普通高等教育“十二五”规划教材

普通高等教育“十一五”国家级规划教材

新编C语言程序设计教程

（第三版）

林碧英　主　编

王默玉　吴耀红　王素琴　副主编

中国电力出版社
CHINA ELECTRIC POWER PRESS

普通高等教育“十二五”规划教材

普通高等教育“十一五”国家级规划教材

新编C语言程序设计教程

（第三版）

主　编　林碧英
副主编　王默玉　吴耀红　王素琴
编　写　韩　霜　辜庭帅　王艳萍　朱　迪　杨　震
主　审　张基温　曲俊华

中国电力出版社
CHINA ELECTRIC POWER PRESS

内 容 提 要

本书为普通高等教育“十二五”规划教材、普通高等教育“十一五”国家级规划教材。

本书共分 12 章，主要内容包括程序设计概述，C 语言概述，数据类型、运算符和表达式，顺序结构程序设计，选择结构程序设计，循环结构程序设计，数组，指针，函数，结构体与共用体，位运算与编译预处理，文件。此外，本书各章的最后都附有丰富的习题，书后还有 4 个附录。本书内容在第二版的基础上做了一定的调整，调整后的内容重点更加突出，在重点章节增补了能够充分体现“计算思维”训练的系列例题，对长期困扰教师和学生的指针和函数参数传递问题，采用了通俗易懂的语言描述、形象逼真的图形展示，使复杂难懂的问题变得简单易学。

本书可作为普通高等院校计算机及相关专业的教材，也可作为国家计算机等级考试二级 C 语言的教学参考书。

图书在版编目（CIP）数据

新编 C 语言程序设计教程 / 林碧英主编. —3 版. —北京：中国电力出版社，2014.9

普通高等教育“十二五”规划教材　普通高等教育“十一五”国家级规划教材

ISBN 978-7-5123-6396-0

Ⅰ. ①新…　Ⅱ. ①林…　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（2014）第 208311 号

中国电力出版社出版、发行

（北京市东城区北京站西街 19 号　100005　http://www.cepp.sgcc.com.cn）

北京丰源印刷厂印刷

各地新华书店经售

*

2006 年 2 月第一版

2014 年 9 月第三版　2014 年 9 月北京第六次印刷

787 毫米×1092 毫米　16 开本　25.75 印张　627 千字

定价 **48.00** 元

前　言

国家教委提出，21 世纪的高等教育不仅要培养合格的人才，重要的是培养创新型人才。熟练掌握一门计算机程序设计语言，能够用计算机的编程技术解决实际问题，是创新型人才应该具备的能力。编者于 2005 年编写了《新编 C 语言程序设计教程》，在近 9 年的教学实践中，教材的使用效果良好，得到了教师和学生的一致好评，2009 年被评为华北电力大学教育教学成果一等奖。2008 年被列为普通高等教育“十一五”国家级规划教材，2010 年正式出版。

目前编者正在探讨如何将计算思维贯穿于计算机的程序设计语言教学。在此次的修订中，编者广泛听取了任课教师和学生的意见，再次对本书的内容进行了深入的分析研究，在重点章节增补了能够充分体现“计算思维”训练的系列例题，对长期困扰教师和学生的指针和函数参数传递问题，采用了通俗易懂的语言描述、形象逼真的图形展示，使复杂难懂的问题变得简单易学，使学生通过本书的学习，能够熟练掌握 C 语言的精华“指针”，全面提升学生的程序设计能力。各章的最后都附有丰富的习题，涵盖各种题型，知识点覆盖全面，从不同的角度测试学生掌握的程度。本书的内容与国家计算机等级二级 C 语言的考试要求紧密相关，适合各个层次的 C 语言程序设计教学，是从事 C 语言程序开发人员的一本实用参考书。

此次修订主要做了如下补充调整：

（1）详细阐述了标准输入函数何时从键盘输入数据，如何从缓冲区中读取数据，留在缓冲区的数据对后续输入函数会产生什么影响。重点分析了“%d”、“%c”、“%s”等常用格式控制符在读取数据时的规则，使学生从根本上理解并正确掌握输入函数的使用。

（2）以一维数组为存储结构，介绍了系列经典算法，以此促进学生“计算思维”和“创新思维”的协同式发展。

（3）以 VC++的编译环境为例，介绍了结构体和共用体变量在存储空间分配时遵守的字节对齐规则。

本书由华北电力大学（北京）教师编写。林碧英担任主编，王默玉、吴耀红、王素琴担任副主编。其中第 1 章~第 3 章由王素琴老师编写，第 4 章~第 6 章、第 10 章由王默玉老师编写，第 7 章~第 9 章由吴耀红老师编写，第 11 章、第 12 章由林碧英老师编写。全书由林碧英老师修改定稿。韩霜、辜庭帅、王艳萍、朱迪和杨震参与了习题的编写与程序调试。

本书由江南大学张基温、华北电力大学（北京）曲俊华担任主审。同时，本书在编写过程中引用、借鉴了相关专家的教材、著作。在此一并致谢。

由于编写时间比较仓促，书中难免存在不足之处，欢迎广大读者批评指正。

《新编 C 语言程序设计教程》编写组

2014 年 7 月

第二版前言

教育部提出，21 世纪的高等教育不仅要培养合格的人才，重要的是培养创新型人才。熟练掌握一门计算机语言，能够将计算机技术渗透于所学专业，解决实际问题，是创新型人才必须具备的能力。现在各个高等院校尤其是工科院校几乎所有的专业都开设了 C 语言程序设计这门课。为了使教师有一本较好的教材，学生有一本易学易懂的课本，软件开发人员有一本比较全面的参考书，我们于 2005 年编写了《新编 C 语言程序设计教程》。在近 6 年的教学实践中，该教材的使用效果非常好，得到了教师和学生的一致好评，2008 年被列为“十一五”国家级规划教材出版计划，2009 年被评为华北电力大学教学教育成果一等奖。

根据目前各个学校学时压缩，时间紧、任务重的特点，此次修订无论从内容的选择，还是问题的引入，以及问题的解决方法，更进一步地体现了由浅入深、循序渐进、通俗易懂的原则，各个章节都配有大量的例题和详细的分析过程、运行结果，很多例题还给出了多种算法设计，使学生全面掌握语法结构和程序设计的基本方法。各章的最后都附有各种题型的习题，知识点覆盖全面，从不同的角度测试学生掌握的程度，与国家计算机等级考试二级 C 语言要求紧密相关，适合各个层次的 C 语言程序设计教学。

此次修订主要对第一版做了如下调整：

（1）补充了输入函数 scanf()的读取数据的分析，使学生真正掌握函数 scanf()从缓冲区如何得到正确的数据，剩余数据对后续输入函数产生的影响。

（2）强调二维数组与一维数组的关系，在函数的参数传递中专门介绍了在函数之间将二维数组当作一维数组传递的方法，解决了行指针做函数参数传递不通用灵活的弊病，大大降低了学生理解和掌握二维数组传递的难度。

（3）计算机各相关专业的核心课程“数据结构”，对结构体、链表有较高的要求，此次修订加大了该部分内容的阐述，以图文方式重点分析了带头结点和不带头结点单向链表的基本操作，并给出了学生信息管理系统的完整案例。

（4）首次提出用自定义类型 typedef 语句，将具有一定长度的数组结构定义为数组类型，用数组类型的变量实现函数之间的数组传递，真正实现形参和实参在调用时共用同一片存储空间，解决了用指针做函数参数的难点。

本书的内容共 12 章，其中第 1～3 章由王素琴老师编写，第 4～6 章和第 10 章由王默玉老师编写，第 7～9 章由吴耀红老师编写，第 11 章、第 12 章由林碧英老师编写。全书由林碧英老师主编。蒲艳和刘硕参与了习题的编写与程序调试。

由于编写时间比较仓促，书中难免还存在不足之处，欢迎广大读者批评指正。

《新编 C 语言程序设计教程》编写组

2011 年 6 月

目　录

第1章　程序设计概述

随着计算机科学与技术的迅速发展，计算机已经深入到人们工作、学习和生活的方方面面。时代的发展对人才的素质也提出了更高的要求，使用计算机语言进行程序设计来解决专业领域中的实际问题已经成为一项对大学生的基本要求。

对于初学者来讲，首先必须弄清楚什么是计算机语言、计算机语言的分类以及利用计算机语言进行程序设计来解决问题的方法和步骤等。

1.1　计算机语言及程序设计

1.1.1　计算机语言

人们使用什么方式同计算机进行交流呢？人与人之间的交流用的是双方都能理解的自然语言（汉语、英语等），同样，人和计算机交流也要用人和计算机都能够理解的语言，这就是计算机语言。可以说，计算机语言架起了人和计算机之间沟通的桥梁。

作为沟通工具的计算机语言应该兼顾人和计算机的特点。但实际上，目前的计算机语言更多的是根据计算机的特点而编制的（趋势是越来越向人的思维特点靠拢）。它没有自然语言那么丰富多彩，只是有限规则的集合，学习起来比较容易。但是，也正因为它是根据计算机的特点编制的，所以在人机交流过程中不能揣摩和意会，更多地表现出“说一不二”的特点，体现了“规则”的严谨。这使得人们在开始学习计算机语言时会有些不习惯，不过只要认识到计算机语言的特点，注意学习方法，把必需的严谨和恰当的灵活结合起来，一切都会变得得心应手。

1.1.2　计算机语言的发展

自计算机诞生以来，先后出现了数百种计算机语言，这直接推动了计算机的应用和普及。按照计算机语言的发展历史，它们大致可分为机器语言、汇编语言和高级语言三类。

1. 机器语言

机器语言是以二进制指令代码表示的指令集合，是计算机能直接识别和执行的语言。用机器语言编写的程序运行速度快，占用内存少，但缺点是面向机器，通用性差。用机器语言编程序不直观，生产效率低，容易出错。现在，除了计算机生产厂家的专业人员外，绝大多数程序员已经不再去学习机器语言了。

2. 汇编语言

为了克服机器语言难读、难编和易出错的缺点，人们使用与指令代码实际含义相近的英文缩写词、字母或数字等符号来取代指令代码，于是就产生了汇编语言。汇编语言是一种用助记符来表示指令的面向机器的计算机语言。

由于采用了助记符来编写程序，比用机器语言的二进制代码编程方便得多，简化了编程

过程。用汇编语言编写的程序称为源程序，计算机不能直接识别和执行，必须翻译成二进制形式的目标程序后才能执行，这一翻译工作由“汇编程序”来完成，翻译的过程称为“汇编”。

汇编语言主要用来开发系统软件和过程控制软件，其目标程序占用内存空间少，运行速度快，可实现一般语言难以实现的功能和性能要求。但汇编语言仍然是面向机器的语言，通用性差。

汇编语言和机器语言都依赖于具体的机器，统称为“低级语言”。

3. 高级语言

机器语言和汇编语言与人们习惯使用的自然语言差别很大，不易学习和掌握。人们一直在寻求与自然语言相近的通用易学的计算机语言，这就是高级语言。高级语言是完全符号化的语言，用类似自然语言的形式描述问题的处理过程，用数学表达式的形式描述数据的计算过程，易于被人们理解和掌握。常见的面向过程的高级语言有 Fortran、Pascal 和 C 等，这些语言只要求人们向计算机描述问题的求解过程，而不必了解计算机的内部结构。常见的面向对象的高级语言有 C++、Java 等，对这些语言的讲述超出了本书的范围，不再赘述。

用高级语言编写的源程序不能直接被计算机识别，必须将它翻译成二进制的目标程序后才能执行。翻译方式一般分为编译和解释两种。

（1）编译方式。它是指将源程序一次性地翻译成目标程序，然后再执行该目标程序。完成翻译工作的程序称为“编译程序”，翻译的过程称为“编译”。

（2）解释方式。它是指将源程序逐句地翻译，翻译一句执行一句。完成翻译工作的程序称为“解释程序”。

这两种翻译方式各有所长，编译方式执行速度快，解释方式执行速度慢，但占用内存少。

随着计算机技术的迅猛发展，自 20 世纪 80 年代以来，出现了众多的面向对象的语言和面向应用的语言（第四代语言）。但是，“面向过程”仍然是所有程序设计的基础。作为初学者，首先应该掌握一门面向过程的语言。在本书中，将以面向过程的 C 语言为例，详细介绍程序设计的基本概念和方法。

1.1.3 程序与程序设计

在日常工作中，人们可以使用计算机进行数学运算、编写文档、绘制表格和收发邮件等。也就是说，尽管计算机本身只是一种批量生产出来的通用电子产品，但是，在计算机上运行不同的程序，就能完成不同的任务。今天，计算机之所以能够产生如此大的影响，主要在于人们开发出了不计其数的能够指挥计算机完成各种工作的程序，正是这些程序给了计算机无穷的生命力。

程序是用计算机语言编写的，因此，计算机语言通常又被称为“程序设计语言”。确切地说，所谓程序，就是用计算机语言对所要解决问题中的数据以及处理步骤做出的完整而准确的描述，而得到这个描述的过程就称为程序设计。对数据的描述就是指明数据结构形式，对处理步骤的描述就是下一节要讨论的算法问题。可见，数据结构与算法是程序设计过程中密切相关的两个方面。因此，著名的计算机科学家 Nikiklaus Wirth 给出了一个公式，即

数据结构+算法=程序

进行程序设计时，一般包含以下四个步骤：

（1）分析问题，建立数学模型。使用计算机解决具体问题时，首先要对问题进行充分的分析，确定已知条件是什么，要得到什么结果，以及解决问题的详细步骤。将解题过程归纳为一系列的数学表达式，建立各种量之间的关系。需要说明的是，有些问题的数学模型非常简单，以至于没有感觉到需要建立模型。但更多的问题需要进行认真的分析，构造出数学模型。模型的好与坏、对与错，在很大程度上决定了程序的正确性和复杂性。

（2）确定数据结构和算法。根据已得到的数学模型以及要求的输入输出数据，确定存放数据的数据结构，然后选择合适的算法。

（3）编制程序。根据已经确定的数据结构和算法，选择合适的程序设计语言，将解决方案正确地描述出来，形成程序。

（4）调试程序。编制程序不太可能一次成功，常常会出现各种各样的错误，需要经过认真分析后加以改正，直到程序能够正确运行为止。

1.1.4 结构化程序设计

长时间以来，人们一直认为程序是给机器执行的，只要程序没有错误，能够给出正确的结果就可以了。但随着计算机技术的迅速发展，程序的规模越来越大，人们发现读程序的时间可能比写程序的时间还要长。不仅程序员需要读程序，测试人员和维护人员也需要读程序。如果程序晦涩难懂，就会给测试和维护带来极大的困难。因此，衡量程序的质量不仅要看它的逻辑是否正确，性能是否满足要求，还要看它是否容易阅读和理解。

结构化程序设计是实现程序清晰易懂的关键技术。其基本思想是规定几种基本控制结构，然后由这些基本控制结构按一定规律组成程序，如同使用一些基本构件搭建房屋一样。

结构化程序设计的概念最早由 E.W.Dijkstra 于 1965 年提出。1966 年 Bohra 和 Jacopini 进一步证明了，只用三种基本控制结构就能实现任何单入口、单出口的程序。这三种基本控制结构分别是顺序结构、选择结构和循环结构。

1. 顺序结构

顺序结构表示程序中的各个操作按照其出现的先后顺序来执行，如图 1.1 所示。图 1.1 中的 A 和 B 表示两个顺序执行的操作，即先执行 A，再执行 B。顺序结构是最简单的控制结构。

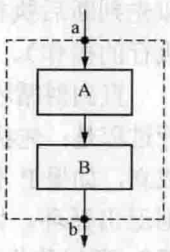


图 1.1 顺序结构

事实上，无论程序中包含什么结构，程序的总流程都是顺序结构。

2. 选择结构

选择结构也称为“分支结构”，如图 1.2 所示。在执行时，先判断给定的条件是否成立，再决定执行哪种操作。选择结构分为双分支的选择结构［见图 1.2（a）、（b）］和多分支选择结构［见图 1.2（c）］两种。

双分支的选择结构在执行时，首先判断条件 P 是否成立，如果成立，则执行 A 操作，否则，执行 B 操作。这里，条件 P 是否成立用英文字母来表示：“T”表示成立，“F”表示不成立。注意，无论条件 P 是否成立，只能执行 A 或 B 其中之一，执行后经出口点 b 离开，不可能既执行 A 又执行 B。A 和 B 两个操作中可以有一个为空，如图 1.2（b）所示。

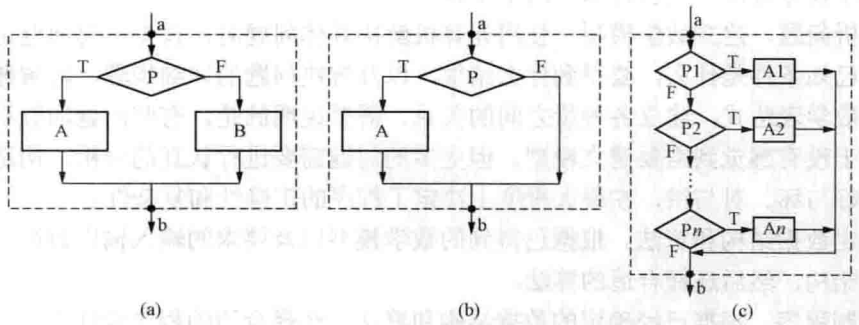


图 1.2 选择结构

（a）、（b）双分支选择结构；（c）多分支选择结构

多分支选择结构是指选择结构中有多种可能的情况，有多个可执行的分支，如图 1.2（c）所示。在执行时，首先判断 P1 是否成立，如果成立，则执行 A1 操作，否则继续判断 P2 是否成立，如果成立，则执行 A2 操作，否则继续判断 P3 是否成立，依次类推。不论执行了哪一个分支，执行后都直接到达出口 b 处，接着执行下面其他的语句，也就是说一次最多只能执行其中的一个分支。如果所有分支的条件都不满足，则直接到达出口 b 处。

3. 循环结构

循环结构又称“重复结构”，即反复执行某个或某些操作。循环结构分为当型循环结构和直到型循环结构两种。

当型循环结构如图 1.3（a）所示。它的执行过程是：先判断条件 P 是否成立，如果成立则执行操作 S，执行 S 后，再判断 P 是否成立，如果仍然成立，再执行 S，如此反复执行操作 S，直到条件 P 不成立为止，退出循环结构。因为是“当条件成立时执行循环”，即先判断后执行，所以称为当型循环。其中，P 称为循环条件，S 称为循环体（一组重复执行的操作）。

直到型循环结构如图 1.3（b）所示。它的执行过程是：先执行操作 S，然后判断条件 P 是否成立，如果 P 不成立则再执行 S，如果 P 成立，则退出循环。因为是“直到条件成立时停止循环”，所以称为直到型循环。同当型循环一样，P 称为循环条件，S 称为循环体。

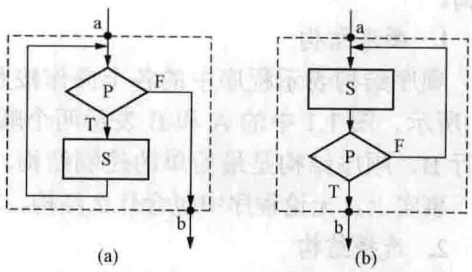


图 1.3 循环结构

（a）当型循环；（b）直到型循环

在使用循环结构时，最主要的是确定什么情况下执行循环（即循环条件），哪些操作需要重复执行（即循环体）。

以上介绍的这些基本控制结构可以顺序组合，也可以嵌套（一个基本控制结构中包含另一个或多个基本控制结构）。一个程序无论功能多么强大、逻辑多么复杂，都可以用这三种基本控制结构嵌套组合而成。

依据结构化思想编写出来的程序称为“结构化程序”，整个程序具有清晰的线索，容易理解、容易维护、可靠性高。

1.2 算 法

1.2.1 算法的概念

无论做什么事情，都有一定的步骤，必须按步骤循序渐进地进行。例如大学的期末考试，考试开始前，监考教师先到考试办公室取试卷，然后到指定教室组织学生进行考试，考试结束后，监考教师收卷，经任课教师阅卷后给出成绩。这些步骤缺一不可，顺序错了还可能导致严重的后果。因此，在做任何事情之前，都必须考虑好实施的步骤，然后按部就班地进行。

广义地说，为解决一个问题而采取的方法和步骤，就称为算法。

当然，同一个问题，可以有多种解决方法或操作步骤。例如：从北京到大连去旅游，可以选择不同的交通工具，如火车、汽车或飞机等。乘坐不同的交通工具，所花费的时间和金钱是大不相同的。人们可以根据自己的实际情况选择最合适的交通工具。再如，求1+2+3+…+100，可以先算1+2，再加3，再加4，一直加到100，一共要做99次加法运算，得到结果5050；也可以用等差数列的求和公式（1+100）×100/2，同样得到结果5050。从中可以看出，同一个问题，可以有多种算法，而算法不同，效率也不同。因此，在确定一个问题的算法时，不仅要保证算法的正确性，还要考虑算法的优劣。

作为一本C语言的教材，本书涉及的算法都是计算机算法。所谓计算机算法，是指用计算机解决一个实际问题的方法和步骤。计算机算法分为数值运算算法和非数值运算算法两大类。数值运算的目的是求解数值，如求方程的根、图形的面积等；非数值运算包含的范围很广，如人员管理、网上购物等。最初，计算机主要应用在数值运算领域，人们对数值运算算法的研究非常深入，各种数值运算都有比较成熟的算法可供选用。后来，计算机更多地用于解决非数值运算问题，本书只介绍一些简单的非数值运算算法，至于其他的非数值运算问题，读者可以参考相关书籍或者自己来设计解决问题的算法。

1.2.2 算法的特征

算法具有以下基本特征：

（1）有穷性：一个算法包含的操作步骤应该是有限的，而不能是无限的，要在执行有限个操作步骤后终止。

事实上，“有穷性” 往往指在合理的范围内，如果让计算机执行一个历时100年才结束的算法，这虽然是有穷的，但超过了合理的限度，人们也不把它视为有效的算法。

（2）确定性：算法中每个步骤的含义都必须是确定的，不能是模糊的、有歧义的。例如有这样一句话：“当i为0，j为0时加1”，通过这句话无法判断要给哪个变量加1，可能是i，也可能是j，或者是其他变量，因此无法执行。

（3）有效性：算法中的每个操作都应该能够有效地执行，并得到确定的结果。一个不可执行的操作是无效的。例如，被0除就是无效的，应当避免这种操作。

（4）有零个或多个输入：输入是指在执行算法时所需要的初始数据。输入数据的多少取决于特定的问题。例如，求$N!$就需要先输入N的值；求一元二次方程的根，需要输入三个

系数 a、b 和 c 的值；求前 100 个自然数的和则不需要输入数据。

（5）有一个或多个输出：算法的目的就是要得到问题的“解”，“解”就是输出。在一个完整的算法中至少应有一个输出。例如，求 $N!$，要输出 $N!$ 的值；求一元二次方程的根，要输出该方程的根；求前 100 个自然数的和，要输出求和的结果。没有输出的算法是没有意义的。

1.2.3 算法的描述

在分析、设计算法的过程中，需要将算法表达出来。表达算法的工具很多，常用的有自然语言、程序流程图、N-S 图和伪代码等。

1. 用自然语言表示算法

自然语言就是人们日常使用的语言，如汉语、英语等。用自然语言描述算法具有通俗易懂的优点。[例 1.1] 就是用自然语言描述了求前 10 个自然数之和的算法。

【例 1.1】 求 $\sum_{i=1}^{10} i$，即 1+2+3+4+5+6+7+8+9+10 的值。

可以用最直接的方法进行计算。

步骤 1 先求 1+2，得到结果 3。

步骤 2 将步骤 1 得到的和加上 3，得到结果 6。

步骤 3 将 6 再加上 4，得 10。

……

步骤 8 将上步得到的结果 36 再加上 9，得 45。

步骤 9 将 45 再加上 10，得 55。

这样的算法虽然正确，但太繁琐了。如果要计算 $\sum_{i=1}^{100} i$，则要写 99 个步骤，显然是不可取的。而且每次都直接使用上一步的计算结果，不太方便。应该找一个更通用的表示方法。

可以设两个变量，代表两个加数。直接将每一步求得的和放在一个加数变量中。现设 *sum* 和 *i* 分别代表两个加数，每一步求得的和也存放在 *sum* 中。用循环算法来求结果。改进的算法如下：

步骤 1 0→*sum*（含义是使 *sum* 的值为 0）。

步骤 2 1→i （含义是使 *i* 的值为 1，依次类推，以后将不再注明）。

步骤 3 求 *sum*+*i* 的和，结果仍然放在变量 *sum* 中，可表示为 *sum*+*i* → *sum*。

步骤 4 *i*+1→*i*（含义是使 *i* 的值加 1）。

步骤 5 如果 *i*>10， 算法结束，否则返回重新执行步骤 3、步骤 4 和步骤 5。

可以看出，步骤 3、4、5 组成一个循环，在实现算法时，要反复多次执行这三个步骤，直到执行步骤 5 时，经过判断，*i* 已经超过规定的数值 10 而不返回步骤 3 为止，算法结束。此时变量 *sum* 的值就是所求的结果。

用这种方法表示的算法具有较强的通用性和灵活性。如果要计算 $\sum_{i=1}^{100} i$，只需要将步骤 5 中的 *i*>10 改成 *i*>100 即可。

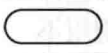
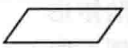
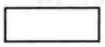
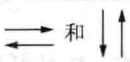
用自然语言描述的算法虽然通俗易懂，但用计算机语言实现该算法时常常要做较大改动，

而且算法描述文字冗长，含义不够清晰，容易产生歧义性（即可能被理解成多种含义）。例如，有这样一句话："张三对李四说他取得了好成绩"，通过这句话无法判断是张三取得了好成绩，还是李四取得了好成绩。因此，除了特别简单的问题，一般不用自然语言来表示算法。

2. 用程序流程图表示算法

程序流程图，又称为程序框图，是最早的用图形来表示算法的工具。它采用一组特定的图形、流程线以及文字说明来表示算法中的基本操作和控制流程，具有形象直观、简单易懂等特点。美国国家标准化协会 ANSI（American National Standard Institute）规定了一些常用的图形符号，这些符号已为世界各国的广大程序设计工作者普遍接受和采用，其中的一部分符号见表 1.1。

表 1.1　　程序流程图的常用图形符号

符号名称	符　号	功　能
起止框		表示算法的开始和结束
输入/输出框		表示算法的输入/输出操作，框内填写需输入或输出的各项
处理框		表示算法中的各种处理操作，框内填写处理说明或算式
判断框		表示算法中的条件判断操作，框内填写判断条件
注释框		表示算法中某操作的说明信息，框内填写文字说明
流程线	 和	表示算法的执行方向
连接点		表示流程图的延续

起止框：表示算法的开始或结束。每个程序流程图中都必须有而且只能有一个开始框和一个结束框，开始框只能有一个出口，没有入口，结束框只有一个入口，没有出口。

输入/输出框：表示算法的输入数据或输出结果，可以是一项或多项数据，多项之间用逗号分隔。输入/输出框只能有一个入口，一个出口。

处理框：表示程序中的各种操作，如计算、赋值等。处理框内填写处理说明或具体的算式。可在一个处理框内描述多个相关的处理。一个处理框只能有一个入口，一个出口。

流程线：用箭头表示，它指出各框的执行顺序。流程线箭头的方向就是算法执行的方向。事实上，流程线非常灵活，可以到达流程图的任意位置。但没有限制的灵活就是随意，程序设计的随意性一定要杜绝，它会使软件的可读性、可维护性大大降低。所以使用程序流程图表示算法时，流程线不能随意地跳转，必须遵守结构化程序设计的要求。

判断框：表示对一个给定的条件进行判断，然后根据判断结果决定其后的操作。

注释框：注释框中包含的内容称为注释。注释是对整个算法或其中部分操作的解释说明。正确的注释有助于对程序的理解。

连接点（小圆圈）：用于将画在不同地方的流程线连接起来，避免流程线的交叉或过长，使流程图更清晰。

用程序流程图表示算法时，必须考虑结构化程序设计的要求。图 1.1~图 1.3 所示为程序

流程图中基本控制结构的表示方法。

【例 1.2】 将［例 1.1］中求 $\sum_{i=1}^{10} i$ 的算法用程序流程图表示，如图 1.4 所示。

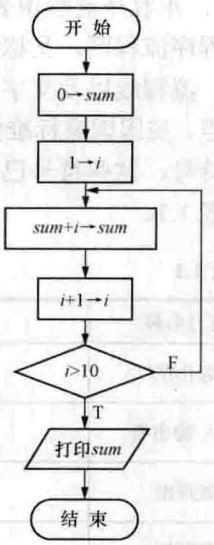


图 1.4 求 $\sum_{i=1}^{10} i$ 的程序流程图

用流程图表示算法形象直观、易于理解。但流程图也存在严重的缺点，它所使用的符号不够规范，常常使用一些习惯性画法；对流程线的使用没有严格的限制，流程线可以不受约束，随意转移控制。这样一来，用程序流程图描述的算法就不一定能满足结构化程序设计的要求。

3. 用 N-S 图表示算法

既然用基本控制结构可以表示任何复杂的算法，那么控制结构之间的流程线就属多余了。1973 年，Nassi 和 Shneiderman 提出了一种结构化的流程图——N-S 图，完全去掉了带箭头的流程线。它的基本单元是矩形框，程序的三种基本控制结构分别用不同的矩形框表示。在矩形框内还可以包含其他从属于它的矩形框，即 N-S 图是由矩形框组合嵌套而成，因此又称为盒图。图 1.5 所示为 N-S 图中基本控制结构的画法。

图 1.5（a）所示为顺序结构，先执行 A，再执行 B。

图 1.5（b）所示为双分支的选择结构：首先判断条件 P 是否成立，如果成立，则执行 A 操作，否则，执行 B 操作。若是空操作，则用箭头“↓”表示。

图 1.5（c）和图 1.5（d）所示为两种类型的循环结构，P 是循环条件，S 是循环体。其中，图 1.5（c）所示为当型循环结构，当条件 P 成立时，反复执行 S 操作，直到条件 P 不成立为止；图 1.5（d）所示为直到型循环结构，先执行 S 操作，然后判断条件 P 是否成立，如果不成立则再执行 S 操作，执行后再去判断条件 P 是否成立，……如此反复，直到条件 P 成立为止。

图 1.5（e）表示多分支选择结构：在执行时，首先判断 P1 是否成立，如果成立，则执行 A1 操作，否则继续判断 P2 是否成立，如果成立，则执行 A2 操作，否则继续判断 P3 是否成立，以此类推。

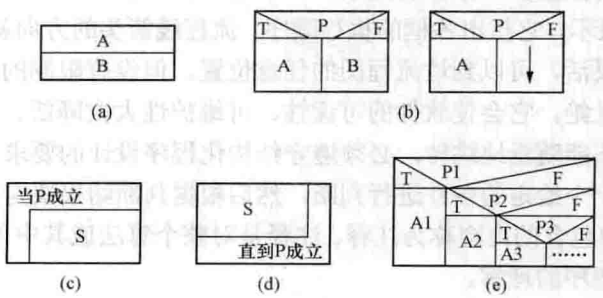


图 1.5 N-S 图的基本控制结构

（a）顺序结构；（b）双分支选择结构；（c）当型循环结构；（d）直到型循环结构；（e）多分支选择结构