程序设计基础
（C 语 言）教程

主　编◎刘媛媛　雷莉霞　胡　平
副主编◎刘美香　甘　岚

西南交通大学出版社

程序设计基础（C 语言）教程

主　编　刘媛媛　　雷莉霞　　胡　平
副主编　刘美香　　甘　岚

西南交通大学出版社
·成　都·

图书在版编目（CIP）**数据**

程序设计基础（C 语言）教程 / 刘媛媛，雷莉霞，胡平主编. —成都：西南交通大学出版社，2022.12
ISBN 978-7-5643-9138-6

Ⅰ. ①程… Ⅱ. ①刘… ②雷… ③胡… Ⅲ. ①C 语言－程序设计 Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字（2022）第 255248 号

Chengxu Sheji Jichu (C Yuyan) Jiaocheng
程序设计基础（C 语言）教程

主编　刘媛媛　雷莉霞　胡　平

责任编辑	黄淑文
封面设计	原谋书装
出版发行	西南交通大学出版社 （四川省成都市金牛区二环路北一段 111 号 西南交通大学创新大厦 21 楼）
发行部电话	028-87600564　028-87600533
邮政编码	610031
网　　址	http://www.xnjdcbs.com
印　　刷	成都蜀通印务有限责任公司
成品尺寸	185 mm × 260 mm
印　　张	20.25
字　　数	503 千
版　　次	2022 年 12 月第 1 版
印　　次	2022 年 12 月第 1 次
书　　号	ISBN 978-7-5643-9138-6
定　　价	58.00 元

课件咨询电话：028-87600533
图书如有印装质量问题　本社负责退换

前　言

C 语言程序设计是一门基础课程，旨在培养学生具有设计计算机程序、编写程序和调试程序的能力。C 语言是一种通用的高级程序设计语言，同时又具有其他高级语言所不具备的低级语言功能，不但可用于编写应用程序，还可用于编写系统程序，具有运算符和数据类型丰富、生成目标代码质量高、程序执行效率高、可移植性好等特点，因而得到广泛的应用。同时，掌握了 C 语言，就可以较为轻松地学习其他任何一种程序设计语言，为后续的课程打下坚实的基础，能较好地训练学生解决问题的逻辑思维能力以及编程思路和技巧，使学生具有较强的利用 C 语言编写软件的能力，为培养学生有较强软件开发能力打下良好基础。

本书针对程序设计思想零基础的同学，以培养学生的计算机思维为目的，从模块化程序设计思想着手，以解决实际问题为课程目标，精心设计了趣味性和实用性较强的案例，由浅入深地介绍了每章所涉及的知识点。全书共分为 10 章，第 1 章通过一个简单的 C 语言程序的概述，介绍了程序的概念和程序设计的流程；第 2 章详细介绍了 C 语言的基础知识；第 3~5 章介绍了 C 程序设计的基本结构；第 6 章介绍了数组，包括一维数组、二维数组、字符数组与字符串；第 7 章介绍了指针；第 8 章介绍了模块化程序设计，包括函数和编译预处理；第 9 章介绍了构造型数据类型的使用；第 10 章介绍了文件系统。

本书可以作为高等院校 C 语言程序设计课程的教材，也可以作为读者自学 C 语言的自学用书。本书由刘媛媛、雷莉霞、胡平担任主编，刘美香、甘岚担任副主编。具体编写分工如下：第 1 章和第 3 章由甘岚编写，第 2 章、第 7 章以及附录由刘媛媛编写，第 6 章和 8 章由雷莉霞编写，第 4 章和 5 章由胡平编写，第 9 章和 10 章由刘美香编写。全书由刘媛媛负责最终统稿。在制订编写大纲及书稿编写过程中，华东交通大学信息工程学院自始至终给予了极大的关心和支持，计算机基础教学部的熊李艳、吴昊、丁振凡、宋岚、周美玲、李明翠、张月园给了作者大力帮助，在此表示由衷的感谢。

本书不仅有配套的实践教材，而且有电子教案、习题答案等教材中涉及的相关教学资源。

由于编者水平有限，编写时间仓促，书中难免有欠妥之处，恳请广大读者提出宝贵意见。

编　者

2022 年 12 月于南昌

目 录

第 1 章　C 语言程序设计概述

C 语言从诞生起，就成为主流的程序设计语言，近几年也一直稳居在编程语言排行榜的前列。C 语言是一种计算机程序设计语言，既具有高级语言的特点，又具有汇编语言的特点。因此它可以作为工作系统设计语言，编写系统应用程序，也可以作为应用程序设计语言，编写不依赖计算机硬件的应用程序。它的应用范围广泛，具备很强的数据处理能力，不仅仅是在软件开发上，而且各类科研都需要用到 C 语言。

本章从介绍程序设计语言的基本概念入手，着重介绍 C 语言的发展与特点、应用领域、C 程序的构成及其运行步骤与运行环境。

1.1　程序与程序设计语言

计算机主要由硬件和软件两大部分构成，硬件就不用解释了，主机、显示器等都属于硬件，但是计算机光有硬件是没有办法使用的，还必须有软件支持。软件又分为系统软件，也就是经常用的操作系统，如 Windows XP，Windows 7，Windows 11 等，以及通用软件和应用软件，如 Office 办公软件与 QQ 等，而软件的主体就是程序，因此程序设计语言是计算机科学技术中非常重要的一个部分。

1.1.1　程序设计语言的发展与分类

程序设计语言（Program Design Language，简称 PDL）又称编程语言，是一组用来定义计算机程序的语法规则。它是一种被标准化的交流技巧，用来向计算机发出指令。一种计算机语言让程序员能够准确地定义计算机所需要使用的数据，并精确地定义在不同情况下所应当采取的行动。

正如人们交流思想需要使用各种自然语言（如汉语、英语、法语等）一样，人与计算机之间交流信息必须使用人和计算机都能理解的程序设计语言。程序设计语言也叫计算机语言，是一套关键字和语法规则的集合，可用来产生由计算机进行处理和执行的指令。计算机语言也有一个发展过程，从最开始的计算机语言，也就是二进制机器语言如 011010111，那个时候程序员编程恐怕是非常痛苦的事，因为你要会用 0 和 1 表示一切。后来逐步发展，把一些常用的指令用英语单词表示出来，形成了汇编语言，这个时候也是比较痛苦的，你要记住那些单词的含义不说，还必须知道计算机硬件的组成，再告诉计算机每一步要怎么做，而计算机又是一个非常笨的东西，只要掉一个步骤它就罢工，由于每台机器的硬件组成不一定相同，所以汇编语言的可移植性差，也就是说你在这台计算机上写的程序到另一台计算机上可能就不能用了。之后为了方便软件移植，高级语言诞生了。高级语言不要求程序员掌握计算机的硬件运行，只要写好上层代码，编译软件会将高级语言翻译成汇编语言，然后再将汇编语言转化成计算机语言，从而在计算机中执行。因此，程序员使用高级语言写的代码可以移植到

其他计算机执行，而不用考虑计算机硬件的组成。

程序设计语言有很多种，常用的不过十多种，按照程序设计语言与计算机硬件的联系程度将其分为三类，即机器语言、汇编语言和高级语言。前两类依赖于计算机硬件，有时统称为低级语言，而高级语言与计算机硬件关系较小。

1. 机器语言

机器语言是用二进制代码表示的计算机能直接识别和执行的一种机器指令的集合。它是计算机的设计者通过计算机的硬件结构赋予计算机的操作功能。机器语言具有灵活、直接执行和速度快等特点。不同型号的计算机其机器语言是不相通的，按照一种计算机的机器指令编制的程序，不能在另一种计算机上执行。例如运行在 IBM PC 机上的机器语言程序不能在 51 单片机上运行。

机器指令由操作码和操作数组成，操作码指出要进行什么样的操作，操作数指出完成该操作的数或它在内存中的地址。

例如，计算 1+2 的机器语言程序如下：

```
10110000 00000001     ；将 1 存入寄存器 AL 中
00000100 00000010     ；将 2 与寄存器 AL 中的值相加，结果放在寄存器 AL 中
11110100              ；停机
```

由此可见，用机器语言编写程序，编程人员必须熟记所用计算机的全部指令代码和代码的含义。编写程序时，程序员必须自己处理每条指令和每一数据的存储分配和输入输出，还得记住编程过程中每步所使用的工作单元处在何种状态。这是一件十分烦琐的工作，编写程序花费的时间往往是实际运行时间的几十倍或几百倍。而且，编出的程序全是些 0 和 1 的指令代码，直观性差，难以记忆，还容易出错。

2. 汇编语言

为了克服机器语言的缺点，人们采用了有助于记忆的符号（称为指令助记符）与符号地址来代替机器指令中的操作码和操作数。指令助记符是一些有意义的英文单词的缩写和符号，如用 ADD（Addition）表示加法，用 SUB（Subtract）表示减法，用 MOV（Move）表示数据的传送等等。而操作数可以直接用十进制数书写，地址码可以用寄存器名、存储单元的符号地址等表示。这种表示计算机指令的语言称为汇编语言。

例如上述计算 1+2 的汇编语言程序如下：

```
MOV AL,1              ；将 1 存入寄存器 AL 中
ADD AL,2              ；将 2 与寄存器 AL 中的值相加，结果放在寄存器 AL 中
HLT                   ；停机
```

由此可见，汇编语言克服了机器语言难读难改的缺点，同时保持了占存储空间小、执行速度快的优点，因此许多系统软件的核心部分仍采用汇编语言编制。但是，汇编语言仍是一种面向机器的语言，每条汇编命令都一一对应于机器指令，而不同的计算机的指令长度、寻址方式、寄存器数目等都不一样，这使得汇编语言通用性差，可读性也差。

3. 高级语言

所谓高级语言就是更接近自然语言、更接近数学语言的程序设计语言。它是面向应用的

计算机语言，与具体的机器无关，其优点是符合人类叙述问题的习惯，而且简单易学。高级语言与计算机的硬件结构及指令系统无关，它有更强的表达能力，可以方便地表示数据的运算和程序的控制结构，能更好地描述各种算法，而且容易学习和掌握。但用高级语言编译生成的程序代码一般比用汇编程序语言设计的程序代码要长，执行的速度也慢。

高级语言并不是特指的某一种具体的语言，而是包括很多编程语言，如目前流行的 Java，C，C++，C#，PASCAL，PYTHON，LISP，PROLOG，FoxPro 等，这些语言的语法、命令格式都不相同。

例如上述计算 1+2 的 BASIC 语言程序如下：

```
A=1+2                ；将 1 加 2 的结果存入变量 A 中
PRINT A              ；输出 A 的值
END                  ；程序结束
```

这个程序和我们平时的数学思维是相似的，非常直观易懂且容易记忆。

1.1.2　程序设计方法

1. 程序设计过程

计算机程序设计的过程包括问题定义、算法设计、程序设计以及调试运行。整个开发过程都要编制相应的文档，以便管理。

（1）问题定义。在计算机能够理解一些抽象的名词并做出一些智能的反应之前，必须要对交给计算机的任务做出定义，并最终翻译成计算机能识别的语言。问题定义的方法很多（对此在软件工程的需求分析中会有更多解释，包括描述方法和工具），但一般包括三个部分：输入、输出和处理。

（2）算法设计。问题定义确定了未来程序的输入、输出、处理，但并没有具体说明处理的步骤，而算法则是对解决问题步骤的描述。

（3）程序设计。问题定义和算法设计已经为程序设计规划好了蓝本，下一步就是用真正的计算机语言表达了。不同的语言写出的程序有时会有较大的差别。

（4）调试运行。程序编写可以在计算机上进行，也可以在纸张上进行，但最终要让计算机来运行则必须输入到计算机中，并经过调试，以便找出错误，然后才能正确地运行。

（5）文档。对于微小的程序来说，有没有文档显得并不怎么重要，但对于一个需要有多人合作，并且开发、维护较长时间的软件来说，文档就是至关重要的。文档记录程序设计的算法、实现以及修改的过程，保证程序的可读性和可维护性。程序中的注释就是一种很好的文档。

2. 结构化程序设计方法

在早期由于计算机存储器容量非常小，人们设计程序时首先考虑的问题是如何减少存储器开销，硬件的限制不容许人们考虑如何组织数据与逻辑，为此程序员使用各种技巧和手段编写高效的程序。其中显著的特点是程序中大量使用 GOTO 语句，使得程序结构混乱、可读性差、可维护性差、通用性差。但是，随着大容量存储器的出现及计算机技术的广泛应用，程序编写越来越困难，程序的大小以算术级数递增，而程序的逻辑控制难度则以几何级数递增，人们不得不考虑程序设计的方法。

结构化程序设计是进行以模块功能和处理过程设计为主的详细设计的基本原则，其概念最

早由荷兰科学家 E.W.Dijkstra 提出，它的主要观点是采用自顶向下、逐步求精的程序设计方法；使用三种基本控制结构构造程序，任何程序都可由顺序、选择、重复三种基本控制结构构造。

3. 面向对象程序设计

虽然结构化程序设计方法具有很多的优点，但它仍是一种面向过程的程序设计方法，它把数据和处理数据的过程分离为相互独立的实体。当数据结构改变时，所有相关的处理过程都要进行相应的修改，每一种相对于老问题的新方法都要带来额外的开销，程序的可重用性差。

同时由于图形用户界面的应用，程序运行由顺序运行演变为事件驱动，使得软件使用起来越来越方便，但开发起来却越来越困难，对这种软件的功能很难用过程来描述和实现，使用面向过程的方法来开发和维护都将非常困难。

由于上述缺陷已不能满足现代化软件开发的要求，一种全新的软件开发技术应运而生，这就是面向对象程序设计（Object Oriented Programming，OOP）。面向对象程序设计方法于 20 世纪 60 年代后期首次提出，80 年代开始走向实用。

面向对象程序设计是一种计算机编程架构。OOP 的一条基本原则是计算机程序由单个能够起到子程序作用的单元或对象组合而成。OOP 达到了软件工程的三个主要目标：重用性、灵活性和扩展性。OOP=对象+类+继承+多态+消息，其中核心概念是类和对象。面向对象程序设计方法是尽可能模拟人类的思维方式，使得软件的开发方法与过程尽可能接近人类认识世界、解决现实问题的方法和过程，也即使得描述问题的问题空间与问题的解决方案空间在结构上尽可能一致，把客观世界中的实体抽象为问题域中的对象。面向对象程序设计以对象为核心，该方法认为程序由一系列对象组成。

1.1.3 程序设计语言翻译系统

计算机硬件只能识别并执行机器指令，即只能直接执行相应机器语言格式的代码程序，而不能直接执行高级语言或汇编语言编写的程序。为了让计算机能够理解高级语言或汇编语言编写的程序，必须要为它配备一个“翻译”，这就是所谓的程序设计语言翻译程序。

程序设计语言翻译程序是一类系统软件，通常所说的翻译程序是指这样一个程序，它能够把某一种语言程序（称为源语言程序）转换成另一种语言程序（称为目标语言程序），而后者与前者在逻辑上是等价的。不同的程序设计语言需要有不同的程序语言翻译系统，同一种程序设计语言在不同类型的计算机上也需要配置不同的程序设计语言翻译系统。

程序设计语言翻译系统可以分成 3 种：汇编语言翻译系统、高级语言翻译系统和高级语言解释系统。这些翻译系统之间的不同之处主要体现在它们生成计算机可以执行的机器语言的过程中。

1. 汇编语言翻译系统

汇编语言翻译系统（汇编程序）的主要功能是将汇编语言书写的源程序，翻译成用二进制代码 0 或 1 表示的等价的机器语言，形成计算机可以执行的机器指令代码，如图 1-1 所示。

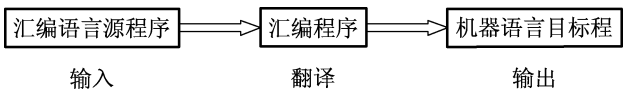


图 1-1 汇编程序功能示意图

2. 高级程序设计语言翻译系统

高级程序设计语言翻译系统是指将用高级语言编写的源程序翻译成等价的汇编语言程序或机器语言程序的处理系统，也称为编译程序。

由此可见，在计算机上用编译方式执行高级语言编写的程序，一般需要两个阶段：第一阶段称为编译阶段，其任务是由编译程序将源程序翻译为目标程序，如果目标程序不是机器语言程序，则尚需汇编程序再行汇编为机器代码程序；第二阶段为运行阶段，其任务是在目标计算机上执行编译阶段所得到的目标程序。编译程序和运行系统合称为编译系统。图 1-2 显示了按编译方式执行一个高级语言的主要步骤。

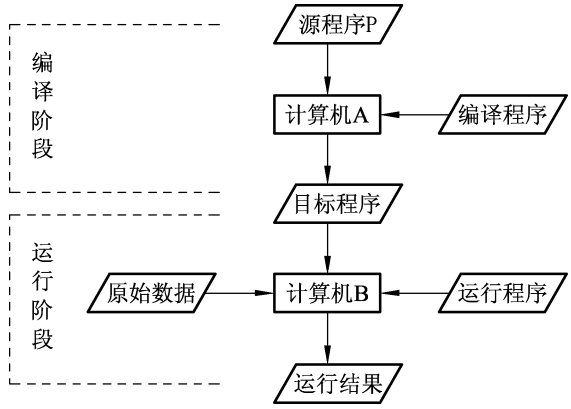


图 1-2　计算机执行高级语言程序的步骤

3. 高级程序设计语言解释系统

高级程序设计语言解释系统是按源程序中的语句的动态顺序逐条翻译并立即执行相应功能的处理系统。它将源语言(如 BASIC)书写的源程序作为输入，解释一句后就提交计算机执行一句，并不形成目标程序。就像外语翻译中的“口译”一样，说一句译一句，不产生全文的翻译文本。这种工作方式非常适合于人通过终端设备与计算机会话，如在终端上打一条命令或语句，解释程序就立即将此语句解释成一条或几条指令并提交硬件立即执行且将执行结果反映到终端，从终端把命令打入后，就能立即得到计算结果。

对源程序边解释翻译成机器代码边执行的高级语言程序，由于它的方便性和交互性较好，早期一些高级语言采用这种方式，如 BASIC。但它的弱点是运行效率低，程序的运行依赖于开发环境，不能直接在操作系统下运行。

1.2　算法及其描述

1.2.1　算法的概念

1. 什么是算法

计算机解题一般可分解成若干操作步骤，通常把完成某一任务的操作步骤称为求解该问题的算法。程序就是用计算机语言描述的算法。

算法是指完成一个任务所需要的具体步骤和方法。也就是说给定初始状态或输入数据，

能够得出所要求或期望的终止状态或输出数据。

既然算法是解决给定问答的方法，算法所处理的对象就是该问题所涉及的数据。程序的目的就是加工数据，而如何加工数据就是算法的目的。

例如给定两个正整数 m 和 n，求它们的最大公约数。在学数学的时候，我们都知道这个问题就是求能同时整除 m 和 n 的最大正整数。但是要在计算机中实现的话，仅有数学的思维是不行的，计算机中实现的步骤如下：

（1）以 n 除 m 并令所得余数为 r，r 必小于 n；

（2）若 $r = 0$ 算法结束，输出结果 n。否则继续步骤（3）；

（3）将 m 置换为 n，n 置换为 r，并返回步骤（1）继续进行。

2. 算法的性质

著名计算机科学家 Donald Knuth 在他的著作《The Art of Computer Programming》中曾把算法的性质归纳为以下 5 点：

（1）输入：一个算法必须有零个或以上输入量。

（2）输出：一个算法应有一个或以上输出量，输出量是算法计算的结果。

（3）明确性：算法的描述必须无歧义，以保证算法的实际执行结果精确地符合要求或期望，通常要求实际运行结果是确定的。

（4）有限性：依据图灵的定义，一个算法是能够被任何图灵完备系统模拟的一串运算，而图灵机只有有限个状态、有限个输入符号和有限个转移函数（指令）。而一些定义更规定算法必须在有限个步骤内完成任务。

（5）有效性：又称可行性。能够实现，算法中描述的操作都可以通过已经实现的基本运算执行有限次来实现。

1.2.2 算法的描述

算法是对解题过程的精确描述。定义解决问题的算法对程序员来说通常是最具挑战性的任务。它既是一种技能又是一门艺术，要求程序员懂得程序设计概念并具有创造性。对算法的描述是建立在语言基础之上的。在将算法转化为高级语言源程序之前，通常先采用文字或图形工具来描述算法。文字工具如自然语言、伪代码等，图形工具如流程图、N-S 流程图等。

1. 自然语言

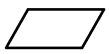
自然语言即人们日常生活中所用的语言，如汉语、英语等。使用自然语言不用专门训练，所描述的算法也通俗易懂。然而其缺点也是明显的：首先是由于自然语言的歧义性容易导致算法执行的不确定性；其次是由于自然语言表示的串行性，因此当一个算法中循环和分支较多时，就很难清晰地表示出来；此外，自然语言表示的算法不便转换成用计算机程序设计语言表示的程序。

2. 流程图

流程图是采用一些框图符号来描述算法的逻辑结构，每个框图符号表示不同性质的操作。流程图可以很方便地表示任何程序的逻辑结构。另外，用流程图表示的算法不依赖于任何具

体的计算机和程序设计语言，从而有利于不同环境的程序设计。早在 20 世纪 60 年代，美国国家标准协会（American National Standards Institute，ANSI）就颁布了流程图的标准，这些标准规定了用来表示程序中各种操作的流程图符号，例如用矩形表示处理，用菱形表示判断，用平行四边形表示输入/输出，用带箭头的折线表示流程，等等。如表 1-1 所示流程图符号及意义。

表 1-1　流程图常用符号

流程图符号	名　称	说　明
	起止框	表示算法的开始和结束
	处理框	表示完成某种操作，如初始化或运算赋值等
	判断框	表示根据一个条件成立与否，决定执行两种不同操作的其中一个
	输入输出框	表示数据的输入输出操作
	流程线	用箭头表示程序执行的流向
	连接点	用于流程分支的连接

3. N-S 流程图

N-S 流程图又称为结构化流程图，于 1973 年由美国学者 I.Nassi 和 B.Shnei-derman 提出。与传统流程图不同的是，N-S 流程图不用带箭头的流程线来表示程序流程的方向，而采用一系列矩形框来表示各种操作，全部算法写在一个大的矩形框内，在大框内还可以包含其他从属于它的小框，这些框一个接一个从上向下排列，程序流程的方向总是从上向下。N-S 结构化流程图比较适合于表达三种基本结构（顺序、选择、循环），适于结构化程序设计，因此很受程序员欢迎。

N-S 流程图用以下不同的流程图符号表示不同的结构：

（1）顺序结构。顺序结构用图 1-3 形式表示。A 和 B 两个框组成一个顺序结构。

（2）选择结构。选择结构用图 1-4 表示，它与图 1-3 相应。当 P 条件成立时执行 A 操作，P 不成立则执行 B 操作。注意：图 1-4 是一个整体，代表一个基本结构。

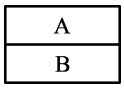


图 1-3　顺序结构

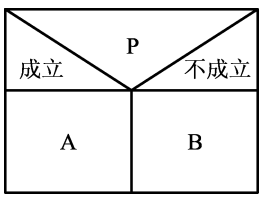


图 1-4　选择结构

（3）循环结构。“当”型循环结构用图 1-5 形式表示。图 1-5 表示当 P1 条件成立时反复执行 A 操作，直到 P1 条件不成立为止。“直到”型循环结构用图 1-6 形式表示。

在初学时，为清楚起见，可如图 1-5 和图 1-6 那样，写明“当 P1”或“直到 P2”，待熟练之后，可以不写“当”和“直到”字样，只写“P1”和“P2”。从图的形状即可知道是“当”型或“直到”型。

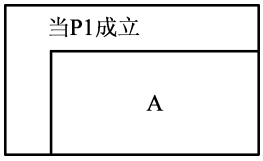


图 1-5 “当”型循环结构

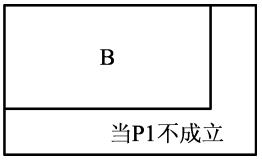


图 1-6 “直到”型循环结构

用以上 3 种 N-S 流程图中的基本框可以组成复杂的 N-S 流程图，以表示算法。

应当说明，在图中的 A 框或 B 框，可以是一个简单的操作（如读入数据或打印输出等），也可以是 3 个基本结构之一。

例如，图 1-7 所表示的顺序结构，其中的 A 框可以又是一个选择结构，B 框可以又是一个循环结构。如图 1-8 所示那样，由 A 和 B 这两个基本结构组成一个顺序结构。

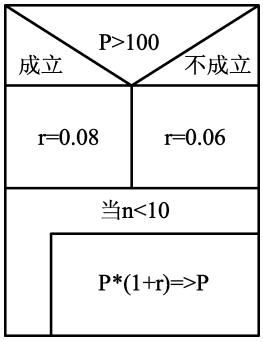


图 1-7 复杂的 N-S 流程图 1

图 1-8 复杂的 N-S 流程图 2

4. 伪代码

伪代码是指不能够直接编译运行的程序代码，它是用介于自然语言和计算机语言之间的文字和符号来描述算法和进行语法结构讲解的一个工具。表面上它很像高级语言的代码，但又不像高级语言那样要接受严格的语法检查。它比真正的程序代码更简明，更贴近自然语言。它不用图形符号，因此书写方便，格式紧凑，易于理解，便于向计算机程序设计语言算法程序过渡。用伪代码书写算法时，既可以采用英文字母或单词，也可以采用汉字，以便于书写和阅读。它没有固定的、严格的语法规则，只要把意思表达清楚即可。用伪代码描述算法时，自上而下地写。每一行（或每几行）表示一个基本操作。用伪代码书写的算法格式紧凑，易于理解，便于转化为计算机语言算法（即程序）。在书写时，伪代码采用缩进格式来表示三种基本结构。一个模块的开始语句和结束语句都靠着左边界书写，模块内的语句向内部缩进一段距离，选择结构和循环结构内的语句再向内缩进一段距离。这样的话，算法书写格式一致，富有层次，清晰易读，能直观地区别出控制结构的开始和结束。

5. 程序设计语言

对一些简单的问题，可以直接使用某种程序设计语言来描述算法。

6. 算法设计举例

【例 1-1】若给定两个正整数 *m* 和 *n*，试写出求它们的最大公约数（即能同时整除 *m* 和 *n* 的最大正整数）的算法。

解：在公元前 300 年左右，欧几里得在其著作《几何原本》（Elements）中阐述了求解两个数最大公约数的过程。

（1）该算法用自然语言描述如下：

第 1 步：读入两个正整数 *m* 和 *n*，大的数存入 *m*，小的数存入 *n*；

第 2 步：求 *m* 除以 *n* 的余数 *r*；

第 3 步：用 *n* 的值取代 *m*，*r* 的值取代 *n*；

第 4 步：判别 *r* 的值是否为零，如果 *r* = 0，则算法结束，输出 *m* 的值，即为最大公因子；否则返回第 2 步。

（2）流程图如图 1-9 所示。

（3）N-S 流程图如图 1-10 所示。

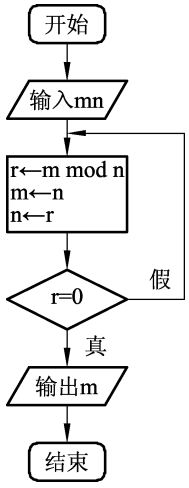


图 1-9　例 1-1 流程图

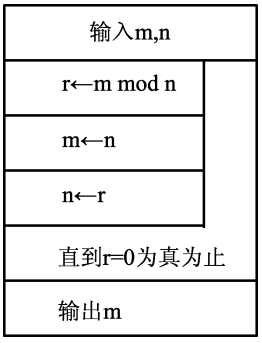


图 1-10　例 1-1 N-S 流程图

（4）伪代码描述表示如下：

```
INPUT m,n
DO
    r=m MOD n
    m=n
    n=r
LOOP UNTIL r=0
PRINT m
END
```

（5）C 语言描述如下：

C 源程序（文件名：li1_1.c）：

li1_1.c

```
#include <stdio.h>
int main()
{
    int m,n,r;
    printf("请输入 m,n:");
    scanf("%d%d",&m,&n);
    do
    {
        r = m%n;
        m = n;
        n = r;
    }while (r);
    printf("最大公约数：%d\n",m);
    return 0;
}
```

【例 1-2】求 1+ 2+ 3+ … + 100 之和。分别用传统流程图、N-S 流程图及自然语言描述其算法，并将该算法转化为 BASIC 语言源程序。设变量 *x* 表示被加数，*y* 表示加数。

解：（1）该算法用自然语言描述如下：

步骤 1：将 1 赋值给 *x*；

步骤 2：将 2 赋值给 *y*；

步骤 3：将 *x* 与 *y* 相加，结果存放在 *x* 中；

步骤 4：将 *y* 加 1 结果存放在 *y* 中；

步骤 5：若 *y* 小于或等于 100 转到步骤 3 继续执行，否则算法结束，结果为 *x*。

（2）流程图如图 1-11 所示。

（3）N-S 流程图如图 1-12 所示。

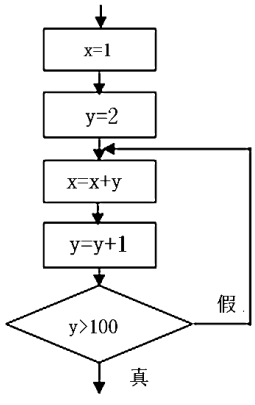


图 1-11　例 1-2 流程图

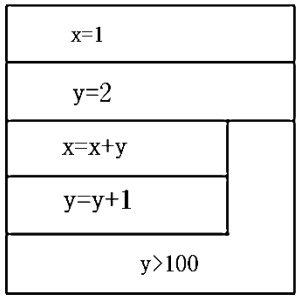


图 1-12　例 1-2 N-S 流程图