WUTP

普通高等学校计算机科学与技术专业新编系列教材

High-Level Language Programming

高级语言程序设计

王 浩 主编

```
#include <stdio.h>
#include <stdio.h>
void main()
void main()
{
   void swap(int *ptr1,int *ptr2);
   void swap(int *ptr1,int *ptr2);
  int x,y,*ptr1,*ptr2;
  int x,y,*ptr1,*ptr2;
  printf("input x,y:");scanf("%d,%d",&x,
  printf("input x,y:");scanf("%d,%d",&x,
  printf("%d\t%d\n",x,y);ptr1=&x;ptr2=
  printf("%d\t%d\n",x,y);ptr1=&x;ptr2=
  if(x<y)
  if(x<y)
  swap(ptr1,ptr2);
  swap(ptr1,ptr2);
  printf("%d\t%d\n",x,y);
  printf("%d\t%d\n",x,y);
}
void swap(int *ptr1,int *ptr2)
```

武汉理工大学出版社
Wuhan University of Technology Press

普通高等学校计算机科学与技术专业新编系列教材

High-Level Language Programming

高级语言程序设计

主　编　王　浩

副主编　周双娥　谭同德

武汉理工大学出版社

Wuhan University of Technology Press

内 容 提 要

全书共分 11 章。第 1 章概述，介绍必要的基本知识；第 2 章介绍 C 语言的基本数据类型及其运算；第 3 章介绍 C 语言简单程序设计，即顺序结构程序设计；第 4 章介绍 C 语言流程控制语句；第 5 章介绍数组及其应用；第 6 章介绍函数及其应用；第 7 章介绍 C 语言编译预处理功能；第 8 章介绍指针及其应用；第 9 章介绍结构与联合；第 10 章介绍 C 语言的输入输出；第 11 章介绍面向对象语言 C++基础知识，两个附录分别给出了 ASCII 字符编码表和 C 标准库函数。

本书可作为高等院校计算机科学与技术专业或其他专业的计算机程序设计教材，适用于只有很少甚至没有编程经验的大专院校学生，也可作为从事计算机应用的科技人员自学或培训教材。

图书在版编目(CIP)数据

高级语言程序设计/王浩主编. —武汉：武汉理工大学出版社，2003.8
普通高等学校计算机科学与技术专业新编系列教材
ISBN 7-5629-1953-4
Ⅰ.高… Ⅱ.王… Ⅲ.高级语言-程序设计-高等学校-教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2002)第 106868 号

出版发行：武汉理工大学出版社(武汉市珞狮路 122 号 邮政编码：430070)
HTTP：//www.whut.edu.cn/chubanl
E-mail：wutp@public.wh.hb.cn
经 销 者：各地新华书店
印 刷 者：武汉理工大印刷厂
开 本：787×960 1/16
印 张：20.5
字 数：400 千字
版 次：2003 年 8 月第 1 版
印 次：2003 年 8 月第 1 次印刷
印 数：1～5000 册
定 价：28.00 元

本书如有缺页、倒页、脱页等印装质量问题，请向出版社发行部调换。本社购书热线电话：(027)87397097 87394412

普 通 高 等 学 校
计算机科学与技术专业新编系列教材
编 审 委 员 会

出版说明

当今世界已经跨入了信息时代，计算机科学与技术正在迅猛发展。尤其是以计算机为核心的信息技术正在改变整个社会的生产方式、生活方式和学习方式，推动整个人类社会进入信息化社会。为了顺应时代潮流，适应计算机专业调整及深化教学改革的要求，充分考虑到不同层次高校的教学现状，满足广大高校的教学需求，武汉理工大学出版社经过广泛调研，与国内近 30 所高等院校的计算机专家进行探讨，决定组织编写“普通高等学校计算机科学与技术专业新编系列教材”。

我们在组织编写新编本套系列教材时，**以培养现代化高级人才为重任，以提高学生综合素质、培养学生应用能力和创新能力为目的，以面向现代化、面向世界、面向未来为准绳，注重系列教材的特色和实用性，反映最新的教学与科研成果，体现本专业的时代特征。**同时，面对教育改革的需要、人才的需要和社会的需要，在编写本教材时，借鉴、学习国外一流大学的先进教学体系，结合国内的实际需要，吸取具有先进性、实用性和权威性的国外教材的精华，以更好地促进国内教材改革顺利进行。从时代和国际竞争要求的高度来思考，为打造一套高起点、高水平、高质量的系列教材而努力。

本套教材具有以下特色：

与时俱进，内容科学先进——充分体现计算机学科知识更新快的特点，及时更新知识，确保教材处于学科前沿，以拓宽学生知识面，培养学生的创新能力。

紧跟教学改革步伐，体现教学改革的阶段性成果——符合全国高校计算机专业教学指导委员会、中国计算机学会教育委员会制订的“计算机学科教学计划 2000”的内容要求。

实现立体化出版，适应教育方式的变革——本套教材努力使用和推广现代化的教学手段，凡有条件的课程都准备组织编写、制作和出版配合教材使用的实验、习题、课件、电子教案及相应的程序设计素材库。

本套教材首批 25 种预定在 2003 年秋季全部出齐。我们的编审者、出版者决不敢稍有懈怠，一定高度重视，兢兢业业，按最高的质量标准工作。教材建设是我们共同的事业和追求，也是我们共同的责任和义务，我们诚恳地希望大家积极选用本套教材，并在使用过程中给我们多提意见和建议，以便我们不断修订、完善全套教材。

武汉理工大学出版社

2002 年 10 月

前　言

C语言是一种结构化、模块化、可编译的通用程序设计语言。C语言具有表达能力强、灵活性好、应用面宽、编译实现容易、代码质量高、语言规模小、通用性和可移植性良好等特点，并兼备高级语言和低级语言的许多特点。自问世迄今，广泛应用于系统软件设计（操作系统、语言处理、系统实用程序）、数据处理、科学计算、计算机辅助工程、办公自动化、计算机控制等领域，不仅成为当前软件开发的主流语言，也成为计算机语言教学的首选语言。

C语言是作为解决实际问题的工具而设计的，而不是用作理论证明或作为某些系统的例子。用它编写的UNIX操作系统同样具有规模小、适应性强、实现容易、通用性好等特点。C语言和UNIX操作系统是20世纪70年代最重要的软件成果之一，其影响和作用至今经久不衰。由于其取得的巨大成功，C语言的创始人D. M. 里奇（Dennis M. Ritchie）和UNIX的主要开发者K. 汤普逊（Ken Thompson）一起荣获了1982年度Electronics最高成就奖，这是该奖项设立以来第一次授予在软件领域里作出杰出贡献的科学家。继之，美国计算机学会也授予他们1983年度计算机科学最高荣誉奖——ACM图灵奖。

本书作为高等学校计算机科学与技术专业入门语言教材，以ANSI C标准为主线，全面、系统地讲述C语言基本概念、原理和方法。本书特别强调理论联系实际，结合大量例题和上机实践，介绍程序设计的基本知识和基本方法，培养掌握用计算机解决问题的思维方法及计算机操作和调试程序的基本技能。在内容安排上遵循深入浅出、由简到繁、循序渐进的原则，并突出C语言的灵活性和实用性。书中例题程序均可在Borland公司的C/C++系列开发环境下（如Turbo C 2.0、Turbo C++ 3.0、Borland C++）运行。

本书第1章、第8章由王浩编写；第2章、第5章由谭同德编写；第3章、第4章、第10章由徐苏编写；第6章、第7章、附录A、附录B由周双娥编写；第9章、第11章由王文才编写。全书由王浩任主编，周双娥、谭同德任副主编。

本书作为普通高等学校计算机科学与技术专业（本科）新编系列教材之一，得到了普通高等学校计算机科学与技术专业新编系列教材编审委员会和武汉理工大学出版社的关心和大力支持，在此一并表示感谢！

由于作者水平有限，本书不妥之处在所难免，恳请读者予以批评指正。

编者

2003年4月

目　录

1 概　述

本章提要

本章首先回顾了程序设计语言的发展，随后介绍了程序、软件、算法和结构化程序设计的基本概念，最后介绍了C语言的发展历史、C语言的特点及程序结构。

程序设计语言随着计算机技术的发展，经历了从机器语言、汇编语言、高级语言等阶段。早期的高级语言如ALGOL60、FORTRAN、COBOL等开创了最初的软件产业，但这些语言的数据类型单调，程序设计主要依赖于程序员的个人技巧，缺乏规范化的设计方法，因此程序规模较大时，其复杂性和可靠性就变得难以控制。20世纪70年代结构化程序设计出现，强调程序的模块性，C语言就是这种结构化程序设计语言的代表作。

1.1 程序设计和程序设计语言

程序设计语言是人与计算机交互的工具，人把需要计算机完成的工作告诉计算机，就需要使用程序设计语言编写程序，让计算机去执行。

1.1.1 程序与软件

程序是对所要解决问题的各个对象和处理规则的描述。或者说是为解决某一问题而设计的一系列计算机能识别的指令(由操作码和操作数组成)。

程序设计是一个使用程序设计语言产生一系列的指令以告诉计算机该做什么的过程。程序设计人员(简称程序员)的工作就是用程序设计语言来生成一系列可以被计算机接受和执行的指令完成各种指定的任务。

软件是计算机程序、方法、规则以及所要求的文档资料和在计算机上运行时所必需的数据。软件发展经历了三个时期：

(1) 程序时期(1947～1960 年初) 程序作为机器运行时必须进行的准备工作。程序设计全凭设计者个人经验和技艺独立进行，是一种典型的手工艺智力劳动。

(2) 软件＝程序＋说明时期(20 世纪 50 年代末～20 世纪 70 年代初) 程序规模较大，需多人协作才能完成，“作坊式”生产；程序的设计与运行维护不能由一个人来承担；程序不再是计算机硬件的附属部分，而是计算机系统中与硬件相互依存的不可缺少的部分。

(3) 软件＝程序＋文档时期(20 世纪 70 年代初至今，即软件工程时期) 用“工程化”的思想作指导来解决软件研究和开发中面临的困难和混乱。

软件的开发必须以工程化的思想为指导，运用标准和规范的方法来进行。程序编码，即我们通常所说的“编程序”，是软件开发过程中的一个关键阶段。程序是软件的本体，软件的质量主要是通过程序的质量来体现的，因此程序设计在软件开发中占有十分重要的地位。程序设计人员既需要有严密的思维能力，又需要有创意甚至是一定的艺术修养。成为一个优秀的程序员需要经过艰苦的努力。

1.1.2 程序设计语言发展

程序设计语言经历了机器语言、汇编语言和高级程序设计语言三个阶段：

(1) 机器语言

机器语言由计算机的指令系统组成，机器语言中的每一条语句(机器指令)实际就是由 0 和 1 组成的二进制代码，它由操作码的二进制编码和操作数的二进制编码组成。机器指令通常随计算机类型不同而不同，编写程序效率低、难理解且不能通用。

(2) 汇编语言

汇编语言是面向机器的程序设计语言。在汇编语言中，使用助记符表示机器指令的操作码和操作数。这种替代使得机器语言“符号化”，所以又称汇编语言是符号语言。与机器语言一样，不同类型计算机具有不同的汇编语言。

由于计算机硬件只能识别机器指令，因此汇编语言书写的程序必须先翻译成机器语言程序才能在计算机上执行，用于翻译的程序称为汇编程序(汇编系统)。汇编语言相对机器语言较易理解和记忆，且能直接对计算机硬件操作，因此在实时控制、实时监测等领域仍有较多的应用。

(3) 高级语言

机器语言和汇编语言都是面向机器的语言，因此人们称之为“低级语言”。

从20世纪50年代中期开始，出现了许多高级程序设计语言。高级语言中的语句用较为接近自然语言的英文字句表示，数据用十进制来表示。它独立于具体的计算机系统，从而容易为人掌握和理解，它是面向人的程序设计语言。高级语言发展依据程序设计方法也经历了三个时期，即早期的线性程序设计、结构化程序设计（又称面向过程程序设计）和面向对象程序设计。

任何一种高级程序设计语言都有严格的词法规则、语法规则和语义规则。词法规则规定了从语言的基本符号集如何构成单词；语法规则规定了从单词如何构成各类语法单位（如表达式、语句、程序等）；语义规则规定了各个语法单位的含义。高级语言的词法、语法和语义可能各不相同，但一般说来，其成分有如下四种：

① 数据成分。用以描述程序中所描述的数据。

② 运算成分。用以描述程序中所包含的运算。

③ 控制成分。用以表达程序中的控制结构。

④ 传输成分。用以表达程序中的数据传输。

高级语言书写的程序称为“源程序”，源程序同样不能在计算机上直接运行，也需翻译成机器指令才能执行。将源程序翻译成机器指令时，通常有两种翻译方式：一种为“解释”方式，另一种为“编译”方式。所谓解释方式是把源程序逐句翻译，翻译一句执行一句，边翻译边执行，翻译工作用解释程序完成，解释程序不产生被执行的目标程序。而编译方式是把源程序先翻译成等价的目标程序，然后再执行目标程序，将源程序翻译成机器语言的程序称为编译程序。将源程序转换成可执行的目标程序一般分为两个阶段：

① 翻译阶段。提供汇编程序或编译程序将源程序转换成目标程序，这一阶段的目标模块由于没有分配存储器的绝对地址，是不能执行的。

② 连接阶段。这一阶段用连接程序把目标程序以及所需的功能库等转换成一个可执行的装入程序。这个装入程序分配有地址，是可执行程序。

从源程序输入到可执行程序生成的过程如图1.1所示。

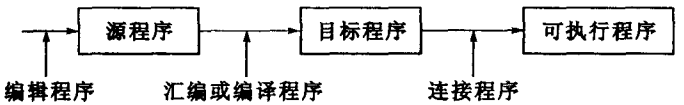

图1.1 源程序编辑运行过程

1.2 算　法

1.2.1 问题求解过程

通过计算机系统来求解问题一般需经过如下步骤：

(1) 问题分析。通过问题分析，可以准确地定义所要解决问题的要求，确定问题的输入输出。

(2) 算法设计。确定待解问题的数据对象和求解步骤。

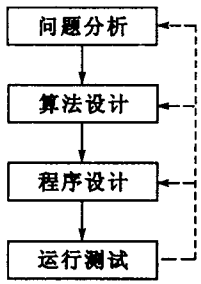

图 1.2　问题求解过程

(3) 程序设计。即使用程序设计语言将算法编写为程序，因此前面两步是与语言无关的。

(4) 运行测试。通过对预先设计的一组测试数据实际的程序运行，对程序代码、模块接口、算法等错误进行检测，发现错误，纠正错误，直到最终得到正确可用的程序为止。

图 1.2 显示了问题求解的过程，可以看出，通过运行测试，我们可以发现上面三个阶段所出现的错误。这主要依靠测试数据集的质量。

1.2.2 算法及其表示

算法是由一系列规则组成的过程，这些规则确定了一个操作的顺序，以便能在有限步骤内得到特定的解。算法具有以下特点：

(1) 有穷性。一个算法应包含有限的操作步骤，而不能是无限的。

(2) 确定性。算法中的每一个步骤都应当是确定的，而不是含糊的，模棱两可的。

(3) 通用性。算法是给出一类问题的求解方法，而不是表示解决某一个特殊的具体的问题。

(4) 有零个或多个输入。所谓输入是指在执行算法时需要从外界取得必要的信息。

(5) 有一个或多个输出。算法的目的是为了求解，“解”就是输出。没有输出的算法是没有意义的。

表示一个算法可以用自然语言、流程图、算法描述语言等。

自然语言可以是中文或英文，用自然语言表示算法通俗易懂，但文字冗长，且由于自然语言语义不严格，往往会出现算法表示的“歧义性”。一般不用自然语言描述算法。

流程图是用一些图形元素来表示算法或程序中的各个要素。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(见图 1.3),已为程序员广泛采用。

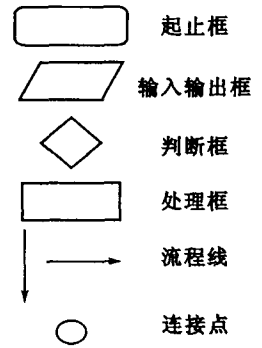

图 1.3　流程图符号

算法描述语言(又称伪码)是一种把自然语言与程序设计语言相结合起来的算法描述工具。它使用程序设计语言语句形式和控制成分,并按严谨的程序结构来描述算法,但无程序设计语言的一些细节成分,因此不是一个正式的程序。

下面我们分别用自然语言和流程图来描述判断一个大于或等于 3 的正整数是不是一个素数的算法。

所谓素数,是指除 1 和该数本身之外,不能被其他任何整数整除的数。因此,判断一个数 $n(n>3)$是否素数的方法很简单:将 n 作为被除数,将 2 到 $n-1$ 各个整数作为除数,如果都不能被整除,则 n 为素数。定义变量 i 记录各个整数。

算法用自然语言描述如下:

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:如果 $r=0$,表示 n 能被 i 整除,打印"n 不是素数",算法结束。否则执行步骤 5。

步骤 5:i 自增 1,即 $i+1\Rightarrow i$。

步骤 6:如果 $i\leqslant n-1$,返回步骤 3。否则打印"n 是素数",算法结束。

算法用流程图描述如图 1.4 所示。

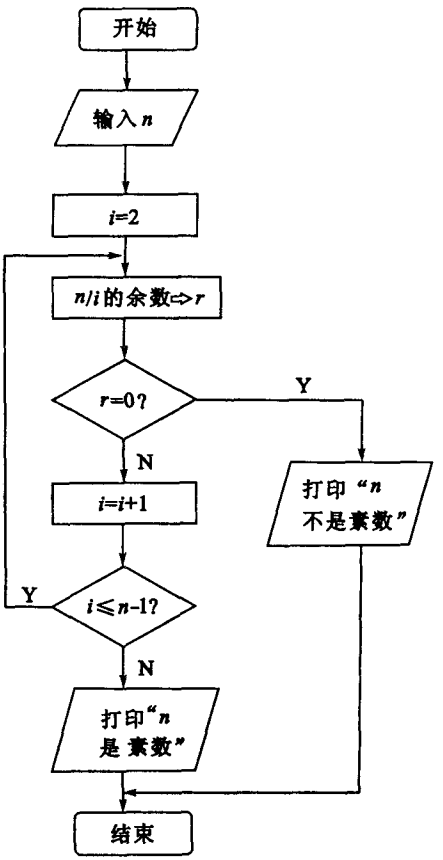

图 1.4　判断素数算法

1.2.3　结构化程序设计

结构化程序设计的原理思想是由 E. W. Dijkstra 等人于 20 世纪 60 年代后期提出的。它采用自顶向下逐步求精的设计方法和单入口单出口的控制结构。结构化程序设计应遵循下列原则: