

高等学校计算机系列教材

# C++ 面向对象 程序设计

王 萍 冯建华 编著  
邓俊辉 审

清华大学出版社



高等学校计算机系列教材

# C++ 面向对象 程序设计

王 萍 冯建华 编著  
邓俊辉 审

清华大学出版社  
北京

## 内 容 简 介

本教材是针对普通高等院校计算机科学与技术专业编写的。教材围绕 C++ 面向对象程序设计,内容覆盖基本概念和方法、基本数据结构和面向对象的概念、方法和技巧。全书共有 14 章,其中第 1 章至第 6 章简要介绍了 C++ 语言的发展历史、特点和程序风格,C++ 语言程序设计的基本结构、步骤、方法和技巧;第 7 章至第 8 章重点介绍了数组与结构等数据结构以及指针及其引用的概念和方法;第 9 章至第 14 章详细介绍了面向对象程序设计的方法和特点。

本书是供普通高等院校计算机科学与技术专业本科生的教材,也可供专科生以及从事计算机软件开发的科研人员使用。

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

C++面向对象程序设计/王萍,冯建华编著. --北京:清华大学出版社,2006.3

(高等学校计算机系列教材)

ISBN 7-302-12411-6

I. C… II. ①王… ②冯… 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(2006)第 005027 号

出版者:清华大学出版社

<http://www.tup.com.cn>

社总机:010-62770175

地 址:北京清华大学学研大厦

邮 编:100084

客户服务:010-62776969

责任编辑:马瑛珺

印 装 者:清华大学印刷厂

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:27.5 字数:644千字

版 次:2006年3月第1版 2006年3月第1次印刷

书 号:ISBN 7-302-12411-6/TP·7954

印 数:1~4000

定 价:34.00元

# 出版说明

清华大学出版社推出的这套《高等学校计算机系列教材》是《清华大学计算机系列教材》的姊妹篇。

《清华大学计算机系列教材》出版以来,多次获得国家和部级奖项。我们经常收到一些师生热情洋溢的来信,强烈感受到他们对新的知识与教育模式的渴求,同时也感受到广大师生对清华大学计算学科教学工作的关注和信任。

随着高等教育规模的持续扩大和高等教育改革的不断深入,不同院校对于计算学科的教学工作提出了新的要求,突出体现在:理论课时的压缩、实践能力的要求提高,以及学科教育与行业需求的不断结合。根据这些发展趋势,清华大学一批学术水平高、教学经验丰富的教授总结了他们几十年的教学和科研经验,有针对性地编写了《高等学校计算机系列教材》。这套教材的特点体现在:

1. 课程内容在《清华大学计算机系列教材》的基础上,进行了适时的修订更新,并且明确了教学基本要求,区分应该熟练掌握和只需一般了解的内容。

2. 强调加强基础理论教育,重视学生实践能力的培养。课程内容为进一步的实践教学既提供了基础知识,又留出了足够的时间。

另外,本套丛书同时出版了相关辅导用书,并为教师免费提供电子课件,便于师生的教学使用。

清华大学计算学科坚持推行具有启发性的、富于创造性的教学工作,为国家源源不断地培养出一批又一批优秀人才。从《清华大学计算机系列教材》中就可以体会到这些艰辛的探索历程,希望作为姊妹篇的《高等学校计算机系列教材》也能得到师生的认可。

清华大学出版社

2005年9月

# 前 言

面向对象的程序设计思想和方法,是近几年来软件设计、开发和维护技术的一次革命。作为这种新技术的典型代表,C++语言以其灵活性、高效性和高可复用性得到广泛的认可、推崇和应用。

《C++面向对象程序设计》是为普通高等院校计算机科学与技术专业的面向对象程序设计课程专门编写的教材。本书是作者在清华大学计算机科学与技术系多次使用的讲义的基础上,依据教育部教学指导委员会建议的我国普通高等院校计算机科学与计算机工程专业课程大纲的要求,经过整理和完善而成的。在内容的取舍上,本书在力求系统全面的同时,根据课时要求,突出和强调了基本概念、基本原理、基本方法和技巧,使学生能够尽快地对C++程序设计形成整体的认识,为以后的进一步学习打下坚实的基础。

本书围绕C++面向对象程序设计,覆盖基本概念和方法(第1章至第6章)、基本数据结构(第7章至第8章)和面向对象的概念、方法和技巧(第9章至第14章)。针对教学的特点,书中包含大量例题和课后思考题,并提供了答案,极大地方便了学生的阅读和自学。教师在讲授时,可以根据学生基础、学时安排等做适当取舍和编排。

与其他的同类教材相比,本书根据学生在接受、领会C++语言过程中的特点,注重通过大量生动实际的例子,深入浅出地讲解和分析复杂的概念,力图使读者通过实际动手,领会和掌握C++的精髓,并能在实践中学以致用,获得良好的学习效果,为今后的进一步学习和应用开发打下坚实的基础。

本书初稿由王萍和冯建华编著完成,由清华大学计算机科学与技术系博士、副教授、清华大学教务处副处长邓俊辉初审,并整理、最终完稿。

在本书的编写过程中,我们参考了不少国内外同行的有关资料。由于计算机技术的迅速发展,对面向对象程序设计的最新成果难以全面论及;限于作者的学识水平,书中尚有很多不足和错误之处,恳请读者惠予批评、指正。

作 者

2005年10月5日于清华园

# 目 录

<b>第 1 章 C++ 编程基础</b> .....	1
1.1 程序语言的发展 .....	1
1.1.1 机器语言 .....	1
1.1.2 汇编语言 .....	1
1.1.3 高级语言 .....	2
1.1.4 C 语言 .....	2
1.1.5 C++ 语言 .....	2
1.2 C++ 语言简介 .....	3
1.2.1 程序 = 数据 + 操作 .....	3
1.2.2 数据类型 .....	3
1.2.3 对数据的操作——表达式 .....	3
1.2.4 数据操作的流程——语句控制 .....	3
1.2.5 操作的复用——函数 .....	3
1.2.6 数据和操作的封装——类 .....	3
1.2.7 类的复用——派生和继承 .....	3
1.2.8 软件模块 .....	4
1.3 如何学习 C++ 语言 .....	4
1.3.1 勤能生巧 .....	4
1.3.2 风格与规范 .....	4
1.4 用 Visual C++ 开发程序 .....	4
1.4.1 程序——从概念到运行 .....	4
1.4.2 Visual C++ 简介 .....	5
1.4.3 建立应用程序 .....	5
1.5 程序风格 .....	7
1.5.1 效率与风格 .....	7
1.5.2 注释——整理编程思路、增加程序的可读性 .....	7
1.5.3 注释的形式 .....	8
1.5.4 命名 .....	13
1.5.5 编排 .....	14
1.5.6 简单性原则 .....	15
1.5.7 一致性原则 .....	16
小结 .....	17

<b>第 2 章 基本 C++ 程序结构</b> .....	18
2.1 C++ 语言成分 .....	18
2.1.1 字符 .....	18
2.1.2 数字、标识符和表达式 .....	18
2.1.3 语法 .....	19
2.1.4 结构 .....	19
2.1.5 模块 .....	19
2.2 程序的基本结构 .....	19
2.2.1 注释 .....	20
2.2.2 编译预处理命令 #include .....	20
2.2.3 main 函数 .....	20
2.2.4 C++ 程序的执行过程 .....	21
2.2.5 终端输出 .....	22
2.3 变量和简单表达式 .....	23
2.3.1 main 函数的返回值 .....	24
2.3.2 常数和常量 .....	24
2.3.3 变量 .....	27
2.3.4 运算符与简单表达式 .....	29
2.4 表达式和语句 .....	31
2.4.1 返回值 .....	31
2.4.2 嵌套 .....	31
2.4.3 优先级 .....	31
2.4.4 结合性 .....	32
2.4.5 括号与次序 .....	32
2.4.6 操作数 .....	32
小结 .....	32
习题 .....	33
<b>第 3 章 数据类型</b> .....	36
3.1 数据及操作 .....	36
3.2 C++ 数据类型 .....	36
3.3 数据类型 .....	37
3.3.1 数据在计算机中的存储 .....	37
3.3.2 基本数据类型的修饰符号 .....	38
3.3.3 基本数据类型 .....	39
3.3.4 C++ 中的特殊字符 .....	41
3.3.5 字符串 .....	42
3.3.6 数据类型长度的确定 .....	42

3.3.7	数据类型的选择 .....	43
3.4	构造数据类型 .....	44
3.4.1	数组 .....	44
3.4.2	枚举 .....	44
3.4.3	结构 .....	46
3.5	数据输入 .....	46
	小结 .....	48
<b>第4章</b>	<b>程序流程控制 .....</b>	<b>50</b>
4.1	复合语句 .....	50
4.2	程序的执行流程 .....	51
4.3	分支流程 .....	51
4.3.1	if 语句 .....	52
4.3.2	switch 语句 .....	55
4.4	循环流程 .....	57
4.4.1	while 语句 .....	58
4.4.2	do...while 语句 .....	62
4.4.3	for 循环 .....	65
4.5	设计举例 .....	68
4.6	其他控制语句 .....	73
4.6.1	break 语句与多重循环 .....	73
4.6.2	goto 语句 .....	74
4.6.3	条件运算符?: .....	74
	小结 .....	75
<b>第5章</b>	<b>函数 .....</b>	<b>77</b>
5.1	函数机制 .....	77
5.2	函数定义 .....	78
5.2.1	函数名 .....	79
5.2.2	函数参数 .....	79
5.2.3	返回类型 .....	80
5.2.4	函数体 .....	80
5.3	函数调用 .....	82
5.4	程序运行时的内存分布 .....	83
5.5	函数调用的实现机制 .....	84
5.6	函数的参数与返回值 .....	88
5.7	函数参数的传递方式 .....	88
5.7.1	指针参数 .....	88



5.7.2	引用参数 .....	90
5.7.3	数组参数 .....	92
5.7.4	数组参数对实在参数的改变 .....	94
5.7.5	字符串作为参数 .....	94
5.7.6	多维数组作为参数 .....	96
5.7.7	main( )的参数 .....	97
5.8	函数返回值 .....	98
5.9	const 参数、const 返回值与 const 函数 .....	100
5.9.1	const 参数 .....	100
5.9.2	const 返回值 .....	101
5.9.3	const 函数 .....	101
5.10	作用域 .....	102
5.10.1	局部作用域 .....	103
5.10.2	文件作用域 .....	104
5.10.3	覆盖问题 .....	106
5.11	函数原型 .....	109
5.11.1	函数作用域 .....	109
5.11.2	函数原型 .....	110
5.11.3	函数原型的作用 .....	110
5.11.4	使用自己定义的函数原型 .....	115
5.12	特殊的函数用法 .....	115
5.12.1	内联函数 .....	115
5.12.2	内联函数的作用 .....	115
5.12.3	内联函数使用的限制 .....	116
5.12.4	函数重载 .....	116
5.12.5	使用函数重载的条件 .....	117
5.12.6	重载函数的使用方法 .....	117
5.12.7	函数的默认参数 .....	118
5.12.8	递归函数 .....	120
5.12.9	设计递归程序的方法 .....	121
5.12.10	递归程序的优缺点 .....	123
5.12.11	函数模板 .....	123
5.12.12	模板的实例化 .....	125
5.12.13	支持多种类型的模板 .....	126
5.12.14	使用模板的注意事项 .....	127
	小结 .....	127

<b>第 6 章 简单程序设计</b> .....	129
6.1 程序开发过程 .....	129
6.2 文件之间的信息共享 .....	130
6.2.1 外部变量.....	130
6.2.2 外部函数.....	132
6.2.3 静态全局变量与静态函数.....	133
6.3 头文件 .....	135
6.3.1 #include 指令 .....	135
6.3.2 头文件内容.....	136
6.3.3 预编译指令.....	138
6.4 生存期 .....	141
6.4.1 生存期与内存.....	142
6.4.2 初始化问题.....	143
6.4.3 局部静态变量.....	143
小结.....	144
<b>第 7 章 数组与结构</b> .....	146
7.1 数组 .....	146
7.1.1 数组的定义.....	146
7.1.2 数组元素的访问.....	147
7.1.3 数组复制.....	149
7.1.4 数组下标越界.....	149
7.1.5 数组元素的初始化.....	150
7.1.6 多维数组.....	151
7.1.7 多维数组元素的访问.....	151
7.1.8 多维数组元素的初始化.....	156
7.1.9 字符串数组.....	156
7.1.10 数组应用举例 .....	158
7.2 结构 .....	162
7.2.1 结构类型的定义.....	162
7.2.2 结构成员的访问.....	163
7.2.3 结构变量的初始化.....	163
7.2.4 结构的赋值.....	164
7.2.5 结构成员.....	166
7.2.6 结构嵌套.....	167
小结.....	167

<b>第 8 章 指针及其引用</b> .....	169
8.1 指针的定义与初始化 .....	169
8.2 指针的赋值与使用 .....	171
8.3 指针的数据类型 .....	174
8.3.1 void 指针* .....	176
8.3.2 NULL 指针值 .....	176
8.4 用 const 来限定指针 .....	177
8.4.1 指向常量的指针.....	177
8.4.2 指针常量.....	178
8.4.3 指向常量的指针常量.....	178
8.5 指针与数组 .....	179
8.5.1 指针运算.....	181
8.5.2 用指针对数组操作.....	183
8.5.3 指针与字符串.....	184
8.6 动态内存申请 .....	185
8.6.1 new 与 delete .....	186
8.6.2 指针与动态数组.....	189
8.6.3 指针与动态结构.....	194
8.6.4 指针小结.....	201
8.7 引用的概念 .....	201
8.7.1 引用的初始化.....	202
8.7.2 能够引用的数据类型.....	203
8.7.3 const 引用 .....	204
8.8 指针和引用 .....	205
8.9 引用的应用 .....	206
小结.....	206
<b>第 9 章 面向对象程序方法</b> .....	207
9.1 面向对象方法概述 .....	207
9.1.1 面向对象的概念.....	207
9.1.2 面向对象方法的形成.....	209
9.2 面向对象方法的基本概念 .....	210
9.2.1 对象.....	211
9.2.2 程序设计语言中的对象.....	211
9.2.3 消息.....	212
9.2.4 程序设计语言中的消息.....	212
9.2.5 类.....	213
9.2.6 程序设计语言中的类.....	213

9.2.7	继承	214
9.2.8	程序设计语言中的继承	215
9.2.9	封装	215
9.2.10	程序设计语言中的封装	216
9.2.11	多态性	216
9.2.12	程序设计语言中的多态性	217
9.3	面向对象的意义	217
9.3.1	模块化——软件质量保证	217
9.3.2	复用——软件快速开发的必由之路	218
9.3.3	走向面对对象的道路	218
9.3.4	结构化程序设计	219
9.3.5	面向对象程序设计	220
9.4	面向对象的分析与设计	221
9.4.1	面向对象开发方法的生命周期	221
9.4.2	面向对象分析	221
9.4.3	面向对象设计	222
9.5	面向对象方法与软件复用	222
9.5.1	复用级别	222
9.5.2	复用的好处	223
9.5.3	面向对象方法对复用的支持	223
9.5.4	复用技术对面向对象软件开发的支持	224
9.6	面向对象程序设计语言	224
小结		225
<b>第 10 章</b>	<b>类与对象</b>	<b>226</b>
10.1	类的定义与使用	226
10.2	成员的访问控制	227
10.3	类的成员函数	231
10.3.1	成员函数的定义位置	231
10.3.2	类作用域	232
10.3.3	内联成员函数与非内联成员函数	235
10.3.4	使用内联函数的条件	236
10.3.5	const 成员函数	238
10.4	类定义与头文件	240
10.4.1	使用类库中的类	243
10.4.2	类与程序结构	244
10.5	对象的创建与使用	245
10.5.1	对象的创建	245

10.5.2	对象作为函数的参数与返回值	246
10.6	this 指针	247
10.7	类与结构的关系	250
10.8	举例	252
10.8.1	用数组实现的队列类	253
10.8.2	用链表实现的队列类	259
小结		265
<b>第 11 章</b>	<b>构造函数与析构函数</b>	267
11.1	构造函数和析构函数的意义	267
11.2	构造函数	270
11.2.1	重载构造函数	270
11.2.2	对象构造的几种方式	274
11.2.3	构造函数的默认参数	274
11.2.4	默认构造函数	275
11.2.5	拷贝构造函数	277
11.2.6	成员初始化参数表	280
11.3	析构函数	285
小结		295
<b>第 12 章</b>	<b>静态成员、友元</b>	297
12.1	静态成员	297
12.1.1	静态数据成员与静态函数成员	298
12.1.2	静态数据成员的初始化	299
12.2	友元	306
12.2.1	友元的说明与使用	306
12.2.2	使用友元的条件	311
12.2.3	使用友元的注意事项	311
小结		313
<b>第 13 章</b>	<b>运算符重载</b>	314
13.1	可以重载的运算符	314
13.2	运算符重载的规则	316
13.3	常用运算符重载举例	319
13.3.1	函数调用运算符的重载	320
13.3.2	赋值运算符的重载	323
13.3.3	双目算术运算符的重载	325
13.3.4	单目算术运算符的重载	327

13.3.5	自增自减运算符的重载	328
13.4	插入抽取运算符的重载	337
13.4.1	C++ 的 I/O 流库	338
13.4.2	插入运算符的重载	339
13.4.3	抽取运算符的重载	341
13.5	类型转换函数	345
13.5.1	构造函数充当类型转换函数	346
13.5.2	直接定义类型转换函数	347
13.5.3	隐式类型转换与显式类型转换	349
	小结	350
<b>第 14 章</b>	<b>继承</b>	<b>352</b>
14.1	单继承	352
14.1.1	公有、保护和私有继承	354
14.1.2	派生类对象的构造	361
14.1.3	继承应用举例	368
14.2	多继承	372
14.2.1	多继承的定义	373
14.2.2	多继承中的构造函数与析构函数	373
14.2.3	二义性	376
14.2.4	虚基类	379
14.2.5	虚基类的初始化	380
14.2.6	多继承的应用	381
14.3	继承与类库	381
14.4	Visual C++ 的类库——MFC	382
	小结	383
	自测题答案	384
	参考文献	421

# 第 1 章 C++ 编程基础

## 本章要点

(1) C++ 是一种面向对象的高级程序设计语言,具有良好的数据、功能封装性,可以提高软件的可复用性。

(2) Visual C++ Studio 是 C++ 语言的一种集成开发环境,提供了高效的编辑、编译、连接和调试功能。

(3) 软件书写应当遵循良好的风格,这包括注释、变量命名、编排、简单性和一致性等方面。

## 1.1 程序语言的发展

### 1.1.1 机器语言

计算机是人们的好帮手,它有能进行快速处理的 CPU,有能快速存取信息的内存,有能存放大量信息的硬盘。但是,计算机毕竟是一台机器,除非我们告诉它做什么,怎么做,否则它什么也不会做。

要让计算机做事,必须给它“指令”。可是计算机很笨,它只认识 0 和 1,它的所有操作和数都用 0 和 1 来表示。如计算机中要表示数 5,6,7,8,9,必须分别用四位二进制数来表示:0101,0110,0111,1000,1001。而让计算机进行加、减、乘、除等操作,也必须用二进制数来表示。如果让计算机计算  $AL=4+5$ ,则需要向计算机发出以下指令:

```
1011 0000 0000 0100
0000 0100 0000 0101
```

这是计算机可以理解的语言,我们称它为机器语言。

### 1.1.2 汇编语言

计算机用数来“思考”,而人不行。于是聪明的程序员想出了一种办法,用一些人易于理解的符号来代替上面的 0、1 序列,如求  $AL=4+5$  可以写成下面两句话:

```
MOV AL, 4
ADD AL, 5
```

上面的每一句话都代表一个指令。这样的语言很容易被翻译成机器语言,因为翻译是一件很机械的事情,非常适合计算机去完成,所以人们编写了汇编程序,专门负责将上

面这种符号语言(称为汇编语言)翻译成机器语言。程序员再也不用与繁琐的 0 和 1 打交道了。

### 1.1.3 高级语言

随着计算机硬件技术的迅速发展,计算机的处理能力不断提高。人们发现,可以用更自然的方式书写程序,如可以直接在程序中写“AL=4+5”来进行计算。然后将这些程序翻译成机器能理解的机器语言。于是产生了高级语言,翻译这些高级语言的程序称为“编译程序”。C、FORTRAN、PASCAL、COBOL 等都是高级语言。

为什么不用人类的自然语言来作为计算机语言呢?因为人类语言有一个很大的缺陷:它是不精确的。即使是像法律、合同这样精心书写的条文,也还是可能会存在二义性。而计算机需要的是准确而详尽的指示,自然语言显然无法胜任。计算机高级语言的发展趋势是向自然化发展,但是自然语言无法作为计算机语言。

### 1.1.4 C 语言

1970 年,两位程序员 Brian Kernighan 和 Dennis Ritchie 在 B 语言的基础上首创了一种新的程序设计语言,取名为 C 语言。设计 C 语言的最初目的是编写操作系统,它是 UNIX 操作系统的开发语言。C 语言有很多优点:与硬件无关,移植方便;语言简洁,使用方便;丰富的运算符和数据类型;可以直接访问内存地址,能进行位操作;生成的目标代码质量高,程序运行效率高。由于这些优点,C 语言很快就被用于编写各种不同类型的程序,从而成为世界上最流行的语言之一。

### 1.1.5 C++ 语言

C++ 源于 C 语言。随着 C 语言的应用,它的缺点也逐渐显示了出来:C 语言的类型检查机制弱,使得程序开发过程中的错误不能在编译时被发现;C 语言本身是面向过程的语言,没有支持代码复用的机制,因此所有的程序都需要从头开始编制,而且当程序规模达到一定程度时,程序员很难控制程序的复杂性。

20 世纪 80 年代初,美国 AT&T 贝尔实验室的 Bjarne Stroustrup 设计并实现了 C 语言的扩充、改进版本,C++ 语言由此诞生。C++ 语言改进了 C 的不足之处,增加了对面向对象的程序设计的支持,在改进的同时,保持了 C 的简洁性和高效性。C++ 包含了 C 的所有语法,大多数 C 程序都可以简单地转化为 C++ 程序(只是里面不包含 C++ 新的特征),所以原来的一大批 C 程序员可以很容易地成为 C++ 语言的拥戴者。更重要的是,植根于 C 语言的 C++ 语言继承 C 的高效简洁的特点,可以使用比其他大多数语言更高效的方法组织信息。所以,C++ 语言获得了巨大的成功。

时至今日,C++ 越来越受到重视并且已经得到了广泛的应用,许多软件公司如 Microsoft, Inprise 公司都为 C++ 设计编译系统,提供不同应用级别的类库和越来越方便的开发环境。利用 C++ 设计并实现应用系统已是日益简单和快捷的事情了。



## 1.2 C++ 语言简介

### 1.2.1 程序 = 数据 + 操作

计算机语言是作为程序员和计算机之间的桥梁而存在的。计算机本身只是一台机器,它所能做的事情就是在指定的数据上执行指定的操作。因此,要让计算机帮人做事,就必须按计算机的思路来告诉它要做什么事情,也就是程序员需要将要做的事情分解成为“数据+操作”。

### 1.2.2 数据类型

计算机中,数据被存储为一系列的字或字节,C++语言则将这些字或字节组织成为有用的数据。不同的组织方式得到不同的数据类型,C++中有简单的数据类型,如整型、浮点型、字符型等,也有由这些简单类型构造而成的复合的数据类型,如结构、联合。数据用变量或常量来存放。常量存放的是不变的数据,而变量存放的数据可以变化。

### 1.2.3 对数据的操作——表达式

类似于我们用“(1+2)\*4=12”来进行数学运算,C++中可以用操作符对数据进行操作。操作符和操作数一起构成了表达式,根据操作符的不同,表达式可以是算术表达式、逻辑表达式、条件表达式等。表达式可以构成表达式语句。

### 1.2.4 数据操作的流程——语句控制

语句是C++最小的可执行单元。程序的运行过程就是对语句的执行过程。语句一般按顺序执行,但流程控制语句可以改变程序的执行流程。语句分为声明语句、表达式语句、流程控制语句等。声明语句定义变量或常量;表达式语句在指定数据上执行指定操作;流程控制语句控制程序的执行顺序。

### 1.2.5 操作的复用——函数

一组相关的语句可以组成函数。函数可以完成特定的功能,如计算一个数的平方根、求N个数的平均值。函数可以供程序员在程序中反复使用。C++提供丰富的标准函数,这些函数可以完成常用的操作,如查找、排序、输入输出、数学运算等。

### 1.2.6 数据和操作的封装——类

传统的程序设计语言都是将数据和操作分开,而C++的重要突破是:它允许将数据和作用在这些数据上的操作(函数)组合在一起,形成一个整体——类,可以将类看成是一种特殊的数据类型,它的特殊性在于这种数据类型带有自己的操作,类的变量就是对象。

### 1.2.7 类的复用——派生和继承

可以在一个类的基础上派生出新的类,新的类继承了原来类的所有特征。继承允许