



普通高等教育“十一五”国家级规划教材

中国高等学校信息管理与信息系统专业规划教材

丛书主编：陈国青

# 程序设计基础与数据结构

戚桂杰 姚云鸿 主编

刘位龙 副主编

张新 主审



根据教育部管理科学与工程类学科专业教学指导委员会主持鉴定的《中国高等院校信息系统学科课程体系》组织编写



与美国ACM和IEEE/CS Computing Curricula 2005同步



清华大学出版社

中国高等学校信息管理与信息系统专业规划教材

丛书主编：陈国青

---

# 程序设计基础与数据结构

---

戚桂杰 姚云鸿 主编

刘位龙 副主编

张新 主审

清华大学出版社

北京

## 内 容 简 介

本书包括程序设计基础与数据结构两部分内容。在程序设计基础部分,讲述了C语言程序设计方法,主要包括程序设计基本概念、数据类型、程序控制、指针、函数、文件和大型程序设计原理与方法等内容。在数据结构部分,介绍了线性表、栈、队列、树、二叉树、查找表等基本数据结构以及相应的算法和相关应用。数据结构部分的所有算法皆以C语言为描述语言。

本书注重理论联系实际,概念清晰,论述严谨,通俗易懂。

本书可作为信息管理与信息系统专业或计算机类相关专业的本科或专科教材,也可供编程爱好者学习参考。

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

程序设计基础与数据结构/戚桂杰,姚云鸿主编.—北京:清华大学出版社,2008.3

(中国高等学校信息管理与信息系统专业规划教材)

ISBN 978-7-302-16068-7

I. 程… II. ①戚… ②姚… III. ①程序设计—高等学校—教材 ②数据结构—高等学校—教材 IV. TP311.1

中国版本图书馆CIP数据核字(2007)第139123号

责任编辑:索梅 李玮琪

责任校对:李建庄

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

[c-service@tup.tsinghua.edu.cn](mailto:c-service@tup.tsinghua.edu.cn)

社总机:010-62770175

邮购热线:010-62786544

投稿咨询:010-62772015

客户服务:010-62776969

印刷者:北京密云胶印厂

装订者:三河市新茂装订有限公司

经 销:全国新华书店

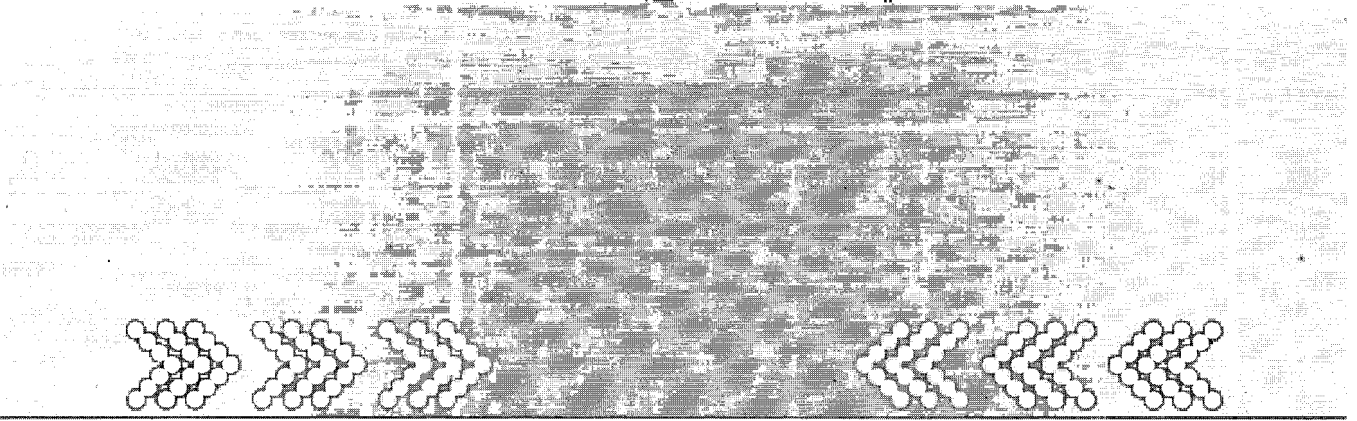
开 本:185×260 印 张:25.5 字 数:616千字

版 次:2008年3月第1版 印 次:2008年3月第1次印刷

印 数:1~4000

定 价:38.00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:024754-01



中国高等学校信息管理与信息系统专业规划教材

## 编写委员会

主 任 陈国贵

副主任 陈 禹

委 员	毛基业	王刊良	左美云	甘仞初	刘 鲁
	朱 岩	严建援	张 新	张朋柱	张金龙
	李 东	李一军	杨善林	陈晓红	陈智高
	崔 巍	戚桂杰	黄丽华	赖茂生	



# 序



在信息技术刚刚兴起的时候,信息系统还没有作为一个专门的学科独立出来,它更多的只是计算机学科的一个附属。但是,随着信息技术的跳跃式发展和计算机系统在生产、生活、商务活动中的广泛应用,信息系统作为一个独立的整体逐渐独立出来,并得到了迅速发展。由于信息系统是基于计算机技术、系统科学、管理科学以及通信技术等多个学科的交叉学科,因此,信息系统是一个跨专业,面向技术和管理等多个层面,注重将工程化的方法和人的主观分析方法相结合的一门学科。

早在1984年,邓小平同志就提出了要开发信息资源,服务四个现代化(工业现代化、农业现代化、国防现代化和科学技术现代化)建设。1990年,江泽民同志曾经指出,四个现代化恐怕无一不和电子信息化有着紧密的联系,要把信息化提到战略地位上来,要把信息化列为国民经济发展的一个重要方针。2004年,胡锦涛同志在APEC(亚洲太平洋经济合作组织)上的讲话明确指出:“信息通信技术改变了传统的生产方式和商业模式,为亚太地区带来了新的经济增长机遇。为把握住这一机遇,我们应抓住加强信息基础设施建设和人力资源开发这两个关键环节。”我国的经济目前正处在迅速发展阶段,信息化建设正在成为我国增强国力的一个重要举措,信息管理人才的培养至关重要。因此,信息系统学科面临着新的、更为广阔的发展空间。

近年来,我国高等学校管理科学与工程一级学科下的“信息管理与信息系统”专业领域的科研、教学和应用等方面都取得了长足的进步,培养了一大批优秀的技术和管理人才。但在整体水平上与国外发达国家相比还存在着不小的差距。由于各所高校在相关专业的历史、特点和背景上的差异以及社会对人才需求的多样化,使得我国信息管理与信息系统专业教育面临着前进中的机遇和挑战。如何适应人才需求变化进行教育改革和调整;如何在基本教学规范和纲要的基础上建立自己的教育特色,如何更清晰地定义教育对象和定位教育目标及体系,如何根据国际主流及自身特点更新知识和教材体系等都是我们在专业教育和学科建设中需要探讨和考虑的重要课题。

2004年,教育部高等学校管理科学与工程类专业教学指导委员会制订了学科的核心课程以及相关各主干课程的教学基本要求(简称《基本要求》)。其中,“管理信息系统”是学科的核心课程之一,“系统分析与设计”、“数据结构与数据库”、“信息资源管理”和“计算机网络”是信息管理与信息系统专业的主干课程。该《基本要求》反映了相关专业所应构建的最基本的核心课程和主干课程系统以及涉及的最基本的知识元素,旨在保证必要的教学规范,提升我国高等学校相关专业教育的基础水平。

2004年6月,IEEE/ACM公布了“计算教程CC2004”(Computing Curriculum 2004),其中包括由国际计算机学会(ACM)、信息系统学会(AIS)和信息技术专业协会(AITP)共同



提出的信息系统学科的教学参考计划和课程设置 (IS2002)。与过去的历届教程相比, IS2002 比较充分地体现出“技术与管理并重”这一当前信息系统学科领域的主流特点。IS2002 中的信息系统学科也涵盖了“信息管理”(IM)、“管理信息系统”(MIS)等相关专业, 与我国的信息管理与信息系统专业也相兼容。

为了进一步提高我国高等学校信息系统学科领域课程体系的规划性和前瞻性, 反映国际信息系统学科的主流特点和知识元素, 进一步体现我国相关专业教育的特点和发展要求, 清华大学经济管理学院与中国人民大学信息学院共同组织, 于 2004 年秋成立了“中国高等院校信息系统学科课程体系 2005”(CISC2005) 课题组, 通过对国内外信息系统的发展现状与趋势进行分析, 参照 IS2002 的模式, 课题组研究探讨了我国信息系统教育的指导思想、课程体系、教学计划, 确定了课程体系的基础内容与核心内容, 制订出了一个符合我国国情的信息管理与信息系统学科的教育体系框架, 我们希望 CISC2005 有助于我国信息管理与信息系统学科的建设, 促进我国信息化人才的培养。

2006 年, 根据 CISC2005 的指导思想编写的系列教材——《中国高等学校信息管理与信息系统专业规划教材》被列入教育部普通高等教育“十一五”国家级规划教材。同年, CISC2005 通过了教育部高等学校管理科学与工程类专业教学指导委员会组织的专家鉴定。为了能够使这套教材尽快出版, 课题组成员和清华大学出版社一道, 对教材进行了详细规划, 并组织了国内相关专家学者共同努力, 力争从 2007 年起陆续使这套教材和读者见面。希望这套教材的出版能够满足国内高等学校对信息管理与信息系统专业教学的要求, 并在大家的努力下, 在使用中逐渐完善和发展, 从而不断提高我国信息管理与信息系统人才的培养质量。

陈国青

2007 年 10 月



# 前言



程序设计基础与数据结构是信息管理与信息系统专业必修的一门核心课程,该课程的目的是培养学生的程序设计理念,使学生能掌握程序设计的基本方法,了解最常用数据结构的逻辑特性、存储表示以及对这些数据结构的操作算法,并通过课程学习与上机实习进行复杂程序设计的训练,为学习本专业后续相关专业课程打下坚实的基础。

作者以《中国高等院校信息系统学科课程体系 2005》为依据,针对该课程所要求的知识点,结合作者多年的教学及应用方面的经验,设计了本书的体系结构并在内容取舍上做了精心的安排。本书包括程序设计基础与数据结构两部分内容,并以 C 程序设计语言为工具介绍相应内容。

全书共 13 章。第 1 章至第 7 章为程序设计基础的内容,包括程序设计的基本概念,数据类型、运算符与表达式,程序控制语句,函数与数组,指针与结构体,文件,大型程序的设计与调试等;第 8 章至第 13 章为数据结构的内容,包括数据结构的基本概念,线性结构,树和二叉树,排序,查找,文件组织等。

本书在内容取舍及写法上力求条理清晰、层次分明、简明扼要、循序渐进,便于读者学习。按照本课程及本书的设想,读者将通过程序设计基础内容的学习,掌握程序设计的基本知识和基本方法,为学习数据结构打下一定的基础;在随后的数据结构内容学习中,不仅掌握数据结构的相关知识,也通过其中数据结构相关操作算法的实现及理解,进一步提高程序设计的能力。

本书由戚桂杰、姚云鸿担任主编,刘位龙、贾晓毅担任副主编,参加本书编写的主要人员还有李楠、王颖、王翠翠、焦丽媛、张海涛、李明明、曹刚、李亚伟。全书由戚桂杰、姚云鸿提出整体框架、组织讨论编写计划并负责统稿工作。本书的出版得到了清华大学经济管理学院常务副院长陈国青教授、中国人民大学信息学院陈禹教授、山东大学管理学院院长徐向艺教授的诸多指导和大力支持,担任本书主审的山东经济学院张新教授对本书提出了许多中肯的意见,清华大学出版社的索梅女士、李玮琪女士为本书的顺利出版提供了莫大的帮助,在此表示衷心的感谢。此外,在本书的写作过程中,参阅和引用了国内外有关专家学者的论著,因篇幅所限,可能未全部一一列出,在此一并致谢。由于时间仓促及水平有限,书中难免有不当之处,敬请读者批评指正。

戚桂杰 姚云鸿

2008 年 1 月于山东大学

# 目录

第1章 程序设计的基本概念	1		
1.1 程序设计和程序设计语言	1		
1.1.1 程序与软件	1		
1.1.2 程序设计语言的发展	2		
1.1.3 计算机如何运行程序	2		
1.2 C语言概述	4		
1.2.1 C语言的发展	4		
1.2.2 C程序的结构	4		
1.2.3 C语言字符集	6		
1.2.4 C语言的特点	7		
1.3 算法	8		
1.3.1 问题求解过程	8		
1.3.2 算法及其表示	9		
1.3.3 简单算法举例	9		
1.3.4 用流程图表示算法	11		
1.3.5 三种基本算法结构和改进的流程图	13		
1.3.6 用N-S流程图表示算法	14		
1.3.7 用伪代码表示算法	16		
1.3.8 用计算机语言表示算法	17		
1.4 结构化程序设计方法	18		
1.5 本章小结	18		
习题1	19		
第2章 数据类型、运算符与表达式	20		
2.1 C语言的数据类型	20		
2.2 标识符	21		
2.3 常量与变量	22		
2.3.1 常量和符号常量	22		
2.3.2 变量	23		
2.4 整型数据	23		
2.4.1 整型常量的表示方法	23		
2.4.2 整型变量	24		
2.5 实型数据	27		
2.5.1 实型常量的表示方法	27		
2.5.2 实型变量	28		
2.5.3 实型常量的类型	29		
2.6 字符型数据	29		
2.6.1 字符常量	29		
2.6.2 转义字符	29		
2.6.3 字符变量	30		
2.6.4 字符数据在内存中的存储形式及使用方法	30		
2.6.5 字符串常量	32		
2.7 变量赋初值	32		
2.8 各类数值型数据之间的混合运算	32		
2.8.1 自动类型转换	33		
2.8.2 强制类型转换	33		
2.9 算术运算符和算术表达式	34		
2.9.1 C运算符简介	34		
2.9.2 算术运算符和算术表达式	35		
2.10 赋值运算符和赋值表达式	37		
2.10.1 赋值运算符	37		
2.10.2 类型转换	37		
2.10.3 复合的赋值运算符	38		
2.11 逗号运算符和逗号表达式	38		
2.12 本章小结	39		
习题2	40		
第3章 C程序语句	42		
3.1 C程序语句概述	42		





<b>3.2 顺序程序设计</b>	<b>43</b>	<b>4.3 函数调用</b>	<b>78</b>
3.2.1 数据输入输出的概念及在 C 语言中的实现	44	4.3.1 函数调用的一般形式	78
3.2.2 字符数据的输入输出	44	4.3.2 函数嵌套调用与递归调用	79
3.2.3 格式输入与输出	46	<b>4.4 函数作用域规则</b>	<b>82</b>
3.2.4 顺序结构程序设计举例	50	4.4.1 局部变量与外部变量	83
<b>3.3 选择结构程序</b>	<b>51</b>	4.4.2 动态存储变量与静态存储变量	86
3.3.1 关系运算符和关系表达式	51	4.4.3 内部和外部函数的说明	88
3.3.2 逻辑运算符和表达式	53	<b>4.5 数组</b>	<b>89</b>
3.3.3 if 语句	54	4.5.1 一维数组	89
3.3.4 switch 语句	59	4.5.2 二维数组	91
3.3.5 程序举例	61	4.5.3 数组应用程序举例	92
<b>3.4 循环控制</b>	<b>62</b>	4.5.4 字符数组	94
3.4.1 概述	62	<b>4.6 预处理命令</b>	<b>98</b>
3.4.2 goto 语句以及用 goto 语句构成循环	62	4.6.1 宏定义	98
3.4.3 while 语句	63	4.6.2 文件包含	104
3.4.4 do-while 语句	64	4.6.3 条件编译	104
3.4.5 for 语句	66	<b>4.7 本章小结</b>	<b>106</b>
3.4.6 循环的嵌套	68	<b>习题 4</b>	<b>107</b>
3.4.7 几种循环的比较	68	<b>第 5 章 指针与结构体</b>	<b>109</b>
3.4.8 break 和 continue 语句	69	<b>5.1 指针概述</b>	<b>109</b>
3.4.9 程序举例	71	5.1.1 指针是地址	109
<b>3.5 本章小结</b>	<b>72</b>	5.1.2 指针变量及其定义	110
<b>习题 3</b>	<b>73</b>	5.1.3 指针变量作函数参数	111
<b>第 4 章 函数与数组</b>	<b>74</b>	<b>5.2 数组与指针</b>	<b>115</b>
<b>4.1 函数概述</b>	<b>74</b>	5.2.1 一维数组与指针	116
<b>4.2 函数定义的一般形式</b>	<b>75</b>	5.2.2 多维数组与指针	119
4.2.1 类型标识符与 return 语句	75	5.2.3 字符串与指针	122
4.2.2 形式参数和实际参数	76	5.2.4 指针数组	124
		<b>5.3 指针与函数</b>	<b>127</b>
		5.3.1 函数指针变量	127



5.3.2 指针型函数	129	6.3.2 数据块读写函数 fread()和 fwrite()	161
5.3.3 多级指针	130	6.3.3 字符串读写函数 fgets()和 fputs()	163
5.3.4 指针数组作 main()的参数	132	6.3.4 格式读写函数 fscanf()和 fprintf()	165
<b>5.4 结构体</b>	133	6.3.5 其他输入输出函数	166
5.4.1 结构体类型的定义	133	<b>6.4 文件的定位</b>	167
5.4.2 结构体类型变量的定义	133	6.4.1 ftell()函数	167
5.4.3 结构体变量的引用	135	6.4.2 文件头复位函数 rewind()	167
5.4.4 结构体变量的初始化	136	6.4.3 文件的随机定位函数 fseek()	168
5.4.5 结构体数组	136	<b>6.5 出错的检测</b>	169
5.4.6 结构体与指针	138	<b>6.6 典型实例</b>	169
<b>5.5 共用体</b>	139	<b>6.7 常见的编程错误</b>	173
5.5.1 共用体的概念	139	<b>6.8 本章小结</b>	173
5.5.2 共用体变量的引用	140	<b>习题 6</b>	174
<b>5.6 其他数据类型</b>	141	<b>第 7 章 大型程序的设计与调试</b>	175
5.6.1 枚举类型	141	<b>7.1 大型程序的开发过程</b>	175
5.6.2 类型定义符 typedef	143	7.1.1 需求分析	175
<b>5.7 位操作</b>	144	7.1.2 软件设计的任务	176
5.7.1 位运算	145	7.1.3 编码	177
5.7.2 位段	147	7.1.4 软件测试	179
<b>5.8 本章小结</b>	149	<b>7.2 大型程序的设计原理</b>	179
<b>习题 5</b>	150	7.2.1 使用抽象管理复杂性	180
<b>第 6 章 文件</b>	153	7.2.2 自顶而下、逐步求精的设计策略	181
<b>6.1 文件概述</b>	153	7.2.3 模块化程序设计	181
6.1.1 与文件有关的概念	153	7.2.4 重用代码	184
6.1.2 文件指针	154	<b>7.3 C 语言大型程序的几种开发机制</b>	184
<b>6.2 文件的打开与关闭</b>	155	7.3.1 模块的实现	184
6.2.1 文件的打开(fopen 函数)	155	7.3.2 多模块程序的编译、连接	187
6.2.2 文件关闭	157	7.3.3 标识符的连接属性	188
<b>6.3 文件的输入输出</b>	158	7.3.4 条件编译	191
6.3.1 读写文件中字符的函数	158		



7.4 大型程序的测试	191	9.2.1 线性表的定义	224
7.4.1 测试技术	192	9.2.2 线性表的抽象数据类型定义	224
7.4.2 测试步骤	194	9.3 线性表的顺序表示和实现	225
7.4.3 调试	195	9.4 线性表的链式表示和实现	230
7.5 程序设计实例	196	9.5 线性表的其他链式表示	235
7.5.1 链表简介	196	9.5.1 静态链表	235
7.5.2 需求分析	197	9.5.2 循环链表	238
7.5.3 总体设计	197	9.5.3 双向链表	238
7.5.4 函数设计	198	9.6 线性表的应用举例	239
7.5.5 头文件设计	199	9.7 栈	243
7.5.6 算法分析	200	9.7.1 栈的定义	243
7.5.7 实现文件设计	201	9.7.2 栈的抽象数据类型定义	243
7.5.8 测试	205	9.7.3 栈的顺序表示和实现	244
7.6 本章小结	209	9.7.4 栈的链式表示和实现	247
习题 7	209	9.8 栈的应用举例	249
第 8 章 数据结构基本概念	211	9.8.1 数制转换	249
8.1 数据结构实例	211	9.8.2 迷宫问题	250
8.2 基本概念和术语	213	9.8.3 栈与递归——汉诺塔问题	253
8.2.1 数据	213	9.9 队列	257
8.2.2 数据结构	214	9.9.1 队列的定义	257
8.2.3 存储结构	215	9.9.2 队列的抽象数据类型定义	258
8.2.4 抽象数据类型	216	9.9.3 队列的顺序表示和实现	259
8.2.5 抽象数据类型的表示与实现	217	9.9.4 队列的链式表示和实现	261
8.3 算法的评价	220	9.10 队列的应用举例	264
8.4 本章小结	221	9.11 本章小结	266
习题 8	221	习题 9	266
第 9 章 线性结构	223	第 10 章 树和二叉树	269
9.1 线性结构概述	223	10.1 树	269
9.2 线性表的类型定义	224	10.1.1 树的定义	269
		10.1.2 树的基本术语	271



10.1.3 树的性质	272	11.6 各种内部排序方法的比较讨论	315
10.1.4 树的表示	273	11.7 外部排序	316
<b>10.2 二叉树</b>	<b>273</b>	11.8 本章小结	317
10.2.1 二叉树的定义	273	习题 11	318
10.2.2 二叉树的性质	274		
10.2.3 二叉树的表示和实现	276		
<b>10.3 线索二叉树</b>	<b>287</b>	<b>第 12 章 查找/检索</b>	<b>320</b>
<b>10.4 树和森林</b>	<b>289</b>	12.1 基本概念	320
10.4.1 树的存储结构	289	12.2 静态查找表	322
10.4.2 森林和二叉树的转化	291	12.2.1 顺序查找	323
10.4.3 树和森林的遍历	293	12.2.2 折半查找	324
<b>10.5 霍夫曼树及其应用</b>	<b>293</b>	12.2.3 分块查找	328
10.5.1 基本术语	293	<b>12.3 二叉查找树</b>	<b>329</b>
10.5.2 霍夫曼树的构造	294	12.3.1 二叉查找树的查找	329
10.5.3 霍夫曼树的应用	296	12.3.2 二叉查找树的插入	331
<b>10.6 本章小结</b>	<b>297</b>	12.3.3 二叉查找树的删除	333
习题 10	298	12.3.4 平衡二叉树	336
		<b>12.4 B-树</b>	<b>342</b>
<b>第 11 章 排序</b>	<b>301</b>	12.4.1 B-树的定义与示例	342
<b>11.1 概述</b>	<b>301</b>	12.4.2 B-树查找	343
<b>11.2 插入排序</b>	<b>302</b>	12.4.3 B-树的插入	344
11.2.1 直接插入排序	302	12.4.4 B-树中的删除	346
11.2.2 改进的插入排序	303	12.4.5 B+树	348
11.2.3 希尔排序	304	<b>12.5 散列表查找</b>	<b>349</b>
<b>11.3 交换排序</b>	<b>306</b>	12.5.1 基本概念	349
11.3.1 起泡排序	306	12.5.2 散列函数	350
11.3.2 快速排序	307	12.5.3 冲突的处理	352
<b>11.4 选择排序</b>	<b>309</b>	12.5.4 散列表查找及性能分析	354
11.4.1 简单选择排序	309	<b>12.6 本章小结</b>	<b>355</b>
11.4.2 堆排序	310	习题 12	357
<b>11.5 归并排序</b>	<b>313</b>		



<b>第 13 章 文件组织</b>	<b>359</b>	<b>13.6 倒排文件</b>	<b>373</b>
<b>13.1 文件概述</b>	<b>359</b>	<b>13.7 本章小结</b>	<b>374</b>
13.1.1 文件的概念	359	<b>习题 13</b>	<b>374</b>
13.1.2 文件的逻辑结构	360	<b>附录 A Turbo C 3.0 集成开发环境的使用</b>	<b>376</b>
13.1.3 文件的物理结构	361	<b>A.1 Turbo C 3.0 简介和启动</b>	<b>376</b>
13.1.4 文件的操作	362	<b>A.2 Turbo C 3.0 集成开发环境</b>	<b>376</b>
<b>13.2 顺序文件</b>	<b>363</b>	<b>A.3 C 程序上机执行过程</b>	<b>382</b>
13.2.1 顺序文件概述	363	<b>A.4 C 程序的调试</b>	<b>383</b>
13.2.2 顺序文件的操作	363	A.4.1 编译、连接时常见错误	383
<b>13.3 直接文件(散列文件)</b>	<b>364</b>	A.4.2 程序的动态调试	384
13.3.1 桶散列	365	<b>附录 B 参考术语表</b>	<b>385</b>
13.3.2 可扩展散列	366	<b>参考文献</b>	<b>392</b>
<b>13.4 索引文件</b>	<b>368</b>		
<b>13.5 索引顺序文件</b>	<b>369</b>		
13.5.1 ISAM	369		
13.5.2 VSAM 文件	371		



## 第 1 章 程序设计的基本概念

随着计算机技术的发展,程序设计语言经历了机器语言、汇编语言、高级语言等阶段。早期的高级语言如 ALGOL60、FORTRAN、COBOL 等开创了最初的软件业,但这些语言的数据类型单调,程序设计主要依赖于程序设计人员的个人技巧,缺乏规范化的设计方法,因此程序规模较大时,其复杂性和可靠性就变得难以控制。到了 20 世纪 70 年代,结构化程序设计兴起,强调程序的模块性,C 语言就是这种结构化程序设计语言的代表。在深入学习程序设计之前,先介绍一些程序设计的基本概念。

### 1.1 程序设计和程序设计语言

程序设计语言是人与计算机交互的工具,人们把需要计算机完成的工作告诉计算机,就需要使用程序设计语言编写程序,让计算机执行以完成相应的工作。

#### 1.1.1 程序与软件

程序是对所要解决的问题的各个对象和处理规则的描述,或者说是为了解决某一问题而设计的一系列计算机能识别的指令(由操作码和操作数组成)。

程序设计是一个使用程序语言产生一系列的指令告诉计算机该做什么的过程。程序设计人员(简称程序员)的工作就是用程序设计语言来生成一系列可以被计算机接收和执行的指令完成各种指定的任务。

软件是计算机程序、方法、规则以及所要求的文档资料和在计算机上运行时所必需的数据的总和。软件发展经历了三个时期:程序设计时期(20 世纪 40 年代末至 20 世纪 50 年代末)、软件=程序+说明时期(20 世纪 50 年代末至 20 世纪 70 年代初)及软件=程序+文档时期(20 世纪 70 年代初至今,即软件工程时期)。

最初的程序设计全凭设计者个人经验和技艺独立进行,是一种典型的手工艺智力劳动。为解决软件研究和开发中面临的困难和混乱,软件的开发必须以工程化的思想为指导,运用标准和规范的方法来进行。

程序编写,即通常所说的“编程序”,是软件开发过程中的一个关键阶段。软件的质量主要是通过程序的质量来体现的,因此程序设计在软件开发中占有十分重要的地位。程序设计人员既需要有严密的思维能力,又需要有创意甚至是一定的艺术修养。成为一个优秀的程序设计者需要经过艰苦的努力。



### 1.1.2 程序设计语言的发展

程序设计语言经历了机器语言、汇编语言和高级程序设计语言三个阶段。

#### 1. 机器语言

机器语言由计算机的指令系统组成,机器语言中的每一条语句(机器指令)实际就是由 0 和 1 组成的二进制代码,它由操作码的二进制编码和操作数的二进制编码组成。机器指令通常随计算机类型不同而不同,编写程序效率低,难理解且不能通用。

#### 2. 汇编语言

汇编语言是面向机器的程序设计语言。在汇编语言中,使用助记符表示机器指令的操作码和操作数。这种代替使得机器语言“符号化”,所以又称汇编语言是符号语言。与机器语言一样,不同类型计算机具有不同的汇编语言。

由于计算机硬件只能识别机器指令,因此汇编语言书写的程序必须先翻译成机器语言程序才能在计算机上执行,用于翻译的程序称为汇编程序。

汇编语言能直接对计算机硬件操作,相对机器语言较易理解和记忆,汇编语言在程序的复杂性、可读性和查错等方面都有很大进步。但是仍然要求使用者了解计算机的指令系统,也就是要对计算机硬件很熟悉。另外,在程序的通用性方面,汇编语言并没有多大改进,在一台机器上编写的汇编语言程序在不同型号的另一台机器上还是不能运行,这就使人们不得不针对每种型号的机器开发不同的程序,程序的开发效率很低。为了解决这些问题,人们设计出了高级语言。

#### 3. 高级语言

机器语言和汇编语言都是面向机器的语言,因此人们称之为“低级语言”。从 20 世纪 50 年代中期开始,出现了许多高级程序设计语言。高级语言的语句用较为接近自然语言的英文字句表示,数据用十进制来表示。它独立于具体的计算机系统,容易被人掌握和理解,是面向人的程序设计语言。高级语言发展依据程序设计方法也经历了三个时期,即早期的线性程序设计、结构化程序设计(又称面向过程程序设计)和面向对象程序设计。

面向过程的程序设计不仅要告诉计算机“做什么”,而且要告诉计算机“怎么做”。面向对象的程序设计只需告诉计算机“做什么”。面向过程的高级语言包括 BASIC、PASCAL、FORTRAN、C 语言等,面向对象的高级语言包括 C++、Java 等。

任何一种高级程序设计语言都有严格的词法规则、语法规则和语义规则。词法规则规定了从语言的基本符号集如何构成单词;语法规则规定了从单词如何构成各类语法单位(如表达式、语句、程序等);语义规则规定了各个语法单位的含义。高级语言的词法、语法和语义可能各不相同,但一般说来,其成分有四种:数据成分,用以描述程序中所描述的数据;运算成分,用以描述程序中所包含的运算;控制成分,用以表达程序中的控制结构;传输成分,用以表达程序中的数据传输。

### 1.1.3 计算机如何运行程序

计算机只能识别和处理由 0 和 1 组成的二进制代码和数据,而由高级语言书写的程序代码,计算机是无法识别的。为此在计算机的执行机构与高级语言编写的代码之间必须架



设一座桥梁,来沟通两者之间的语言。

用高级语言编写的程序代码称为“源程序”。把源程序变为计算机可执行的二进制代码,这项工作就相当于上述所说的桥梁作用,具体实现方式有两种:一种是解释方式,另一种是编译方式。

### 1. 解释方式

BASIC 语言也属于高级语言,它采用的就是解释方式,计算机在处理 BASIC 语言编写的源程序时,整个处理过程如图 1.1 所示。

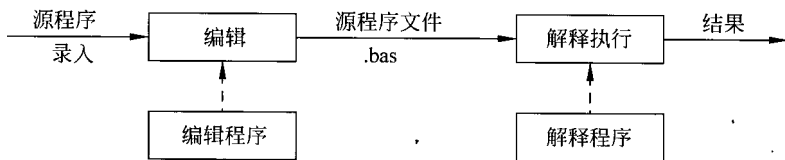


图 1.1 解释执行方式

首先通过“编辑程序”,将源程序录入到计算机中,按 BASIC 语言格式要求最终建立一个源程序文件,然后再通过“解释程序”对源程序进行翻译,每翻译一条源程序代码语句,就转换成对应二进制代码来执行一句,如果发现错误则立即终止解释过程,需要修改源程序后重新运行。如果程序没有错误,则一直解释执行完全部源程序,这中间用到的编辑过程和解释过程是通过软件来完成的。这种方式,非常类似于外语翻译中的“口译”方式,说一句,翻译一句,并没有形成译文。

### 2. 编译方式

C 语言采用的是编译方式。计算机在处理由 C 语言编写的源程序时,整个处理过程如图 1.2 所示。

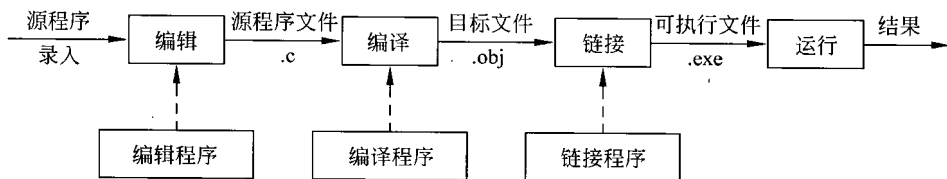


图 1.2 编译执行方式

在编译方式下,源程序经过编辑生成源文件(.c),然后由“编译程序”软件,对源程序的语法进行检查,如出现错误,给予提示,操作者对源程序进行修改,在源程序没有错误的情况下,生成目标文件(.obj),再经过“链接程序”软件,将目标文件与系统提供的函数库及其他目标程序装配链接成一个完整的可执行程序文件(.exe),此时,该可执行程序文件才能运行,并得到程序执行后的结果。这种方式类似于外语翻译中的“笔译”,只有反复修改定稿后才形成最终的译文。编译方式中所用到的编辑程序、编译程序和链接程序都属于系统软件。目前,很多软件都将这些程序集成在一起,以方便使用。Turbo C 就是这样一种具有集成开发环境的 C 语言系统软件,参见附录 A。





## 1.2 C 语言概述

本书以 C 语言为主线介绍程序设计与数据结构。为了更直观地了解程序设计语言、程序的结构与语法规则以及更容易理解后面所涉及的程序设计的相关概念,下面对 C 语言进行简单介绍。

### 1.2.1 C 语言的发展

C 语言是目前国际上广泛流行的一种结构化的程序设计语言。它是一种通用结构程序设计语言,集汇编语言和高级语言的优点于一身,不仅能够开发系统软件,而且也可用于编写应用软件。因此,它深受广大程序设计者的欢迎。

早期的操作系统等系统软件主要是用汇编语言编写的(包括 UNIX 操作系统在内)。由于汇编语言依赖于计算机硬件,程序的可读性和可移植性都比较差。为了提高可读性和可移植性,最好改用高级语言,但一般的高级语言难以实现汇编语言的某些功能(汇编语言可以直接对硬件进行操作,例如:对内存地址的操作、位操作等)。人们设想能否找到一种既具有一般高级语言特性、又具有低级语言特性的语言,集它们的优点于一身。于是,C 语言就在这种情况下应运而生了。

C 语言是在 20 世纪 70 年代初由美国贝尔实验室的 Dennis M. Ritchie 设计并首先在安装 UNIX 操作系统的 DEC PDP-11 计算机上实现。到了 1973 年,K. Thompson 和 Dennis M. Ritchie 两个人合作把 UNIX 的 90% 以上内容用 C 语言进行了改写,即大家熟知的 UNIX 第五版。多年来,UNIX V 系统配备的 C 语言一直是 C 的公认标准,因此,当初的 C 语言是为开发 UNIX 操作系统而研制的,它随着 UNIX 的出名而闻名。1983 年 ANSI 成立了专门定义 C 语言标准的委员会,将 C 语言标准化,随后 C 语言的广泛应用又不断推出新的 C 语言版本,其功能也越来越强。

到了 1975 年 UNIX 第六版的推出和随着面向对象程序设计技术的出现,C 语言的突出优点已引起了人们的普遍关注,ANSI C 标准于 1989 年被采用,该标准一般称为 ANSI/ISO Standard C。于是,1989 年定义的 C 标准定义为 C89。到了 1995 年,出现了 C 语言的修订版,其中增加了一些库函数,出现了初步的 C++,在此基础上,C89 成为 C++ 的子集。此后,C 语言不断发展,在 1999 年又推出了 C99,C99 在基本保留了 C 语言的特性的基础上增加了一系列新的特性,随后又几经修改和完善,它也从面向过程的编程语言发展到面向对象的程序设计语言。目前可在计算机上运行的 C 语言版本主要有 Microsoft C/C++、Turbo C、Quick C、Visual C/C++ 等版本。

### 1.2.2 C 程序的结构

#### 1. C 程序的结构特点

用 C 语言编写的程序称为 C 语言源程序,简称 C 程序。为了说明 C 语言源程序结构的特点,先看以下一个程序示例。这个程序虽然简单,但也表现了 C 语言源程序在组成结构上的特点。

**【例 1.1】** 从键盘输入一个数  $x$ ,求  $x$  的正弦值,然后输出结果。