丛书主编　陈松乔

高等学校计算机专业规划教材

第3版

# C语言程序设计

主编　成奋华　陆惠民

中南大学出版社
www.csupress.com.cn

# C语言程序设计

主　编　成奋华　陆惠民
副主编　刘　钢　向南平　唐　强
编　委　张　敏　黄　婕　张敏波　莫雅波
　　仇振中　秦戈亮　谢汀芬

中南大学出版社

**图书在版编目(CIP)数据**

C 语言程序设计/成奋华,陆惠民主编. —长沙:中南大学出版社,2005.6

ISBN 978-7-81105-098-1

Ⅰ.C... Ⅱ.①成...②陆... Ⅲ.C 语言—程序设计

Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2005)第 047299 号

**C 语言程序设计**

主编 成奋华 陆惠民

□责任编辑 刘 辉

□责任印制 文桂武

□出版发行 中南大学出版社

社址:长沙市麓山南路 邮编:410083

发行科电话:0731-88876770 传真:0731-88710482

□印 装 湖南大学印刷厂

□开 本 787×1092 1/16 □印张 23.75 □字数 557 千字

□版 次 2009 年 9 月第 3 版 □2009 年 9 月第 1 次印刷

□书 号 ISBN 978-7-81105-098-1

□定 价 48.00 元

# 前　言

C 语言具有语言简洁紧凑，运算符、表达式类型及数据类型丰富，表达能力强，使用方便灵活，支持结构化程序设计，可移植性好，目标代码运行效率高，兼容高，低级语言功能等众多优点，是目前世界上最受欢迎、应用最广泛的程序设计语言之一，成为目前几乎所有计算机专业学生学习计算机语言的入门课程，很多高等学校都把 C 语言作为计算机语言的首选教学语言，并把它作为数据结构和各种操作算法的描述工具。因此，计算机专业人员大多数都懂 C 语言，但没有几个人敢夸口说精通 C 语言，至少本书作者也不敢说。由此可看出 C 语言的通俗与深奥、精简与博大。

为便于计算机专业学生在有限的教学时间内尽快掌握 C 语言的主要内容，经高等学校计算机专业规划教材编委会策划，我们编写了本书。

本书编写的总原则是：根据专业人才培养规格的需要，突出职业素质教育和技术应用能力教育主线，强调理论与实践教学相结合，注重创新精神、综合素质、实践能力和可持续发展能力的培养。编写中力求“基础知识够用，注重应用能力”。各章主体内容均以需求实例为引导，首先介绍相关的基本知识和基本规则，重点介绍其应用方法与注意事项，继而解决需求实例(书中完整的例题都在 Turbo C 2.0 和 Visual C++ 6.0 环境下调试通过，便于读者直接上机验证)，最后辅以上机实训指导，使基本概念介绍、应用方法学习、上机实训操作一气呵成，即学即用，帮助初学者一步一个脚印地踏实前进。

本书参照当前最新的 2008 年版全国计算机等级考试二级 C 语言程序大纲安排章节，内容涵盖该大纲的要求并有较大延伸。编写时注意对该大纲要求的内容从概念到应用均作重点介绍，以期望对初学者参加全国计算机等级考试有较大帮助。

本书内容可分为四部分：第 1 章到第 5 章为第一部分，主要包括程序设计基础知识，C 语言的特点、开发过程与开发环境，C 语言源程序的组成，C 语言的数据类型、运算符与表达式、基本语句和控制结构等，属于基本概念部分，以使读者初步掌握用 C 语言进行程序设计所需的基本知识；第 6 章到第 10 章为第二部分，主要包括 C 语言的数组应用、函数应用、指针应用、构造数据类型应用与文件操作等，属于应用基础部分，以使读者初步掌握用 C 语言进行程序设计的基本方法和常用算法，进而具有用 C 语言解决实际问题的初步能力；第 11 章为第三部分，简要介绍 C++ 基础，以使读者对面向对象这一当前软件开发的主流方法有个初步印象；附录为第四部分，供读者在编制 C 语言程序解决实际问题时参考。

本书内容安排由简到繁、循序渐进、深入浅出、强调实用，既考虑到系统性、严密性、先进性，又兼顾目前学生的整体水平和初学者的接受能力。因此，本书不仅可作为高等学校计算机专业的规划教材，也可作为计算机专业的成人教育、自学考试和培训辅导教材，还可作为高职高专学校计算机教学参考书及有关人员的自学参考书。

虽然本书的完整例题都在 Turbo C 2.0 和 Visual C++ 6.0 环境下调试通过，但随着 Windows 环境下的 C++ 及 C#的迅速崛起，面向对象与可视化编程已成为当前软件开发的主流，

DOS 环境下的 Turbo C 已较少使用，即使学习 C 语言的人目前也多在 C++ 环境中上机调试运行。为此，编者推荐读者在 Visual C++ 6.0 开发环境下学习 C 语言。在一些 Turbo C 2.0 和 Visual C++ 6.0 两种环境下差别较大的地方，书中给出了简要说明，以便读者尽快熟悉 Visual C++ 6.0 环境，有利于读者在后续学习中向面向对象与可视化编程方向发展。

本书的参考教学时间是 72 课时，其中上机 18 课时。如果作为高职高专和培训辅导教材，教学时可酌情对一些章节进行删减。

参加本书编写的都是长期工作在教学一线、具有丰富教学经验、熟悉高等学校教学模式的教师。其中，陆惠民教授承担了大纲审定和第 1 章、第 2 章、第 7 章编写；成奋华老师承担了全书统稿和第 10 章、第 11 章编写；刘钢老师编写了第 8 章；龙玉辉老师编写了第 9 章；谭林海老师编写了第 4 章；刘桂林老师编写了第 5 章；高登老师参编了第 7 章；周艳香老师编写了第 6 章；彭铁光老师编写了实训内容；赵敏之老师编写了第 3 章。

在本书的编写出版发行过程中，相关单位的许多领导和同仁给予了大量关心、指导和帮助。在此，谨向为本书付梓提供了大量帮助和便利的主编院校湖南涉外经济学院、湖南科技职业学院，参编院校湘潭职业技术学院、湖南工业职业技术学院、长沙航空职业技术学院、湖南商务职业技术学院、长沙民政职业技术学院、湖南软件职业技术学院、湖南外国语职业学院、湖南对外经济贸易职业学院、南方职业技术学院，新加坡国立大学博士后叶施仁等表示衷心的感谢和诚挚的祝福。

虽然我们力求准确无误、尽善尽美，但由于时间仓促和水平有限，加之计算机技术的飞速发展，书中的内容取舍难免顾此失彼，也肯定有错误或不妥之处，恳请专家和读者批评指正。您的使用是对我们工作最好的鼓励，您的批评是对我们工作最好的帮助。联系方式（E-mail 地址）：cfh897@163.com

**编　者**

**2009 年 6 月**

# 目　录

# 第 1 章 C 语言程序设计概述

## 教学目标

- 了解程序、程序设计、程序设计语言的概念及程序设计语言的分类；
- 了解算法的概念、特性与基本要求，掌握用流程图和伪码描述算法的方法；
- 理解程序的 3 种基本结构；
- 了解 C 语言的特点，掌握 C 语言程序的基本构成和书写规则；
- 掌握 C 语言的字符集、标识符与关键字；
- 掌握 C 语言程序的开发过程与 Turbo C 2.0 和 MS Visual C++ 6.0 两种常用开发环境下开发 C 语言程序的方法。

电子数字计算机系统(简称计算机系统)是一种能自动、高速、精确地进行信息处理的现代工具。与人类历史上拥有的其他工具相比，计算机系统具有运算速度快、记忆能力强、有逻辑判断能力、能自动进行计算等基本特点，这些特点根源于计算机系统的组成。

计算机系统的组成包括两大部分：硬件和软件。计算机硬件可以定义为“组成计算机系统的各种物理元器件”；软件则可以定义为“计算机程序及说明程序的各种文档”。软件与硬件一起构成完整的计算机系统，硬件是载体，软件是灵魂，它们相互依存、缺一不可。

程序是关于计算任务的处理对象和处理规则的描述；文档则是关于计算机程序功能、设计、编制、使用的各种资料的集合。

要让无生命的计算机系统运行起来，为人类完成各种工作，就必须让它执行相应的程序。这些程序都是采用程序设计语言结合相应的数据结构编制而成，且预先存储在计算机的存储器中，是使计算机完成任务的关键。

# 1.1 程序设计基础知识

## 1.1.1 程序与程序设计语言

要让计算机按人的意图处理事务，人们必须预先编制好相应的程序并预先将它存放在计算机存储器中。计算机的工作过程就是自动、连续、有序地读取程序指令，按照指令执行相应操作的过程。

1. 程序、程序设计与程序设计语言的概念

程序是算法与数据结构的和谐结合，由有限的指令序列构成，它的作用是告诉计算机当前的处理对象与处理步骤，使计算机完成相应的任务。

程序设计是指人们借助程序设计语言，将某一任务的处理对象、处理方法及处理步骤转化为计算机能够处理的符号序列(语句、命令、指令)，以及指挥计算机完成相应任务的过程。

程序设计语言是建立在一定语法规则之上、用于向计算机传递程序设计人员思想的一批特殊符号的集合。它与人类自然语言的根本区别在于它能被计算机硬件识别并执行，或经相应的工具软件(编译程序、解释程序、汇编程序等)处理后能被计算机硬件识别并执行。自 20 世纪 40 年代电子计算机诞生至今，已有许多种程序设计语言面世，且发展非常迅速。新的程序设计语言层出不穷，其功能越来越强大，人机界面越来越友好。

2. 程序设计语言的分类

按照其与计算机硬件联系的紧密程度或与人类自然语言的接近程度，程序设计语言可以大致分为低级语言(如机器语言、汇编语言)和高级语言两大类(如 Fortran 语言、Pascal 语言、BASIC 语言、C 语言等)。由于 C 语言兼有高级语言的成分和低级语言的功能，也有人将 C 语言称为中级语言。按照其发展阶段与编程特性，可以分为面向机器的语言、面向过程的语言、面向任务的语言及当今最为流行的面向对象的语言等几大类。

(1) 面向机器的程序设计语言：该类语言与硬件密切关联，执行速度快，占用空间少，可以直接操作硬件以完成高级语言无法完成的特殊功能，适合于编制对程序执行时间要求高的软件系统(如实时处理系统)，属于低级语言范围。该类语言主要有：①机器语言[全部用 0 和 1 组成的符号序列表示的机器指令的集合(指令系统)是惟一能被硬件直接理解和执行的语言。用机器语言编制软件难写、难读、易出错、不通用]；②汇编语言(采用助记符表示机器指令程序设计语言，其指令与机器指令一一对应，但并不能被硬件直接理解和执行。用汇编语言源程序比机器语言程序较为方便书写、识别和记忆，是迄今为止许多实时处理系统的常用程序设计语言)。

(2) 面向过程的程序设计语言：又称算法语言，不依赖于机器硬件，形式上较为接近人类自然语言，属于高级语言范围。与低级语言比较，用该类语言编制的程序较为易写、易读、易记忆、可移植，程序设计人员编程时不需熟悉繁杂难记的计算机指令系统，可以把精力集中在解决问题的算法上，从而提高程序设计的效率和质量。该类语言的代表有 Fortran 语言、Pascal 语言、C 语言等。

(3) 面向任务的程序设计语言：该类语言以各种数据库管理系统(DBMS)为代表，具有算法语言的优点，而且是一种非过程化的语言，具有自动解决部分算法过程的能力，编程时不需详细描述问题的求解过程，而只需描述需要求解的问题即可。例如，在支持结构化查询语言(SQL)的DBMS中，从数据表“学生表”中查找“计软0501”班学生的姓名、年龄信息，使用SELECT语句作如下描述：

SELECT 姓名，年龄 FROM 学生表 WHERE 班级 = '计软0501'，即可得到所要信息，而具体怎样从数据表中筛选出'计软0501'学生及如何将所要的信息提取出来等具体算法过程均不必详细描述。

(4) 面向对象的程序设计语言：面向对象的方法把数据、操作封装在一起形成类，以对象和类间的抽象/细化为核心，通过派生、继承、重载和多态性等重要特色机制实现了较好的抽象性、封装性和共享性，极大地改善了软件的可维护性、可扩充性、可重用性，适合于开发大型复杂的软件，已成为当今程序设计的主流方法。目前该类语言较多，由C语言发展而来的C++以及C#、JAVA和VB.NET等是其代表。

### 1.1.2 程序算法描述

1. 算法的概念、特性与基本要求

算法(algorithm)是对特定问题求解方法与步骤的描述，是程序求解问题的核心内容。著名的瑞士计算机科学家、PASCAL语言发明者Niklaus Wirth教授曾提出程序定义的著名公式：程序 = 算法 + 数据结构。这个公式的重要性在于它说明了程序与算法的关系，同时也说明数据结构的选择亦十分重要。

算法由有限的指令(语句)序列构成，其中每条指令(语句)表示一个或多个操作。

算法有5个重要特性：

(1) 有穷性：对任何合法输入值，算法必须在有穷时间内、执行有穷步之后结束；

(2) 确定性：算法中的每一个步骤必须有确切含义，不产生二义性，任何条件下对相同的输入只能得出相同的输出；

(3) 可行性：算法中描述的操作都是可通过已实现的基本运算执行有限次实现，任何条件下都不允许出现算法不能进行的现象；

(4) 输入：输入指算法执行时从外界取得的信息。一个算法可有零个或多个输入；

(5) 输出：输出是指算法执行后得到的结果。一个算法可以没有输入，但必须有输出，无输出的算法是无意义的。

在程序设计语言中，与算法密切相关的便是语句，包括功能处理语句(如输入输出语句、赋值语句、调用语句等)和流程控制语句(如条件转移语句、循环语句等)。对程序设计而言，算法的确定也就是如何合理安排这些语句以完成特定功能。

一个好的算法应当达到以下要求：

(1) 正确性：算法应满足具体问题的需求，除了不含语法错误这一起码要求外，还应当对精心选择的典型、苛刻甚至带有刁难性的几组输入数据能够得出满足用户提出的规格说明要求的结果。正确性是对算法的最基本要求，不具备正确性的算法毫无意义。

(2) 可读性：算法主要是为了人的阅读与交流，其次才是机器执行。良好的可读性是根

据算法编制的软件具有较好的可维护性的前提，晦涩难懂的算法易于隐藏较多错误且难以调试和修改。

（3）健壮性：算法对来自程序外部的错误（如非法输入数据）和内部的缺陷（如未能考虑到的运算条件时）应能适当作出反应或处理，不能出现莫名其妙的结果或停机等现象。

（4）高效性：在保证正确性和可读性的前提下，算法应尽可能追求较少的时间与空间占用。

2. 简单算法举例

**例 1－1** 求解自然数前 10 项的代数和。

**算法 1.1.1** 采用最原始的方法求解：

第 1 步：求 0 +1，得结果 1；

第 2 步：将第 1 步的结果再加上 2，得结果 3；

第 3 步：将第 2 步的结果再加上 3，得结果 6；

第 4 步：将第 3 步的结果再加上 4，得结果 10；

⋮

第 10 步：将第 9 步的结果再加上 10，得最后结果 55；

第 11 步：输出最后结果 55。

算法 1.1.1 是正确的，但太繁琐，如果要求解自然数前 1000 项的代数和，则需要写 1001 个步骤，显然不可取，可以改用算法 1.1.2。

**算法 1.1.2** 采用循环的方法求解：设变量 add 表示加数（兼作循环控制变量），变量 sum 开始时存放第 1 个数 0，然后存放和：

第 1 步：sum = 0；

第 2 步：add = 1；

第 3 步：sum = sum + add；

第 4 步：add = add + 1；

第 5 步：如果 add 不大于 10，返回到第 3 步，重复执行第 3 步、第 4 步和第 5 步（循环）；否则，sum 中的值就是最后结果；

第 6 步：输出 sum。

相对于算法 1.1.1，算法 1.1.2 有了很大的进步，如果要求解自然数前 1000 项的代数和只要将第 5 步中的“add 不大于 10”修改成“add 不大于 1000”即可。

3. 算法的描述

算法考虑的是特定问题的求解方法与步骤，是在较高、较抽象层次上的解决问题的框架流程；编程则是根据算法给出的框架流程进行语言细化，实现问题求解的具体过程。因此，算法的描述不像编写程序那样有严格的、受限于具体语言的语法。可以使用下面几种工具描述算法。

（1）自然语言：即人们日常进行交流所使用的语言，如英语、汉语等。虽然用自然语言描述算法对用户之间的交流较为方便，但由于容易出现歧义性、不方便描述包含分支和循环结构、目前尚难于直接在计算机上实现等原因，所以一般不用自然语言描述算法。

（2）流程图：流程图是用一些特定的图形符号表示相应操作及其顺序的图表。表 1－1

列举了常用的一些流程图符号。本书的算法描述主要采用流程图。

**表1-1 常用流程图符号**

| 符号 | 名 称 | 说 明 |
|---|---|---|
|  | 开始框 | 程序或算法开始 |
|  | 结束框 | 程序或算法结束 |
|  | 输入/输出框 | 输入/输出数据，可注明数据名称、来源、用途或其他文字说明 |
|  | 处理框 | 能改变数据值或位置的各种处理，可注明处理名称或其简要功能 |
|  | 特定处理框 | 可注明特定处理名称或简要功能，表示已命名的处理 |
|  | 判断框 | 可注明判断条件。只有一个入口，有2个或2个以上可供选择的出口 |
|  | 流程线 | 连接其他符号，指明程序流向 |
|  | 连接点 | 转到图的另一部分或从图的另一部分转来，通常在同一页上 |
|  | 换页连接 | 转到另一页图上或由另一页图转来 |
|  | 文档 | 通常表示打印输出，也可表示用打印终端输入数据 |
|  | 磁盘 | 磁盘输入或输出，也可表示存储在磁盘上的文件或数据库 |

注：在流程图中人们习惯上用  表示开始框，本书也按照这种习惯。

**例1-2** 用流程图描述算法1.1.2，见图1-1。

一个流程图至少包括3部分内容：表示相应操作的框、流程线和框内外必要的文字说明。应特别注意的是流程线必须带箭头以指明流程图上各框所示操作执行的先后次序。

流程图的优点是直观清晰、易于使用，是开发者普遍采用的工具，但它有严重的缺点：①由于流程线的存在，设计时可随心所欲地画流程线的流向，容易造成非结构化的程序结构。编码时势必频繁使用goto语句，导致基本控制块多入口多出口，使流程图难于理解，继而影响软件质量；②不易反映逐步求解过程，往往反映的是最后结果；③不易表示数据结构。

为解决流程线导致程序流程随意流转问题，可以采用N-S图、PAD图等来描述算法流程。

(3) N-S流程图：简称N-S图，是由美国学者I. Nassi和B. Shneiderman提出的一种结构化流程图。N-S图没有流程线，而是针对程序的3种基本结构(见本章图1-2、图1-3、图1-4)，使用4种基本框图加以表示，全部算法均写在1个矩形框内。由于没有流程线，

N－S图从根本上杜绝了由流程线导致的非结构化程序流程，非常适合于结构化程序设计。

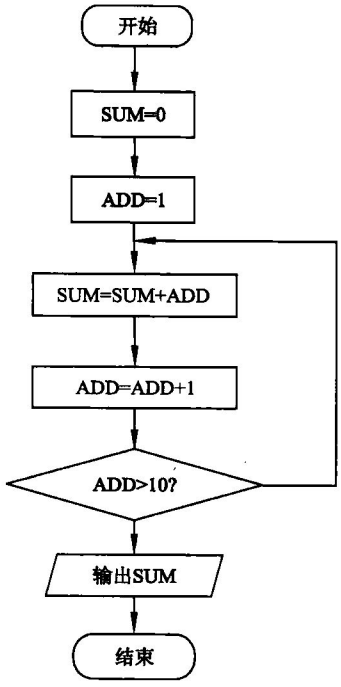



**图 1－1　算法 1.1.2 的流程图**

图 1－2、图 1－3、图 1－4 分别是 N－S 图的 4 种基本框图。其中图 1－2 中的“A”、“B”两个框组成“顺序结构”，意为先执行“A”操作，然后执行“B”操作；图 1－3 描述的是“选择结构”，意为如果条件“P”成立则执行“A”操作，否则执行“B”操作；图 1－4 描述的是“循环结构”，其中(a)图是“当循环”，意为当条件“P”成立时反复执行“A”操作，(b)图是“直到循环”，意为反复执行“A”操作，直到条件“P”成立为止。

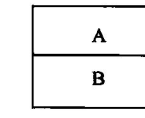



**图 1－2　N－S 图的顺序结构**

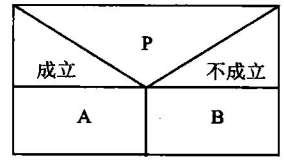



**图 1－3　N－S 图的选择结构**

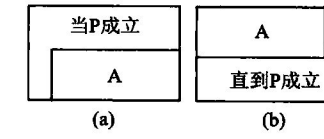



**图 1－4　N－S 图的循环结构**

**例 1－3**　用 N－S 图描述算法 1.1.2，见图 1－5。

（4）PAD：PAD（problem analysis diagram，问题分析图）是日立公司提出的一种算法描述工具，如图 1－6、图 1－7、图 1－8 所示。PAD 也采用几种基本图形表示 3 种基本程序结构（3 种基本程序结构的逻辑意义解释见本章 1.1.3 节）。

PAD 是一种由左往右展开的树型结构，其控制流程为自上而下、从左到右地执行。

图 1－9 为采用 PAD 描述的算法 1.1.2。从图 1－9 可看出 PAD 有以下优点：①清晰反映程序算法的层次结构：图中的竖线为程序的层次线，最左竖线是程序的主线，其后一层一层展开，层次关系一目了然；②支持逐步求精的设计方法：左边层次中的内容可以抽象，然后由左到右逐步细化[例如图 1－9(b)是对图 1－9(a)中“A”处理框的细化]，因而设计时可以先集中精力设计算法的主干部分，然后再逐步考虑各个细节部分；③易读易写，使用方便；④支持结构化程序设计；⑤可自动生成程序：PAD 图有对照 Fortran，Pascal，C 等高级语言的标准图式，在有 PAD 系统的计算机上(如日立公司的 M 系列机)，可以直接输入 PAD 图，由机器自动通过走树生成相应的源代码，大大提高软件的生产率。

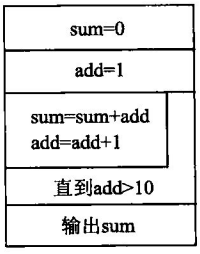



图 1－5　算法 1.1.2 的 N－S 图

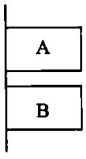
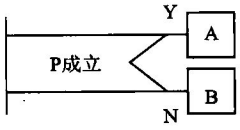
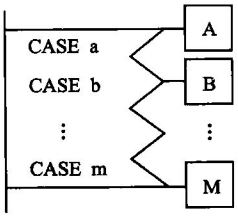



图 1－6　PAD 的顺序结构

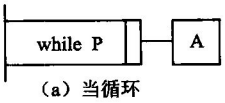



（a）简单选择

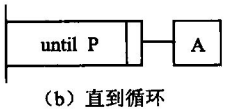



（b）多重选择

图 1－7　PAD 的选择结构




（a）当循环




（b）直到循环

图 1－8　PAD 的循环结构

**例 1－4**　用 PAD 描述算法 1.1.2，见图 1－9。

（5）伪代码：简称伪码，是介于自然语言和计算机语言之间的文字和符号。用伪码描述算法如同写文章，自上而下地写下来。每一行(或几行)表示一个基本操作。它书写方便、格式紧凑，也比较好理解。

伪码的结构一般分为内外两层，外层语法应符合一般程序设计语言常用的语法规则，内层语法则用一些简单的句子、短语和通用的数学符号描述程序应执行的功能。PDL（process design language，过程设计语言）就是一种用于描述模块算法设计和处理细节的伪码语言。PDL 具有严格的关键字外层语法，用于定义控制结构、数据结构和模块接口，而它表示实际

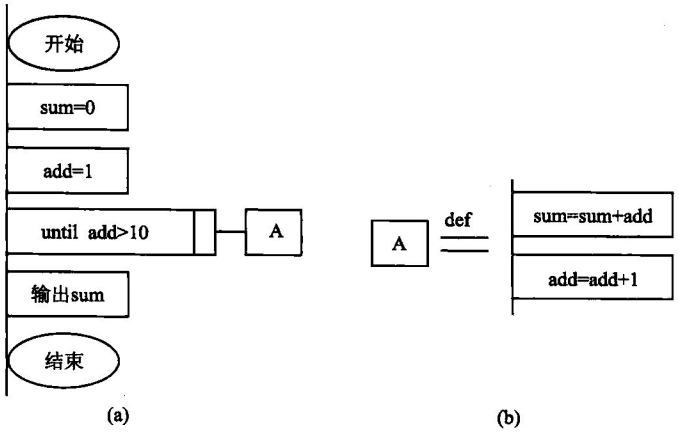



**图 1-9 算法 1.1.2 的 PAD 图**

操作和条件的内层语法又是灵活自由的，可使用自然语言的词汇。

**例 1-5** 用伪码描述算法 1.1.2。

```
开始
sum = 0
add = 1
while add < = 10
  sum = sum + add
  add = add + 1
endwhile
输出 sum 的值
结束
```

(6) 程序设计语言：算法也可以用程序设计语言描述。为了算法的可读性，一般不用低级语言而采用高级语言来描述算法。用程序设计语言描述算法必须严格遵循所用语言的语法规则，这是和伪码描述的根本区别。

**例 1-6** 用 C 语言描述算法 1.1.2，见程序 1-1。

**程序 1-1**

```
#include <stdio.h>
main()
{
  int sum, add;                    /* 定义变量 */
  sum = 0;                         /* 语句 1 */
  add = 1;                         /* 语句 2 */
  while (add < = 10)               /* 语句 3 */
    {sum = sum + add;              /* 语句 4 */
      add = add + 1;               /* 语句 5 */
```

```
    }
  printf("sum = %d\n", sum);    /* 语句6 */
}
```

程序运行结果：

```
sum = 55
```

### 1.1.3　程序的3种基本结构

如前所述，程序由有限的语句(指令)序列构成，这些语句在执行时有时是按顺序依次进行且不重复(如例1-6中的语句1、语句2)，有时要根据条件判断的结果，选择执行部分语句，或重复执行部分语句(如例1-6中的语句3、语句4、语句5)，从而呈现出不同的流程结构。1966年巴纳和杰克匹尼(Bohra and Jacopini)提出了程序的3种基本结构，用这3种基本结构可以组合出任何复杂的程序结构。

1. 顺序结构

顺序结构是一种最简单的基本结构，其特点是结构内的语句(指令、操作)均按其存储顺序依次执行。例1-6中的语句1和语句2构成一个顺序结构；图1-2中的“A”操作框和“B”操作框，图1-6中的“A”操作框和“B”操作框亦分别构成一个顺序结构，表示程序运行时先执行“A”操作，然后执行“B”操作。

2. 选择结构

选择结构又称为分支结构，其特点是结构内含有判断框，程序运行时根据给定的条件是否成立来选择执行结构内的部分语句(指令、操作)。图1-3、图1-7(a)均为简单选择结构的描述，表示程序运行时如果条件P成立则执行“A”操作，否则执行“B”操作；图1-7(b)描述的是多重选择结构，表示程序运行时如果条件运算结果为a，b，…，m之一时，选择执行“A”，“B”，…，“M”中的一个操作。

3. 循环结构

循环结构又称为重复结构，其特点是结构内含有判断框，程序运行时根据给定的条件是否成立来反复执行结构内的语句(指令、操作)。结构内的这些被反复执行的语句(指令、操作)构成循环结构的“循环体”。图1-4、图1-8描述的即是循环结构。其中(a)图是“当循环”，表示当条件P成立时反复执行A操作；(b)图是“直到循环”，表示反复执行A操作，直到条件P成立为止。“当循环”与“直到循环”的主要区别在于“当循环”是先判断后执行，而“直到循环”是先执行后判断。因此，“当循环”的循环体可能1次也不执行，而“直到循环”的循环体至少会执行1次。另外，典型“当循环”是当条件成立时执行循环，而典型直到循环是当条件成立时结束循环。

## 1.2　C语言的历史沿革及特点

### 1.2.1　C语言的产生与发展概况

C语言的出现与UNIX操作系统密切相关。1963年剑桥大学推出了CPL(combined