

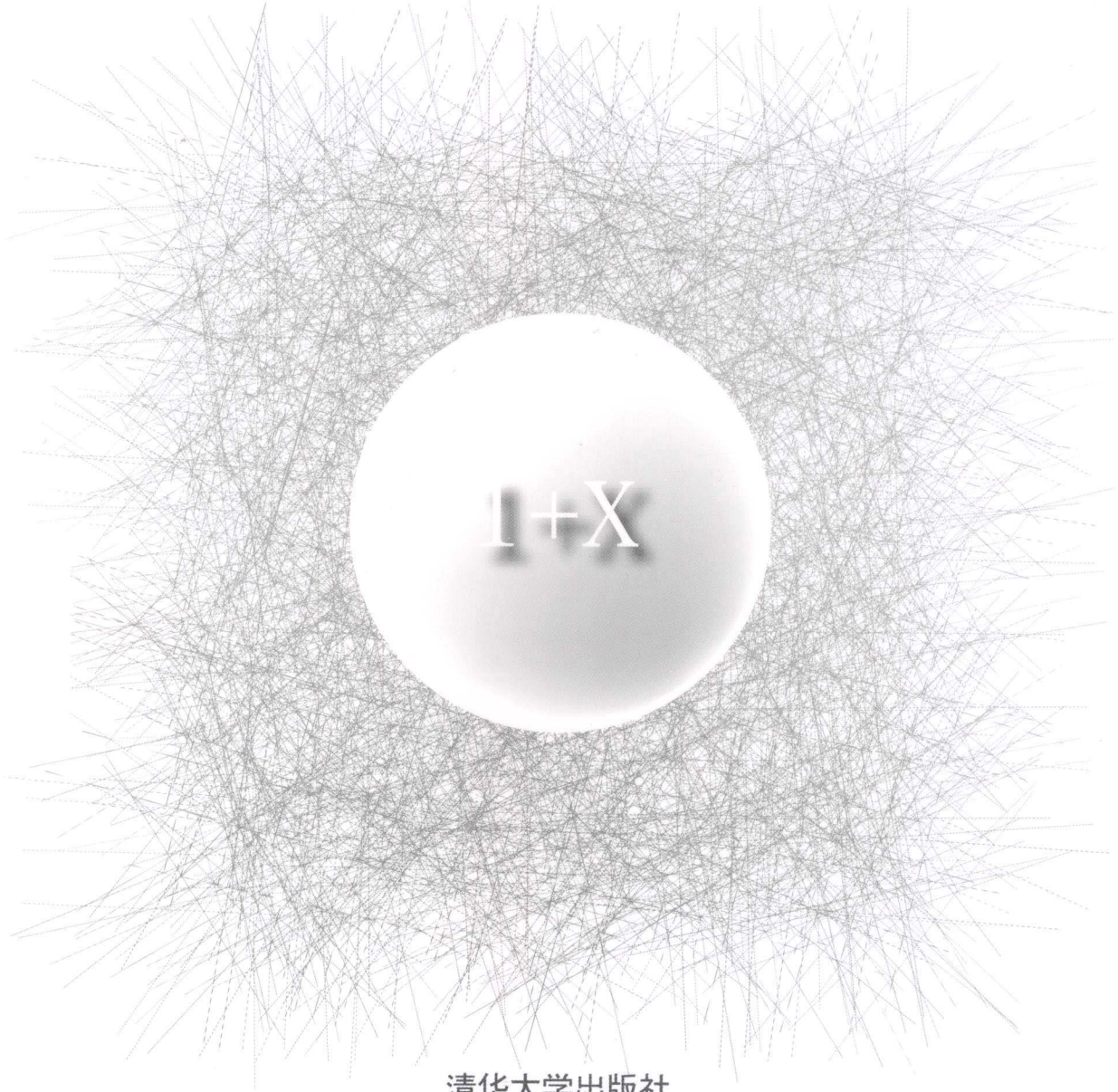
大学计算机基础教育规划教材

C++ 程序设计教程

赵英良 主编

冯博琴 审

仇国巍 夏秦 贾应智 乔亚男 编著



1+X

清华大学出版社

大学计算机基础教育规划教材

C++ 程序设计教程

赵英良 主编

仇国巍 夏秦 贾应智 乔亚男 编著

1+X

清华大学出版社
北 京

内 容 简 介

本书以问题求解的过程为主线,以 C++ 语言为载体,介绍计算机程序的基本结构、信息的表示、流程的控制、模块化方法、指针操作、面向对象的编程方法、输入输出格式控制与文件操作和基本数据结构及应用等内容。本书采用“精讲多练”的教学模式,有丰富的例题和习题。例题从题目描述、问题分析、源程序、运行结果、程序分析、思维扩展等方面进行讲解。本书的特点是层次清晰、循序渐进、清楚易懂。书中源码有丰富的注释,能有效帮助学生理解解题思路。

本书不仅涵盖了 C++ 语言的基本语法知识,而且更注重讲解计算机程序求解问题的思想方法;目的在于既培养编程能力,又启发思维。本书既可作为高等学校理工类专业计算机程序设计课程的教材或参考书,也可供程序设计爱好者、工程技术和软件开发人员学习、参考。

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。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C++ 程序设计教程/赵英良主编. —北京:清华大学出版社, 2013

大学计算机基础教育规划教材

ISBN 978-7-302-33057-8

I. ①C… II. ①赵… III. ①C 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2013)第 145944 号

责任编辑:焦虹

封面设计:常雪影

责任校对:梁毅

责任印制:刘海龙

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京嘉实印刷有限公司

经 销:全国新华书店

开 本:185mm×260mm

印 张:23

字 数:575 千字

版 次:2013 年 8 月第 1 版

印 次:2013 年 8 月第 1 次印刷

印 数:1~2000

定 价:35.00 元

产品编号:055009-01



2010年7月,C9高校联盟在西安召开了“计算机基础课程研讨会”,“计算思维”一词成为大家讨论的热点。会后,C9高校联盟发表联合声明,“计算思维”一词在2660字的声明中出现了18次,可见“计算思维”的重要性和地位。从此我国开始了“计算思维”研究和教学改革的热潮,特别是在计算机基础教育领域。结合程序设计课程中存在的问题,我们也进行了思考,编写了本书。

本书以思维能力培养为目的,以提高编程能力为目标,以C++语言为载体,以问题求解的过程为主线,介绍计算机程序的基本结构、信息的表示、流程的控制、模块化方法、指针操作、面向对象的编程方法、输入输出格式控制与文件操作和基本数据结构及应用等内容。

本书采用“精讲多练”的教学模式,有丰富的例题和习题。例题从题目描述、问题分析、算法描述、编程指南、源程序、运行结果、测试指南、程序分析、思维扩展等方面进行讲解。本书的特点是层次清晰、循序渐进、清楚易懂。

本书希望对解决C++程序设计学习中的以下问题有所帮助。

(1) 提高独立编程的能力。程序设计课程常常会使学生陷入语法的复杂规则中,使其在问题、求解方法和程序之间,无法建立清晰的关联。这就使得他们在看到问题时,用手工是会做的,看别人的程序也懂,但自己写就不行了。为此,本书对稍微复杂的例题,一是给出问题的分析,分析问题求解的关键;二是写出步骤详细的算法,这是问题和程序之间的桥梁;三是在源程序中给出详细清晰的注释,并与算法之间有一致的对应关系,能有效帮助学生理解解题思路。希望同学们思考:对于待求解的问题,关键是什么,其中的物理量如何表达,如何将方法写成算法,如何将算法“翻译”成程序。

(2) 提高程序调试的能力。调试方法本身不在本书中讲述,是在实验中渗透的。从第二次实验开始,在第2章、第3章对应的实验中,会教给学生跟踪程序的方法,强调跟踪、调试的重要性。这是每个学生必须学会的。

(3) 提高自学和独立解决问题的能力。本课程要求学生必须学会使用帮助,认识程序设计中的英语词汇。遇到问题先尝试到教材、网络、MSDN以及同学那儿去获取帮助,然后再问老师。学生遇到英文的编译错误信息和帮助时,在理解上还是很有困难。本书对大部分术语都列出了对应的英文词汇,在配套的《C++程序设计实验指导》的附录中列出了编译中见的英文词汇。

(4) 提高思维能力。本书的例题绝大部分都有“思路扩展”一项,对求解的思路、方法进行概括,进一步介绍这种方法的适用场合或提出问题让学生思考。本书作为讲义已使

用了两届。调查结果显示,认为本课程的教学对解决问题的一般方法“很有启发”的占 31.71 % ,认为“有启发”的占 58.54 % ,两项合计占 90.25% 。

本书不仅涵盖了 C++ 语言的基本语法知识,而且更注重讲解计算机程序求解问题的思想方法;目的在于既培养编程能力,又启发思维。本书可作为高等学校理工类各专业的计算机程序设计教材或参考书,也可供程序设计爱好者、工程技术和软件开发人员学习、参考。

本书由赵英良主编,冯博琴教授审阅。第 1~4 章由赵英良编写,第 5、6 章由贾应智编写,第 7~9 章由夏秦编写,第 10 章由仇国巍编写,第 11 章由乔亚男、仇国巍编写。本书由赵英良、仇国巍统稿。在编写过程中还得到了西安交通大学计算机教学实验中心许多同事的关心、指导和帮助,2011 级、2012 级的许多同学提出了很多建议,在此表示感谢。本书编写过程中参考了很多资料,向这些图书的作者表示诚挚的谢意。由于作者水平有限,书中可能会有错误和不当之处,恳请读者指正。

编 者

大学计算机基础教育规划教材

近 期 书 目

- 大学计算机基础(第4版)(“国家精品课程”、“高等教育国家级教学成果奖”配套教材、普通高等教育“十一五”国家级规划教材)
- 大学计算机基础实验指导书(“国家精品课程”、“高等教育国家级教学成果奖”配套教材)
- 大学计算机应用基础(第2版)(“国家精品课程”、“高等教育国家级教学成果奖”配套教材、普通高等教育“十一五”国家级规划教材)
- 大学计算机应用基础实验指导(“国家精品课程”、“高等教育国家级教学成果奖”配套教材)
- 计算机程序设计基础——精讲多练 C/C++ 语言(“国家精品课程”、“高等教育国家级教学成果奖”配套教材、普通高等教育“十一五”国家级规划教材)
- C/C++ 语言程序设计案例教程(“国家精品课程”、“高等教育国家级教学成果奖”配套教材)
- C 程序设计导引
- C 程序设计实验教程
- C++ 程序设计实验教程
- Visual Basic 2005 程序设计(“国家精品课程”、“高等教育国家级教学成果奖”配套教材、普通高等教育“十一五”国家级规划教材)
- Visual Basic 程序设计语言
- Java 语言程序设计基础(第2版)(普通高等教育“十一五”国家级规划教材)
- Java 语言应用开发基础(普通高等教育“十一五”国家级规划教材)
- 微机原理及接口技术(第2版)
- 单片机及嵌入式系统(第2版)
- 数据库技术及应用——Access
- SQL Server 数据库应用教程(第2版)(普通高等教育“十一五”国家级规划教材)
- Visual FoxPro 8.0 程序设计
- 多媒体技术及应用(“高等教育国家级教学成果奖”配套教材、普通高等教育“十一五”国家级规划教材)
- 多媒体文化基础(北京市高等教育精品教材立项项目)
- 网络应用基础(“高等教育国家级教学成果奖”配套教材)
- 计算机网络技术及应用(第2版)
- 计算机网络基本原理与 Internet 实践
- Web 应用程序设计基础
- Web 标准网页设计与 ASP
- MATLAB 基础教程



第 1 章 程序设计与 C++ 概述	1
1.1 程序设计与计算机语言	1
1.1.1 计算机语言的发展	1
1.1.2 C++ 语言	3
1.2 第一个 C++ 程序	4
1.2.1 在屏幕上显示“Hello World!”	4
1.2.2 C++ 的程序结构和 C++ 程序的执行顺序	4
1.2.3 C++ 程序的基本要素	7
1.3 C++ 的编程步骤	9
1.4 编程实例	13
1.4.1 打印中秋贺卡	13
1.4.2 计算存款利息	15
1.5 小结	17
习题 1	17
第 2 章 简单信息的表达与运算	19
2.1 信息的表达	19
2.1.1 数据类型与常量	19
2.1.2 单项特征的表达——变量	26
2.1.3 多个相同类型的特征的表示——数组	28
2.1.4 文字信息的表达——字符串	30
2.2 数据的运算	32
2.2.1 算术运算	32
2.2.2 关系运算	34
2.2.3 逻辑运算符和表达式	35
2.2.4 自增运算符和自减运算符	36
2.2.5 位运算符和位运算表达式	37
2.2.6 三目条件运算符	41
2.2.7 运算符的优先级	43

2.2.8 不同类型数据的混合运算	45
2.3 程序设计实例	46
2.3.1 已知三边计算三角形面积	46
2.3.2 从反序数到回文数	47
2.3.3 数字符号的数值形式和 ASCII 形式	49
2.3.4 启闭指定设备	50
2.3.5 加密解密	51
2.4 小结	53
习题 2	55
第 3 章 运算的流程控制	57
3.1 程序的执行顺序	57
3.2 不同情况分别处理——分支	58
3.2.1 特殊情况特殊处理(单路分支)	59
3.2.2 不同情况分别处理(两路分支)	60
3.2.3 多种情况分类处理(多重分支 switch)	63
3.3 多次加工——循环程序设计	65
3.3.1 已知次数的循环	65
3.3.2 依据条件进行循环	68
3.3.3 终止循环和直接进入下次循环	71
3.4 综合实例	72
3.4.1 数组的输入、排序和输出	72
3.4.2 字符串的处理	74
3.4.3 有趣的数字	77
3.5 小结	80
习题 3	81
第 4 章 复杂信息的表达与处理	85
4.1 多个事物的多项特征	85
4.1.1 二维数组	85
4.1.2 多维数组	88
4.1.3 二维字符数组及字符串的其他表示方法	89
4.2 多项特征整体描述	92
4.2.1 结构体类型的定义和使用	93
4.2.2 结构体数组	97
4.3 取有限值的特征的描述——枚举	101
4.4 综合实例	103
4.4.1 矩阵运算	103

4.4.2 字符串处理·····	105
4.4.3 统计词频·····	107
4.5 小结·····	110
习题 4·····	112
第 5 章 问题的模块化求解·····	114
5.1 模块化程序设计·····	114
5.2 函数的定义和声明·····	116
5.2.1 函数的定义·····	116
5.2.2 函数的声明·····	118
5.3 函数调用·····	119
5.3.1 函数调用的格式·····	119
5.3.2 参数的传递方式·····	123
5.3.3 为形参指定默认值·····	125
5.3.4 数组名作为函数参数·····	128
5.3.5 结构体变量作为函数参数·····	130
5.4 递归函数·····	131
5.5 内联函数·····	135
5.6 函数重载·····	136
5.7 变量的作用域和存储类型·····	139
5.7.1 局部变量·····	139
5.7.2 全局变量·····	140
5.7.3 变量存储类型·····	141
5.8 程序设计实例·····	144
5.8.1 使用递归求斐波那契序列的前 30 项·····	144
5.8.2 求非线性方程的根·····	145
5.8.3 有趣的数·····	146
5.8.4 二分查找法·····	148
5.9 小结·····	150
习题 5·····	151
第 6 章 按址操作——指针·····	153
6.1 地址与指针·····	153
6.1.1 地址·····	153
6.1.2 指针和指针变量·····	154
6.2 指针变量的定义和使用·····	155
6.2.1 指针变量的定义·····	155
6.2.2 指针变量的使用·····	157

6.2.3	结构体变量的指针	158
6.2.4	二级指针	159
6.3	地址传递和函数的指针	159
6.3.1	函数调用时的地址传递	159
6.3.2	指向函数的指针变量	162
6.4	数组的指针和字符串的指针	164
6.4.1	一维数组的地址	164
6.4.2	二维数组的地址	168
6.4.3	字符串的指针	169
6.5	动态申请存储空间	172
6.5.1	动态申请存储空间	172
6.5.2	定义动态数组	173
6.6	程序设计实例	174
6.6.1	指针变量作为函数的形参	174
6.6.2	32 位的二进制 IP 地址转换成点分十进制地址	177
6.6.3	指针数组	178
6.6.4	带参数的 main 函数	180
6.7	小结	181
	习题 6	181
第 7 章	数据的抽象与封装——类	183
7.1	类的定义和使用	183
7.1.1	类的定义	183
7.1.2	类的使用	186
7.2	面向对象的方法简介	188
7.2.1	对象和面向对象	188
7.2.2	面向对象方法	189
7.3	构造函数和析构函数	190
7.3.1	构造函数	191
7.3.2	析构函数	193
7.3.3	类的组合	199
7.4	对象与指针	203
7.4.1	指向对象的指针	203
7.4.2	指向对象成员的指针	204
7.4.3	this 指针	205
7.5	多文件结构	206
7.5.1	类的定义文件	206
7.5.2	类的实现文件	207

7.5.3 类的使用	208
7.5.4 编译预处理	209
7.6 程序设计实例	212
7.6.1 学生信息类	212
7.6.2 日期类	216
7.7 小结	222
习题 7	222
第 8 章 取其精华 发挥优势——继承	225
8.1 继承和派生	225
8.1.1 派生类的定义	225
8.1.2 派生类的构成	226
8.2 继承方式	230
8.2.1 公有继承	231
8.2.2 私有继承	231
8.2.3 保护继承	234
8.3 派生类的构造函数与析构函数	236
8.3.1 派生类的构造函数	236
8.3.2 派生类的析构函数	237
8.4 虚基类	240
8.5 程序设计实例	244
8.5.1 从学生到本科生、硕士生、博士生	244
8.5.2 从 U 盘到 MP3	249
8.6 小结	251
习题 8	252
第 9 章 统一接口 不同实现——多态性	253
9.1 多态性概述	253
9.2 虚函数	256
9.3 抽象类	258
9.4 运算符重载	261
9.5 程序设计实例	267
9.5.1 从几何形状到点、圆和矩形	267
9.5.2 向量的加减运算	272
9.5.3 高校员工管理系统	276
9.6 小结	285
习题 9	286

第 10 章 标准输入输出与文件操作	287
10.1 数据的输入输出	287
10.1.1 输入输出流及流库	287
10.1.2 标准输入流	288
10.1.3 标准输出流	292
10.2 文件操作	296
10.2.1 文件输入输出流类	296
10.2.2 文件的打开与关闭	297
10.2.3 文本文件和二进制文件的读写	299
10.2.4 文件操作典型例题	307
习题 10	312
第 11 章 数据结构、算法与应用	314
11.1 数据结构概览	314
11.1.1 数据结构的基本概念	314
11.1.2 线性数据结构	316
11.1.3 非线性数据结构	319
11.2 模板与标准模板库	323
11.2.1 函数模板和类模板	323
11.2.2 标准模板库	325
11.2.3 简单应用举例	327
11.3 常见算法策略	332
11.3.1 枚举法	332
11.3.2 分治法	335
11.3.3 回溯法	337
11.3.4 贪心算法	343
习题 11	345

第1章

程序设计与C++概述



计算机是一种自动计算装置。然而计算什么？怎么计算？需要人们以命令的形式告诉计算机。所有命令、符号及使用规则的集合，就是计算机语言。人们用自然语言写成文章，记录事实，表达意愿，交流信息。为了用计算机解决问题，人们需要按一定的顺序使用计算机命令，这种为解决问题，用计算机语言表达的命令的序列就是计算机程序。计算机程序及相关文档的集合称为计算机软件。没有计算机软件，再好的硬件也无法发挥其性能。因此，有人称软件是计算机的“灵魂”。

1.1 程序设计与计算机语言

计算机程序设计(简称程序设计)就是用某种计算机语言编写程序。计算机语言并不唯一，即使能被计算机直接执行的机器语言也是如此。根据不同的需要，人们已经开发出上千种计算机语言，而且仍有新语言不断产生。每种语言都有它的特点。没有哪种语言是最好的，能替代其他所有语言。下面先概括性地介绍计算机语言。

1.1.1 计算机语言的发展

计算机语言的发展经历了机器语言、汇编语言和高级语言等几个阶段。

1. 机器语言

电子计算机是由电子元件和线路组成的，用电子信号表示数据和要执行的操作(也就是命令，计算机中称指令)。命令的表现形式就是0、1组成的序列。不同的序列，可以表示不同的指令，称为**指令的编码**。这样的编码系统称为**机器语言**。人们把要做的事情用机器语言的指令序列表达出来，这便是计算机程序(机器语言程序)。机器语言是计算机可以直接“理解”的语言，机器语言的程序是计算机可以“看得懂”的“文件”，它可以遵照执行，结果就完成了人们交给它的任务。

机器语言用二进制数表示命令，计算机可以直接执行。然而，无论是程序的编写还是阅读都是一件困难的事情。特别是当程序有错误时，要想查找并修改错误，是非常困难的，因为程序员看到的是一系列数字。

2. 汇编语言

20 世纪 40 年代,研究人员为了简化程序设计的过程,开发了记号系统。使用单词的缩写符号来表示指令,而不再使用数字形式,这些符号称为**指令助记符**。同时也用符号表示数据(汇编语言中称操作数)、数据的存放地址以及 CPU 中暂时存放数据的装置——寄存器(register)等。例如使用 ADD 表示加,MOV 表示移动数据,JZ 表示转移等。

用指令助记符、地址符号等符号表示的指令称为**汇编格式指令**(assemble instruction)。汇编格式指令及其表示和使用这些指令的规则,称为**汇编语言**(assembly language)。用汇编语言编写的程序称为**汇编语言程序**或**汇编语言源程序**,或简称**源程序**。

由于汇编语言使用了助记符和用符号表示的数据的存储位置(称为存储单元的地址,简称地址),自然比机器语言容易掌握和使用。然而,机器只能识别机器语言表示的指令,比如,“MOV CX,E024”要翻译为 B924E0H。所以用汇编语言编写的程序并不能被计算机直接执行,还需要将它们翻译为一系列机器指令。实际上,这个工作不需要人来做,可以用机器语言编写一个程序来做这项工作。这个程序称为**汇编程序**(assembler)。翻译的过程称为**汇编**(assemble),翻译的结果称为**目标程序**(object program)。

汇编语言是在机器语言基础上的巨大进步,以至于被称为第二代程序设计语言。然而,由于汇编格式指令是机器指令的符号表示,而不同的 CPU 能识别的机器指令可能是不同的,所以汇编格式指令与机器有着密切的关系。也就是说,为一种 CPU 编写的求解某一问题的程序在另一种 CPU 的机器上不一定能正确执行。另一个缺点是程序员在编写求解问题的程序时,需要考虑计算机中的寄存器、数据的存储位置、内存的容量等细节问题,而不仅需要关心例如如何求解方程的问题。因此机器语言、汇编语言又被人们称为**低级语言**。

3. 高级语言

1953 年,美国 IBM 公司约翰·贝克斯(John W. Backus)向他的主管提出一项建议,开发一种更实用的计算机语言代替汇编语言为他们的计算机 IBM 704 编写程序。这就是 FORTRAN 语言,是 IBM mathematical formula translating system 的缩写。1954 年,完成了计算机语言的详细说明书。1956 年 10 月第一本 FORTRAN 指南问世。1957 年 4 月开发出第一个 FORTRAN 编译器。约翰·贝克斯说:“我的工作来源于懒惰。我不喜欢写程序,所以当我参加 IBM704 项目,为计算弹道写程序时,我开始设计一套编程系统以使写程序更容易。”

从 FORTRAN 开始,以后计算机科学家还开发了多种语言,如 COBOL、BASIC、PASCAL、C、C++ 等。它们一是与机器无关,使用这些语言编写的程序可以较容易地移至到不同的计算机上;二是其命令注重描述解决问题的方法和步骤,而不是某种机器的指令。因此它们又称为**高级语言**。

高级语言的命令也是用单词或缩写符号来表示的,但更加接近于问题的求解方法,因而容易被人理解,但这样的程序也不能被计算机直接识别,所以,也需要翻译成机器语言命令的程序,这样的程序称为**编译器**(compiler)。通常,一条高级语言的命令(有时称为

语句)编译后会对应几条机器指令。对不同的计算机系统,可能翻译后的机器指令序列也不同。

编译器一次将高级语言程序翻译成可执行的机器指令序列,以后再执行程序时不再需要翻译。还有另外一种翻译的策略,就是翻译的同时执行指令;实际是翻译一条高级语言命令,接着就执行这些机器指令,然后再翻译下一条高级语言命令并执行。这样的翻译方式称为**解释执行**,这样的翻译程序称为**解释器**(interpreter)。使用解释器的高级语言有 BASIC、DBASE、Python 等,编译执行的语言有 Fortran、C、C++ 等。

本书以 C++ 语言为例,介绍计算机程序设计的基本方法。

1.1.2 C++ 语言

C++ 语言的鼻祖可以追溯到 20 世纪 50 年代末的 ALGOL 60。1963 年,剑桥大学将 ALGOL 60 语言发展成为 CPL(Combined Programming Language)语言。1967 年,剑桥大学的 Martin Richards 对 CPL 语言进行了简化,于是产生了 BCPL 语言。1969 年,美国贝尔实验室的 Ken Thompson 将 BCPL 进行了修改,为它起了一个有趣的名字“B 语言”,并用 B 语言写了第一个 UNIX 操作系统。1973 年,同在贝尔实验室的 Dennis Ritchie 在 B 语言的基础上设计实现了 C 语言。1973 年,Ritchie and Thompson 在 PDP-11 计算机上用 C 语言重新改写了 UNIX 操作系统。与此同时,C 语言的编译器也被移植到 IBM360/370、Honeywell 11 以及 VAX-11/780 等多种计算机上。此后,C 语言经过多次修改,迅速成为应用最广泛的计算机语言。

1. C++ 的诞生

1983 年,贝尔实验室的 Bjarne Stroustrup 博士在 C 语言基础上引入并扩充了面向对象的概念,设计出了 C++ 语言。后来 Stroustrup 和他的同事又为 C++ 引入了运算符的重载、虚函数、模板和异常处理、名字空间等许多特性。1997 年,C++ 语言成为美国国家标准。1998 年 C++ 语言成为国际标准(ISO/IEC: 1998—14882),各软件厂商推出的 C++ 编译器都支持该标准。

2. C++ 的特点

1) C++ 全面兼容 C

C++ 保持了 C 的简洁、高效和接近汇编语言等特点,对 C 的类型系统进行了改革和扩充,因此 C++ 比 C 更安全,C++ 的编译系统能检查出更多的类型错误。

由于与 C 保持兼容,因此许多 C 代码不经修改就可以在 C++ 的编译环境下使用。大多数用 C 编写的库函数和实用软件也可以用于 C++ 中。

2) C++ 是面向对象的语言

面向对象之前是面向过程的程序设计语言和面向过程的程序设计。面向过程的程序设计以数据为操作、处理对象,对数据进行计算、加工、处理,输出处理结果。面向对象的程序设计认为,真实世界是由对象(各种事物)组成的,多种对象的作用组合起来完成一项任务。面向对象的程序设计用程序设计语言描述客观事物,称为对象。每个对象均有其

特征(称为属性)和功能(称为行为),多个对象的组合及作用,完成一项软件任务。这就是面向对象程序设计的基本思想。C++ 支持对象的描述,同类对象的一般描述称为“类”(class)。通过类的继承、多态等一系列特征技术,面向对象的程序设计能够方便地实现代码重用,提高编程效率,缩短开发周期,提高软件的可维护性。软件开发人员能够利用人类认识事物的一般思维方法来进行软件开发。

1.2 第一个 C++ 程序

“Hello World!”程序是几乎所有程序设计语言教学的第一个程序。该程序最早在 1972 年由贝尔实验室的成员布莱恩·柯林汉(Brian Kernighan)撰写在内部技术文件《Introduction to the Language B》之中,不久他又在 1974 年所撰写的《Programming in C: A Tutorial》中使用了这个范例。“Hello World!”就是编写程序在屏幕上显示这行字。该程序虽然简单,但却包含了一种语言编程的最基本的组成、内容、结构和语言的使用方法。“Hello World!”没有一行是可有可无的,它的所有内容在今后几乎所有的程序中都要用到。

1.2.1 在屏幕上显示“Hello World!”

高级语言的程序是语句的有序集合,要使程序能够被计算机执行,需要再经过编译、链接两个步骤。需要借助其他计算机软件。不同的编译环境,操作方法会略有不同。以 VC6 为例的实现过程见与本书配套的实验指导。

【例 1-1】 编写程序,在屏幕上显示“Hello World!”。

解

① 在记事本或 C++ 编程环境中输入如下的程序:

```
/* Example1-1 hello world! */           //程序注释
#include <iostream>                       //包含基本输入输出库头文件
using namespace std;                   //使用名字空间
int main()                             //主函数
{
    cout<<"Hello World!";               //在屏幕显示 Hello World!
    cout<<endl;                         //换行
    return 0;                           //程序结束
}
```

② 编译、链接。

③ 运行此程序,屏幕显示文字:

```
Hello World!
```

1.2.2 C++ 的程序结构和 C++ 程序的执行顺序

例 1-1 是一个最简单的 C++ 程序,展示了 C++ 程序的基本结构,包含了以后编写程

序时的必备内容。

程序的第1行是注释。/*和*/之间的内容为注释,其中的注释可以连续写在多行中。每行从//开始到末尾的内容也是注释。这是注释的两种形式。注释用于说明或解释其后面的程序段或本行程序的功能、变量的作用以及需要向程序的阅读者说明的任何内容。注释对完成程序的功能没有作用,但却对程序员理解程序的功能有很大帮助。在程序中适当添加注释是好的编程习惯。

第2行是编译预处理命令,用来指示编译器将输入输出的基本程序添加到当前程序中。程序中的“cout”就是iostream中的一个对象。由于几乎每个程序都需要有输入和输出,所以,这一行是以后编写的所有程序都要有的一行。

第3行说明使用std名字空间,这一行也是以后编写基本程序时都要有的一行。当编写的程序较大时,需要多个人的参与,多人可能使用相同的符号表示不同的意义。为了避免产生混淆,C++允许每个程序员为自己使用的符号的集合起一个名字,这就是名字空间(namespace)。当使用这些符号时,采用“名字空间::符号”的形式,例如“std::cout”,这样就知道使用的是哪个人编写的程序中的符号了。然而,这样使用符号有时觉得太烦琐。比如,C++的标准程序库为其使用的符号起的名字空间的名字叫“std”。如果在使用其中的符号时都要加上“std::”,这样例1-1程序中的cout都要写成“std::cout”,“endl”也要写成“std::endl”。如果程序再长,其他符号也都要加上“std::”。当使用了同一个名字空间中的符号时,可以在程序前加上“using namespace <名字空间名>;”,如“using namespace std;”。这样在程序中就不必再加名字空间名的前缀了。不加前缀就是使用“<名字空间名>”中的符号。

第4行到最后一行称为主函数,其中第4行的main称为**函数名**。前面的int表示函数的计算结果是整数,第5行和最后一行组成的一对大括号{}之间的部分称为**函数体**。它们是完成函数的功能的程序,例如其中的cout…。

函数是C++的最小功能单位。一个复杂的C++程序,可以由很多函数组成,但有且仅能有一个名字叫main的函数。一个C++程序被执行时,就是从main函数开始执行的,称为**程序执行的起点**(或开始点)。

main函数的函数体内,程序的第6行完成输出功能。其中cout称为**输出流对象**,<<称为**提取运算符**。简而言之,“cout<<”将后面的内容输出到屏幕上。第6行和第7行可以合并为一行:“cout<<“Hello World!”<<endl;”,可以理解为每加一对小于号“<<”,后面就可以加一项输出的内容。

第7行也是输出行,其中endl是一个控制符号,表示换行,这样其后输出的内容写到下一行,而不是在“Hello World!”后面与“Hello World!”在同一行。

第8行“return 0;”称为函数的**返回语句**。其中的0表示函数的计算结果,可以改变,如写成1,2,3等。本程序里,0没有实际意义。这一行程序也是几乎所有的简单程序照抄的一行。

另外必须知道,C++程序的每一条语句都必须以分号“;”结束。本例中#include一行不是可执行语句,不加分号“;”。大括号说明程序段的开始和结束,不是语句,不加分号。其他各行均有分号。int main()和后面的一对大括号可以看做一个整体,是函数的定义。