 机器语言

机器语言的内容实际上就是 0 和 1，它通过一系列的指令代码来完成操作，例如：
000×××表示加，001×××表示移动，010×××表示读取，011×××表示储存，111表示待机（×××表示地址码）等。

很明显，用机器语言编写程序要求程序员必须熟记机器指令的二进制代码和地址码，这样写出来的程序很不直观且编写难度大，且不同型号的计算机机器语言也是不同的，但它是直接在计算机硬件上执行的，所以效率比较高。

2. 汇编语言

因为机器语言不直观，所以人们想到了用一些助记符来代替二进制代码，例如：

ADD 表示加，MOV 表示移动，LOAD 表示读取，STORE 表示储存，HALT 表示待机等。

这样，每条指令就易于理解和记忆，用汇编语言编写的程序也直观易于理解，但是计算机却并不能识别和直接运行汇编语言程序，必须由一种翻译程序将汇编语言程序翻译成机器语言程序，这种翻译程序即称为汇编程序。

3. 高级语言

高级语言是一种比较接近自然语言的计算机语言，因为它对具体的算法进行描述，所以又称为算法语言。

它是一类面向问题的程序设计语言，且独立于计算机的硬件，其表达方式接近于被描述的问题，易于人们的理解和掌握。用高级语言编写程序，可简化程序编制和测试，其通用性和可移植性好。

在计算机上，高级语言程序（一般称为源程序）不能直接执行，必须将它们翻译成具体机器的机器语言程序（目标程序）才能执行。这种翻译是由编译程序来完成的。

1.1.2 程序

“程序”这个词有两种含义：其一由程序员编写的源代码，其二是可执行软件。

任何语言编制的程序，最后一定都需要转换成机器语言程序，才能被计算机执行。语言处理程序的任务，就是将各种高级语言编写的源程序翻译成机器语言表示的目标程序。

不同语言编写的源程序, 有不同的语言处理程序。语言处理程序, 按其处理方式的不同, 可分为解释型程序与编译型程序两大类。前者对源程序的处理采用边解释边执行的方法, 并不形成目标程序, 称为对源程序的解释执行; 后者必须先将源程序翻译成目标程序才能执行, 称作编译执行。两者所用的分别是解释器和编译器。使用编译执行的语言相对于使用解释执行的语言优势在于, 编译产生的程序每次运行都非常快, 而解释语言每次运行都必须再将源程序解释一遍, 编译所产生的程序可以给没有编译器的人用, 而使用解释器的语言则必须有运行程序的解释器。

许多年前, 计算机程序员的基本目标是编写小段可快速运行的代码。程序必须小是因为存储器比较贵, 程序必须快是因为运行成本也比较高。随着计算机小型化、廉价和高速, 存储器价格下降, 原来的目标已经改变。今天在商业应用中, 程序员的时间成本在价值上远远超过大多数计算机的成本。现在非常重视的是易写、易维护的代码。所谓易维护是指当用户需求改变时程序能够较容易的扩展和增强。

1.2 结构化程序设计

以前, 人们把程序看成是处理数据的一系列过程。过程或函数定义为一个接一个顺序执行的一组指令。数据完全从过程中分开, 编程的思想在于跟踪哪些函数调用哪些函数, 哪些数据发生了变化。为解决其中可能存在的问题, 结构化编程应运而生。

结构化程序设计的主要思想是功能分解并逐步求精。当一些任务十分复杂以至无法描述时, 可以将它拆分为一系列较小的功能部件, 直到这些自完备的子任务小到易于理解的程度。例如, 计算一个公司中每一个员工的平均工资是一项较为复杂的任务。可以将其拆分为以下的子任务:

- 1) 找出每个雇员的工资。
- 2) 计算雇员人数。
- 3) 合计全部工资。
- 4) 用人数除合计工资。

合计工资可分解为下列步骤:

- 1) 获取每个雇员的档案。
- 2) 读取工资金额。
- 3) 将金额加到合计中。
- 4) 取出下一雇员的档案。

获取所有雇员档案中的记录也可分解为下列步骤:

- 1) 打开雇员的档案。
- 2) 找出正确记录。
- 3) 从磁盘上读取数据。

结构化编程提供了解决复杂问题的有效手段。然而到了 20 世纪 80 年代末, 结构化编程的一些不足都暴露无遗。

首先, 原本希望将数据 (如雇员档案) 和用数据做什么 (如分类、编辑等) 认为是一个整体。过程编程正好相反, 它将数据从要用来处理该数据的函数中分离后工作。其次,

程序员发现每一种解决老问题的新方法都要带来额外的开销，相对于重用性而言。重用性的思想是建立一些具有已知特性的功能部件，当程序需要的时候将其插入。这是在仿效硬件世界，当工程师需要一个新的晶体管时，他不需去制造，而是直接从晶体管盒中找出所需的晶体管。而对于软件工程师来说，在面向对象程序设计出现之前，一直缺乏这样的工具。

1.3 面向对象程序设计

1.3.1 面向对象的基本特征

面向对象的系统包含了三个要素：对象、类和继承。

1. 对象

要想具体地说清楚对象的概念实际是很困难的，如同给“人”这个概念下定义，我们用一个不求甚解的办法使大家形成一个大概的标准。汽车、电视、花、鸟、鱼、虫、树等都是对象。对象是具体的事物。

2. 类

类是创建对象的样板，类包含着所创建对象的状态描述和方法的定义。类并不是针对某一个具体的事物，它只是对某个群体的共性的描述，如同汽车的概念，它并不是针对某一辆汽车，而是针对其所有的汽车的共性。类是抽象的，用类创建一个实体就是对象。

3. 继承

类提供了说明一组对象结构的机制，继承提供了创建新类的一种方法，这就是说，可以通过对已有的类进行扩充或修改来得到一个新类。新类共享已有的类的行为，无需再次定义，而只要定义自己特有的部分即可。例如现在已有汽车类，又需要定义小轿车类，利用继承汽车类的办法，现只需写出小轿车额外具有的特性即可。继承的思想就是对重用性的体现。

1.3.2 面向对象的三类基本机制

所有面向对象的语言都会提供三类机制：封装、继承和多态。

1. 封装

封装就是把数据和操作这些数据的函数放在一起，这样做的好处主要有两个：

- 1) 外部不能随意地更改对象中的数据，提高了数据的安全性。
- 2) 用户使用对象时，将不必在意其中的实现细节，这样就大大减少了程序员的负担。

2. 继承

继承就是一个类获取另外一个类属性或方法的过程。

3. 多态

多态是指一个接口能够做多种用途，而其特定的用途由其特定的环境所决定。例如，一个学生类应该有一个计算成绩的操作。大学生继承了中学生，或者说是中学生的延伸。对于中学生，计算成绩的操作表示语文、数学、英语等课程的计算，而对于后继的大学生，

计算成绩的操作表示高等数学、政治公共课、计算机等课程的计算。

1.4 C++概念

1.4.1 C++的发展

C++语言是C语言的扩充语言。C语言是贝尔实验室的Dennis Ritchie在B语言的基础上开发出来的。C语言是作为多用户多任务分时操作系统UNIX的开发语言而为人们所认识。C语言是与硬件无关的,这样把用C语言编写的程序移植到大多数计算机上成为可能。

1978年Kernighan和Ritchie(K&R)的名著“The C Programming Language”问世,该书全面介绍了传统的C语言。1983年,美国国家标准化协会(ANSI)根据C语言问世以来的各种版本对C语言的发展和扩充,开始制定新的标准,即ANSI C,并于1989年正式公布。1988年,K&R按照ANSI C的标准又重写了《The C Programming Language》一书。1990年,国际标准化组织(ISO)接受87 ANSI C为ISO C的标准。

C语言的主要优点在于表达简洁、生成代码质量高、程序运行效率高。C语言可直接访问内存地址,能进行位操作,使其能够胜任开发操作系统的工作。

但C语言的缺点也逐渐暴露出来。C语言类型检查机制相对较弱;C语言本身几乎没有支持代码重用的语言结构,当程序的规模达到一定程度时,程序员很难控制程序的复杂性。

为了满足管理程序复杂性的要求,1980年,贝尔实验室的Bjarne Stroustrup开始对C语言进行改进和扩充。最初的成果称为“带类的C语言”。1983年正式取名C++。在经历了三次修订后,于1994年制定了ANSI C++标准的草案。以后又经过不断完善,成为目前的C++。

C++仍在不断的发展中。C++保持C语言的所有优点并进行了增强,同时增加了对面向对象编程(OOP)的完全支持。适用于大型系统软件和应用软件的开发。

C++语言对新语言的形成也有较大的影响力,20世纪90年代中期以来,Internet日益普及,用于Internet开发的Java语言日益为人们所熟知,事实上,Java语言与C++语言极为相似,以至于熟悉其中的任何一门语言就可以在很短的时间内掌握另一门语言。

1.4.2 C++与面向对象的关系

C++包括了第三节所述的面向对象程序设计的所有性质,面向对象是C++中的一个最重要的概念,但是我们也可以使用C++来编写传统的结构化C语言程序。C++是一种典型的混合型语言。

1.5 C++的一个简单实例

1.5.1 一个简单的示范程序

我们从一个最简单的示范程序来了解C++的程序结构。

下面这个程序是用来求两个浮点数之和,这两个浮点数是从键盘输入而得到,求得的和由屏幕显示输出。

例 1-1 示范程序。

```
//This is an example.  
#include<iostream.h>  
void main()  
{  
double x,y;  
cout<<" Enter two float numbers:" ;  
cin>>x>>y;  
double z=x+y;  
cout<<" x+y=" <<z<<endl;  
}
```

运行结果:

Enter two float numbers:5.5 3.2

x+y=8.7

1.5.2 C++程序的组成部分

结合上面的示范程序, 看出 C++程序有如下的基本组成部分。

1. 注释

注释是程序员为阅读方便所做的说明, 是提高程序可读性的一种手段。

C++的注释为“//”之后的内容, 直到换行。注释仅供阅读程序使用, 是程序的可选部分, 即有和无并不影响程序的运行。在生成可执行程序之前, C++忽略注释, 并把每个注释都视为一个空格。

2. 处理命令

以“#”开头的行, 称为编译预处理行。“#include”称为文件包含预处理命令。

“#include<iostream.h>”表示在编译之前将头文件(以.h 结尾的文件称之为头文件)“iostream.h”的内容增加到程序中, 以作为其一部分。头文件的作用就相当于工具箱, 它里面包含着我们需要的工具(函数)。例如, 在示范程序中我们需要用到输入输出流对象 cout 和 cin, 而 cout 和 cin 就包含在头文件 iostream.h 中, 所以我们需要声明调用头文件 iostream.h。

3. 程序主体

程序主体就是 void main()和其后的{}之间的部分。程序主体就是一个函数, 是一个主函数。C++程序是由若干个文件组成的, 每个文件又是由若干个函数组成的。函数与函数之间是相对独立的, 并且是并行的, 函数之间可以相互调用。若干个函数不能群龙无首, 所以必须有一个主函数, 并且只能有一个主函数。程序在执行的时候, 系统会先找主函数, 并从主函数开始执行。

并不要因为在 C++中经常看到 void main()就认为这是固定的格式, 每个程序都一样。其实不然, 这里只有 main 和其后的括号是固定的, 前面的 void 只是表示无返回类型, 这里不必深究无返回类型是什么, 以后会讲到, 括号里面是可以添加内容的, 如果不可添加内容, 那么要括号干什么呢? 这里举出几种形式供大家参考。例: