工业和信息化普通高等教育“十二五”规划教材立项项目
21世纪高等学校计算机规划教材
21st Century University Planned Textbooks of Computer Science

C语言程序设计（第2版）

The C Programming Language (2nd Edition)

姚琳 主编
屈微 副主编

- 精华再版，推陈出新，贯彻以学生为主的教学思想
- 通俗易懂，由浅入深，透彻解析C语言的来龙去脉
- 案例导入，环环相扣，将获得的知识灵活应用到实践中

高校系列

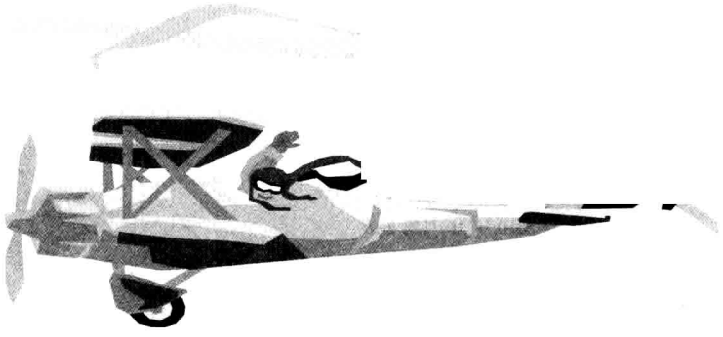
人民邮电出版社
POSTS & TELECOM PRESS

工业和信息化普通高等教育“十二五”规划教材立项项目
21世纪高等学校计算机规划教材
21st Century University Planned Textbooks of Computer Science

C语言程序设计（第2版）

The C Programming Language (2nd Edition)

姚琳 主编
屈微 副主编
黄晓璐 刘莲英 齐悦



高校系列

人民邮电出版社
北京

图书在版编目（CIP）数据

C语言程序设计 / 姚琳主编. -- 2版. -- 北京 : 人民邮电出版社, 2010.10
21世纪高等学校计算机规划教材
ISBN 978-7-115-23790-3

Ⅰ. ①C… Ⅱ. ①姚… 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(2010)第172008号

内 容 提 要

本书根据教育部非计算机专业计算机基础课程教学指导分委员会提出的《高等学校非计算机专业计算机基础课程教学基本要求》中的关于“程序设计”课程教学要求，根据当前学生的实际情况，结合一线教师的教学实际经验编写而成。

本书主线清晰、重点明确、内容恰当、概念通俗、表述简洁、举例实用，既注重基础理论，又突出实践性。全书共分9章，内容包括计算机的组成与程序设计基础、C语言基础、C语言控制语句、函数与预处理、数组、指针、其他自定义数据类型、文件和一个完整案例的设计和实现。

本书适合各类大专院校作为程序设计教材使用，也可作为学习计算机知识的自学参考书或培训教材。

工业和信息化普通高等教育“十二五”规划教材立项项目

21世纪高等学校计算机规划教材

C语言程序设计（第2版）

◆ 主　　编　姚　琳
副 主 编　屈　微
责任编辑　武恩玉

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街14号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司印刷

◆ 开本：787×1092　1/16
印张：18.25　　2010年10月第2版
字数：482千字　　2010年10月河北第1次印刷

ISBN 978-7-115-23790-3

定价：32.00元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

自 20 世纪 80 年代开始，随着我国教育事业的不断发展，在非计算机专业的大学生中普及计算机知识与应用技能的计算机基础教育也在不断发展和完善。在我国大学计算机基础教育中，目前普遍采用 1+X 课程体系结构。在这个课程体系结构中，程序设计是一门重要的基础性课程。而在众多的程序设计语言中，C 语言因其具有完备的高级语言特性，并具有丰富、灵活的控制和数据结构，简洁而高效的语句表达，清晰的程序结构和良好的可移植性等特点，被许多高校列为程序设计课程的首选计算机语言，不仅在计算机专业开设了 C 语言课程，而且在非计算机专业也开设了 C 语言课程。

本书根据教育部非计算机专业计算机基础课程教学指导分委员会提出的《高等学校非计算机专业计算机基础课程教学基本要求》中的关于“程序设计”课程教学要求，根据当前学生的实际情况，结合一线教师的教学实际经验编写而成。

1. 读者对象明确，内容紧贴教育部提出的大学计算机教育基本要求

本书按照教育部高等学校非计算机专业计算机基础课程教学指导委员会提出的最新教学要求和教学大纲的精神，在充分总结大学计算机基础教育事业 20 多年发展经验的基础上，制定了带有指导意义和规范作用的一本课程结构和教学大纲。

2. 在教学方法上，贯彻以学生为主的教学思想

在教学方法上，遵循初学者学习程序设计语言的规律和特点，采用理论和案例相结合的教学方式，运用通俗易懂的文字，每一章都通过一些能吸引学生的案例和问题引入教学内容，由浅入深、由易到难、循序渐进，力求做到符合学习规律。既注重基础理论，又突出实用性。

3. 提供立体化教材，方便教与学

本套教材包括主讲教材和辅助教材，另外可从人民邮电出版社教学服务与资源网（http://www.ptpedu.com.cn）免费下载该教材的电子课件及习题答案。本书的例题均在 Visual C++ 6.0 中调试通过。

本书的第 1 章和第 9 章由姚琳编写；第 2 章和第 3 章由齐悦编写；第 4 章由屈微编写；第 5 章由刘莲英编写；第 6 章由黄晓璐编写；第 7 章和第 8 章由姚亦飞编写。全书由姚琳最后审阅统稿。

由于编者水平有限，加上时间仓促，书中难免存在疏漏与不当之处，恳请广大读者批评指正。

编　者

2010 年 7 月

目 录

第 1 章 计算机的组成与程序设计基础

【本章内容提要】

本章在介绍具体的 C 语言之前，简单介绍一下计算机的组成、计算机的基本工作原理、计算机语言的发展、程序设计的基础知识、C 语言程序的基本结构、一个程序的开发过程和具体的操作步骤。

【本章学习重点】

- 重点掌握有关计算机的基本知识、基本概念、基本原理。
- 掌握一个程序的开发过程和具体的操作步骤。

1.1 计算机的组成及基本工作原理

一个完整的计算机系统应包括两个部分，即硬件系统和软件系统，如表 1-1 所示。硬件系统是计算机实现自动控制与运算的物质基础，软件系统加载在硬件系统之上控制硬件完成各种功能。无论是系统软件还是应用软件，都是用程序设计语言编写的。程序设计语言的发展经历了从机器语言到计算机各种高级程序设计语言的过程。机器语言是计算机语言发展过程的原点，而高级程序设计语言是计算机语言发展的重要阶段。C 语言是高级程序设计语言中的经典之作，也是深入掌握其他程序设计语言的基础。

硬件系统一般指计算机的装置，软件系统一般指管理和指挥硬件运行的程序。计算机硬件系统和计算机软件系统是相辅相成的，人们都形象的把没有软件的计算机（裸机）比喻为“没有灵魂的躯体”，同样软件离开了硬件的支持，也就失去了它的作用。

表 1-1　　计算机系统构成表

<table>
<tr><td colspan="9">计算机系统</td></tr>
<tr><td colspan="5">硬件系统</td><td colspan="3">软件系统</td><td></td></tr>
<tr><td colspan="3">主机</td><td colspan="2">外设</td><td colspan="3">系统软件</td><td rowspan="2">应用软件</td></tr>
<tr><td colspan="2">中央处理器</td><td rowspan="2">内存储器</td><td rowspan="2">输入设备</td><td rowspan="2">输出设备</td><td rowspan="2">外存储器</td><td rowspan="2">操作系统</td><td rowspan="2">语言处理程序</td><td rowspan="2">服务程序</td></tr>
<tr><td>运算器</td><td>控制器</td></tr>
</table>

1.1.1 计算机的硬件系统

计算机的硬件（Hard ware）系统是组成计算机的各种电子的、磁的、机械的部件和设备的总称。

计算机硬件的基本结构

当今计算机已发展成由巨型机、小巨型机、大型机、小型机、微型机组成的一个庞大“家族”。这个家族中的成员尽管在规模、结构、性能、应用等方面存在着一定差异，但它们的基本硬件结构仍沿用着冯·诺依曼设计的传统结构，即由运算器、控制器、存储器、输入设备和输出设备 5 部分组成。

计算机的基本硬件结构如图 1-1 所示。

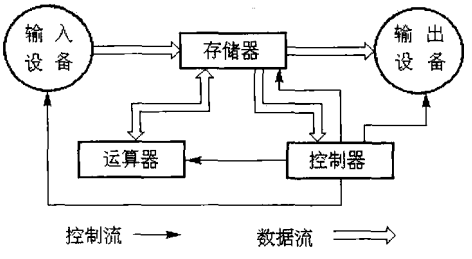


图 1-1 计算机的基本硬件结构

图 1-1 所示为这 5 部分之间的连接关系，也显示了计算机中数据和控制信息的流动方向，反映了计算机的基本工作原理。这种结构就是依据著名科学家冯·诺依曼提出的存储程序计算机的基本结构的设计思想，其基本特点是将程序和数据都以二进制的形式存储在存储器中，在控制器的指挥下，自动地从存储器中取出指令并执行，以完成计算机的各种工作。

（1）运算器

运算器是对数据进行处理和运算的部件。运算器的主要部件是算术逻辑单元（Arithmetic Logic Unit，ALU），另外还包括一些寄存器。它的基本操作是进行算术运算和逻辑运算。算术运算是按算术规则进行的运算，如加、减、乘、除等。逻辑运算一般指非算术性质的运算，如比较大小、移位、逻辑“与”、逻辑“或”、逻辑“非”等。在计算机中，一些复杂的运算往往是通过大量简单的算术运算和逻辑运算来完成的。

（2）存储器

存储器是用来存储程序和数据的部件，可以分为内存储器（主存储器）和外存储器（辅助存储器）两类。内存储器简称内存，用来存储当前要执行的程序和数据以及中间结果和最终结果。存储器由许多存储单元组成，每个存储单元都有自己的地址。根据地址就可找到所需的数据和程序。

外存储器简称外存，用来长期存储大量暂时不参与运算的数据和程序以及运算结果。

（3）控制器

控制器的主要作用是指挥计算机各部件协调的工作。它是计算机的指挥中心，在控制器的控制下，将输入设备输入的程序和数据，存入存储器中，并按照程序的要求指挥运算器进行运算和处理，然后把运算和处理的结果再存入存储器中，最后将处理结果传送到输出设备上。

控制器一般由程序计数器（Program Counter，PC）、指令寄存器（Instruction Regisiter，IR）、指令译码器（Instruction Decoder，ID）和操作控制器（Operation Controller）等组成。程序计数器用来存放当前要执行的指令地址，它有自动加 1 的功能。指令寄存器用来存放当前要执行的指令代码。指令译码器用来识别 IR 中所存放要执行指令的性质、操作。控制器是根据指令译码器对要执行指令的译码，产生出实现该指令的全部动作的控制信号。

（4）输入设备

输入设备是将用户的程序、数据和命令输入到计算机的内存的设备，标准的输入设备是键盘。

常用的输入设备还有鼠标、扫描仪等。目前，市场上还出现了汉字语音输入设备和手写识别输入设备，使得汉字输入变得更为方便。

（5）输出设备

输出设备是显示、打印或保存计算机运算和处理结果的设备。标准的输出设备是显示器。常用的输出设备还有打印机、绘图仪等。目前，其他类型的输出设备也有不同程度的发展，最为常见的是数据投影设备，与计算机直接相连，计算机在屏幕上的输出结果就可直接传到投影仪上输出。投影仪可用于多媒体教育、大型场合的计算机演示等。

通常把运算器和控制器合称为中央处理单元（Central Processing Unit，CPU），它是计算机的核心部件。将 CPU 和内存合称为“主机”，把输入设备和输出设备及外存合称为外部设备，简称外设。

1.1.2　计算机的软件系统

没有装入任何软件的计算机称作“裸机”。裸机是无法工作的，需要计算机软件系统来支撑，所以计算机的软件系统是计算机系统中必不可少的组成部分。

所谓软件（Software）是计算机系统中各类程序、有关文档以及所需要的数据的总称。其中，程序只是软件的一部分。

1. 程序的基本概念

程序简单地说，就是为了解决某一问题而设计的一系列指令或语句。

要想使计算机按人们的意愿去工作，目前还要进行程序设计。也就是说必须把解决问题的方法、步骤等编写成程序，输入到计算机，然后再由计算机执行这个程序，完成程序中所指定的工作。

2. 软件的分类

计算机软件可分为系统软件和应用软件两大类。

（1）系统软件

系统软件一般是用来管理、维护计算机及协调计算机内部更有效工作的软件。主要包括操作系统、语言处理程序和一些服务性程序。系统软件中的核心软件就是操作系统。

操作系统是对计算机系统进行控制及管理的大型程序。它有效地统管计算机的所有资源（包括硬件和软件资源）。合理地组织计算机的整个工作流程，从而提高资源的利用率，并为用户提供强有力的使用功能和灵活方便的使用环境。

操作系统是计算机系统的重要组成部分，是计算机系统所有软件、硬件资源的组织者和管理者。任何一个用户都是通过操作系统使用计算机的。

操作系统的基本任务主要有两点：其一是管理好计算机的全部资源（包括 CPU、存储器、各种外设、程序和数据）；其二是担任用户与计算机之间的接口，让用户使用方便，操作简单，而且不必过问计算机硬件的具体细节。

操作系统的主要功能包括 CPU 管理、存储管理、文件管理、设备管理和作业管理。目前，微机上使用的操作系统主要有：Windows XP、UNIX、Linux 等系统。

（2）应用软件

应用软件一般是为某个具体应用开发的软件，如文字处理软件、杀毒软件、财务软件、图形软件等。应用软件的种类很多，包括各种游戏程序、字处理程序（如 Word、WPS）、电子表格（如 Excel、Lotus1-2-3）及各种工具软件（如 WinRAR）等。

3. 计算机语言的发展

在日常生活中，人与人之间交流一般是通过语言，人类所使用的语言一般称为自然语言。而人与计算机之间的“沟通”，或者说人们让计算机完成某种任务，也需用一种语言，这就是计算机语言。在使用计算机时，必须把要解决的问题编成一条条语句，这些语句的集合就称为程序。用户为解决自己的问题编制的程序，称为源程序（Source Program）。随着计算机硬件的不断发展，计算机软件也飞速发展，计算机所使用的“语言”也在不断的发展，并形成了一种体系。下面就介绍它们的发展情况及其特点。

（1）机器语言

指令通常分为操作码（Operation Code）和操作数（Operand）两大部分。操作码表示计算机执行什么操作；操作数表示参加操作的数本身或操作数所在的地址。

因为计算机只能识别二进制数，所以计算机的指令系统中的所有指令，都必须以二进制编码的形式来表示，也就是一串 0 或 1 排列组合而成。例如，某种型号的微机系统中加法指令的编码为 78H（此处 H 表示十六进制），减法指令的编码为 77H 等，它们相对应的二进制编码为：

```
加法（78H）：  01111000B
减法（77H）：  01110111B
```

这就是指令的机器码（Machine Code）。这种指令功能与二进制编码的关系是人为规定的，计算机按照规定进行识别。

计算机发展的初期，就是用指令的机器码来编制用户的源程序，这就是机器语言阶段。也就是说用 0 和 1 组成的二进制的代码形式写出机器的指令，把这些代码按用户的要求顺序排列起来，这就是机器语言的程序。在机器语言中，每一条指令的地址、操作码及操作数都是用二进制数表示的，显然机器语言是计算机“一看就懂”的语言，是计算机能唯一识别和可直接执行的语言。因此，它占用内存少，执行速度快，效率高，而且无须“翻译”。机器语言对机器方便，但人们用机器语言编写程序就很麻烦。对人来讲，机器语言存在许多不足，如很不直观（难读），难懂、难记、易出错、难修改等。它的致命的弱点是无通用性。也就是说不同类型的计算机各有自己的指令系统，所以人们称机器语言是面向机器的语言。

（2）汇编语言

由于用机器语言编写程序时存在许多不足。为了克服这些缺点，人们想到是否能用一些符号（如英文字母、数字等）来代替难读、难懂、难记的机器语言。于是人们就用一些助记符（Mnemonic）来代替操作码，这些助记符通常使用指令功能的英文单词的缩写，这样更便于记忆。如某种型号的微机系统中加法指令用助记符 ADD 来表示，减法指令用助记符 SUB 来表示等。操作数用一些符号（Symbol）来表示。如 ADD AX，BX（此条指令的作用是把累加器 AX 和寄存器 BX 中的内容相加后的结果送到累加器 AX 中）。这样每条指令都有明显的特征，易于理解和记忆，这就是汇编语言阶段。

用汇编语言编写的程序称为汇编语言源程序，指令的操作数和地址不直接使用二进制代码编写的程序，而是用符号或用十六进制数表示。这样对人们来讲汇编语言比机器语言容易理解，便于记忆，使用起来方便多了。但对机器来讲，必须将汇编语言编写的程序翻译成机器语言程序，然后再执行，用汇编语言编写的源程序被翻译成机器语言程序，一般称之为目标程序。将汇编语言源程序翻译成目标程序的软件称为汇编程序，具体翻译过程如图 1-2 所示。

虽然汇编语言比机器语言前进了一步，使用起来方便了许多，但是汇编语言是一种由机器语言符号化而成的语言，因此仍没有完全解决机器语言的致命弱点，就是通用性差，也就是说汇编

语言和机器语言都是面向机器的语言。

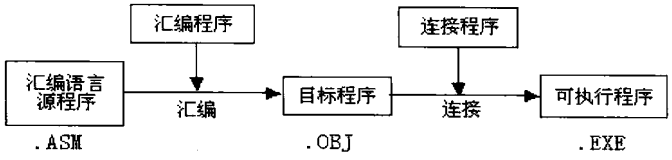


图 1-2　汇编语言的翻译过程

（3）高级语言

高级语言又称算法语言。为了克服机器语言和汇编语言依赖于机器，通用性差的弱点，人们发明创造了高级语言。高级语言有两个特点：一是和人类的自然语言（指英语）及数学语言比较接近，比如在 BASIC 语言中，“INPUT”表示输入，“PRINT”表示打印，用符号+、-、*、/代表算术运算符中的加、减、乘、除；二是与计算机的硬件无关，无须熟悉计算机的指令系统。这样用高级语言编写程序时，只需考虑解决什么问题和怎样解决，而无须考虑机器，所以称高级语言是面向过程的语言。

目前，计算机高级语言分为两大类：一类为面向过程的高级语言，如 BASIC、FORTRAN、Pascal、C 等；另一类为面向对象的高级语言，如 C++、Java 等。

用高级语言编写的源程序在计算机中也不能直接执行，必须翻译成机器语言程序才能执行，通常翻译的方式有两种，一种是编译方式，一种是解释方式。

在“编译”方式中，将高级语言源程序翻译成目标程序的软件称为编译程序，这种翻译过程称为编译。在翻译过程中，编译程序要对源程序进行语法检查，如有错将给出相关的错误信息，如无错才翻译成目标程序。还要注意一点就是目标程序虽然已是二进制文件，但还不能直接执行，还需经过连接和定位生成可执行程序文件后，才能执行。用来进行连接和定位的软件称为连接程序。具体的编译方式过程如图 1-3 所示。

在“解释”方式中，将高级语言源程序翻译和执行的软件称为解释程序。解释程序不是对整个源程序进行翻译，也不生成目标程序，而是解释一条语句执行一条语句。如果发现错误就给出错误信息，并停止解释和执行，如果没有错误就解释执行到最后的语句。具体的解释方式过程如图 1-4 所示。

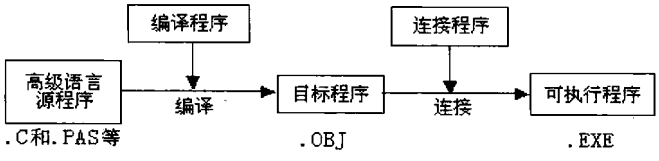


图 1-3　编译方式过程

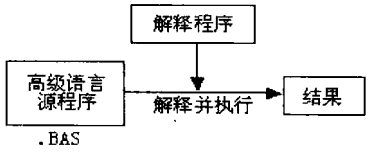


图 1-4　解释方式过程

无论是编译方式还是解释方式都起着将高级语言编写的源程序翻译成计算机可以识别与运行的二进制代码的作用。但这两种方式是有区别的，即编译方式就是把高级语言的源程序文件用此种语言的编译程序翻译成相应的机器语言的目标程序，然后再通过连接程序，将目标程序连接成可执行程序文件，再运行可执行程序文件得到结果。而解释方式在执行时，源程序和解释程序必须同时参与才能运行，而且不产生目标文件和可执行程序文件，下次运行此程序时还要重新解释执行。所以解释方式的效率低，执行速度慢，它的唯一特点是便于人机对话。编译方式和解释方

式两种翻译方式与我们日常生活中的翻译很相似，日常生活中的翻译也分两种，即笔译和口译，编译方式如同笔译，而解释方式如同口译。

1.1.3 计算机工作原理

计算机的工作过程实质上就是执行程序的过程，程序中的每一个操作步骤都是指示计算机做什么和如何做的命令，这些用以控制计算机、告诉计算机进行怎样操作的命令称为计算机指令。只要这些指令能被计算机理解，则将程序装入计算机并启动该程序后，计算机便能自动按编写的程序一步一步地取出指令，根据指令的要求控制机器各个部分运行。这就是计算机的基本工作原理，这一原理最初由美籍匈牙利科学家冯·诺依曼（Von Neumann）提出。

可以看出，冯·诺依曼结构的计算机必须具有如下部件。

- 把要执行的程序和所需要的数据送至计算机中存储起来的存储器；
- 需要具有输入程序和数据的输入设备；
- 能够完成程序中指定的各种算术、逻辑运算和数据传送等数据加工处理的运算器；
- 能够根据运算的结果和程序的需要控制程序的走向，并能根据指令的规定控制机器各部分协调操作的控制器；
- 能按人们的需求将处理的结果输出给操作人员使用的输出设备。

冯·诺依曼结构计算机的工作原理最重要之处是“存储原理”。即如果要让计算机工作就是要先把编制好的程序输入到计算机的存储器中存储起来，然后依次取出指令执行。每一条指令的执行过程又可以划分成如下 3 个基本操作。

- 取出指令：从存储器某个地址中取出要执行的指令。
- 分析指令：把取出的指令送到指令译码器中，译出指令对应的操作。
- 执行指令：向各个部件发出控制操作，完成指令要求。

1.2 程序设计基础

1.2.1 程序设计的风格

程序设计是一门技术，需要相应的理论、技术、方法和工具来支持。一般认为程序设计方法和技术经历了结构化程序设计和面向对象的程序设计阶段。

除了好的程序设计方法和技术之外，程序设计风格也是很重要的。因为程序设计风格会影响软件的质量和可维护性，良好的程序设计风格可以使程序结构清晰合理，使程序代码便于维护，因此，程序设计风格是保证程序质量的重要因素之一。

要形成良好的程序设计风格，应考虑源程序文档化、数据说明的方法、语句构造以及输入和输出几个因素。

（1）源程序文档化

源程序文档化一般要考虑：标识符的命名、程序注释信息、视觉组织等几个方面。

- 标识符的命名应遵循“见名知义”的原则。
- 程序注释信息应能帮助读者正确理解整个程序。注释信息一般包括序言性注释和功能性注释。所谓序言性注释一般位于程序的开始部分，主要包括：标题、程序的主要功能、主要算法、

程序作者等。所谓功能性注释一般位于程序的中间，主要描述变量的含义、语句的作用等。

- 视觉组织主要是书写程序时应尽量清晰，便于阅读，一般使用空格、空行、缩进等技巧。

（2）数据说明的方法

在编写程序时，为了更好的理解和维护程序，数据说明应注意次序规范化。当一个说明语句说明多个变量时，变量按照字母顺序排列。使用注释来说明复杂数据的结构等。

（3）语句构造

除非对效率有特殊要求，否则程序编写要做到清晰第一，效率第二。例如，要将两个变量 A 和 B 的值交换一下，我们给出两种方法：

方法一：	方法二：
A=A+B	T=A
B=A−B	A=B
A=A−B	B=T

从表面上看方法一只用了 A 和 B 两个变量，而方法二用了 A、B 和 T 3 个变量。但方法二显然比方法一更清晰，更容易理解。

（4）输入和输出

输入和输出是一个程序不可缺少的部分，输入/输出方式和格式应尽可能方便用户的使用。

1.2.2　结构化程序设计

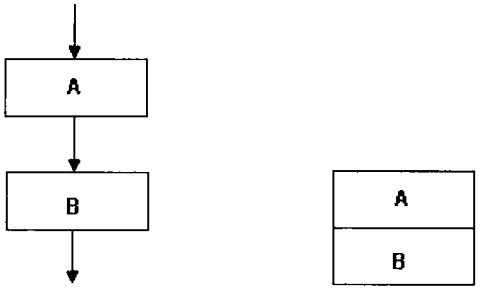
由于软件危机的出现，人们开始研究程序设计方法，其中结构化程序设计方法和面向对象程序设计方法最受关注。“结构化程序设计（structured programming）是软件发展的一个重要里程碑”，它是以模块化设计为中心，将待开发的软件系统划分为若干个相互独立的模块，这样使完成每一个模块的工作变得单纯而明确，为设计一些较大的软件打下了良好的基础。

结构化程序设计方法的基本原则是：采用自顶向下、逐步细化的方法进行设计；采用模块化原则和方法进行设计；限制使用 goto 语句。

结构化程序的基本结构包括顺序结构、选择结构和循环结构。

1. 顺序结构

顺序结构是程序的最基本、最常用的结构，也是最简单的程序结构。它是按照书写顺序依次执行语句的结构，如图 1-5 所示。



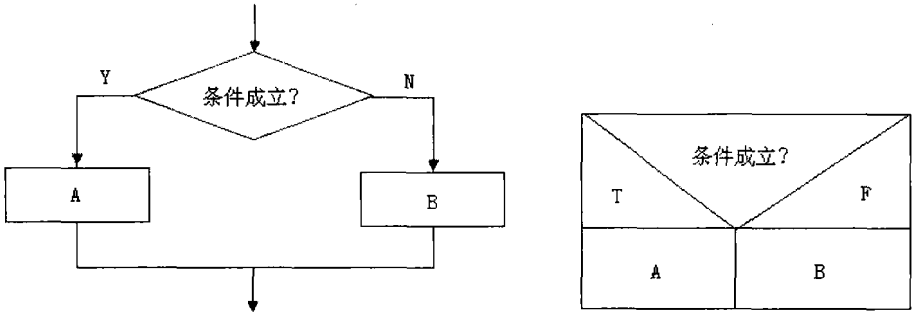
（a）流程图　　（b）N-S 结构图

图 1-5　顺序结构

2. 选择结构

选择结构又称为分支结构，这种结构是按照给定的条件判断选择执行相应的语句序列，如

图 1-6 所示。



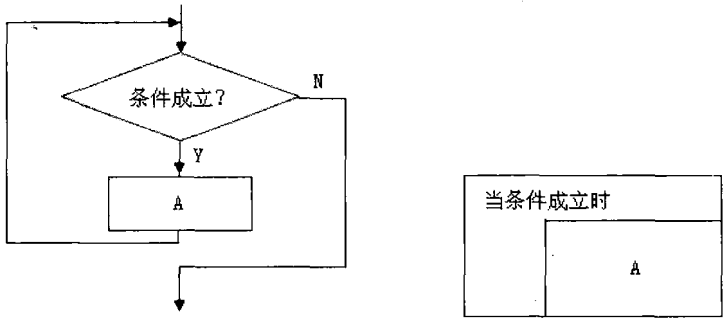
（a）流程图　　（b）N-S 结构图

图 1-6　选择结构

分支结构一般根据条件判别来决定执行哪一个程序分支，满足条件则执行语句序列 A，不满足条件，则执行语句序列 B。通常，CPU 每执行完一条指令后，便自动执行下一条指令，但分支结构的执行可以改变程序的执行流程。

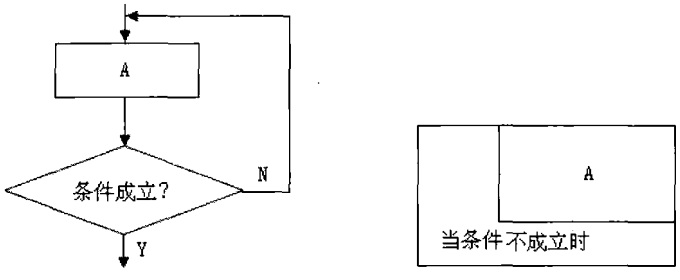
3. 循环结构

循环结构又称为重复结构，通过循环控制条件来决定是否重复执行相同的语句序列。在计算机程序设计语言中，一般包括两种类型的循环：当型循环（见图 1-7）和直到型循环（见图 1-8）。



（a）流程图　　（b）N-S 结构图

图 1-7　当型循环结构



（a）流程图　　（b）N-S 结构图

图 1-8　直到型循环结构

循环结构每次测试循环条件，当满足条件时，重复执行这一段程序，否则结束循环，顺序往下执行。循环结构一般由以下 3 部分构成。

① 初始化部分：为循环作准备，如为循环变量赋初值。这一部分往往位于循环语句的前面。

② 循环控制部分：循环控制条件，控制循环继续或结束，以保证循环按预定的次数或特定条件正常执行。

③ 循环体部分：实现循环的基本操作，是循环工作的重复部分。这一部分从初始化部分设置的初值开始，反复执行相同或相似的操作。

在循环体中一定有一个语句能够更改循环控制条件的逻辑值，使得在恰当的时候结束循环。

结构化程序的主要特点是：

- 程序易于理解、使用和维护；
- 提高了编程工作的效率，降低了软件开发成本。

1.3　C 语言程序的基本结构及开发过程

1.3.1　C 语言程序的基本结构

请看下面 2 个示例。

【例 1-1】　计算圆的面积。

```
#include "stdio.h"
void main( )                                    /* main（主）函数 */
{ int r;                                        /*定义 1 个整型变量 r*/
  float area ;                                  /*定义 1 个单精度变量 area*/
  r=8 ;                                         /*输入半径为 8*/
  area=3.14*r*r;                                /*计算圆的面积*/
  printf ("%f\n ", area) ;                      /*输出结果*/
 }
```

【例 1-2】　输入年份，判别该年是否为闰年。

```
#include "stdio.h"
int leap(int year)                              /* leap 函数 */
{ int flag;                                     /*定义 1 个整型变量 flag*/
  if (year%4==0 && year%100!=0)
        flag =1;                                /*满足闰年的条件 1，则 flag =1*/
  else  if (year%400==0)
        flag =1;                                /*满足闰年的条件 1，则 flag =1*/
  else
       flag =0;                                 /*不满足闰年的条件，则 flag =0*/
  return flag;                                  /*返回变量 flag 的值*/
}
void main( )                                    /* main（主）函数 */
{ int  year;
   scanf("%d", &year);                          /*输入 year 的值*/
   if(leap(year)== 1)                           /*调用 leap 函数，并判断函数的返回值
是否等于 1*/
       printf("%d is a leap year \n", year);    /*等于 1 的输出结果*/
   else
       printf("%d is not a leap year \n", year);/*不等于 1 的输出结果*/
}
```

从例 1-1 和例 1-2 两个简单 C 语言程序的例子中可以看出，虽然它们的功能互不相同，程序中语句的条数也不相同，但它们都反映了一般 C 语言程序的基本组成以及主要特点。下面对一般的 C 语言程序做几点说明。

① 在 C 语言中，一个 C 语言程序是以函数作为模块单位的，一个完整的 C 语言程序可以由一个或多个函数组成，但有且仅有一个 main()函数（或称主函数）。一个 C 语言程序总是从 main 函数开始执行，最终在 main 函数中结束。

② 一个 C 函数模块分两大部分，即函数的说明部分和函数体部分。

函数的说明部分又称为函数首部，它包括函数类型、函数名和函数参数。通常，在函数名后面有一对圆括号，根据需要在圆括号中可以有函数的参数，函数的参数是与主调函数交换的参数（如例 1-2 中的 year）。

函数体是用左右花括号括起来的部分，左“{”表示开始，右“}”表示结束。函数内包括若干条语句，语句序列是实现函数的预定功能。

③ C 语言程序中的每一个语句必须以“;”结束，但书写格式是自由的。在 C 语言中是区分大小写的，一般习惯使用小写字母，书写时按缩进格式。在 C 语言程序中，一行上可以写多条语句，一条语句也可以占多行，但在实际编写时应注意程序的可读性。

值得注意的是，在一个 C 语言语句中，并不是所有的部分都可以拆开写在两行上，句中的有些部分作为整体只能写在同一行上而不允许拆开，如例 1-3 所示。

【例 1-3】 输出字符串“Welcome to Beijing”。

```
#include "stdio.h"
void main()
{
   printf("Welcome to Beijing\n");
}
```

printf() 中用双引号括起来的字符串是作为格式输出函数的一个独立参数（用以指定输出数据的格式），它作为一个整体是不能被拆开的。因为 C 语言允许一条 C 语言语句可以写在多行上，所以，例 l-3 中的程序还可以书写成如下的格式：

```
#include "stdio.h"
void main()
{
 printf(
 "Welcome to Beijing\n"
 );
}
```

但例 l-3 中的程序不能写成如下的格式：

```
#include "stdio.h"
void main()
 {
   printf(
   "Welcome to
   Beijing\n"
  );
}
```

因为"Hello World!\n"是函数 printf() 的一个独立参数，它只能作为整体写在同一行上，而不能拆开写在两行上。