

高等学校
计算机教材

面向应用与实践系列

郭有强 等编著

C++面向对象程序设计



清华大学出版社

郭有强 王 磊 姚保峰 编著

C++面向对象程序设计



清华大学出版社
北京

内 容 简 介

本书采用通俗易懂的语言,全面系统地介绍了 C++ 语言的基本概念;通过大量精选的具有典型性的例题,重点讲述解决问题的思路,帮助读者理解编程思想及相关概念;通过综合性的实训,提高读者的实际编程能力。本书以结合实例讲解基本概念和方法为主,力求将复杂的概念用简洁浅显的语言来描述,做到深入浅出。

本书中所有的例程都在 Visual C++ 6.0 下调试通过。清华大学出版社出版的《C++ 面向对象程序设计实验指导与课程设计》为本书配套教材。

本书既可作为高等院校本科生程序设计基础教材,也可作为计算机爱好者自学用书和各类工程技术人员的参考书。

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++ 面向对象程序设计 / 郭有强,王磊,姚保峰编著. —北京:清华大学出版社,2009.2
(高等学校计算机教材——面向应用与实践系列)

ISBN 978-7-302-19353-1

I. C… II. ①郭… ②王… ③姚… 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(2009)第 010848 号

责任编辑:袁勤勇 赵晓宁

责任校对:白 蕾

责任印制:何 芊

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

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

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

印 刷 者:北京季蜂印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185×260 印 张:24.25

字 数:583 千字

版 次:2009 年 2 月第 1 版

印 次:2009 年 2 月第 1 次印刷

印 数:1~4000

定 价:33.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。
联系电话:010-62770177 转 3103 产品编号:030164-01

读者意见反馈

亲爱的读者：

感谢您一直以来对清华版计算机教材的支持和爱护。为了今后为您提供更优秀的教材，请您抽出宝贵的时间来填写下面的意见反馈表，以便我们更好地对本教材做进一步改进。同时如果您在使用本教材的过程中遇到了什么问题，或者有什么好的建议，也请您来信告诉我们。

地址：北京市海淀区双清路学研大厦 A 座 602 计算机与信息分社营销室 收
邮编：100084 电子邮件：jsjic@tup.tsinghua.edu.cn
电话：010-62770175-4608/4409 邮购电话：010-62786544

教材名称：C++面向对象程序设计

ISBN：978-7-302-19353-1

个人资料

姓名：_____ 年龄：_____ 所在院校/专业：_____

文化程度：_____ 通信地址：_____

联系电话：_____ 电子信箱：_____

您使用本书是作为：指定教材 选用教材 辅导教材 自学教材

您对本书封面设计的满意度：

很满意 满意 一般 不满意 改进建议_____

您对本书印刷质量的满意度：

很满意 满意 一般 不满意 改进建议_____

您对本书的总体满意度：

从语言质量角度看 很满意 满意 一般 不满意

从科技含量角度看 很满意 满意 一般 不满意

本书最令您满意的是：

指导明确 内容充实 讲解详尽 实例丰富

您认为本书在哪些地方应进行修改？（可附页）

您希望本书在哪些方面进行改进？（可附页）

电子教案支持

敬爱的教师：

为了配合本课程的教学需要，本教材配有配套的电子教案（素材），有需求的教师可以与我们的联系，我们将向使用本教材进行教学的教师免费赠送电子教案（素材），希望有助于教学活动的开展。相关信息请拨打电话 010-62776969 或发送电子邮件至 jsjic@tup.tsinghua.edu.cn 咨询，也可以到清华大学出版社主页（<http://www.tup.com.cn> 或 <http://www.tup.tsinghua.edu.cn>）上查询。

序

1. 关于本书

随着计算机科学技术的迅速发展,程序设计技术和程序设计语言也得到不断发展。目前,面向对象程序设计是软件开发领域的主流技术。这种技术从根本上改变了人们以往设计软件的思维方式,它集抽象性、封装性、继承性和多态性于一体,实现了代码重用和代码扩充,极大地减少了软件开发的繁杂性,提高了软件开发的效率。C++(本书为方便将C++语言简称为C++)为面向对象技术提供全面支持,是主流的面向对象程序设计语言。因此C++在当前程序设计领域中的地位是很重要的。它是一个可编写高质量的用户自定义类型库的工具。其核心应用领域是最广泛意义上的系统程序设计。此外,C++还被成功地运用到系统程序设计的应用领域中。几乎所有操作系统上都有C++的应用。

C++是“带类的C”,是面向对象的程序设计(object-oriented programming)语言。高等院校计算机专业和相关专业基本上都开设了该课程,其目的是让学生掌握面向对象程序设计的概念和方法,深刻理解面向对象程序设计的本质,并用面向对象技术来编写程序、开发软件。长期以来,通过教学实践发现学生在学习C++的过程中,普遍感觉到C++不好掌握,尤其是它的面向对象的设计思想及相关概念在接受时感到困难,实际动手编程的能力较差,更谈不上应用了。鉴于此,笔者结合十多年本课程的教学经验和体会,特编写这本符合当前学生接受能力的、通俗易懂的教材。

本书的目标是帮助读者深刻理解面向对象程序设计的思想,掌握C++程序分析能力和设计技能,学会运用MFC进行Windows程序设计。

2. 本书结构

本书共13章,分成如下三个部分。

第1部分(第1~第6章)概述了面向过程和面向对象技术的基本概念、不同的编程思维方式;介绍了如何利用Visual C++ 6.0调试C++控制台应用程序;讲授了C++面向过程编程部分,包括C++程序设计基础、程序流程控制、数组、模块设计和指针等内容。

第2部分(第7~第11章)主要讲授C++面向对象编程思想及方法,是本书的核心。这部分内容以面向对象技术的特征为线索展开对C++的讨论,为第3部分的应用奠定理论基础。

第3部分(第12章和第13章)是面向对象技术在Visual C++中的应用部分。这部分主要介绍了利用MFC开发Windows应用程序的基本编程模式和程序设计方法。为什么要加强本部分内容呢?原因是C++加入可视化概念后,并不像一些可视化程序设计语言(如VB、Delphi语言等)那样容易上手。Visual C++结构复杂,代码量大,专业化程度高,初学者很难把握Visual C++的脉搏。第12章通过循序渐进的方式讲述了利用MFC编程的模式、方法,配以具体应用的详细实训步骤,读者在学习了本章后即可基本入门,便于日后进阶。第13章给出了基于两种编程模式的综合实训,以提高读者的综合编程能力。

3. 本书特点

本书通过比较的方法介绍了面向过程和面向对象程序设计的概念和方法,读者可以从中理

解到思维方式的变化;在讲授面向对象编程时,紧紧围绕面向对象程序设计的特征展开;在讲述每部分内容时,采用先说明开设这部分内容的理由的方法,使读者对所学概念有充分的理解;每部分在具体阐述的过程中既重点突出,又注重细节用法;为便于读者自学,全书采用了大量的注释;每部分都精选了大量的典型实例;为了与当前流行的可视化编程接轨,最后讲述了 MFC 编程模式,以使读者深刻地体会到 Visual C++ 功能的强大,从而提高读者对软件编程的学习兴趣和能力,使读者感受到学习后的成就感。本书追求用通俗易懂的语言、生活中的现象来阐述基本理论,突出重点,解释到位,讲深讲透“难点”部分;所有内容均采用“提出问题,说明问题,解决问题”的模式编写,更多地考虑到学习者的接受能力和接受方式;采用先例题后理论的模式,用程序例子来说明难懂的抽象概念;对许多问题都描述多个编程方案,以使读者在比较中学习,进而深刻理解相关设计思想和分析相关算法;本书除了正面阐述说明什么是正确的,还注意从反面来说明什么是不正确的;在讲解 MFC 编程的过程中,注重操作步骤及细节,以便于读者模仿和操作。

本书内容丰富,结构紧凑,概念阐述清楚,例题典型丰富,注重能力的培养。提供了大量实例、实训、习题和实验,突出实践环节,以期读者深刻理解编程思想和增强实际动手编程能力。

本书中所有的例程都在 Visual C++ 6.0 下调试通过。本书中的程序和算法较多,为方便阅读书中的变量均用正体。

4. 适用对象

本书是作者总结多年教学经验、参考大量的国内外有关资料并结合自身的实际工程项目经验编写而成,融会贯通了 C++ 面向过程、C++ 面向对象以及利用 VC++ 进行 Windows 程序设计三部分,内容丰富,结构紧凑,概念阐述清楚,注重能力的培养,是一本内容全面的教材。可作为高等院校本科生程序设计基础教材,以及计算机爱好者自学用书和各类工程技术人员的参考书。

本书由郭有强编写第 1、第 7~第 13 章,并负责总体设计并统稿;王磊编写第 4 和第 5 章,并负责本书全部例题源代码的测试和电子讲稿的制作;姚保峰编写第 2、第 3 和第 6 章。本教材电子讲稿与全部例题的源代码可从 www.bbxu.edu.cn 网站下载。

本书的编写大纲及内容均经合肥工业大学信息与计算机学院的王浩教授审阅,在此谨致谢忱。感谢胡学钢教授对本书给予了极大的关注和支持,提出了宝贵的建设性意见。感谢本书所列参考文献的作者!感谢清华大学出版社各位编辑,他们为本教材的出版倾注了大量的心血和热情。也正是由于他们前瞻性的眼光和忙碌,才让读者有机会看到本教材。

由于作者水平有限,加之时间仓促,错误与疏漏之处在所难免,敬请读者不吝赐正。在使用该书时如遇到什么问题需要与作者商榷,或想索取其他相关资料,请与作者联系。联系方式: bbxyguo@163.com。

郭有强

2008 年 12 月

目 录

第 1 章 程序设计与C++ 概述	1
1.1 程序与程序设计语言	1
1.1.1 程序	1
1.1.2 程序设计语言	2
1.1.3 程序设计	3
1.1.4 结构化程序设计	4
1.2 算法概述	4
1.2.1 算法的含义	4
1.2.2 算法的特性	6
1.2.3 算法的表示	6
1.3 面向过程和面向对象编程概述	8
1.4 C++ 语言简介	10
1.4.1 C++ 语言的发展	10
1.4.2 一个简单的C++ 程序	11
1.5 面向过程和面向对象的程序设计方法比较	12
1.6 用 Visual C++ 开发程序	14
1.6.1 C++ 程序的开发步骤和上机调试流程	14
1.6.2 Visual C++ 6.0 调试C++ 程序的操作过程	14
本章小结	18
习题 1	18
第 2 章 C++ 程序设计基础	20
2.1 数据与数据类型	20
2.1.1 数据与类型	20
2.1.2 C++ 中的数据类型	20
2.2 常量	21
2.2.1 数值常量	22
2.2.2 字符常量	23
2.2.3 字符串常量	24
2.2.4 符号常量	24
2.2.5 const 常量	25
2.3 变量	26
2.3.1 变量的定义	26
2.3.2 整型变量	27
2.3.3 实型变量	27
2.3.4 字符变量	28

2.4	各类数值型数据间的混合运算及数据类型转换	29
2.4.1	自动转换	29
2.4.2	强制转换	30
2.5	数据的输入与输出	30
2.5.1	输出流对象 cout	30
2.5.2	输入流对象 cin	32
2.6	基本运算符和表达式	32
2.6.1	运算符、表达式、优先级和结合性	32
2.6.2	算术运算符与算术表达式	34
2.6.3	赋值运算符与赋值表达式	36
2.6.4	自增、自减运算符与表达式	37
2.6.5	关系运算符与关系表达式	38
2.6.6	逻辑运算符与逻辑表达式	39
2.6.7	逗号运算符及逗号表达式	40
2.6.8	条件运算符与条件表达式	41
2.6.9	sizeof 运算符	42
	本章小结	42
	习题 2	43
第 3 章	程序流程控制	46
3.1	顺序结构程序设计	46
3.2	选择结构程序设计	46
3.2.1	if 语句	46
3.2.2	switch 语句	50
3.3	循环结构程序设计	51
3.3.1	while 语句	51
3.3.2	do-while 语句	52
3.3.3	for 循环	52
3.3.4	循环的嵌套	53
3.3.5	几种循环的比较	54
3.4	几种控制语句	54
3.4.1	break 语句	54
3.4.2	continue 语句	55
3.4.3	exit 函数和 abort 函数	55
3.5	程序设计举例	56
	本章小结	59
	习题 3	60
第 4 章	数组	65
4.1	一维数组	65
4.1.1	一维数组的定义、初始化	65
4.1.2	一维数组元素的引用	66
4.1.3	程序设计举例	67

4.2	二维数组与多维数组	69
4.2.1	二维数组定义、初始化	69
4.2.2	二维数组元素的引用	71
4.2.3	多维数组	71
4.2.4	程序设计举例	72
4.3	字符数组和字符串	74
4.3.1	字符数组的定义与初始化	74
4.3.2	字符串与字符数组	75
4.3.3	字符数组的输入和输出	76
4.3.4	字符串处理函数	78
4.3.5	程序设计举例	80
4.4	综合程序设计举例	82
	本章小结	84
	习题 4	85
第 5 章	模块设计	89
5.1	模块的实现——函数	89
5.1.1	函数的概念	89
5.1.2	函数的定义	91
5.1.3	函数的调用	93
5.1.4	被调用函数的声明	93
5.2	函数间的参数传递	94
5.2.1	值传递	94
5.2.2	地址传递	96
5.3	变量的存储类别和作用域	99
5.3.1	变量的作用域：局部变量和全局变量	99
5.3.2	变量的存储类别	103
5.4	函数的嵌套调用与递归调用	108
5.4.1	函数的嵌套调用	108
5.4.2	函数的递归调用	109
5.5	程序设计举例	110
5.6	重载函数	112
5.6.1	进行函数重载的原因	112
5.6.2	使用函数重载的条件	113
5.6.3	重载函数的使用方法	113
5.6.4	函数的默认参数	114
5.7	函数模板与模板函数	115
5.8	内联函数	117
5.9	const 函数	118
5.10	编译预处理	118
5.10.1	文件包含命令	118

5.10.2	条件编译命令	119
5.11	宏定义与宏替换	121
5.11.1	宏定义	121
5.11.2	撤销已定义的宏	122
5.11.3	使用宏需注意的问题	122
	本章小结	124
	习题 5	124
第 6 章	指针	128
6.1	地址与指针的概念	128
6.2	指针变量的定义与引用	129
6.2.1	指针变量的定义	129
6.2.2	指针变量的引用	129
6.3	指针的运算	132
6.3.1	指针的算术运算和比较运算	132
6.3.2	指向指针变量的指针与多级指针	132
6.3.3	指向 void 类型的指针	133
6.4	指针变量作为函数参数	134
6.5	指针与数组	136
6.5.1	指向一维数组的指针变量	136
6.5.2	数组指针作函数参数	139
6.5.3	指向二维数组的指针变量	139
6.6	内存的动态分配	142
6.6.1	new 运算符的两种用法	142
6.6.2	delete 运算符	143
6.7	指针数组	144
6.7.1	指针数组的概念	144
6.7.2	带参数的 main 函数	146
6.8	指针与函数	147
6.8.1	指针型函数	147
6.8.2	指向函数的指针变量	148
6.9	const 类型的指针	150
6.9.1	常量指针(const 对象)	150
6.9.2	指针常量(const 指针)	151
6.9.3	指向常量的指针常量	151
6.10	引用	151
6.10.1	引用的概念	151
6.10.2	使用引用的注意事项	152
6.10.3	引用作为函数参数	153
6.10.4	函数返回引用	155
6.11	程序设计举例	156

本章小结	159
习题 6	162
第 7 章 类和对象	168
7.1 类的定义和使用	168
7.1.1 类定义格式	168
7.1.2 成员函数的定义位置	169
7.1.3 内联成员函数	170
7.1.4 常量成员函数	172
7.2 对象的创建和使用	173
7.2.1 对象的种类和创建	173
7.2.2 对象作为函数的参数和返回值	173
7.2.3 this 指针	175
7.3 构造函数的概念和使用	177
7.3.1 为什么引入构造函数	177
7.3.2 重载构造函数	179
7.3.3 默认参数的构造函数	180
7.3.4 复制构造函数	181
7.3.5 成员初始化参数表	184
7.4 析构函数的概念和使用	186
7.4.1 为什么引入析构函数	186
7.4.2 析构函数的使用	187
7.5 堆对象的概念和使用	188
7.5.1 创建和删除单个堆对象的方法	188
7.5.2 创建和删除堆对象数组的方法	190
7.6 静态数据成员和静态成员函数	190
7.6.1 为什么需要静态数据成员	190
7.6.2 静态数据成员的访问和初始化	192
7.6.3 静态成员函数的概念和使用	194
7.7 友元函数和友元类	195
7.7.1 友元的概念和使用	195
7.7.2 使用友元的注意事项	199
7.8 程序设计举例	199
本章小结	202
习题 7	203
第 8 章 继承与派生	205
8.1 为什么要引入继承的概念	205
8.1.1 继承与派生问题举例	205
8.1.2 继承与派生的概念	206
8.2 基类和派生类	206

8.2.1	基类与派生类的概念	206
8.2.2	派生类的定义(单继承)	206
8.3	三种派生方式	207
8.3.1	public 派生	207
8.3.2	private 派生	209
8.3.3	protected 派生	211
8.4	三种派生方式的区别	211
8.5	派生类的构造函数和析构函数	213
8.5.1	派生类的构造函数	213
8.5.2	基类构造函数的调用方式	213
8.5.3	派生类的析构函数	214
8.6	多继承和虚基类	215
8.6.1	多继承的定义	215
8.6.2	多继承中的构造函数和析构函数	216
8.6.3	二义性与虚基类	218
8.7	程序设计举例	224
	本章小结	228
	习题 8	228
第 9 章	多态性与虚函数	230
9.1	为什么需要多态性	230
9.1.1	多态性的实现方法	230
9.1.2	静态多态性和动态多态性	231
9.2	对虚函数的限制	236
9.2.1	声明虚函数的限制	236
9.2.2	虚函数的使用限制	237
9.3	在成员函数中调用虚函数	240
9.4	在构造函数中调用虚函数	241
9.5	纯虚函数和抽象类	242
9.6	程序设计举例	244
	本章小结	245
	习题 9	246
第 10 章	运算符重载与类模板	249
10.1	为什么要进行运算符重载	249
10.1.1	运算符重载的例子	249
10.1.2	注意事项	252
10.2	赋值运算符和四则运算符的重载	253
10.3	自增 1 和自减 1 运算符重载	256
10.4	关系运算符的重载	258
10.5	算术赋值运算符的重载	259

10.6	下标运算符的重载	261
10.7	插入与抽取运算符的重载	262
10.7.1	插入运算符的重载	262
10.7.2	抽取运算符的重载	263
10.8	类型转换	265
10.8.1	基本类型转换和自定义类型的相互转换	266
10.8.2	自定义类型之间的转换	267
10.9	类模板与模板类	271
10.10	程序设计举例	273
	本章小结	279
	习题 10	279
第 11 章	流	281
11.1	C 的标准 I/O 函数的缺陷	281
11.2	I/O 流的概念	282
11.3	I/O 流类库结构	283
11.4	标准 I/O 流	283
11.4.1	标准 I/O 流的类层次	283
11.4.2	预定义流对象	284
11.4.3	预定义的插入类型	284
11.4.4	预定义的抽取类型	285
11.5	用于无格式 I/O 的 ios 类成员函数	286
11.6	格式化 I/O 流	290
11.6.1	使用格式状态标志或调用格式化成员函数	290
11.6.2	使用操纵算子	294
11.7	文件操作	296
11.7.1	文件与文件流概述	296
11.7.2	文件流的类层次	297
11.7.3	文件的打开和关闭	297
11.7.4	文件读写操作举例	299
11.8	随机访问数据文件	303
11.9	程序设计举例	304
	本章小结	306
	习题 11	306
第 12 章	利用 MFC 开发 Windows 应用程序	309
12.1	Windows 应用程序的特点与消息驱动机制	309
12.1.1	基于 Windows 操作系统的应用程序的特点	309
12.1.2	典型的 Windows 应用程序结构	310
12.1.3	学习 MFC 的方法	310
12.2	利用 MFC AppWizard 创建 Windows 应用程序	310

12.3	MFC 应用程序的类和文件	311
12.3.1	类说明	311
12.3.2	文件说明	312
12.4	在窗口的客户区输出文字和图形	314
12.5	Windows 消息处理	315
12.5.1	利用 ClassWizard 编制消息处理函数	315
12.5.2	Windows 消息	318
12.5.3	消息的发送与接收的基本过程和机制	320
12.6	文档/视图结构	320
12.6.1	视图类	321
12.6.2	文档类	322
12.7	菜单、工具栏	325
12.8	对话框与控件	327
12.8.1	对话框	327
12.8.2	控件	334
12.9	数据库访问	335
12.10	程序设计举例	336
	本章小结	344
	习题 12	344
第 13 章	综合实训	346
13.1	实训 1 C++ 控制台应用程序：商品信息管理系统	346
13.2	实训 2 MFC 应用程序：班级信息管理系统	352
	本章小结	366
	习题 13	367
附录	标准 ASCII 码表	370
参考文献	371

第 1 章 程序设计与 C++ 概述

学习目标

- (1) 了解程序和程序设计语言,理解算法的含义并掌握其表示方法。
- (2) 掌握面向过程和面向对象编程的特点和不同。
- (3) 了解 C++ 的发展过程。
- (4) 掌握利用 Visual C++ 6.0 集成开发环境调试 C++ 控制台应用程序。

1.1 程序与程序设计语言

1.1.1 程序

当今的计算机系统仍采用冯·诺依曼(Von Neumann, 1903—1957)的体系结构,即存储程序结构,这说明计算机的执行必须有程序的控制;因此利用计算机解决问题,就要编写计算机程序。程序是由数据和处理数据的操作组成的。数据是操作的对象,操作的目的是对数据进行加工处理,以得到期望的结果。计算机程序是许多指令的集合,每一条指令让计算机执行完成一个具体的操作,一个程序所规定的操作全部执行完后,就能产生计算结果。

以多媒体软件播放电影为例,程序运行的基本过程如图 1.1 所示。

(1) CPU 发出指令读硬盘或外存中数据(可以在多媒体软件的打开文件对话框中输入电影文件名)。

(2) 从硬盘读数据到内存中,为程序的下一步运行做数据准备(打开文件后,读电影数据的一部分)。

(3) CPU 发出指令处理内存中的数据(多媒体播放软件处理内存中电影数据)。

(4) CPU 计算后的结果放入内存中(处理后的电影画面和音频数据暂存于内存)。

(5) 显示内存中的结果到显示器(即将内存中的电影画面在显示器中显示,音频数据送到音箱输出声音)。

(6) 继续处理内存中的数据(内存中电影数据)。

这一系列步骤由多媒体播放软件控制,多媒体播放软件由一系列程序组成。

计算机系统由软件系统和硬件系统两大部分组成,完成各种不同的任务,需要不同的软件。程序当然属于软件,但程序与软件不是同一个概念,利用计算机解决问题,需要从算法到程序,再到软件。

一个程序应包括如下信息。

(1) 对数据的描述。在程序中要指定数据的类型和数据的组织形式,即数据结构(data structure)。

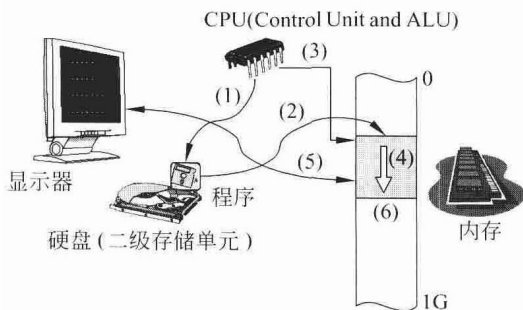


图 1.1 程序的运行过程

(2) 对操作的描述,即操作的方法和步骤,也就是算法(algorithm)。

数据是操作的对象,操作的目的是对数据进行加工处理,以便得到结果。比如,程序设计就像厨师做菜肴时需要的菜谱,通常菜谱包括两个方面:

(1) 配料:制作菜肴所需的原料(类似于程序设计中的数据结构)。

(2) 操作的方法和步骤:制作菜肴时将原料按规定的步骤加工成所需的菜肴(类似于程序设计中的算法)。

因此,著名计算机科学家沃思(Nikiklaus Wirth)提出一个公式:

数据结构+算法=程序

通常认为:

程序=算法+数据结构+程序设计方法+语言工具和环境

由此可见,编写程序是让计算机解决实际问题的关键。编制计算机程序必须具备两个基本条件:一是要掌握一门计算机高级语言的规则,二是要掌握解题的方法和步骤。

1.1.2 程序设计语言

编写程序所使用的语言称为程序设计语言,它是人与计算机之间进行信息交流的工具。从1946年第一台计算机 ENIAC 诞生至今,计算机技术突飞猛进地发展,程序设计语言的发展从低级到高级经历了机器语言、汇编语言和高级语言。

1. 机器语言

机器语言是最早出现的程序设计语言,是用0和1组成的二进制指令(机器指令)集合。程序指令直接被计算机识别和执行,运行速度快,但指令难于理解和记忆。

2. 汇编语言

为了便于理解记忆,引入助记符号来表示机器指令(如 ADD 来表示加法)。助记符替代了二进制指令代码,计算机不能直接识别汇编语言,需要编译后才能识别运行。机器语言和汇编语言是计算机低级语言,都是面向机器的语言,对于大多数用户来说,不方便理解和使用。

3. 高级语言

高级语言采用接近自然语言的命令或语句进行编程。它具有学习容易、使用方便、通用性强、移植性好等特点,便于各类人员学习和应用。早期应用比较广泛的几种高级语言有 BASIC、FORTRAN、PASCAL 和 C 等,在此之后,又诞生了上百种高级程序设计语言,并根据应用领域的不同和语言本身侧重点的差异,分成了许多种类别。

高级语言的运行方式有三种:解释执行、编译执行、先编译后解释。第一种是通过语言解释程序将源程序解释翻译一句,执行一句,如 BASIC、JavaScript、VBScript、PHP、ASP、Perl 和 PYTHON 等语言,这些语言的共同特点是运行速度慢,但简单。第二种是通过编译程序将源程序编译成二进制目标代码程序,最后执行的是目标代码程序,如 C 语言、C++ 等。显然,后者程序执行效率高、速度快,并且能够脱离语言环境独立运行。第三种是一种新型的语言,是先编译后解释执行的,如 Java 和 C# 语言。

20 世纪 60 年代末开始出现的结构化程序设计语言提高了语言的层次。结构化数据、结构化语句、数据抽象、过程抽象等概念使程序更便于体现客观事物的结构和逻辑含义。结构化程

序设计是当时程序设计的主流语言,几乎为所有的程序员所接受和使用。它的产生和发展形成了现代软件工程的基础。

面向对象技术从1986年以来逐步走出了实验室和研究部门,开始进入实际应用。面向对象语言比面向过程语言更接近于自然语言。程序员设计程序时,基本避开“怎么做”的解题过程,而是更多地考虑计算机“做什么”,计算机就会完成具体的解题过程。面向对象语言有很多种,可进行如下分类:

(1) 全新的面向对象语言。其代表是 Object-C、Smalltalk 和 Eiffel,这类语言的缺点是程序员需要从头学习一门全新的语言。

(2) 对传统语言进行面向对象的扩充。这类语言又称为混合型语言,其主要代表就是C++语言。

程序设计技术是随着编程语言的发展而发展的,编程语言的发展趋于相互同化,往往彼此相似,所以熟练掌握一种语言后,再学其他语言就不太难了。

1.1.3 程序设计

计算机毕竟不是人,要计算机干什么事情,人们都要事先替它想好解决问题的方法(叫做算法),再用计算机能理解的语言(计算机语言或程序设计语言)详细描述出来(这个描述过程即程序设计)。在计算机科学中,把解决问题的方法、步骤描述出来,这个描述的过程就是程序设计。

程序设计与软件设计不是一个概念,它们的侧重点不同。如果说软件设计是工程项目设计,则程序设计是工程中核心部分的实施过程。从另一个角度看,程序设计与软件设计之间又难以找出明显的分界线,交叉地带有很多共同的问题。整个程序设计过程为软件开发的一部分,程序设计的一般步骤如下。

1. 确定数据结构

依据所需要处理的任务要求,规划输入的数据和输出的结果,确定存放数据的数据结构。

2. 确定算法

算法是指为解决某一特定问题而采取的确定的、有限的步骤。算法有优有劣,本书实例中介绍的均为一些常用的算法。

3. 编写程序

在充分论证数据结构和算法以后才能考虑编写程序,编写程序需要结合程序设计方法(面向过程的或是面向对象的)和程序设计语言(C语言、C++语言和 Delphi 等),不同的集成开发环境写出的程序代码是有所区别的。

4. 程序调试

程序开发人员编写的程序称为源程序或源代码,源代码不能直接被计算机执行。源代码要经过编译程序编译,生成目标程序,然后链接其他相应的代码,最后生成可被计算机执行的可执行文件(.EXE 或 .COM 文件)。一个源代码有时要经过多次的修改才能编译通过,因此这一步有时是很困难的。程序在编译时,如果不能通过,则会有错误提示信息,程序员要根据错误提示信息调试程序。