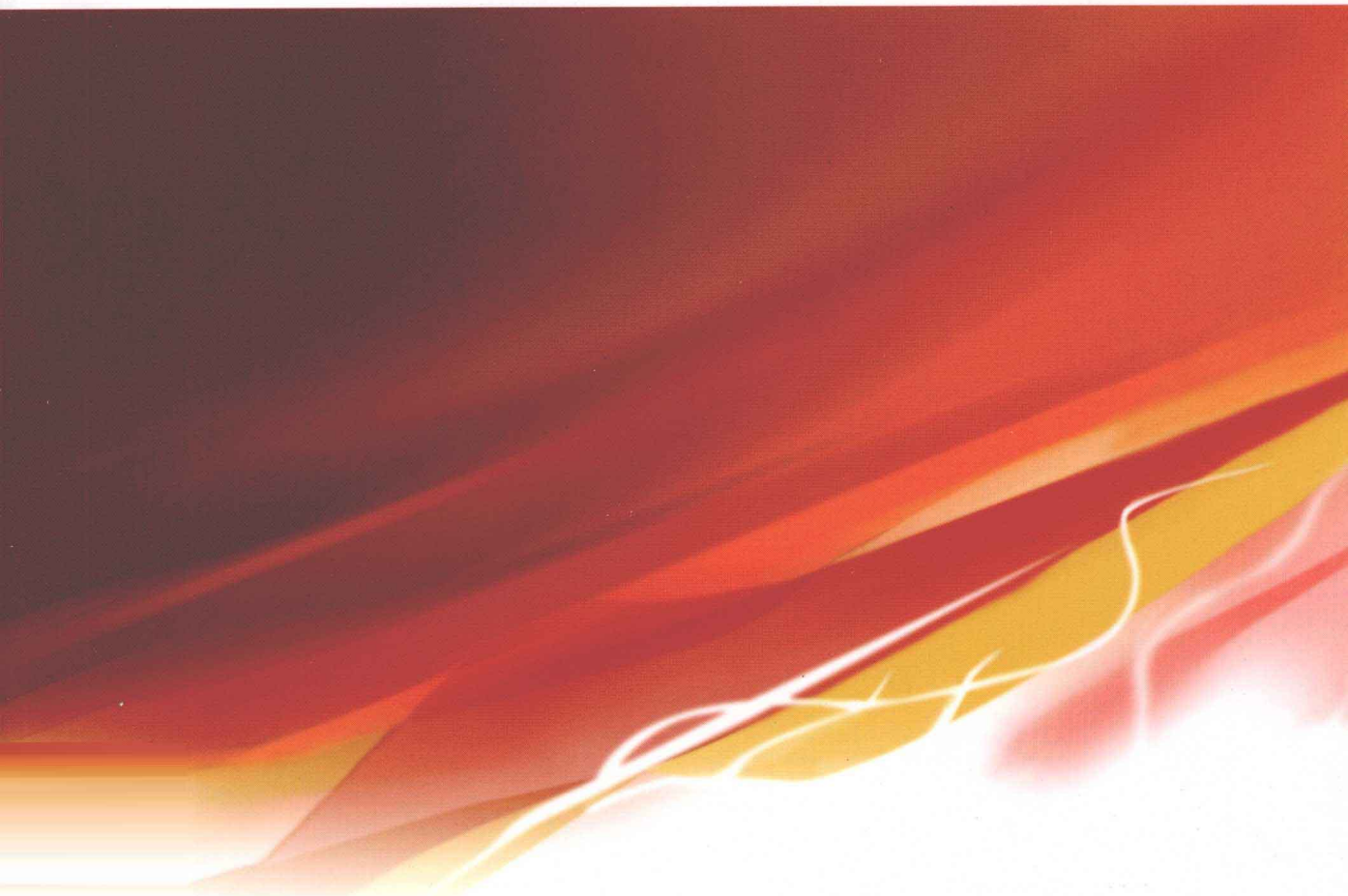
C语言程序设计

主　编　杨晓波　要路岗



国防工业出版社
National Defense Industry Press

C语言程序设计

主编　杨晓波　要路岗
副主编　陈洁　樊瑶　胡永　张环　张兆基

国防工业出版社
·北京·

前　言

C 语言是目前最流行的程序设计语言之一，具有简洁、紧凑、灵活、实用、高效、可移植性好等优点，深受广大用户欢迎。

C 语言是许多后续专业课程的基础，如数据结构、编译原理、操作系统等；同时 C 语言也是学习面向对象程序设计的基础。在计算机教育方面，C 语言是为数不多的与国外保持内容同步的课程之一，它在本科生教学中的地位之重要不言而喻。

本教材的定位是应用型本科层次，着眼于 C 基本语法和程序设计思想，重点训练和培养学生算法设计的思维、风格和能力。

教材共分为 11 章。第 1 章程序设计基础，介绍程序设计的基本概念和思想；第 2 章数据类型、运算符与表达式，介绍各种数据类型、常量和变量、运算法及表达式；第 3 章简单的 C 程序设计，介绍 C 的各种语句和输入/输出函数；第 4 章程序的控制结构，介绍 3 种基本结构顺序、选择和循环；第 5 章函数，介绍模块化程序设计思想、函数、变量的作用域和生命周期等内容；第 6 章数组，介绍一维数组、二维数组、字符串和数组作函数参数的用法；第 7 章指针，介绍各种指针、指针与数组、指针作函数参数；第 8 章结构体与共用体，介绍结构体类型、结构体数组和链表；第 9 章文件操作，介绍流的概念、文件的打开及读写等操作；第 10 章位运算，介绍计算机内数据的表示方法、各种位运算和位域的使用；第 11 章 C 语言程序设计常见错误及解决方法，介绍 C 程序设计常见错误、示例程序设计中各种常见错误及解决方案、程序的调试技巧等内容。另外书后给出了 4 个附录，附录 A 给出常用字符与 ASCII 码对照表；附录 B 列出 C 关键字；附录 C 列出 C 运算符的优先级与结合性；附录 D 列出常用的 ANSI C 标准库函数。

教材每章之后都配有一定数量的习题，习题种类丰富，既考虑了 C 语言各章节基本概念和知识点的掌握，也注重学生算法设计能力的训练和提高，与本教材配套出版的还有上机指导书，该书中提供习题参考答案、课内上机实验、课外上机实验和课程设计。教材采用 Visual C ++ 6.0 作为语言开发环境，所有程序在 VC 6.0 下调试通过，并附有程序运行结果，使程序设计过程和运行过程更加形象直观。使用本教材推荐讲授学时为 50，上机实验学时为 40，实验题目可采用配套上机指导书的内容。

本教材的作者都有着多年的 C 语言程序设计课程教学经验，并对教材的编写和算法的设计倾注了大量的精力，但课程所涉及的内容较多，算法数量大、细节多，难免存在错误和不足，敬请同行专家和读者批评指正。电子邮箱为 hzyangxb@126.com。

目　录

第1章　程序设计基础

C语言是目前世界上最为流行的计算机高级程序设计语言之一。它可读性强，便于按模块化方式组织程序，易于调试和维护，同时很多其他程序设计语言也是以此为基础的，所以掌握C语言是非常必要的。本章主要介绍C语言的功能特点及结构、算法的基本概念和描述方法以及C语言的开发环境。

1.1　程序设计语言简介

1.1.1　什么是程序设计语言

众所周知，计算机是由硬件系统和软件系统两大部分组成的，硬件是物质基础，软件则是计算机的灵魂，没有软件的计算机称为“裸机”。软件由一系列程序和相关的数据文件组成。而用来编写程序的工具就称为程序设计语言，程序设计需要在一定的语言环境中进行。人与计算机通过程序设计语言进行交流及信息交互。程序设计语言也称为计算机语言。

从1946年第一台计算机诞生起，随着计算机技术的飞速发展，程序设计语言也经历了从机器语言到高级语言的转变。

1. 机器语言

机器语言是最早的计算机语言，也称为二进制代码语言。机器语言直接用二进制代码指令表示，指令由一串0和1序列组成。下面是某CPU指令系统中的一条加法指令：

10000000

机器语言是计算机唯一可直接识别的语言，但是由于每台机器的指令格式和代码所代表的含义都是硬性规定的，程序序列长，难记、难理解且不易查错，只有少数专业人员才能掌握。而且大量细节牵制着程序员，使程序的可开发性差、效率低。

2. 汇编语言

由于机器语言编程的弊端，人们用一些容易记忆的助记符来代替机器语言的二进制码，这种程序设计语言就称为汇编语言，也称为符号语言。例如用“ADD”代表加法，“SUB”代表减法。这样一来，就容易读懂并理解程序功能，提高编程效率，纠错及维护也变得容易了。然而计算机不能直接识别这些符号，需要一种程序将汇编语言翻译成机器语言，这种起翻译作用的程序叫汇编程序，汇编程序把汇编语言翻译成机器语言的过程称为汇编。

机器语言和汇编语言都是面向机器的语言，优点是能够较好地发挥机器的特性，缺点和机器语言一样，可移植性差。一般我们把机器语言和汇编语言称为低级语言。

3. 高级语言

由于汇编语言依赖于硬件，且助记符量大难记，1954年第一个完全脱离机器硬件的高级语言——FORTRAN诞生了。高级语言的出现是计算机技术发展道路上的一个里程碑。高级语言接近于数学语言或自然语言，同时又不依赖于计算机硬件，具有良好的可移植性。程序员

从根本上摆脱了指令系统的束缚，能够把精力集中在问题的描述和求解上，大大提高了编程效率。用高级语言编写的源程序，计算机不能直接运行，必须通过编译程序将其翻译成计算机能够识别的目标程序，才能被计算机运行。所以与低级语言相比，目标程序代码长、占用内存大、执行时间长。

几十年来，共有几百种高级语言出现，高级语言程序的设计思想也经历了从早期语言到结构化程序设计语言，从面向过程到面向对象程序语言的发展过程。面向过程就是分析出解决问题所需要的步骤，然后用函数把这些步骤一步步实现，使用的时候一个一个依次调用即可，因此程序的执行顺序是由程序员决定好的。而面向对象是把现实实体抽象为对象，建立对象的目的不是为了完成一个步骤，而是为了描述某个事物在整个解决问题的步骤中的行为。高级语言的下一个发展目标是面向应用，也就是说：只需要告诉程序你要干什么，程序就能自动生成算法，自动进行处理。

比较流行的高级语言有 BASIC、PASCAL、FORTRAN、C 等。随着 Windows 操作系统的普及，又出现了面向对象的可视化编程语言，比较流行的有 Visual Basic、C ++ 、Java 等。C ++ 是支持面向对象功能的程序设计语言，而 C 语言是面向过程的程序设计语言。

1.1.2 C 语言简介

C 语言是目前世界上最为流行的计算机高级程序设计语言之一，它集高级语言和低级语言的功能于一体，既适用于系统软件的开发，也适用于应用软件的开发。同时它灵活性强、功能齐全，因此被称为当代最优秀的程序设计语言。

C 语言的产生根源可以追溯到 1960 年出现的 ALGOL 60。ALGOL 60 结构严谨，对于后来许多重要的程序设计语言产生过重要的影响。1963 年英国剑桥大学在 ALGOL 60 的基础上推出更接近硬件的 CPL 语言，但 CPL 太复杂，难于实现。1967 年，剑桥大学的 Martin Richards 在 CPL 的基础上进行简化，推出 BCPL(Basic Combined Programming Language)语言。

1970 年，贝尔实验室的 Ken Thompson 根据 BCPL 设计出较先进的 B 语言，而 C 语言就是在此基础上发展而来的。1972 年至 1973 年间，贝尔实验室的 D. M. Ritchie 设计出了 C 语言。1973 年 Ken Thompson 和 D. M. Ritchie 把 UNIX 系统用 C 语言重写了一遍，增加了多道程序设计功能，并在 PDP - 11 计算机上加以实现，即 UNIX 版本 5。随着 UNIX 的巨大成功和被广泛移植到各种机器上，C 语言也被人们所接受。

1978 年由美国电话电报公司(AT&T)的贝尔实验室正式发表了 C 语言，同时由 B. W. Kernighan 和 D. M. Ritchie 合著了影响深远的《The C Programming Language》一书，此书中介绍的 C 语言成为后来广泛使用的 C 语言基础版本，它被称为标准 C 语言。随着微型计算机的普及，许多开发机构推出了自己的 C 语言版本，这些版本之间的微小差别不时引起兼容性的问题，因此美国国家标准学会 ANSI(American National Standard Institute)在各种 C 语言版本的基础上制定了一个 C 语言标准，于 1983 年发表，称为 ANSI C。1987 年 ANSI 又公布了新标准——87 ANSI C。目前广泛流行的各种 C 编译系统都是以它为基础的。

C 语言在发展过程中，既具有高级语言的特性，能够编写可读性高和移植性强的程序，又具有某些必要的低级语言的特性，能描述对硬件的操作。因此，现在 C 语言已经成为世界上最广泛的计算机语言之一。

目前最流行的 C 语言开发环境有以下几种：

(1) Microsoft 公司：Microsoft C 和 Microsoft Visual C ++ 。

(2) Borland 公司:Turbo C、Turbo C ++ 、Borland C ++ 和 Borland C ++ Builder。

1.4 节中将对 Microsoft Visual C ++6.0 集成开发环境进行介绍。

1.1.3 C 语言的功能特点

C 语言如此成功是有其自身特点的,其主要特点有以下几个方面。

(1) C 语言是介于汇编语言与高级语言之间的一种描述程序语言,也有人称之为中级语言,这是很自然的。从它的发展历程来看,它正是由开发操作系统(UNIX)的需要而发展起来的,因此它具有许多通常只有像汇编语言才具备的功能,如对内存地址的操作、位操作、字的移位操作以及对寄存器的操作等等,这为编写系统软件提供了方便。同时,为便于加快开发速度,提高工作效率,它又具有类似于高级语言面向用户、容易记忆、便于书写和阅读的优点。使程序员得以减轻负担、提高效率,写出的程序具有更好的可移植性。

(2) C 语言是一种精练、灵巧和使用方便的程序设计语言。它只有很少的核心,一共只有 32 个标准关键字和 9 种控制语句。用 C 语言编写的程序通常比用其他高级语言编写的程序更简练,代码行更少。

(3) C 语言表达能力强,包含丰富的运算符(34 种)和数据类型。范围广泛的运算符使 C 语言的运算类型极其丰富,生成的表达式简练、灵活。C 语言能在整数、字符、浮点数等基本类型的基础上构造数组、结构体和共用体等各种结构化的数据类型;并引入指针概念,可指向各种数据,有助于构造复杂的数据结构,使程序效率更高。

(4) C 语言是结构化的语言,结构化程序设计思想是以模块化设计为中心,程序结构按功能划分为若干个基本模块,这样使每一个模块的工作变得单纯而明确,这种结构化方式可使程序层次清晰,便于使用、维护以及调试。C 语言的主要结构成分是函数。函数允许将一个程序中的各个功能分别编码,在某种程度上实现了数据的隐藏和程序的模块化。C 程序由若干程序文件组成,一个程序文件由若干函数构成。各个部分除了必要的数据交流外彼此独立。同时,C 语言还提供了多种结构化的控制语句(如 if…else 语句、while 语句、do…while 语句、switch 语句、for 语句)和专门的函数库,以满足结构化程序设计的要求。

(5) C 语言生成的目标代码质量高,程序执行效率高。许多高级语言相对汇编语言而言其代码的执行效率要低得多,但 C 语言的代码执行效率只低 15% 左右。

(6) C 语言是一种可移植性很强的语言,程序的可移植性是指在一个环境下运行的程序可以不加或稍加修改后在另一个完全不同的环境上运行。汇编语言是依赖于机器硬件的,而在 C 语言所提供的语句中,没有直接依赖于硬件的语句,与硬件有关的一些操作都是通过调用系统提供的库函数来实现的。因此 C 语言程序稍加修改就能移植到各种不同型号的计算机上。

虽然 C 语言有以上许多优点,但也有一些不足之处。C 语言在编程上提供了较大的自由度,例如,对数组下标越界不做检查,对变量的类型约束不够严格等。虽然提高了开发效率,但是限制了编译程序对语言语法检查,因此增加了开发者出错的机率。

1.2 算法及其描述方法

1.2.1 算法的概念

从小到大,每个人都面临过很多问题。在长期的学习中,我们学会了采取一些方法和步骤

来解决这些问题。例如在数学中学习的二分法、秦九韶算法、割圆术、求两个数的最大公约数等。这种解决问题的方法，在程序设计中称为算法。

算法重在用一个统一的方法有步骤地解决一类问题，但它不是唯一的。大家都听过数学家高斯小时候的一则故事：老师让大家计算 1 +2 +3 +4 +… +100 的值，大部分小朋友采用 1 +2，再加 3，一直加到 100 的方法进行运算，而高斯采用(1 +100) +(2 +99) +… +(49 +52) +(50 +51) =101 ×50 =5050 的方法计算。高斯和小朋友们采用了不同的方法解决了同一个问题，但高斯用时比别人少，所以方法有优劣之分。对于相同问题，有的方法需要很少的步骤，而有些方法则需要较多的步骤才能解决问题。虽然最终的结果都一样，但一般希望采用方法简单、运算步骤少的方法。因此，在解题过程中，不仅需要保证算法正确，还要考虑算法的质量，选择合适的算法。就如同你现在打算从西安到北京旅游，每个人可以选择不同的交通工具和线路，但是不管你怎么选择，大家最终的目的是一样的。可是在不同的过程中所花费的时间、金钱等代价是不同的。

总之，计算机语言只是一种工具，只学习语言的规则还不够，最重要的是学会针对各种类型的问题，拟定出有效的解决方法和步骤。有了正确而有效的算法，可以利用任何一种计算机高级语言编写程序。

著名计算机科学家沃思提出一个公式：

程序 = 数据结构 + 算法

因此，设计算法是程序设计的核心。

【例 1.1】将两个分别盛有糖和盐的杯子进行互换。

步骤 1：先将 A 杯中的糖倒在 C 杯中；

步骤 2：再讲 B 杯中的盐倒在 A 杯中；

步骤 3：最后将 C 杯中的糖倒在 B 杯中。

该算法在今后进行程序设计时可用来解决两个变量之间的互换问题。

【例 1.2】计算分段函数

$$y = \begin{cases} 2x + 1 & x \geqslant 0 \\ 5 - x & x < 0 \end{cases}$$

步骤 1：输入 x 的值；

步骤 2：判断 x 是否大于或等于 0，若大于或等于 0 则 y 为 2x +1，否则 y 为 5 -x，因为条件 x≥0 不成立，则 x <0 肯定成立；

步骤 3：输出 y 的值后结束。

【例 1.3】求两个正整数 m 和 n 的最大公约数。

辗转相除法：

步骤 1：将两个正整数存放到变量 m 和 n 中，使得 m > n。

步骤 2：求余数。m 除以 n，将所得余数存放到变量 r 中。

步骤 3：判断余数 r 是否为 0。若 r =0，则 n 为求得的最大公约数，算法结束。否则执行步骤 4。

步骤 4：更新被除数和除数。将 n 的值存放到 m 中，将 r 的值存放到 n 中，再重复执行步骤 2。

例如：设 m 为 20，n 为 8，余数用 r 表示。采用辗转相除法求它们的最大公约数的方法如下。

20 除以 8,商为 2,余数为 4,余数不为 0,以 n 作 m,以 r 作 n,(即 m = 8,n = 4)继续相除;8 除以 4,商为 2,余数为 0。余数为零,则算法结束,4 为 20 和 8 的最大公约数。

当然除此方法外,也可采用辗转相减法来求解:

步骤 1:计算 $r = \begin{cases} m - n & m > n \\ n - m & n > m \\ 0 & m = n \end{cases}$

步骤 2:判断 r 是否为 0。若 r = 0,则 n 或 m 为求得的最大公约数,算法结束。否则执行步骤 3;

步骤 3:更新减数和被减数。将 min(m,n)存放到 m 中,将 r 的值存放到 n 中,再重复执行步骤 1。

小贴士

min(m,n)表示取 m 和 n 当中较小的数。

读者可根据此方法计算 20 和 8 的最大公约数,写出相关步骤。

1.2.2 算法的特征

算法是为了解决某类问题的一个运算序列。对于该类问题的任何初始输入值,它都能机械地一步一步执行计算,经过有限个步骤后终止计算并产生输出结果。归纳起来,算法具有以下基本特征。

(1) 有穷性。算法中所包含的步骤必须是有限的,不能无穷无止,应该在所能接受的合理时间段内产生结果。在设计算法时,要对算法的执行效率作一定的分析。特别要注意的是循环结构中的死循环,如例 1.3 中若没有 r = 0 这个算法结束条件,那么算法将一直执行下去,无法结束。

(2) 确定性。算法中的每一步所要实现的目标必须是明确无误的,不能有二义性;如例 1.3 中的辗转相除法的步骤 2 如果写成“将 m 和 n 相除”,这是“不确定”的,它没有说明 m 和 n 谁做除数谁做被除数,因此无法执行。也就是说,算法中的每一个步骤都应是确定的,而不应是含糊、模棱两可的。

(3) 有效性。算法中的每一步如果被执行了,就必须被有效地执行,并得到确定的结果。例如,一个数被 0 除的操作就是无效的,应当避免这种操作。

(4) 有零或多个输入。根据算法的不同,有的在实现过程中需要输入一些原始数据,而有些算法可能不需要输入原始数据。如例 1.2 中需要输入 x 的值,例 1.3 中则需要输入 m 和 n 的值。而计算 $\sum_{n=1}^{100} n$ 的值则不需要输入任何信息。

(5) 有一个或多个输出。设计算法的最终目的是为了解决问题,为此,一个完整的算法至少应有一个输出结果。如例 1.2 中输出 y 的值就是输出信息。如果一个算法没有输出,那么这个算法将没有任何意义。

1.2.3 算法的描述方法

有多种方法来描述具体算法,常用的有自然语言、流程图、N - S 图、PAD 图、伪代码等。

1.2.3.1 自然语言

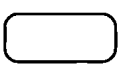
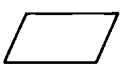
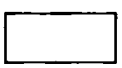
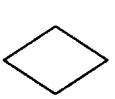
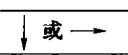
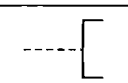
描述算法最简单的一种工具就是自然语言。自然语言指人们在日常生活中使用的语言，如汉语、英语等。在1.2.1节中的例子就是用自然语言来描述算法。虽然通俗易懂，易于掌握，但是缺乏直观性和简洁性，且易产生歧义，对比较复杂的问题，难以表述准确。因此，用自然语言描述算法一般较少采用。

1.2.3.2 流程图

1. 传统流程图

流程图又称程序框图，是最早提出的用图形表示算法的工具，所以也称为传统流程图。这种方法的特点是用一些图框表示各种类型的操作，用流程线表示这些操作的执行顺序。在程序框图中，一个或几个程序框的组合表示算法中的一个步骤；带有方向箭头的流程线将程序框连接起来，表示算法步骤的执行顺序。美国国家标准化协会ANSI规定了一些常用的流程图符号，这些符号已被世界各国的广大程序设计工作者普遍接受和采用，具体如表1-1所列。

表1-1 常用的流程图符号

程序框	名称	功　能
	起止框	起止框是任何流程图都不可缺少的，它表明程序的开始和结束，所以一个完整的流程图的首末两端必须是起止框
	输入/输出框	表示数据的输入或输出结果，用在算法中的任何需要输入/输出的位置
	处理框	矩形表示各种处理功能，算法中各种赋值、执行计算语句等操作所需要的公式或处理说明等分别写在处理框中
	判断框	菱形表示判断，判断框一般有一个入口和两个出口，有时也有多个出口，它是唯一的具有两个或两个以上出口的符号，在只有两个出口的情形中，通常都分成“是”与“否”（也可用“Y”与“N”）两个分支
 或 →	流程线	表示算法的走向，流程线箭头的方向就是算法执行的方向
	连接点	连接程序框的两部分
	注释框	注释是为了对流程图中某些程序框的操作做必要的备注，是程序的编写者向阅读者提供的说明，并不是提供给计算机的

流程图灵活方便，它将算法用形象化的图形直观地展现出来。不仅可以指导编写程序，而且可以在调试程序时用来检查程序的正确性。作为一个程序设计者，在学习具体的程序设计语言之前，必须学会针对问题进行算法设计，而流程图是目前应用较广的一种设计、阅读和分析程序的方法。

2. 算法的基本结构和改进的流程图

传统的流程图用流程线指出各框的执行顺序，对流程线的使用没有严格限制。这条简单的流程线是很灵活的，它可以到达流程的任意位置，容易使流程图毫无规律，阅读困难。

1966年，Bohra和Jacopini提出了以下三种基本结构，目的是使算法的流程更清晰简洁，增强可读性，以提高算法质量。已经证明任何简单或复杂的算法都可以由顺序结构、选择结构和

循环结构这三种基本结构组合而成。

（1）顺序结构：该结构中各个操作是按书写的顺序执行的。这是一种最简单的算法结构，也是任何一个算法必不可少的逻辑结构。如图1-1所示，框内是一个顺序结构，A和B两个框是顺序执行的。即在执行完A框所指定的操作后，接着执行B框所指定的操作。

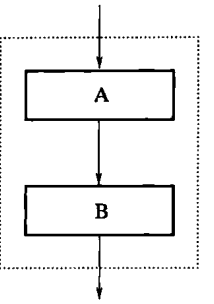


图1-1　顺序结构

（2）选择结构：又称分支结构。根据指定的条件进行判断，根据判断的结果在两条分支路径中选择其中的一条执行。如图1-2所示，框内是一个双分支的选择结构。根据给定的条件P是否成立而选择执行A或B。成立则执行A框指定操作，不成立则执行B框指定操作。在执行完A或B之后将脱离选择结构。A或B两个框中可以有一个是空的，即不执行任何操作，也称为单分支结构，如图1-3所示。

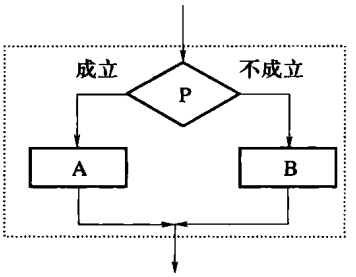


图1-2　选择结构

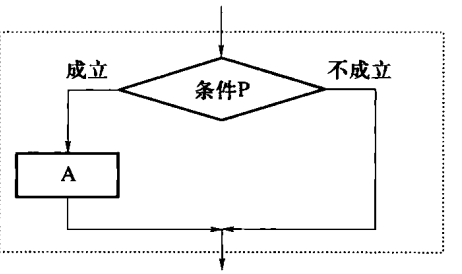


图1-3　单分支结构

需要说明的是，选择结构也可以扩展成多分支结构，但是要注意的是选择结构最多只可能执行两个或多个分支中的一个。

（3）循环结构：又称重复结构。根据是否满足给定的条件来决定是否继续执行循环体中的操作。循环结构可细分为两类：

① 当型循环结构。如图1-4所示，它的功能是先判断条件P，当给定的条件P成立时执行循环体A框，A框执行完毕后，再判断条件P是否成立，如果仍然成立，再执行A框，如此反复执行A框，直到某一次条件P不成立后为止，此时不再执行A框，结束循环。

② 直到型循环结构。如图1-5所示，它的功能是先执行循环体A框，然后判断给定的条件P是否成立，如果P仍然不成立，则继续执行A框，A框执行完毕后，再判断条件P是否成立，直到某一次给定的条件P成立为止，此时不再执行A框，结束循环。

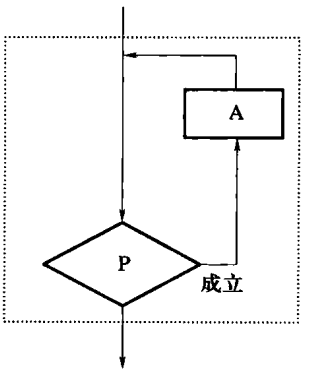


图1-4　当型循环结构

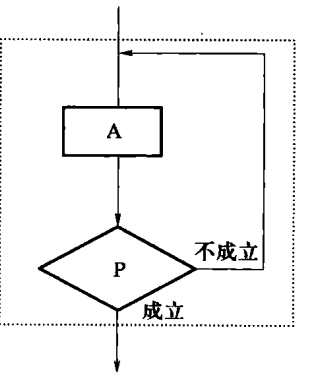


图1-5　直到型循环结构

【例 1.4】用流程图表示例 1.1 中描述的两数交换的算法，如图 1-6 所示。

【例 1.5】用流程图表示例 1.2 中求分段函数的算法，如图 1-7 所示。

【例 1.6】用流程图表示例 1.3 中求两数最大公约数的算法，如图 1-8 所示。

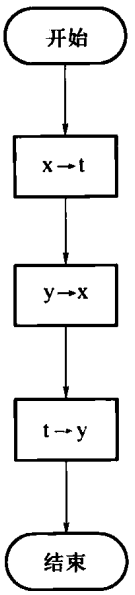


图 1-6　例 1.4 流程图

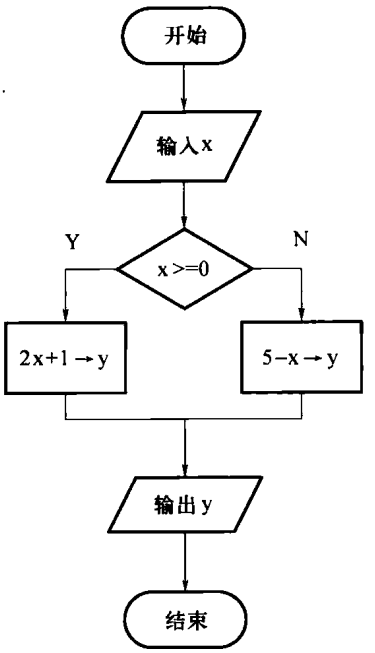


图 1-7　例 1.5 流程图

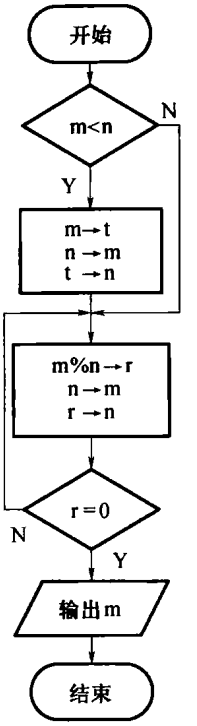


图 1-8　例 1.6 流程图

实际上，任何复杂的处理过程都可以用这三种基本控制结构组合实现，因此，掌握这三种控制结构的基本思想，是学习程序设计的基础。在以后的章节中会做详细介绍。

1.2.3.3　N－S 图

两位美国学者 Nassi 和 Shneiderman 于 1973 年提出了一种新的流程图形式，这就是 N－S 流程图，也称为盒图或 CHAPIN 图。N－S 图的每一种基本结构都是一个矩形框。完全去掉了带有方向的流程线，限制了随意的控制转移，保证了程序的良好结构。整个算法可以像堆积木一样，由一些基本的矩形框图顺序排列组成一个大矩形。N－S 图形象直观，有效地保证了设计的质量，但是确定后进行修改比较麻烦。

顺序、选择和循环这三种基本结构相对应的 N－S 流程图如图 1-9 所示。

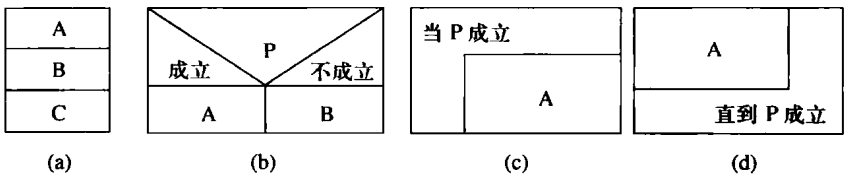


图 1-9　三种基本结构的 N－S 图

（a）顺序结构；（b）选择结构；（c）当型循环结构；（d）直到型循环结构。

【例 1.7】用 N－S 图表示例 1.1 中两数交换的算法，如图 1-10 所示。

【例 1.8】用 N－S 图表示例 1.2 中求分段函数的算法，如图 1-11 所示。

【例 1.9】用 N－S 图表示例 1.3 中求两数最大公约数的算法，如图 1-12 所示。

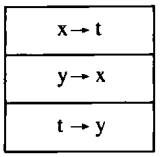


图 1-10　例 1.7 N－S 图

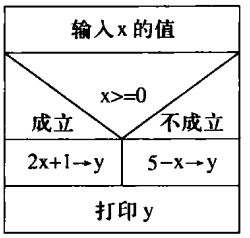


图 1-11　例 1.8 N－S 图

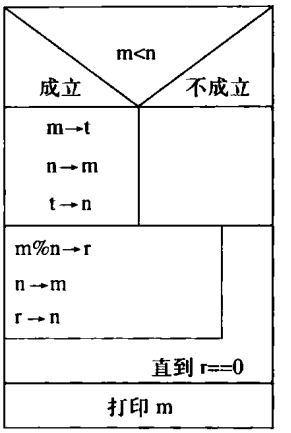


图 1-12　例 1.9 N－S 图

在结构化的程序设计方法中，常用的流程图、N－S 图和 PAD 图都是表示算法的图形工具。对于程序设计的初学者来说，流程图有其显著的优点，流程线非常明确地表示了算法的执行方向，便于初学者对程序控制结构的学习和理解。N－S 图和 PAD 图是软件工程中使用的图形工具。

1.2.3.4　伪代码

伪代码(Pseudo code)是另一种算法描述语言，介于自然语言与编程语言之间。通过前面两个小节的学习，我们发现流程图和 N－S 图表示算法直观易懂，但是绘制过程比较麻烦，并且修改起来比较复杂。所以相对于它们来说，伪代码结构性较强，比较容易书写和理解，修改起来也相对方便。

伪代码没有统一的标准，不拘泥于语言的语法结构，可以用英文、汉字、中英文混合表示算法，也可以采用与程序设计语言类似的形式，但要保证算法正确与清晰易读。

【例 1.10】用伪代码表示例 1.1 交换 x 和 y 的值。

```
开始
x→t
y→x
t→y
结束
```

【例 1.11】用伪代码表示例 1.2 求分段函数的算法。

```
开始
输入 x 的值
if  x>=0
    2x+1→y
else
    5-x→y
```

```
print y
结束
```

【例 1.12】用伪代码表示例 1.3 求两数最大公约数的算法。

以辗转相除法为例：

```
开始
if m<n
{    m→t
     n→m
     t→n}
m%n→r
while r<>0
{    n→m
     r→n
     m%n→r}
print n
结束
```

前面我们讲到的是描述算法的方法，计算机是无法识别它们的。但是只要有了正确的算法，再将其转换成一种计算机语言程序，就能够用计算机实现算法，得到我们想要的结果。

1.3 简单 C 程序介绍

用 C 语言语句编写的程序称为 C 程序或 C 源程序。我们先看几个简单的 C 程序。

1.3.1 C 程序示例

【例 1.13】显示输出“How are you?”

```
#include <stdio.h> /*文件包含操作*/
void main(){ /*定义主函数*/
    printf("How are you? \n");/*输出语句*/
}
```

说明：

（1）该程序仅由 main( )函数构成。main 表示“主函数”。每个 C 语言程序都必须有一个 main 函数，它是每一个 C 语言程序的执行起点。无论 main 函数在程序中的哪个位置，C 程序总是从 main 函数处开始执行。

（2）用“{}”括起来的是 main 函数的函数体。main 函数中的所有语句都写在这一对“{}”之间。本例 main 函数中只有一条语句。

（3）printf 是 C 语言的格式输出函数，用于程序的输出。“\n”是换行符，即在输出完“How are you?”后回车换行。

（4）每条语句用“;”号结束。

（5）/*……*/括起来的部分是注释部分，便于人们的理解和阅读，并不会编译、运行。注释可以放在程序任何位置。

（6）#include <stdio.h>是文件包含操作。是指一个源文件可以将另外一个源文件的全部内容包含进来。因为程序中用到格式输出函数 printf，所以要调用 stdio.h(标准输入/输出头