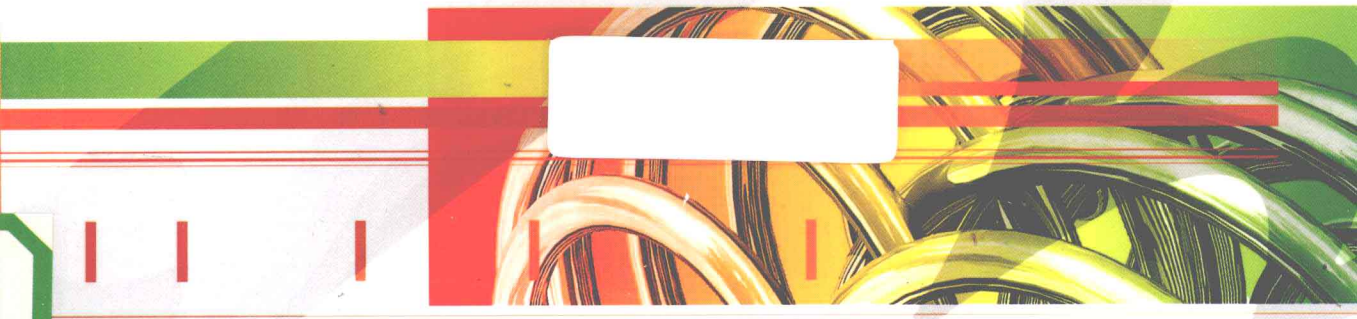
高等学校计算机规划教材

C语言程序设计

◆ 呼克佑 主编 ◆ 林福平 邓红霞 段利国 曹 棣 副主编



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http: //www.phei.com.cn

高等学校计算机规划教材

C 语言程序设计

呼克佑　主编

林福平　邓红霞　段利国　曹　棣　副主编

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书系统地介绍了C语言的基本概念、语法和语义，包括数据类型、常量、变量、运算符和表达式、语句、数组、函数、结构体、指针、文件等。将C语言的介绍和结构化程序设计方法有机地结合在一起，通过大量实例的分析、编程，帮助读者尽快掌握C语言和用C语言编写程序，通过基本算法思想介绍和应用实例掌握用C语言描述算法和基本算法在程序设计中的应用。教材提供了大量精心设计的习题和实验，通过完成习题和实验学习C语言和使用C语言编写程序。本书配有电子课件、源代码等教学资源。

本书可作为高等院校学生学习C语言程序设计课程的教材，也可作为广大计算机程序设计人员和计算机程序设计爱好者的参考书，同时可供参加相关考试的读者参考。

图书在版编目(CIP)数据

C语言程序设计 / 呼克佑主编. — 北京：电子工业出版社，2013.2
高等学校计算机规划教材
ISBN 978-7-121-19284-5

I. ①C… II. ①呼… 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303998号

策划编辑：史鹏举
责任编辑：王凌燕
印　　刷：涿州市京南印刷厂
装　　订：涿州市京南印刷厂
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编：100036
开　　本：787×1092　1/16　印张：17.75　字数：454千字
印　　次：2013年2月第1次印刷
定　　价：35.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010)88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：(010)88258888。

前　言

近年来，学习计算机程序设计的人几乎都选择C语言作为首选语言。《C语言程序设计》不仅是计算机类专业的一门专业基础课，也是大多数理工类专业的必修课。

为什么要学习C语言？大致有以下几个理由：

- C语言可以作为学习计算机程序设计语言的入门语言。
- C语言是编写操作系统的首选语言，与计算机硬件打交道时灵巧且高效。
- C语言具有现代高级程序设计语言的基本语法特征。
- 常用的面向对象程序设计语言如C++和Java，其基本语法来源于C语言。
- 许多用C语言编写的系统需要维护。
- 用于要求程序高速运行领域的编程，如嵌入式系统、通信程序。
- 游戏设计者和黑客少不了C语言。
- C语言使用者和爱好者众多。

学习C语言，需要理解和掌握C语言中诸如关键字、标识符、数据类型、常量、变量、运算符和表达式等基本概念，也需要掌握语句、数组、函数、指针、结构体、文件的使用。同时需要了解和掌握计算机的解题过程、算法的设计和表示、结构化程序设计方法等内容。

本书全面、系统地介绍了C语言的语法、语义和程序设计技术，内容丰富，结构完整，力求精练实用。对抽象概念的叙述通俗易懂，内容安排突出重点、分散难点，实例充实全书。全书共分10章。第1章介绍了C语言的简史、特点，计算机解题过程和算法的概念。第2章介绍了C语言的基本数据类型、常量、变量、运算符和表达式。第3章介绍了结构化程序设计的三种基本结构及其控制语句。第4章介绍了数组的使用及其程序设计方法。第5章介绍了指针的使用及其程序设计方法。第6章介绍了用户自定义函数的定义、调用和声明方法，以及变量的作用域规则。第7章介绍了C语言的复杂数据类型结构体和动态数据结构。第8章介绍了文件的基本概念和操作方法。第9章介绍了宏、文件包含、条件编译等预处理命令的使用。第10章是上机指导。每章均给出了覆盖几乎所有知识点的习题供学生练习。最后提供了非常实用的附录，如常用库函数介绍、编译连接常见错误信息表等。

本书作为教材使用时，建议讲授学时不少于40课时，实验学时不少于16课时。

本书配有电子课件、源代码等教学资源，可登录电子工业出版社华信教育资源网（www.hxedu.com.cn），免费注册、下载。

参加本书编写的人员都是长期工作在教学第一线的教师，从事计算机专业课程教学多年，有丰富的教学经验。本书由呼克佑老师主编并编写第4、8、9章，第1、2章由林福平老师编写，第3章由邓红霞老师编写，第5、10章由段利国老师编写，第6章由曹棣老师编写，第7章由阎志中老师编写，全书由呼克佑老师统稿，由冯秀芳老师主审。

在本书的编写过程中得到了太原理工大学相关领导的大力支持和帮助，在此表示衷心的感谢。由于作者水平有限，书中疏漏在所难免，殷切盼望读者批评指正。

编　者

目　录

第1章　C语言与程序设计

本章主要内容

- C语言发展简史与特点
- 计算机解题过程
- 算法及其表示
- 常用算法介绍
- 结构化程序设计方法

众所周知，计算机系统是由硬件和软件组成的，程序是软件系统中不可或缺的重要组成部分，而程序是用计算机语言编写的。学习计算机语言和程序设计技术，就是要学会用计算机语言编写解决实际问题的程序，这对于软件开发人员来说是重要的一步。

指示计算机完成特定操作的命令称为指令，计算机硬件所有指令的集合称为该计算机的指令系统或机器语言。使用机器语言编写程序效率低且不方便，没有人直接使用机器语言编写程序。C语言是一种既低级又高级的计算机程序设计语言，用C语言代替机器语言编写程序，可以达到机器语言程序的效果——程序执行速度快且使用存储空间小，同时C语言又是一种高级程序语言，编写程序的效率与其他高级程序语言并无差异。

学习“C语言程序设计”主要包括两个方面：一是学习C语言的语法规则；二是学习程序设计方法。在掌握C语言的语法规则的基础上，运用程序设计方法设计解决问题的算法，从而设计出解决问题的C语言程序。

学习计算机语言，当然也包括C语言，主要是学习语法和语义两个方面。计算机语言的语法是严格定义的，不允许有语法错误。语法是一些规定，如各种语句及各种语法成分都有各自的规定。同时计算机语言的语义是唯一的，没有二义性的，各种语句及语法成分有各自的语义。只有掌握计算机语言的语法，理解语义，才能正确使用计算机语言编写没有语法错误和逻辑错误的程序。

程序设计是给出解决特定问题程序的过程，包括问题分析、算法设计、程序源代码设计、测试、调试和维护。源代码可以用C语言编写，也可以用其他计算机语言编写。程序设计的目的是得到一个指令序列，提供给计算机来执行，使得问题得到解决。在程序设计过程中，往往需要不同方面、不同领域的专业知识，包括应用领域知识、特定算法知识和形式逻辑知识等。

学习程序设计时，需要了解和掌握计算机的解题过程、算法的设计和表示、结构化程序设计等内容。同时需要用具体的计算机语言编写程序。在这个过程中，需要学习和掌握程序的编辑、编译、调试和运行等有关操作。

C 语言程序设计是一门实践性很强的课程。在学习过程中，不仅要掌握基本概念、语法和语义等内容，还需要注重实验环节，通过自己动手编写程序，在完成 C 语言程序的编辑、编译、调试和运行的过程中，加深对 C 语言的理解，体会编写程序过程及有关的关键技术。与此同时，培养编程兴趣，提高编程能力。

通过学习 C 语言，可以为进一步学习其他计算机语言，包括目前常用的面向对象的语言如 C++、Java 等各种计算机语言奠定基础。

1.1 C 语言发展简史

C 语言是一种广泛使用的面向过程的计算机程序设计语言，既适合于系统程序设计，又适合于应用程序设计。C 语言在操作系统、软件工程、图形学、数值分析等诸多领域得到了广泛的应用。

早期的计算机语言，除了汇编语言之外，高级程序设计语言有 FORTRAN 语言、ALGOL 语言和 COBOL 语言等。后来出现了大量的各种各样的高级语言，较有影响的有 BASIC 语言、PASCAL 语言、C 语言、C++语言和 Java 语言等。C 语言的发展历程大致如图 1-1 所示。

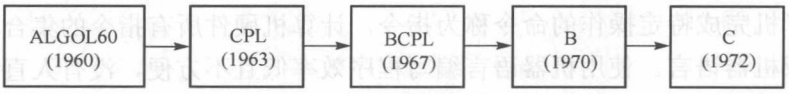


图 1-1　C 语言的发展历程

1960 年出现的 ALGOL60 很受程序设计人员欢迎，用 ALGOL60 描述算法很方便，但硬件操作能力较弱，不宜用来编写系统程序。1963 年英国剑桥大学在 ALGOL 语言的基础上推出了 CPL(复合程序设计语言)，增加了直接对硬件的处理能力，但由于规模大，因此实现困难。1967 年剑桥大学的马丁·理查德(Matin Richards)对 CPL 语言进行了简化，推出了 BCPL(基本复合程序设计语言)。1970 年美国贝尔实验室的肯·汤普逊(Ken Thompson)对 BCPL 语言进一步简化，突出了硬件处理能力，取名 B 语言(BCPL 的第一个字母)，并用 B 语言写了第一个 UNIX 操作系统程序，但 B 语言过于简单，功能有限。1972 年贝尔实验室的丹尼斯.M.里奇(Dennis.M.Ritchie)对 B 语言进行了完善和扩充，保留了 B 语言的硬件处理能力，扩充了数据类型，强调了通用性，这就是 C 语言(BCPL 的第二个字母)。1973 年肯·汤普逊和丹尼斯. M. 里奇两人合作，用 C 语言重写了 UNIX 操作系统，并在 PDP-11 计算机上加以实现。C 语言伴随着 UNIX 操作系统成为一种最受欢迎的计算机程序设计语言。

UNIX 操作系统因简洁、实用和高效率而受到人们欢迎，C 语言功不可没。1977 年出现了不依赖于具体机器的 C 语言编译版本“可移植 C 语言编译程序”，使 C 语言移植到各种不同的机器变得非常简单。1978 年，贝尔实验室的布朗. W. 卡尼汉(Brian. W. Kernighan)和丹尼斯. M. 里奇(Dennis.M.Ritchie)合著了“The C Programming Language”一书，对 C 语言的语法进行了规范化的描述，成为以后广泛使用的 C 语言的基础。随着微型计算机的普及，产生了各种不同的 C 语言版本，为了统一标准，美国国家标准学会(ANSI)于 1983 年

制定了 C 语言标准，这就是 ANSI C 标准。1990 年，C 语言成为国际标准化组织(ISO)通过的标准语言，这一 C 语言版本称为 C89 或 C90。1999 年通过的 ISO/IEC9899:1999 新标准称为 C99。

目前已经有一些 C 语言的编译器支持 C99。在实际编写程序过程中，应该了解和使用 C 语言的标准，使得编写的程序具有可移植性，可以运行于不同的计算机系统。

1.2　C 语言的特点

C 语言是一种通用的程序设计语言，语言本身简洁、灵活、表达能力强，被广泛用于系统软件和应用软件的开发，并且具有良好的可移植性。

C 语言的特点可概括如下：

(1) C 语言简洁、紧凑、灵活。C 语言的核心内容很少，只有 32 个关键字，9 种控制语句；程序书写格式自由，压缩了一切不必要的成分。

(2) 表达方式简练、实用。C 语言有一套强有力的运算符，达 44 种，可以构造出多种形式的表达式，用一个表达式就可以实现其他语言可能需要多条语句才能实现的功能。

(3) 具有丰富的数据类型。数据类型越多，数据的表达能力就越强。C 语言具有现代语言的各种数据类型，如字符型、整型、实型、数组、指针、结构体和共用体等，可以实现诸如链表、栈、队列、树等各种复杂的数据结构。其中的指针类型使得参数的传递简单并且迅速，同时节省内存空间。

(4) 具有低级语言的特点。C 语言具有与汇编语言相近的功能和描述方法，如地址运算和二进制数位运算等，还可以对硬件端口等资源进行直接操作，充分使用计算机的资源。C 语言既具有高级语言便于学习和掌握的特点，又具有机器语言或汇编语言对硬件的操作能力。因此，C 语言既可以作为系统描述语言，又可以作为通用的程序设计语言。

(5) C 语言是一种结构化语言，适合于大型程序的模块化设计。C 语言提供了编写结构化程序的基本控制语句，如 if–else 语句、switch 语句、while 语句和 do–while 语句等。C 语言程序是函数的集合，函数是构成 C 语言程序的基本单位，每个函数具有独立的功能，函数之间通过参数传递数据。程序员可以编写自己的函数。同时，不同操作系统的编译器都为程序员提供了大量的标准库函数，如输入/输出函数、数学函数和字符串处理函数等。灵活地使用标准库函数可以简化程序设计，提高编写程序效率。

(6) 各种版本的编译器都提供了预处理命令和预处理程序。预处理扩展了 C 语言的功能，提高了程序的可移植性，为大型程序的调试提供了方便。

(7) 可移植性好。程序从一个环境不经改动或稍加改动就可以移植到另一个完全不同的环境中运行。这是因为标准库函数和预处理程序将可能出现的与机器有关的因素与源程序分隔开来，使得针对不同的计算机硬件环境，可以重新定义有关的内容。

(8) 生成的目标代码质量高。由 C 源程序编译和链接得到的目标代码的运行效率比用汇编语言编写的也不过只低 10%～20%，可充分发挥机器的效率。

(9) C 语言语法限制不严，程序设计自由度大。C 语言程序在运行时不做诸如数组下

标越界和变量类型兼容性等检查，而是由编程者自己保证程序的正确性。C 语言几乎允许所有的数据类型的转换，字符型和整型可以自由混合使用，所有类型均可作逻辑型，可自己定义新的类型，还可以把某类型强制转换为指定的类型。实际上，这使编程者有了更大的自主性，能编写出灵活、优质的程序，同时也给初学者增加了一定的难度。所以，只有在熟练掌握 C 语言程序设计后，才能体会到其灵活性。

可以先简单了解以上特点，等掌握 C 语言之后，再次阅读就会有更为深刻的领会。

C 语言有许多优点，是一种优秀的计算机语言，但也存在以下的缺点。了解这些缺点，有助于实际使用 C 语言编写程序时做到扬长避短。

(1) C 语言程序的错误更隐蔽。C 语言的灵活性使得用它编写程序时更容易出错，而且 C 语言的编译器不检查这样的错误。与汇编语言类似，需要程序运行时才能发现这些逻辑错误。C 语言还会有一些隐患，需要引起程序员的重视，如将比较的“==”写成赋值“=”，虽然语法上没有错误，但是这样的逻辑错误往往不易发现，想要找出错误往往十分费时。

(2) C 语言程序有时会难以理解。C 语言语法成分相对简单，是一种小型语言。但是，其数据类型多，运算符丰富且结合性多样，使得对其理解有一定的难度。有关运算符和结合性，人们最常说的一句话是“先乘除，后加减，同级运算从左到右”，但是相比较而言 C 语言的运算符及表达式更加复杂。发明 C 语言时，为了减少字符输入，C 语言比较简明，同时也使得 C 语言可以写出常人几乎难以理解的程序。

(3) C 语言程序有时会难以修改。考虑到程序规模的大型化或者巨型化，现代编程语言通常会提供“类”和“包”之类的语言特性，这样的特性可以将程序分解成更加易于管理的模块。然而 C 语言缺少这样的特性，维护大型程序则显得比较困难。

1.3 计算机解题过程

使用计算机解决现实世界的问题是一种必然的选择。如何使用计算机解题？也就是如何编写解决问题的程序或软件？这是一个比较复杂的问题，需要使用软件工程的方法来解决，超出本书的范围。这里从学习 C 语言的角度简要认识计算机解题的过程。

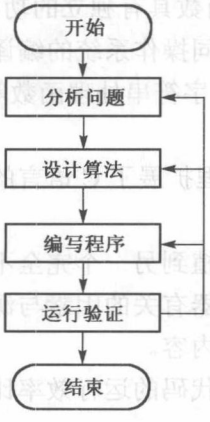


图 1-2 计算机解题过程

计算机的解题过程如图 1-2 所示，大致分为 4 个阶段，分别是分析问题、设计算法、编写程序和运行验证。

1. 分析问题

详细分析需要解决的问题，清楚地了解问题的需求。已知的原始数据有哪些，使用什么方法或数学模型，要得到什么结果。分析问题就是明确做什么(What to do)。

2. 设计算法

明确了做什么之后，就需要考虑怎么做(How to do it)。通常将解决问题的方法或数学模型转换为解决问题的步骤，即设计算法。设计算法是计算机解

题过程中的一个具有创造性的重要环节，所设计的算法是否合理、正确直接关系到问题能否解决。同时，解决问题的方法往往不是唯一的，可能有多种解决方法，需要设计出多种算法，通过分析比较，从中找出合适的或最优的算法。

3. 编写程序

确定了解决问题的算法之后，需要使用一种计算机程序设计语言编写程序。编写程序就是将设计的算法等价映射(转换)为计算机语言的程序，所编写的程序从逻辑上看是算法的一种表现形式。

4. 运行验证

编写的程序有时不能正确地满足解决实际问题的需求，还需要调试，即在计算机上运行并且排除潜在错误。必要时，还要使用测试数据对程序进行测试，验证程序的正确性。如果程序测试不充分，则有可能使得潜在的错误存在。除此之外，问题需求也可能随着时间的推移而变化，使得程序使用一段时间后还需要进行修改完善，这个过程称为维护。

1.4　算法及其表示

大型软件的开发一般采用软件工程的方法，经过分析、设计、编码、调试和运行等阶段。每个阶段都有明确的任务，后一阶段的工作是在前一阶段结果的基础上进行的。对于初学者而言，需要解决的问题和编写的程序一般比较简单，只要将问题分析清楚，找到解决问题的方法，编写程序不是一件困难的事情。

C 语言解题时，在程序中有两方面的描述，即数据描述和处理步骤(算法)描述，后者处理前者的数据。因此，编写 C 语言程序之前，应该将问题所涉及的数据、已知条件和问题的解决步骤分析清楚。以下通过两个简单的实例来说明这一点。

【例 1-1】 求任意两个数的和与平均值。

(1) 数据描述

问题中有 4 个数据，即两个任意数、任意数之和与它们的平均值。在程序中定义 4 个浮点型变量存储这些数据，如 float a,b,sum,average;。

(2) 处理步骤描述

第 1 步：输入两个任意数，存储在变量 a 和 b 中。

第 2 步：计算两个数之和与平均值，存储在变量 sum 和 average 中。

第 3 步：输出变量 sum 和 average 的值，即得到问题所要求的结果。

根据上述分析，编写程序如下：

```
#include<stdio.h>
void main()
{   float a, b, sum, average;                    /* 定义变量 */
    scanf("%f,%f", &a, &b);                      /* 输入两个实型数 */
    sum = a + b;                                 /* 求两个数的和 */
```

```
    average = (a + b) / 2;                          /* 求两个数的平均值 */
    printf("sum=%f,average=%f\n", sum, average);    /* 输出结果 */
}
```

【例 1-2】 某体育比赛中，有 10 个裁判为参赛选手打分，参赛选手最后得分的计算方法是：去掉一个最高分和一个最低分后其他分数的平均值。求参赛选手的最后得分。

(1) 数据描述

问题中的原始数据有 10 个，解题过程中需要计算最高分、最低分和最后得分。在程序中可定义一个数组 s 存储 10 个分数、三个浮点类型变量 max、min、score 分别用来存储最高分、最低分和最后得分，另外还需要若干辅助变量。

(2) 处理步骤描述

第 1 步：输入 10 个任意数，存储在数组 s 中。

第 2 步：计算 10 个数的最高分、最低分与它们的和，并存储在变量 max、min 和 sum 中。

第 3 步：从 sum 中减去 max 和 min 并且除以(10–2)(值为 8)求得最后得分，并将其存储在变量 score 中。

第 4 步：输出变量 score 的值，则得到问题所要求的结果。

对于第 1 步和第 2 步还需给出更详细的步骤，以便程序实现。根据上述分析，编写程序如下：

```
#include<stdio.h>
void main()
{   float s[10],max, min, sum, score; /* 定义变量 */
    int i;
    for(i=0;i<10;i++)                 /* 输入 10 个数 */
       scanf("%f", &s[i]);
    max = min = sum = s[0];           /* 求最高分、最低分及和值 */
    for(i=1;i<10;i++)
    {
        if(max<s[i]) max=s[i];
            if(min>s[i]) min=s[i];
                sum+=s[i];
    }
    score=(sum - max - min )/ 8;      /* 求最后得分 */
    printf("score=%.4f", score);      /* 输出结果 */
}
```

使用计算机解题时，解题步骤归纳如下：

(1) 了解题目要做什么。

(2) 明确处理的数据及数据特征，确定处理方案。程序处理的对象是数据，程序中如何存储数据？输入的数据有哪些？输出的结果有哪些？这些都应该明确并且做到心中有数；而选择适当的处理方案(数学模型)，对于设计处理步骤是至关重要的。

(3) 清楚地描述对数据的处理步骤——算法设计，以便得到预期的结果。算法设计是解决问题的核心，是程序的灵魂，是程序设计过程中的一个重要环节。

(4) 对处理步骤进一步细化——逐步求精算法，直到能够使用程序设计语言编写程序实现。算法设计和逐步求精算法是使用计算机解题过程中最为困难的一步，是唯一的有创造性的工作。

(5) 将算法转换为程序。将算法转换为程序是一项包含众多技巧的工作，需要完整的计算机程序设计语言与程序设计的知识。

(6) 运行程序，分析结果。通过对源程序进行编辑、编译和连接，得到可执行的程序，并且运行所得到的可执行程序。如果出现编译错误或运行结果不正确，则要修改源程序，重新进行编译、连接和运行，直到得到满意的结果。

对于一个复杂的问题，可以将其分解成若干个容易处理、可单独解决的子问题，然后分别加以解决。

1.4.1　算法的概念

算法是精确定义的一系列规则的集合，这些规则规定了解决特定问题的一系列操作，以便在有限的步骤内产生出问题的答案。【例1-1】和【例1-2】中的处理步骤就是一种用自然语言描述的算法。

【例 1-3】求两个正整数 m 和 n 的最大公约数(即同时能够整除 m 和 n 的最大正整数)。

欧几里得阐述了求两个数的最大公约数的过程——欧几里得算法。

第 1 步：以 n 除 m，并令 r 为所得余数(显然 n＞r≥0)。

第 2 步：若 r=0，算法结束，n 即为 m 和 n 的最大公约数。

第 3 步：置 m←n，n←r，返回第 1 步。

算法中的符号←表示将该符号右边变量或表达式的值送到左边的变量中。

算法中不仅各步骤间的顺序是重要的，在每步内的动作次序也同样重要。例如，上述算法的第 3 步中，“置 m←n，n←r”绝对不能写成“置 n←r，m←n”，因为这样做的结果是在变量 m、n 和 r 中存储了同一个值，即都变成了第 1 步除法运算的余数。

对于同一个问题，可以有不同的解题方法，即可以有不同的算法。如【例1-2】求参赛选手的最后得分问题，可以先将 10 个数排序，求中间 8 个数的平均值，还可以有其他方法。为了有效地解题，不但要求算法正确，还要考虑算法的质量，通常选择简单明了、结构清晰和计算高效的算法。

求解问题时，算法可以千变万化、简繁各异，但是算法必须具有以下特性。

- 有穷性：算法在执行了有限步骤之后结束，并且每一步都可以在有穷的时间内完成。
- 确定性：算法中每种操作必须有确切的含义，即无二义性。同时，无论如何算法都只有唯一的一条执行路径，即相同的输入只能得出相同的输出。
- 可行性：算法中描述的操作都可以通过已经实现的基本操作执行有限次数来实现。
- 输入：有零个或多个输入，即算法需要的必要信息。
- 输出：有一个或多个输出，输出的是与输入有某些特定关系的信息。没有输出的算法是无意义的。

1.4.2 算法的表示

算法的基本组成结构有三种，即顺序结构、分支结构和循环结构。或者说，任何一个算法，无论其多么简单或多么复杂，都可由这三种基本结构组合而成。

算法有多种表示方法，常用的表示方法有：

(1) 自然语言描述；

(2) 传统流程图；

(3) N–S 流程图；

(4) 伪代码。

1. 自然语言描述

【例1-2】和【例1-3】介绍的算法是用自然语言描述的。自然语言可以是中文、英文、其他民族语言或数学表达式等。用自然语言描述算法通俗易懂，其缺点是文字有可能冗长，不太严格，容易产生歧义，表达分支和循环结构不方便等。

2. 传统流程图

流程图是用约定的图框和流程线表示运算或操作流程的图示形式。其优点是直观形象、易于理解。美国国家标准学会 ANSI 规定的一些常用流程图符号如图 1-3 所示。

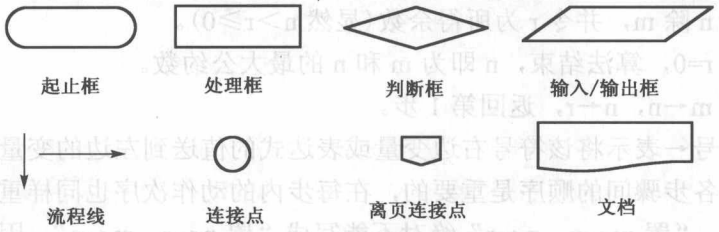


图 1-3 常用流程图符号

可以画出【例1-3】求最大公约数算法的传统流程如图 1-4 所示。

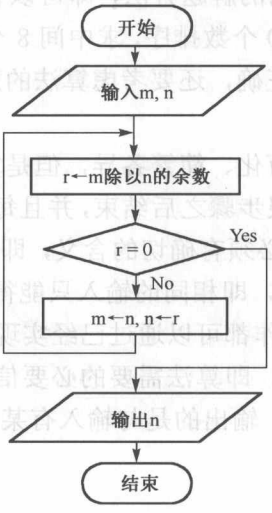


图 1-4 求最大公约数的传统流程图