

21 世纪高等学校计算机**基础**实用规划教材

C++ 语言程序设计 教程与实验 (第二版)

温秀梅 丁学钧 李建华 主编

清华大学出版社



21 世纪高等学校计算机**基础**实用规划教材

C++语言程序设计 教程与实验 (第二版)

温秀梅 丁学钧 李建华 主编

清华大学出版社
北京

内 容 简 介

本书在体系结构的安排上,将 C++ 语言的基础知识和一般的编程思想有机地结合起来,对于典型例题进行了详细的分析解释,除在每章后附有习题外,还在附录中整合了实验设计。

本书由 3 个部分组成。第 1 部分(第 1~8 章)是对 C++ 语言基本特性的介绍,有些是和 C 语言有类似的地方。第 2 部分(第 9~12 章)是关于 C++ 语言面向对象的基本思想及设计方法。这些是 C 语言中所没有的。正是这一部分,使得许多人认为 C++ 语言太复杂,为了使普通读者易于理解,我们力争把这些内容写得简明扼要,通俗易懂,而又比较完整。第 3 部分是本书的附录部分,包括有重要的实验内容设计及 Visual C++ 6.0 环境介绍,这是掌握一种编程语言的重要环节。

本着少而精的原则,全书版面清晰、结构紧凑,知识信息含量高,特别适合作为非计算机专业本科生教学或计算机应用培训班的教材,同时,还可以作为自学或函授学习的参考书。

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++ 语言程序设计教程与实验/温秀梅,丁学钧,李建华主编. —2 版. —北京:清华大学出版社,2009.3

(21 世纪高等学校计算机基础实用规划教材)

ISBN 978-7-302-19378-4

I. C… II. ①温… ②丁… ③李… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2009)第 012410 号

责任编辑:魏江江

责任校对:李建庄

责任印制:李红英

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 装 者:北京国马印刷厂

经 销:全国新华书店

开 本:185×260 印 张:19 字 数:456 千字

版 次:2009 年 3 月第 2 版 印 次:2009 年 3 月第 1 次印刷

印 数:17501~21500

定 价:26.00 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:028852-01

出版说明

随着我国改革开放的进一步深化,高等教育也得到了快速发展,各地高校紧密结合地方经济建设发展需要,科学运用市场调节机制,加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度,通过教育改革合理调整和配置了教育资源,优化了传统学科专业,积极为地方经济建设输送人才,为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是,高等教育质量还需要进一步提高以适应经济社会发展的需要,不少高校的专业设置和结构不尽合理,教师队伍整体素质亟待提高,人才培养模式、教学内容和教学方法需要进一步转变,学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007年1月,教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》,计划实施“高等学校本科教学质量与教学改革工程(简称‘质量工程’)”,通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容,进一步深化高等学校教学改革,提高人才培养的能力和水平,更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中,各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势,对其特色专业及特色课程(群)加以规划、整理和总结,更新教学内容、改革课程体系,建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上,经教育部相关教学指导委员会专家的指导和建议,清华大学出版社在多个领域精选各高校的特色课程,分别规划出版系列教材,以配合“质量工程”的实施,满足各高校教学质量和教学改革的需要。

本系列教材立足于计算机公共课程领域,以公共基础课为主、专业基础课为辅,横向满足高校多层次教学的需要。在规划过程中体现了如下一些基本原则和特点。

(1) 面向多层次、多学科专业,强调计算机在各专业中的应用。教材内容坚持基本理论适度,反映各层次对基本理论和原理的需求,同时加强实践和应用环节。

(2) 反映教学需要,促进教学发展。教材要适应多样化的教学需要,正确把握教学内容和课程体系的改革方向,在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养,为学生的知识、能力、素质协调发展创造条件。

(3) 实施精品战略,突出重点,保证质量。规划教材把重点放在公共基础课和专业基础课的教材建设上;特别注意选择并安排一部分原来基础比较好的优秀教材或讲义修订再版,逐步形成精品教材;提倡并鼓励编写体现教学质量和教学改革成果的教材。

(4) 主张一纲多本,合理配套。基础课和专业基础课教材配套,同一门课程有针对不同层次、面向不同专业的多本具有各自内容特点的教材。处理好教材统一性与多样化,基本教材与辅助教材、教学参考书,文字教材与软件教材的关系,实现教材系列资源配套。

(5) 依靠专家,择优选用。在制定教材规划时依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主题。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平教材编写梯队才能保证教材的编写质量和建设力度,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

21 世纪高等学校计算机基础实用规划教材

联系人:魏江江 weijj@tup.tsinghua.edu.cn

第二版前言

《C++ 语言程序设计教程与实验》教材 2004 年出版后,多次重印,受到读者好评,被许多高校选做教材。根据师生反馈意见,我们及时修编原书,形成了本版教材。

修编后的教材保持了与原书内容、风格一致,使采用原书的教师可以自然地过渡到新版教材。这次修编主要就以下几个方面进行了改进。

根据教学改革、实践教学的需要和教师多年的教学经验,主要修改、增补了第 3 章、第 4 章的例题;在第 5 章中增加了函数的引用调用机制和函数调用中的数据流;修改了第 12 章中的文件输入输出流;修改了实验内容;新增了附录 E 课程设计任务书,供课程设计者使用。

作为教材,使用者可以根据教学大纲和学时安排,选取相应的内容进行教学。如果课时不够,第 9 章到第 12 章面向对象的内容可以不讲,而只讲结构化程序设计;第 12.2 节“文件输入输出流”的内容可以提前到前面的任一章节中讲授。

由于作者水平有限,书中难免有疏漏和错误之处,恳请广大专家和读者指正。

编 者

2009 年 2 月

第一版前言

作为一种计算机语言,C++语言有很多优点。它既可以进行过程化程序设计,也可以进行面向对象程序设计,很多复杂的算法和设计可以比较容易地用 C++ 来实现。当前,C++ 已经被普遍地应用于科学技术和日常生活的各个领域。

在编写本书之前,本书作者已在高校从事多年的“C 语言及 C++ 语言程序设计”教学及研究工作,对于该语言的概念、功能及应用有着较深入的理解和丰富的实践经验。在教学过程中,我们发现目前已有的大部分教材不能很好地适应教学需求,故组织编写了这本教材,旨在通过本教材规范本科非计算机专业“C++ 语言程序设计”课程的教与学。

作为一本教程,本书有以下一些特点。

(1) 本书在体系结构的安排上将 C++ 语言基础知识和一般的编程思想有机结合,对于典型例题进行了详细的分析解释,除在每章后附有习题外,还在附录中整合了实验设计。因此,全书结构严谨,通俗易懂,兼有普及与提高的双重功能。

(2) 计算机等级考试是面向社会推出的一种客观、公正和科学的水平测试,用以测试非计算机专业人员的计算机应用知识和技能。本书参考全国及河北省计算机二级考试(C/C++ 语言)大纲的要求编写而成,内容覆盖大纲的所有内容,编排上由浅入深,重点难点突出,并在书后附有两套等级考试的模拟题。

(3) 本书中对于语言的描述是与平台无关的,只要有标准的 C++ 编译器的支持,它适合于不同的操作系统,如 Windows(Visual C++ 6.0)及 DOS(Borland C++ 3.1)。学习本书的前提只需要基本的微积分和线性代数知识。因为本书包含很多常用的计算方法及其 C++ 源程序,所以特别适合于各类非计算机专业的本科生使用。

(4) 本书由 3 个部分组成。第 1 部分(第 1~8 章)是对 C++ 语言基本特性的介绍,有些是和 C 语言类似的地方。第 2 部分(第 9~12 章)是关于 C++ 面向对象的基本思想及设计方法。这些是 C 语言中所没有的,正是这一部分,使得许多人认为 C++ 太复杂。为了使普通读者易于理解,我们力争把这些内容写得简明扼要,而又比较完整。第 3 部分是本书的附录部分,包括有重要的实验内容设计及 Visual C++ 6.0 集成开发环境介绍,这是掌握一种编程语言的重要环节。

(5) 本着少而精的原则,全书版面清晰、结构紧凑,知识信息含量高,特别适合作为非计算机专业本科生教学或计算机应用培训班的教材,同时,还可以作为自学或函授学习的参考书。

本书由温秀梅、丁学钧任主编及完成统稿,李耀辉、刘建臣任副主编。参加编写者有温秀梅(第 1、9、10、11、12 章,附录 A、B、C)、丁学钧(第 2 章、附录 E)、祁爱华(第 3 章)、周丽莉

(第4章)、李耀辉(第5、6章)、王振岩(第7、8章)、王利霞(附录D),刘建臣担任本书的审校工作。参加本书大纲讨论及部分编写工作的老师还有陈树有、李建华、刘瑞梅、岳杰和赵巍等。由于时间仓促,加之我们的水平有限,书中难免有疏漏和不足之处,恳请广大读者和专家指正。

编 者

2004 年 1 月

目 录

第 1 章 绪论	1
1.1 C++语言发展史简介	1
1.2 程序设计概述	2
1.2.1 程序设计的发展历程	2
1.2.2 结构化程序设计	3
1.2.3 面向对象程序设计	3
1.3 基本语法成分	4
1.3.1 字符集	4
1.3.2 标识符	5
1.3.3 关键字	5
1.3.4 运算符	6
1.3.5 分隔符	6
1.3.6 空白符	6
1.4 一个简单的 C++ 程序	7
第 2 章 基本数据类型、运算符与表达式	10
2.1 数据类型概述	10
2.2 常量与变量	11
2.2.1 常量	11
2.2.2 变量	11
2.3 基本类型	12
2.3.1 整型	12
2.3.2 实型	14
2.3.3 字符型	15
2.3.4 布尔类型	16
2.3.5 void 类型	16
2.4 运算符和表达式	17
2.4.1 赋值运算符和赋值表达式	17
2.4.2 算术运算符和算术表达式	19
2.4.3 关系运算符和关系表达式	20

2.4.4	逻辑运算符和逻辑表达式	21
2.4.5	条件运算符和条件表达式	22
2.4.6	逗号运算符和逗号表达式	22
2.5	类型转换	22
2.5.1	自动类型转换	23
2.5.2	强制类型转换	23
习题		24
第3章	结构化程序设计	25
3.1	C++语言输入输出流	25
3.1.1	C++语言无格式输入输出	25
3.1.2	C++语言格式输入输出	27
3.2	结构化程序设计概述	32
3.3	顺序结构程序设计	33
3.3.1	顺序结构	33
3.3.2	程序举例	34
3.4	选择结构程序设计	35
3.4.1	用 if 语句实现选择结构设计	35
3.4.2	用 switch 语句实现选择结构设计	39
3.5	循环结构程序设计	41
3.5.1	while 语句	42
3.5.2	do-while 语句	43
3.5.3	for 语句	44
3.5.4	跳转语句 break 和 continue	47
3.5.5	循环的嵌套	49
3.6	程序设计举例	50
习题		53
第4章	数组	56
4.1	一维数组	56
4.1.1	一维数组的定义	56
4.1.2	一维数组元素的引用	57
4.1.3	一维数组的初始化	58
4.1.4	一维数组程序举例	58
4.2	二维数组	61
4.2.1	二维数组的定义	61
4.2.2	二维数组元素的引用	62
4.2.3	二维数组的初始化	63
4.2.4	二维数组程序举例	65

4.3 字符数组	66
4.3.1 字符数组的定义	66
4.3.2 字符数组的初始化	66
4.3.3 字符数组的使用	68
4.3.4 字符数组程序举例	70
4.3.5 字符串处理函数	71
4.3.6 字符串程序举例	73
习题	74
第5章 函数	75
5.1 函数的定义	75
5.1.1 定义函数	75
5.1.2 函数原型	77
5.2 函数的调用	78
5.2.1 调用函数	78
5.2.2 参数传递机制	81
5.2.3 函数返回值	84
5.2.4 函数调用中的数据流	85
5.3 函数的嵌套调用	86
5.4 递归函数	90
5.5 作用域与生命期	93
5.5.1 作用域	93
5.5.2 全局变量和局部变量	95
5.5.3 生命期	98
习题	103
第6章 指针	106
6.1 指针的基本概念	106
6.1.1 指针的概念	106
6.1.2 指针变量的定义	107
6.1.3 指针变量运算符	108
6.1.4 指针的初始化与赋值	110
6.1.5 指针的运算	113
6.2 指针与数组	116
6.2.1 指向数组的指针	116
6.2.2 指针与字符数组	119
6.2.3 多级指针与指针数组	123
6.2.4 指针与多维数组	127
6.2.5 数组指针	130

6.3 指针与函数	131
6.3.1 指针作为函数参数	131
6.3.2 函数调用中数组的传递	133
6.3.3 函数指针	135
习题	138
第7章 编译预处理命令	139
7.1 宏定义	139
7.1.1 不带参数的宏定义	139
7.1.2 带参数的宏	141
7.2 文件包含	143
7.3 条件编译	144
习题	146
第8章 结构体、共用体和枚举类型	148
8.1 结构体类型	148
8.2 定义结构体类型变量	149
8.2.1 先声明结构体类型再定义变量	149
8.2.2 声明结构体类型的同时定义变量	150
8.2.3 直接定义结构体类型变量	150
8.2.4 结构体变量的初始化	151
8.3 结构体变量成员的引用	152
8.4 结构体数组	154
8.4.1 结构体数组的定义	154
8.4.2 结构体数组的初始化	154
8.4.3 结构体数组应用举例	155
8.5 结构体指针	156
8.5.1 指向结构体变量的指针	156
8.5.2 指向结构体数组的指针	157
8.5.3 用结构体变量和指向结构体的指针作为函数参数	157
8.6 用指针处理链表	158
8.6.1 链表的概述	158
8.6.2 动态内存分配	159
8.6.3 建立单向动态链表	160
8.6.4 输出链表	162
8.6.5 对链表的删除操作	163
8.6.6 对链表的插入操作	164
8.7 共用体	165
8.7.1 共用体的概念	165

8.7.2	共用体类型和共用体类型变量的定义	165
8.7.3	共用体变量引用的方式	166
8.7.4	共用体类型的特点	166
8.8	枚举类型	167
8.8.1	枚举类型及枚举变量的定义	167
8.8.2	枚举元素的引用	168
8.8.3	用 typedef 定义类型	169
	习题	169
第 9 章	面向对象程序设计基础	171
9.1	面向对象程序设计概述	171
9.1.1	面向对象是软件方法学的返璞归真	171
9.1.2	面向对象程序设计语言的四大家族	171
9.1.3	面向对象程序分析 OOA 与设计 OOD 的基本步骤	172
9.2	类和对象	173
9.2.1	类	173
9.2.2	对象	176
9.2.3	名字解析和 this 指针	177
9.3	带默认参数的函数和函数重载	178
9.3.1	带默认参数的函数	178
9.3.2	函数重载	179
9.4	构造函数和析构函数	181
9.4.1	构造函数	181
9.4.2	析构函数	187
9.5	对象成员和静态成员	189
9.5.1	对象成员	189
9.5.2	静态成员	190
9.6	友元	191
9.7	类模板和模板类	195
	习题	196
第 10 章	继承与派生	198
10.1	单一继承	198
10.1.1	继承与派生	198
10.1.2	派生类的定义	198
10.1.3	类的继承方式	200
10.1.4	派生类的构造函数和析构函数	204
10.2	多重继承	207
10.2.1	多重继承的概念和定义	207

10.2.2	二义性和支配规则	208
10.2.3	赋值兼容规则	209
10.3	虚基类	209
10.3.1	虚基类的概念	209
10.3.2	多重继承的构造函数和析构函数	210
	习题	213
第 11 章	多态性与虚函数	214
11.1	运算符重载	214
11.1.1	什么是运算符重载	214
11.1.2	用成员函数重载运算符	215
11.1.3	用友元函数重载运算符	216
11.1.4	几个运算符的重载	219
11.2	虚函数	223
11.2.1	为什么要引入虚函数	223
11.2.2	虚函数的定义与使用	224
11.3	纯虚函数和抽象类	228
11.3.1	纯虚函数的概念	228
11.3.2	抽象类的概念	228
11.4	虚析构函数	230
	习题	232
第 12 章	输入输出流	234
12.1	标准输入输出流	234
12.1.1	输入输出流的概念	234
12.1.2	C++ 所有输入输出类的继承关系	234
12.2	文件输入输出流	238
	习题	241
附录 A	程序的调试与运行	242
附录 B	标准字符 ASCII 表	252
附录 C	实验	254
附录 D	模拟考试题	259
附录 E	课程设计任务书	283
附录 F	参考课时安排	286

1.1 C++ 语言发展史简介

C++ 语言是从 C 语言发展演变而来的,因此在介绍 C++ 语言之前,首先介绍一下 C 语言。1972 年至 1973 年间,美国贝尔实验室的 Denis. M. Ritchie 在一台 DEC PDP-11 计算机上实现了基于 B 语言的最初的 C 语言。C 语言的产生基于两个方面的需要:一是为满足 UNIX 操作系统开发的需要,二是为拉近高级语言与硬件之间距离的需要。目前,比较流行的 C 语言版本基本上都是以 ANSI C 为基础的。

C 语言具有以下一些特点:

- (1) 语言简洁、紧凑,使用方便、灵活。
- (2) 运算符和数据结构丰富,具有结构化的控制语句,生成目标代码质量高,程序执行效率高。
- (3) 语法限制不太严格,程序设计自由度过大。
- (4) 与其他高级语言相比,具有可以直接访问物理地址,能进行位运算的优点。
- (5) 与汇编语言相比,具有良好的可读性和可移植性。

C 语言盛行的同时,其局限性也逐渐暴露出来:

(1) C 语言类型检查机制相对较弱,这使得程序中的一些错误不能在编译时被发现,这些错误若是遗留到程序的运行阶段由程序员来检查,将是很困难的。

(2) C 语言本身几乎没有支持代码重用的语言结构,因此一个程序员精心设计的程序,很难为其他程序所用。

(3) C 语言不适合开发大型程序,当程序的规模达到一定的程度时,程序员很难控制程序的复杂性。

C 语言毕竟是一个面向过程的编程语言,因此与其他面向过程的编程语言一样,已经不能满足目前运用面向对象方法开发软件的需要。为解决上述问题,并保持 C 语言的简洁、高效和接近汇编语言的特点,1980 年贝尔实验室的 Bjarne Stroustrup 博士及其同事开始对 C 语言进行改进和扩充,最初称为“带类的 C”,1983 年取名为 C++,以后又经过不断完善和发展成为目前的 C++ 语言。C++ 语言包含了 C 语言的全部特征、属性和优点,同时,C++ 语言添加了对面向对象程序设计(Object Oriented Programming, OOP)的完全支持。C 语言是建立 C++ 语言的基础,这就使许多 C 语言代码稍加修改甚至不经修改就可以为 C++ 语言所用,用 C 语言编写的众多的库函数和实用软件可以用于 C++ 语言中。另外,用 C++ 语言编写的程序,可读性更好且代码结构更为合理,可以更直接地在程序中映射问题空间的结构。更重要的是,C 语言程序员仅需学习 C++ 语言的新特征就可以很快地用 C++ 语言编写程序。

C++语言已被应用于程序设计的众多应用领域,它尤其适用于中等和大型程序开发项目。C++语言是C语言发展的新阶段,是一种更好的C语言,是应用广泛的面向对象的程序设计语言。

1.2 程序设计概述

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。操作步骤也就是算法,即求解问题的方法和步骤。下面简单介绍程序设计的发展历程以及结构化和面向对象两种程序设计方法。

1.2.1 程序设计的发展历程

回顾程序设计的发展史,大体上可划分为以下几个不同的时期。

20世纪50年代的程序都是用指令代码或汇编语言编写的,这种程序的设计相当麻烦,编制和调试一个稍大一点的程序常常要花费很长时间,培养一个熟练的程序员更需经过长期训练和实习,这种局面严重影响了计算机的普及应用。

20世纪60年代高级语言的出现大大简化了程序设计,缩短了解题周期,因此显示出强大的生命力。此后,编制程序已不再是只有软件专业人员才能做的事了,一般工程技术人员花上较短的时间学习,也可以使用计算机解题。这个时期,随着计算机的应用日益广泛地渗透到各学科和技术领域,发展了一系列不同风格的、为不同目标服务的程序设计语言。其中较为著名的有FORTRAN、COBOL、ALGOL、LISP、PL/1、PASCAL等十几种语言。高级语言的蓬勃兴起,使得编译和形式语言理论相应日趋完善,这是该时期的主要特征。但就整个程序设计方法而言,并无实质性的改进。

自60年代末到70年代初,出现了大型软件系统,如操作系统、数据库,这给程序设计带来了新的问题。大型系统的研制需要花费大量的资金和人力,可是研制出来的产品却可靠性差、错误多,并且不易维护和修改。一个大型操作系统有时需要几千人/年的工作量,而所获得的系统又常常会隐藏着几百甚至几千个错误。当时,人们称这种现象为“软件危机”。

为了克服60年代出现的软件危机,1968年北约组织提出“软件工程”的概念。对程序设计语言的认识从强调表达能力为重点转向以结构化和简明性为重点,将程序从语句序列转向相互作用的模块集合。1969年,E. W. Dijkstra首先提出了结构化程序设计的概念,他强调了从程序结构和风格上来研究程序设计。在软件工程的迫切要求下,70年代结构化语言获得蓬勃发展并得到广泛应用。使用结构化程序设计方法可显著地减少软件的复杂性,提高软件的可靠性、可测试性和可维护性。经过几年的探索和实践,结构化程序设计的应用确实取得了成效,用结构化程序设计的方法编写出来的程序不仅结构良好,易写易读,而且易于证明其正确性。

进入80年代,由于一系列高技术的研究,如第五代计算机、CAM和知识工程等领域的研究都迫切要求大型的软件系统作为支撑。在这些技术研究中所用的数据类型也超出了常规的结构化数据类型的范畴,提出了对图像、声音及规则等非结构化信息进行管理的要求。为了适应这些应用领域的需要,迫切要求软件模块具有更强的独立自治性,以便于大型软件的管理、维护和重用。由于结构化语言的数据类型较为简单,所以不能胜任对非结构化数据

的定义与管理,采用过程调用机制也不够灵活,独立性较差。

为了适应高技术发展的需要,消除结构化编程语言的局限,自 80 年代以来,出现了面向对象程序设计流派,研制出了多种面向对象编程语言(简称为 OOPL),如 Ada、Smalltalk、C++ 和当前使用在 Internet 上与平台无关的 Java 语言等。

由于 OOPL 的对象、类具有高度的抽象性,所以它能很好地表达任何复杂的数据类型,也允许程序员灵活地定义自己所需要的数据类型。类本身具有很完整的封装性,可以使用它作为编程中的模块单元,满足模块独立自主的需求。再加上继承性和多态性,更有助于简化大型软件和大量重复定义的模块,增强了模块的可重用性,提高了软件的可靠性,缩短了软件的开发周期。

1.2.2 结构化程序设计

采取以下一些方法就可以得到结构化的程序。

1. 自顶向下,逐步细化

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,直到这些完备的子任务小到易于理解的程度。这种方法称为“自顶向下,逐步细化”。

2. 模块化设计

在程序设计中常采用模块化设计的方法,尤其是当程序比较复杂时,更有必要。在拿到一个程序模块(实际上是程序模块的任务书)以后,根据程序模块的功能将它划分为若干个子模块,如果嫌这些子模块的规模大,还可以划分为更小的模块。这个过程采用自顶向下方法来实现。结构化程序设计方法可以解决人脑思维能力的局限性和所处理问题的复杂性之间的矛盾。

3. 结构化编码

在设计好一个结构化的算法之后,还要善于进行结构化编码,即用高级语言的语句正确地实现顺序、选择和循环三种基本结构。

1.2.3 面向对象程序设计

1. 对象

对象是计算机内存中的一块区域,通过将内存分块,每个模块(即对象)在功能上相互之间保持相对独立。当对象的一个成员函数被调用时,对象便执行其内部的代码来响应这个调用,这使对象呈现出一定的行为。行为及其结果就是该对象的功能。

2. 面向对象

面向对象是一种认识世界的方法也是一种程序设计方法。面向对象的观点认为,客观世界是由各种各样的实体,也就是对象组成的。每种对象都有自己的内部状态和运动规律,不同对象间的相互联系和相互作用就构成了各种不同的系统,进而构成整个客观世界。按照这样的思想设计程序,就是面向对象的程序设计。“面向对象”不仅仅作为一种技术,更作为一种方法贯穿于软件设计的各个阶段。

3. 面向对象程序设计

面向对象程序设计是在面向过程的程序设计基础上的质的飞跃。面向对象方法的产