普通高等教育“十二五”规划教材

C语言程序设计
（含习题与实验指导）

主　编　海　燕
副主编　王　卉　闫雒恒

科学出版社

普通高等教育“十二五”规划教材

C语言程序设计

（含习题与实验指导）

主　编　海　燕
副主编　王　卉　闫雒恒

科学出版社
北　京

内 容 简 介

全书由主教材和配套“习题与实验指导”组成，它是编者通过长期教学实践编写而成的。主教材分为12章，包括：C语言概述，数据类型、运算符和表达式，语句与顺序，选择，循环程序结构，数组，函数，编译预处理，指针，结构体和共用体，位运算，文件操作等。

习题与实验指导分三部分，包括理论与指导、实验与指导、全国计算机等级考试二级C语言试题。其中理论与指导包含本章要点、典型例题解析、测试练习、测试练习参考答案四个模块。

全书内容编排由浅入深、循序渐进、注重实践、实例丰富，可作为大学各专业公共课教材和全国计算机考试参考用书，同时也可作为相关领域的工程技术人员的学习参考用书。

图书在版编目(CIP)数据

C语言程序设计：含习题与实验指导/海燕主编. —北京：科学出版社，2012.9

（普通高等教育“十二五”规划教材）

ISBN 978-7-03-035502-7

Ⅰ. ①C… Ⅱ. ①海… 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(2012)第206531号

责任编辑：张丽花 于海云 / 责任校对：宋玲玲

责任印制：闫 磊 / 封面设计：迷底书装

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

北京市安泰印刷厂印刷

科学出版社发行 各地新华书店经销

*

2012年9月第 一 版 开本：787×1092 1/16

2012年9月第一次印刷 印张：31

字数：643 000

定价：66.00元（含习题与实验指导）

(如有印装质量问题，我社负责调换)

前　言

C语言是目前用得最为广泛的程序设计语言，具有代码效率高、运算符和数据结构丰富、表达和运算能力强、程序精练等特点。C语言不仅具有低级语言可以直接对硬件进行操作的特性，也有其他高级语言所具有的良好的可读性和可移植性，因此既可以用其替代汇编语言来编写系统软件，也可以用其来编写应用软件。目前很多高等学校把C语言作为计算机专业或非计算机专业的程序设计和开发课程，各类计算机等级考试也将C语言列为重点考试科目。

全书由主教材和配套“习题与实验指导”组成，它是编者通过长期教学实践编写而成的，内容编排由浅入深、循序渐进、重点突出、难点分散、注重实践、实例丰富、面向应用，各章附有小结和适量习题，便于自学。本书具体特点如下：

(1) 应用型教材定位。本书以丰富的实例讲述 C 语言程序设计，深化读者对程序设计的理解，使之学会用程序设计的思维方法指导软件开发的实践，提高读者的计算机应用能力。

(2) 与等级考试紧密结合。目前大多数选用的教材很大程度上与学生参加国家计算机等级考试的结合性不强。本书附有近两年计算机等级考试二级 C 语言的真题及答案，不仅使学生更好地理解和掌握知识点，同时能够与等级考试的重点、难点紧密结合，满足学生等级考试的实际需求。

(3) 教辅资源配套。本书配套有习题与实验指导，习题可以帮助学生更好地理解知识点，实验可以提高其操作和应用能力。

主教材共 12 章。第 1 章概要介绍了 C 语言及在 Visual C++ 6.0 环境下如何运行 C 语言程序。第 2 章介绍了 C 语言的语法基础：数据类型、运算符和表达式。第 3~5 章分别介绍顺序、选择、循环结构程序设计的基本语句与方法。第 6 章介绍了数组的概念、定义方法与程序编写。第 7 章介绍了模块化程序设计思想、函数的定义与使用方法、变量的作用域和存储类型等。第 8 章介绍了编译预处理命令。第 9 章介绍了指针、指针变量与程序编写。第 10 章介绍了结构体、共用体、枚举类型，以及链表及其应用。第 11 章介绍了位运算的概念、各种位运算及其运算规则。第 12 章介绍了 C 文件的基础知识、基本的文件操作与程序编写。

习题与实验指导分三大部分：理论与指导、实验与指导、全国计算机等级考试二级 C 语言试题。其中第一部分是按主教材章节编写的习题指导，主要包含本章要点、典型例题解析、测试练习、测试练习参考答案四个模块；第二部分是按照主教材章节组织的实验；第三部分提供了近两年的全国计算机等级考试二级 C 语言试题。

全书由华北水利水电学院计算机基础教研室的教师共同编写完成，海燕担任主编，王卉、闫雒恒担任副主编。其中，第 1、2 章由张贞贞编写，第 3、4 章由石秋华编写，第 5、11 章由王卉编写，第 6、12 章由海燕编写，第 7 章由杨雪青编写，第 8、10 章由赵凯编写，第 9 章由闫雒恒编写；习题与实验指导的第三部分由杨雪青整理提供，附录由石秋华整理

提供。

在本书的写作过程中，得到同事们与同行们的许多帮助，在此一并致以深深的谢意！

百密一疏，虽有严谨的编写、细致的审校，然时间、水平所限，难免存在疏漏和不妥之处，敬请广大师生、同行专家批评指正，谨表谢忱。

编　者

2012 年 8 月

目　录

第1章 概 述

C语言是一种国内外广泛流行的、已经得到普遍应用的程序设计语言，它既可以用来编写系统软件，又可以用来编写应用软件。

本章是C语言程序设计的概述内容，主要讲述了C语言的产生与发展、C语言程序的基本结构及特点、算法的概念及描述方法。在本章最后，通过一个示例介绍C语言程序设计的完整过程，包括问题分析、算法设计、编写程序，以及使用Visual C++ 6.0编辑运行程序等内容。

1.1 C语言概况

1.1.1 C语言的产生与发展

早期的操作系统软件主要是用汇编语言编写的。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。改用高级语言编写系统软件，可以提高程序的可读性和可移植性，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，如对内存地址的操作、位操作等)。因此，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。C语言就是在这种情况下应运而生的。

C语言是在B语言的基础上发展起来的，它的根源可以追溯到ALGOL 60。1960年出现的ALGOL 60是一种面向问题的高级语言，它离硬件比较远(也就是说不能编写程序直接对计算机的硬件进行控制)，因而不宜用来编写系统程序。1963年，英国剑桥大学推出了CPL (Combined Programming Language)语言。CPL语言在ALGOL 60的基础上更接近硬件一些，但它规模比较大，难以实现。

1967年，英国剑桥大学的Matin Richards对CPL语言做了简化，推出了BCPL(Basic Combined Programming Language)语言。1970年，美国贝尔实验室的Ken Thompson以BCPL语言为基础，又进一步做了简化，设计出了很简单的而且很接近硬件的B语言(取BCPL的第一个字母)，并用B语言写出了UNIX操作系统，在PDP-7计算机上实现。1971年，在PDP-11/20计算机上实现了B语言，并写出了UNIX操作系统。

1972~1973年，贝尔实验室的D. M. Ritchie在B语言的基础上设计出了C语言(取BCPL的第二个字母)。C语言既保持了BCPL和B语言的优点(精练、接近硬件)，又克服了它们的缺点(过于简单、无数据类型等)。最初的C语言只是为描述和实现UNIX操作系统而提供的一种工作语言。1973年，K. Thompson和D. M. Ritchie两人合作把UNIX的90%以上代码用C改写(UNIX第5版。原来的UNIX操作系统是1969年由美国的贝尔实验室的K. Thompson和D. M. Ritchie开发成功的，使用汇编语言编写的)，实现了C语言的现实应用。

后来，C语言多次做了改进，但主要还是在贝尔实验室内部使用。直到1975年UNIX

第 6 版公布后，C 语言的突出优点才引起人们普遍注意。1977 年出现了不依赖于具体机器的 C 语言编译版本，使将 C 程序移植到其他机器上时所做的工作得到大大简化，这也推动了 UNIX 操作系统迅速在各种机器上实现。例如，VAX、AT&T 等计算机系统都相继开发了 UNIX。随着 UNIX 的日益广泛使用，C 语言也迅速得到推广。C 语言和 UNIX 可以说是一对孪生兄弟，在发展过程中相辅相成。1978 年以后，C 语言已先后移植到大、小、微型机上，并独立于 UNIX 和 PDP 计算机。现在 C 语言已风靡全世界，成为世界上应用最广泛的几种计算机语言之一。

Brian W. Kernighan 和 Dennis M. Ritchie(简称 K&R)以 1978 年发表的 UNIX 第 7 版中的 C 编译程序为基础，合著了影响深远的名著 *The C Programming Language*，该书介绍的 C 语言成为后来广泛使用的 C 语言的基础，也被称为标准 C。1983 年，美国国家标准化协会(ANSI)根据 C 语言问世以来各种版本对 C 语言的发展和扩充，制定了新的标准，称为 ANSI C。ANSI C 比原来的标准 C 有了很大的发展。1987 年，ANSI C 又公布了新标准——87 ANSI C。目前流行的 C 编译系统都是以它为基础的。K&R 在 1988 年修改了他们的经典著作 *The C Programming Language*，按照 ANSI C 的标准重新写了该书。

1.1.2 C 语言的特点

C 语言既有高级语言的特点，又具有汇编语言的特点；既是一个成功的系统设计语言，又是一个实用的程序设计语言；既能用来编写不依赖计算机硬件的应用程序，又能用来编写各种系统程序；是一种受欢迎、应用广泛的程序设计语言。C 语言具有以下基本特点：

1. 简洁紧凑、灵活方便

C 语言一共只有 32 个关键字，9 种控制语句，其程序书写自由，主要用小写字母表示。它把高级语言的基本结构和语句与低级语言的实用性相结合，使得 C 语言可以像汇编语言一样对位、字节和地址进行操作，而这三者是计算机最基本的工作单位。

2. 运算符丰富

C 语言的运算符包含的范围很广泛，共有 34 个运算符。C 语言把括号、赋值、强制类型转换等都作为运算符处理，从而使 C 的运算类型极其丰富，表达式类型多样化。灵活地使用其中的各种运算符可以实现其他高级语言中难以实现的运算。

3. 数据结构丰富

C 语言的数据类型有整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等，能用来实现各种复杂的数据结构；引入了指针概念，使程序效率更高。另外，C 语言具有强大的图形功能，支持多种显示器和驱动器，且计算功能、逻辑判断功能强大。

4. 结构化程序设计语言

结构化程序设计语言的显著特点是代码及数据的分隔化，即程序的各个部分除了必要的信息交流外彼此独立。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 语言是以函数形式提供给用户的，这些函数可方便地调用，并具有多种循环、条件语句控制程序流向，从而使程序完全结构化。

5. C 语言的语法限制不太严格、程序设计自由度大

一般的高级语言语法检查比较严，几乎能够检查出所有的语法错误，而 C 语言允许程序编写者有较大的自由度。

6. C 源程序的结构特点

①一个 C 语言源程序可以由一个或多个源文件组成。

②每个源文件可由一个或多个函数组成。

③一个源程序不论由多少个文件组成，都有一个且只能有一个 main()函数，即主函数。

④源程序中可以有预处理命令(include 命令仅为其中的一种)，预处理命令通常放在源文件或源程序的最前面。

⑤每一个说明、每一个语句都必须以分号结尾。但预处理命令、函数头和花括号“}”之后不能加分号。

⑥标识符、关键字之间必须至少加一个空格以示间隔。若已有明显的间隔符，则也可不再加空格来间隔。

C 语言有很多的优点，指针就是 C 语言的一大特色，可以说 C 语言优于其他高级语言的一个重要原因，就是因为它有指针操作可以直接进行靠近硬件的操作，但是 C 语言的指针操作也给它带来了很多不安全的因素。C++在这方面做了很好的改进，在保留了指针操作的同时又增强了安全性。

当然，C 语言也有自身的不足，比如：C 语言的语法限制不太严格，对变量的类型约束不严格，影响程序的安全性，对数组下标越界不做检查等。从应用的角度，C 语言比其他高级语言较难掌握。

1.1.3 C 语言程序创建的基本步骤

C 语言程序的创建有 5 个基本步骤或过程：编辑→编译→连接→调试→运行。

这些过程大部分都会在语言所提供的编译器软件中得到实现。所有的高级语言都是不能直接在计算机上运行的，需要一个称为编译器的软件来把高级语言转换为在计算机中可以直接运行的机器代码。

1. 编辑

编辑过程就是创建和修改 C 语言程序代码的过程，大部分的 C 语言编译器都有一个编辑器，通常编辑器提供了编写、管理、开发与测试程序的环境，我们一般称其为集成开发环境(简称 IDE)。也可以采用其他的纯文本格式的编辑器来编写程序，这里一定要注意：类似 Word 等带有格式控制功能的编辑器是不能作为程序编辑器来使用的。大部分的程序编辑器都带有大量的编辑程序编辑的辅助功能，例如，程序自动缩进、重要的语言元素用高亮显示、函数的参数提示等。

在 UNIX 或 Linux 环境下，最常用的是 vi，也有人采用 emace。在 Windows 环境下有很多集成编辑环境可以采用，例如，微软提供的 Visual C++集成开发环境就是被用户广泛应用的 IDE。

2. 编译

编译器可以将源代码转换成机器语言，在编译过程中，会找出并报告错误。这个阶段的输入是在编辑阶段产生的源程序文件，输出为对象代码文件，这些文件在 Windows 环境下扩展名是.obj，在 UNIX 环境下是.o。对于命令行编译器需要了解编译命令的使用格式和参数，这些可以在编译器的使用手册中找到，对于集成环境都有编译菜单选项或快捷方式供选择。

3. 连接

连接器将源文件中由编译器产生的各种模块组合起来，再从 C 语言提供的程序库中添加必要的代码模块，将它们组合成一个可执行的文件。连接器也可以检测和报告错误信息，例如，遗漏了程序的某个部分，或引用了一个根本不存在的库组件。

通常情况下，由于问题本身的复杂性，大部分的源程序文件都不可能用一个文件来完成，因此，人们经常把源程序文件分成多个部分分别进行编辑和编译调试，最后再通过连接器连接成为一个可执行文件，这样做可以避免简单输入错误的发生。

程序库所提供的例程，可以扩展程序的功能，或借鉴其他程序中的功能模块。例如，C 语言程序库中包含输入、输出、计算平方根、比较字符串、读取日期和时间信息等操作。

连接阶段出现错误，表明我们必须修改源程序代码；反过来，如果成功，就会产生一个可执行文件。在 Windows 环境下，这个执行文件的扩展名为.exe；现在 UNIX 环境下没有扩展名。

大多数的 IDE 都有 Build 选项，可以一次完成编译和连接的功能。

4. 调试

调试是程序创建过程中必不可少的重要环节，人们经常说："程序是调试出来的"，足见调试在程序创建过程中的作用。程序调试就是根据编译和连接过程中给出的错误和警告信息查找和修改源程序中错误的过程。要提高程序的调试能力必须对编译器所给出错误的类型、错误的表述、错误的位置等迅速做出正确的判断。程序的调试能力只能在调试程序的过程中得到提高。

5. 运行

运行就是在完成以上 4 个步骤后，在相应的操作系统环境下运行生成的可执行文件。但是，在这个阶段也有可能出现各种错误，不管出现哪种错误，都需要返回到编辑阶段，查找原因，然后修改错误。

1.2 简单的 C 语言程序

1.2.1 一个简单的 C 程序

我们先通过一个简单的 C 程序来了解 C 语言程序。

【例 1.1】 在计算机屏幕上输出"Hello World!"。

```
#include <stdio.h>
void main( )
{
  printf("Hello World!\n");
    }
```

程序的运行结果如图 1-1 所示。

图 1-1 例 1.1 的运行结果

程序中 main 是主函数名，C 语言规定必须用 main 作为主函数名，而且函数名后的一对圆括号不能省略，圆括号中内容可以是空的。一个 C 程序可以包含任意多个函数，但必须有且只有一个主函数。一个 C 程序总是从主函数开始执行的，最后在主函数结束。函数体需用花括号括起来，左括号表示函数体的开始，右括号表示函数体的结束。其间可以有定义(说明)部分和执行语句部分；每一条语句都必须用分号“;”结束，语句的数量不限，程序中由这些语句向计算机系统发送指令。本程序函数体内只有一条输出语句，双引号内的内容原样输出，“\n”表示输出字符后换行。

main( )前面的 int 表示主函数的数据类型是整型，return 0 表示函数返回值为 0。

#include <stdio.h>是一条预处理命令，用“#”号开头，后面不能加“;”号。stdio.h 是系统提供的头文件，其中包含有关输入输出函数的信息。

1.2.2 C 语言程序的基本结构

每种计算机语言都有自己关于程序结构的基本规则，C 语言规定，一个完整的 C 程序应该包含以下三部分：

1. 必要的包含命令和预处理命令

这一部分主要定义一个程序中引用了哪些标准函数，也就是引导编译程序到指定的位置调用相应的程序代码添加到用户程序中。包含命令是实现多文件程序设计的基本手段，包含文件也称为库文件，分为系统提供的和用户自定义的两种。系统提供的库文件一般保存在安装目录下的 include 目录中；用户在实现大型程序时也定义了很多的库文件，这些文件的保存可以根据用户的要求来存储。在阅读标准的 C 程序时，我们一定会遇到很多包含命令，它们反映了程序中不同文件之间的相互关系，是理解程序结构的基础。

2. 一个唯一的 main 函数

main 函数的基本格式为

```
void   main( )
{

}
```

main( )函数称为主函数，一个程序只能有一个。main( )函数可以带参数，也可以有返回值，具体形式在第 7 章介绍。

程序中的一对大括号表示主程序的开始和结束。这样的大括号在 C 语言中表示一个处理阶段的开始和结束，称为复合语句的开始和结束，必须在程序中成对出现。规范地表示大括号及高效地确定大括号之间的对应关系是 C 程序的一个基本要求，在一个标准的、实用的 C 程序中会出现大量的大括号。

3. 用户自定义的函数

由于用 C 语言开发的程序可能解决各种各样的问题，实际问题的需求是多种多样的，任何 C 语言编译系统都不可能提供所有的、能实现各种需求的库函数，因此人们一定要在自己的程序中开发适合自己需求的程序段，这样的程序段称为函数。关于函数的具体定义和使用将在第 7 章介绍。

一个标准的 C 程序由一个主函数和大量的自定义函数组成。

1.2.3 C 语言程序的书写规范

从书写清晰以及便于阅读、理解和维护的角度出发，在书写 C 语言程序时应遵循以下规则：

①一个说明或一个语句占一行。

②函数与函数之间加空行，以清楚地分出程序中有几个函数。

③用{}括起来的部分，通常表示了程序的某一层次结构。{}一般与该结构语句的第一个字母对齐，并单独占一行。

④低一层次的语句或说明可比高一层次的语句或说明缩进若干格后书写。同一个层次的语句向左对齐，以便看起来更加清晰，可增加程序的可读性。

⑤在有数据的输入时，运行时最好要出现输入提示；对于数据的输出，也要有一定的提示和格式。

⑥为了增加程序的可读性，对语句和函数应加上适当的注释。

在编写程序时应力求遵循以上规则，以养成良好的编程风格。

1.3 算　　法

1.3.1 算法概述

1. 算法的含义

人们在科学实验、生产实践中有大量问题需要求解，如科学计算、数据处理及各种管理问题等。要解决这些问题，首先需要分析所研究的对象，提出对问题的形式化定义，给出求解方法的形式描述。对问题的形式化定义叫做数学模型，而对问题求解方法的形式描述称为算法。

广义地讲，算法是解决问题的逻辑步骤，是对特定问题求解步骤的一种描述。简单地说，任何解决问题的过程都是由一定的步骤组成的，解决问题确定的方法和有限的步骤称为算法。只有通过算法能够描述出来的问题，才能够通过计算机求解。对同一个问题，可以有不同的解题方法和步骤，也就有不同的算法。

计算机算法是用程序解决问题的逻辑步骤，是指令的有限序列。

计算机算法可分为两大类。

①数值运算算法：求解数值。

②非数值运算算法：事务管理领域。

正确的算法有三个条件：

①每个逻辑步骤有可以实现的语句来完成。

②每个步骤间的关系是唯一的。

③算法要能中止(防止死循环)。

【例 1.2】 求 1×2×3×4×5。

最原始方法：

步骤 1：先求 1×2，得到结果 2。

步骤 2：将步骤 1 得到的乘积 2 乘以 3，得到结果 6。

步骤 3：将 6 再乘以 4，得 24。

步骤 4：将 24 再乘以 5，得 120。

这样的算法虽然正确，但太繁琐。

改进的算法：

步骤 1：使 t=1。

步骤 2：使 i=2。

步骤 3：使 t×i，乘积仍然放在变量 t 中，可表示为 t×i→t。

步骤 4：使 i 的值加 1，即 i+1→i。

步骤 5：如果 i≤5，返回重新执行步骤 3 以及其后的步骤 4 和步骤 5；否则，算法结束。

如果计算 100！只需将步骤 5 中 i≤5 改成 i≤100 即可。

如果要计算 $1\times3\times5\times7\times9\times11$，算法也只需做很少的改动。

步骤 1：1→t。

步骤 2：3→i。

步骤 3：t×i→t。

步骤 4：i+2→i。

步骤 5：若 i≤11，返回步骤 3，否则，结束。

该算法不仅正确，而且是较好的计算机算法、因为计算机是高速运算的自动机器，实现循环轻而易举。

2. 算法的基本特征

算法是一个有穷规则的集合，这些规则确定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、一步一步地执行计算，经过有限步骤后终止计算并产生输出结果。归纳起来，正确的算法有以下基本特征。

①可行性：算法中要执行的运算和操作是最基本的，它们都能够精确地进行。这样，经过算法描述的一系列步骤的确切动作，最后能得到正确的结果。

②确定性：算法中每一步骤都必须有明确定义，即算法中所要执行的动作应有严格的规定，而没有歧义性。

③有穷性：算法必须能在有限的时间内做完，即能在执行有限个步骤后终止，包括合理的执行时间的含义；算法要能终止，不能造成死循环。

④输入：算法有零个或多个输入，即算法开始执行之前，要设置好初始值。

⑤输出：算法有一个或多个输出，这一输出就是算法的最终结果。该结果是与输入有关并经过确定的计算步骤后得到的。

下列过程就不是一个正确的算法。

步骤 1：令 n 等于 0。

步骤 2：n 加 1。

步骤 3：转向步骤 2。

如果利用计算机执行此过程，从理论上讲，计算机将永远执行下去，即形成死循环。

而下列过程就是一个正确的算法。

步骤 1：令 n 等于 0。

步骤 2：n 加 1。

步骤 3：如果 n 小于 100，则转向步骤 2；否则停止。

实质上，算法反映的是解决问题的思路。许多问题，只要仔细分析对象数据，就能够找到处理方法。

1.3.2 算法的表示

算法是解题过程的精确描述，那么，用什么描述这一解题过程呢？要描述必须采用某种语言，而语言有多种形式，如自然语言、伪代码、流程图、N-S 图、计算机程序设计语言等。

1. 用自然语言表示

自然语言是人们日常使用的语言，如汉语、英语、日语等。自然语言的规则是人们所使用语言的规则。用自然语言描述算法是最受欢迎的，因为它直观、通俗易懂、为人们所熟悉。当然，用自然语言来描述解题过程，可能会产生歧义性，容易导致算法执行的不确定性。另外，用自然语言表示的算法要翻译成计算机能认识、理解的形式是不容易的，要做大量的工作。

【例 1.3】 对一个大于或等于 3 的正整数 n，判断它是不是一个素数。

算法可表示如下。

步骤 1：输入 n 的值。

步骤 2：i=2。

步骤 3：n 被 i 除，得余数 r。

步骤 4：若 r=0，表示 n 能被 i 整除，则打印 n “不是素数”，算法结束；否则执行步骤 5。

步骤 5：i+1→i。

步骤 6：若 $i \leqslant n-1$(或 $i \leqslant \sqrt{n}$),返回步骤 3；否则打印 n “是素数”，算法结束。

2. 传统流程图

用图形表示算法，直观形象，易于理解。流程图用一些图框来表示各种操作。美国国家标准化协会 ANSI 规定了一些常用的流程图符号，如图 1-2 所示。

图 1-2 中菱形框的作用是对一个给定的条件进行判断，根据给定的条件是否成立来决定如何执行其后的操作。它有一个入口、两个出口，其流程如图 1-3 所示。

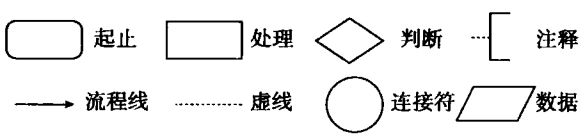

图 1-2 流程图符号

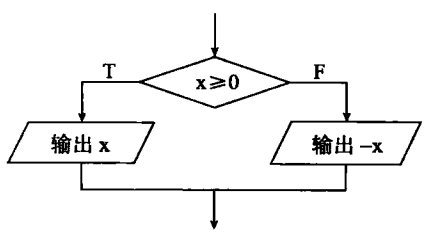

图 1-3 条件判断示意图

菱形框两侧的 T 和 F 表示“真”(TRUE)和“假”(FALSE)。

【例 1.4】 画出求 1+2+3+ … +100 之和的流程图。

流程图如图 1-4 所示。