



iCourse · 教材
高等农林院校基础课程系列



自主创新
方法先行

C 语言程序设计 实验教程

主 编 任洪娥 高心丹
副主编 宋宏光 董本志

高等教育出版社



iCourse · 教材
高等农林院校基础课程系列



自主创新
方法先行



C 语言程序设计 实验教程

主 编 任洪娥 高心丹
副主编 宋宏光 董本志

高等教育出版社·北京

内容提要

本书为《C 语言程序设计教程》配套教材。全书共分为 13 章和 1 个附录。第 1 章介绍实验前预备知识,第 2 章到第 13 章遵照教材《C 语言程序设计教程》的内容组织安排,每章包括 3 个小节:本章要点、实验指导、习题及参考答案。第 1 节给出该章的学习重点、难点和要点;第 2 节组织多种类型的上机实验题目,并给出解题思路和注意事项;第 3 节安排多种题型的习题和答案,编程题给出一个或多个参考程序、运行结果、分析及提示。所给出的参考程序具有全面、细致和书写格式规范的特点,阅读和调试这些程序,可以促使读者在初学 C 语言时就建立良好的编程风格,并从中学会全面、严谨地考虑和解决问题。附录列出了编程过程中易出现的典型错误,以帮助初学者在编写和调试程序过程中改正错误。

本书配有数字资源,可作为 C 语言类课程的配套教材使用,也可作为教师的教学参考书。

图书在版编目(CIP)数据

C 语言程序设计实验教程 / 任洪娥,高心丹主编. --
北京:高等教育出版社,2015.8

iCourse·教材.高等农林院校基础课程系列

ISBN 978-7-04-042836-0*



I. ①C… II. ①任… ②高… 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(2015)第 169299 号

C Yuyan Chengxu Sheji Shiyan Jiaocheng

项目策划 王 瑜 李光跃 陈琪琳 李艳霞 吴雪梅

策划编辑 陈 哲

责任编辑 陈 哲

封面设计 张 楠

责任印制 田甜

出版发行 高等教育出版社

社 址 北京市西城区德外大街 4 号

邮政编码 100120

印 刷 三河市吉祥印务有限公司

开 本 850mm × 1168mm 1/16

印 张 21.25

字 数 450 千字

购书热线 010-58581118

咨询电话 400-810-0598

网 址 <http://www.hep.edu.cn>

<http://www.hep.com.cn>

网上订购 <http://www.landaco.com>

<http://www.landaco.com.cn>

版 次 2015 年 8 月第 1 版

印 次 2015 年 8 月第 1 次印刷

定 价 31.50 元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换

版权所有 侵权必究

物 料 号 42836-00

iCourse · 数字课程 (基础版)

C 语言程序设计 实验教程

主编 任洪娥 高心丹

<http://abook.hep.com.cn/1880151>

登录方法:

1. 访问 <http://abook.hep.com.cn/1880151>, 单击“注册”。在注册页面输入用户名、密码及常用的邮箱进行注册。已注册的用户直接输入用户名和密码登录即可进入“我的课程”界面。
2. 课程充值: 登录后单击右上方“充值”图标, 正确输入教材封底标签上的明码和密码, 单击“确定”按钮完成课程充值。
3. 在“我的课程”列表中选择已充值的数字课程, 单击“进入课程”即可开始课程学习。

账号自登录之日起一年内有效, 过期作废。

使用本账号如有任何问题, 请发邮件至:

ecourse@hep.com.cn



iCourse · 教材
高等农林院校基础课程系列



自主创新
方法先行

C 语言程序设计实验教程

任洪娥 高心丹

用户名

密码

验证码

8298

进入课程

注册

数字课程介绍

纸质教材

版权信息

联系方式

“C语言程序设计实验教程”数字课程与纸质教材一体化设计, 紧密配合。数字课程提供了实验程序和习题程序, 可以让读者更直观地学习程序设计。数字课程丰富了知识的呈现形式, 拓展了教材内容; 在提升教学效果的同时, 为学生学习提供思维与探索的空间。

因系统升级, 所有用户都需要先注册

(不能用书后的明码暗码直接登录)。

注册后的用户登录后, 请先点击页面右上方“充值”, 正确输入教材封底标签上的明码和密码完成课程选择。

注册

登录

充值

资源说明

与本书配套的数字课程资源按照章、节知识树的形式构成,部分章节配有实验程序和习题程序,内容标题为:

1. 实验程序:实验程序为书中实验所用到的代码,可供学生下载用于课前预习或课后复习。
2. 习题程序:书中所列出的习题程序在数字课程中有对应的资源包可供下载,方便学生课后练习并理解程序设计思路。

出版说明

“十二五”是继续深化高等教育教学改革、走以提高质量为核心的内涵式发展道路和农林教育综合改革深入推进的关键时期。教育教学改革的核心是课程建设,课程建设水平对教学质量和人才培养质量具有重要影响。2011年10月12日教育部发布了《教育部关于国家精品开放课程建设的实施意见》(教高[2011]8号),开启了信息技术和网络技术条件下校、省、国家三级精品开放课程建设的序幕。作为国家精品开放课程展示、运行和管理平台的“爱课程(iCourse)”网站也逐渐为高校师生和社会公众认知和使用。截至目前,已启动2911门精品资源共享课和696门精品视频公开课的立项建设,其中的1000多门精品资源共享课和600多门精品视频公开课已经在“爱课程(iCourse)”网站上线。

高等教育出版社承担着“十二五”本科教学工程”国家精品开放课程建设的组织实施和平台建设运营的重要任务,在与广大高校,特别是高等农林院校的调研和协作中,我们了解到当前高校的教与学发生了深刻变化,也真切感受到课程和教材建设所面临的挑战和机遇。如何建设支撑学生自主学习和校际共建共享的课程和新形态教材成为现实课题,结合我社2009年以来在数字课程建设上的探索和实践,我们提出了“高等农林院校基础课程精品资源共享课及系列教材”建设项目,并获批列入科技部“科学思维、科学方法在高等学校教学创新中的应用与实践”项目(项目编号:2009IM010400)。项目建设理念得到了众多农林高校的积极响应,并于2012年12月-2013年6月,分别在北京、扬州、武汉、哈尔滨、福州等地陆续召开了项目启动会议、研讨会和编写会议。2014年,项目成果“iCourse·教材:高等农林院校基础课程系列”陆续出版。

本系列教材涵盖数学、物理、化学化工、计算机、生物学等系列基础课程,在出版形式、编写理念、内容选取和体系编排上有不少独到之处,具体体现在以下几个方面:

1. 采用“纸质教材+数字课程”的出版形式。纸质教材与丰富的数字教学资源一体化设计,纸质教材内容精练适当,并以新颖的版式设计和内容编排,方便学生学习和使用;数字课程对纸质教材内容起到巩固、补充和拓展作用,形成以纸质教材为核心,数字教学资源配合的综合知识体系。

2. 创新教学理念,引导自主学习。通过适当的教学设计,鼓励学生拓展知识面和针对某些重要问题进行深入探讨,增强其独立获取知识的意识和能力,为满足学生自主学习和教师教学方法的创新提供支撑。

3. 强调基础课程内容与农林学科的紧密联系,始终抓住学生应用能力培养这一重要环节。教材和数字课程中精选了大量有实际应用背景的案例和习题,在概念引入和知识点讲授上也总是从实际问题出发,这不仅有助于提高学生学习基础课程的兴趣,也有助于加强他们的创新意识和创新能力。

4. 教材建设与资源共享课建设紧密结合。本系列教材是对各校精品资源共享课和教学改革成果的集成和升华,通过参与院校共建共享课程资源,更可支持各级精品资源共享课的持续建设。

建设切实满足高等农林教育教学需求、反映教改成果和学科发展、纸质出版与资源共享课紧密结合的新形态教材和优质教学资源,实现“校际联合共建,课程协同共享”是我们的宗旨和目标。将课程建设及教材出版紧密结合,采用“纸质教材+数字课程”的出版形式,是一种行之有效的方法和创新,得到了高校师生的高度认可。尽管我们在出版本系列教材的工作中力求尽善尽美,但难免存在不足和遗憾,恳请广大专家、教师和学生提出宝贵意见与建议。

高等教育出版社

2014年7月

前 言

C 语言以其简洁、高效、功能强并兼有高级语言和低级语言功能等特点,自 20 世纪 70 年代诞生以来,一直被广泛使用。因此,C 语言程序设计也被设为高等学校许多理工类学科学生的必修科目。

对于计算机语言的学习,学生除了要掌握语法规则,还应该学习并掌握程序设计的灵魂——算法设计的思想。作为一门实践性很强的计算机语言,只有通过大量的动手实践,才能将硬性的语法规则与算法相结合,设计出能解决实际问题的程序。为了有效地指导学生学习 C 语言,编者编写了与教材《C 语言程序设计教程》相配套的本实验教程,并配有数字资源。

除了第 1 章的实验前预备知识,其余各章遵照教材《C 语言程序设计教程》的内容组织安排,共安排了 12 章的内容。

本书中每章包括 3 个小题:本章要点、实验指导、习题及参考答案。第 1 节除了给出该章的学习重点和难点外,还将这一章的主要理论内容较为详细地归纳出来,特别对一些不易理解和易出现错误的地方给出了特别的说明,方便并加强学生的理解和记忆。第 2 节结合这一章的教学重点和难点,由易至难组织了阅读分析程序、程序改错、程序填空和编写程序几种类型的题目供学生上机实验完成,并对一些题目给出了解题思路或注意事项,以加深学生的理解和知识内容的拓展及延伸。第 3 节根据章节内容的不同,分别安排了单项选择题、填空题、阅读程序和编写程序几种题型。题目的组织遵照该章的各个知识点。单项选择题和填空题注重学生对基础知识和语法规则的理解;阅读程序注重学生对本章和之前章节知识结构的理解和掌握;编写程序题注重学生独立编写程序能力的提高。书中对于每一道编程题均给出了参考程序及运行结果,有的编程题目还给出了分析及提示,大部分参考程序采用交互式编写,从而提高程序的灵活性,并在程序中通过加入注释信息帮助学生。同时,因为编程题目解法不一,编者有意识地对于同一个题目给出了多种解法,以帮助学生分析、理解、掌握。在做编程题目时,建议学生提前独立编写程序,然后再与本书提供的参考程序及其答案对比,以发现不足之处并改进。程序设计类题目的解法很多,本书也仅给出了一种或几种解法,读者可以尝试用其他方法解决,举一反三,对提高学生的编程能力是很有裨益的。本书最后的附录,将学生编程过程中易出现的典型错误列出来,用于帮助初学者在编写和调试程序过程中改正错误。

参加本书编写工作的教师均是长期工作在教学第一线、从事主讲及指导实验经验丰富的教师。

本书由任洪娥、高心丹任主编,宋宏光、董本志任副主编。第1章、第2章、第3章、第7章、第12章、第13章由高心丹编写;第4章、第5章、第6章、第8章由宋宏光编写;第9章、第10章由董本志编写;数字资源由王海丰、高心丹编写制作;任洪娥负责其余部分的编写和全书的统稿。徐秋红、刘丽杰、蔡娟、尹淑欣、王丹也参与了本书编写。

限于作者水平和学识,书中难免有不足之处,恳请广大同行及读者不吝赐教。

最后,谨向每一位关心和支持本书编写工作的各方面人士表示感谢!

编 者

2015年4月

目 录

第 1 章 预备知识、开发环境介绍	1	第 5 章 数据的输入与输出	69
1.1 实验的目的	2	5.1 本章要点	70
1.2 实验的基本要求	2	5.1.1 C 语言语句	70
1.3 常用 C 语言运行环境介绍	8	5.1.2 数据输入/输出	71
1.3.1 Turbo C 2.0 集成开发环境	9	5.2 实验指导	72
1.3.2 Visual C++ 6.0 开发 环境介绍	17	5.3 习题及参考答案	75
第 2 章 计算机系统概述	27	第 6 章 结构化程序设计	83
2.1 本章要点	28	6.1 本章要点	84
2.2 实验指导	30	6.1.1 算法的概念及特征	84
2.3 习题及参考答案	39	6.1.2 算法的表示方法	84
第 3 章 基本数据类型	43	6.1.3 程序设计中的常用算法	85
3.1 本章要点	44	6.2 习题及参考答案	86
3.1.1 C 语言的数据类型	44	第 7 章 程序控制	93
3.1.2 标识符、常量和变量	44	7.1 本章要点	94
3.1.3 基本数据类型	46	7.1.1 关系运算符及关系表达式	94
3.2 习题及参考答案	48	7.1.2 逻辑运算符及逻辑表达式	94
第 4 章 运算符与表达式	53	7.1.3 选择结构控制语句	95
4.1 本章要点	54	7.1.4 循环控制语句	97
4.1.1 运算符及表达式	54	7.2 实验指导	99
4.1.2 不同数据类型的转换	56	7.3 习题及参考答案	114
4.2 实验指导	57	第 8 章 函数	145
4.3 习题及参考答案	60	8.1 本章要点	146
		8.1.1 概述	146

8.1.2 函数的定义和调用	146	11.2 实验指导	233
8.1.3 函数的递归调用	148	11.3 习题及参考答案	238
8.1.4 变量的作用域和生存期	149		
8.2 实验指导	150		
8.3 习题及参考答案	156	第 12 章 结构体、联合体和位运算	259
第 9 章 数组	177	12.1 本章要点	260
9.1 本章要点	178	12.1.1 结构体类型	260
9.1.1 一维数组	178	12.1.2 动态存储分配和链表	262
9.1.2 二维数组	179	12.1.3 共用体类型	263
9.1.3 字符数组与字符串	180	12.1.4 枚举类型	263
9.1.4 字符串处理函数	181	12.1.5 利用 typedef 自定义类型	264
9.1.5 数组与函数	182	12.2 实验指导	264
9.2 实验指导	183	12.3 习题及参考答案	269
9.3 习题及参考答案	191	第 13 章 文件	295
第 10 章 编译预处理	221	13.1 本章要点	296
10.1 本章要点	222	13.1.1 文件的相关概念	296
10.1.1 宏定义	222	13.1.2 文件的打开、关闭	297
10.1.2 文件包含	223	13.1.3 文件的读写	297
10.1.3 条件编译	224	13.1.4 文件处理的其他函数	299
10.2 习题及参考答案	224	13.2 实验指导	300
		13.3 习题及参考答案	302
第 11 章 指针	227	附录 A 常见上机错误信息	313
11.1 本章要点	228	A.1 严重错误	314
11.1.1 地址与指针的概念	228	A.2 一般错误	314
11.1.2 指针的定义与引用	228	A.3 警告信息	324
11.1.3 指针与数组	230	参考文献	329
11.1.4 指针数组和指向指针 的指针	231		
11.1.5 指针与函数	232		

第1章

预备知识、开发环境介绍

- 1.1 实验的目的
- 1.2 实验的基本要求
- 1.3 常用 C 语言运行环境介绍

1.1 实验的目的

上机实验是学习程序设计语言必不可少的实践环节,C 语言功能丰富、表达能力强,并且特别灵活、简洁,更是程序设计语言中一门实践性很强的计算机语言,需要通过编程实践才能真正掌握它。在学习过程中,检查学习好坏的标准,不是记住了多少语法规则,而是会不会使用,学习的目的全在于应用。因此,希望读者一定要重视实践环节,多上机练习,从而进一步提高 C 语言程序设计能力。对计算机语言的学习可以概括为以学习语法规则、掌握程序设计的方法为基础,提高程序的编写和开发能力为目的,这些都必须通过充分的实际上机操作才能完成。

作为计算机专业的软件基础课程,“C 语言程序设计”这门课程对提高学生的逻辑分析、抽象思维和程序设计能力,培养优良的程序设计风格具有很重要的作用,同时它也是后续多门课程的基础,学习效果的优劣,直接影响后续课程的学习及本专业专业能力的提高。

在课堂上要讲授许多关于 C 语言的语法规则,听起来十分枯燥无味,也不容易记住,死记硬背也不可取,只有通过多次上机练习,对于语法知识有了感性的认识,加深对它的理解,在理解的基础上就会自然而然地掌握 C 语言的语法规则。通过 C 程序设计实验,培养学生对学习计算机程序设计的兴趣,加深对讲授内容的理解,尤其是通过上机来掌握语法规则,使学生全面了解 C 语言的特点,熟练掌握 C 语言程序设计的基本方法和编程技巧。学习计算机语言的过程,应该是从上课听讲的学习语法和基本算法,到仿写同类型程序,再到编写程序的过程。因此,只有动手编写、上机实践才是学会编程的根本方法。

1.2 实验的基本要求

对于 C 语言程序设计的学习,要从了解和熟悉 C 语言程序开发的环境,学会上机调试程序开始,要善于发现程序中的错误,并且能很快地排除这些错误,使程序能正确运行,达到实验知识和理论知识的融会贯通。上机实验前,学生必须事先根据题目的内容要求进行分析,选择适当的算法并编写程序。上机前一定要仔细检查程序,直到找不到错误(包括语法错误和逻辑错误)。分析可能遇到的问题及解决的对策。准备几组测试程序的数据和预期的正确结果,以便发现程序中可能存在的错误。

在实验时输入程序、调试程序,直至运行结果正确为止。初学者切不可不经过思考、设计直接上机“现写”程序,盲目地实验既浪费了宝贵的时间,又无法达到提高逻辑思维能力和算法设计分析能力的目的。为了节省时间、提高效率,在上机实验时应做好以下准备。

1. 重视程序设计的方法和程序的分解

编写程序的目的是为了计算机能完成人们预定的任务,在程序设计过程中最为关

心的是程序的正确性和效率,而计算机是机械的,程序的正确性和效率的根本保证是程序的结构和算法。因此要完成人们的预定任务,首先必须为如何完成预定任务设定一个合理的算法,然后再根据算法去编写有良好结构的程序。

计算机要对完成任务的每个对象和处理规则给出正确详尽的描述,其中程序的数据结构和变量用来描述问题的对象,函数和语句用来描述问题的算法。算法和数据结构是程序的两个重要方面。算法是问题求解过程的精确描述,通常求解一个问题可能有多种算法可供选择,选择的主要标准首先是算法的正确性、可靠性、简易性和易理解性,其次是算法所需要的存储空间的大小和执行效率。算法设计是一项非常困难的工作,需要在实践过程中慢慢地积累和摸索,初学者刚刚开始学习程序设计,不可能一下子就掌握程序的算法设计方法,但对一些常用的或经典的算法应在理解的基础上掌握,并应具有在完成程序设计过程中,对完成的任务进行合理分析、尽量选取适当的算法的意识。

处理复杂问题的基本方法就是设法将其分解为一些相对简单的子问题,分别处理这些子问题,然后用各个子问题的解去构造整个问题的解。所以分析问题时,应考虑如何将复杂问题分解成简单问题的组合,如何进一步分解,在此基础上考虑如何用算法实现计算过程。要特别注意具有独立逻辑意义的片段,将它们孤立出来定义为函数。将程序适当分解为一组互相调用的函数是处理复杂问题的基本方法。一个函数不宜过长,有人曾建议一个函数的定义不应超过 100 行。一组较小的、本身有逻辑独立意义的模块的组合比一个大而复杂、没有良好结构的长程序更容易把握,容易写正确,出现了问题也更容易发现和改正。

2. 熟练使用集成开发工具

为了使程序的开发更加方便,人们开发了一类被称为集成式程序开发环境(Integrated Development Environment, IDE)的软件。许多计算机软件公司把 IDE 作为计算机语言处理系统的基本部分。IDE 把编写程序过程中需要的所有软件集成起来统一管理和使用,提供友好的编辑、调试和运行环境。通过 IDE 编写程序,开发过程中的各种工作变得非常方便,从而提高了工作效率。各种计算机系统都有很完善的 C 语言 IDE,编程者可以根据具体的情况选取适宜的 IDE。本书中所采用的 C 语言开发环境为 Turbo C 2.0 和 Visaul C ++6.0 集成开发环境(以下简称 VC 环境)。一个 C 语言程序从编辑、编译、连接到运行,都要在一定的外部操作环境下进行。所谓“环境”就是所用的计算机系统硬件、软件条件,只有学会使用这些环境,才能进行程序开发工作。通过上机实验,熟练地掌握 C 语言开发环境,为以后真正编写计算机程序解决实际问题打下基础。同时,在今后遇到其他开发环境时就会触类旁通,很快掌握新系统的使用方法。

3. 养成良好的编码习惯

一般来说,一个 C 语言程序由特定意义的字符序列组成,可以用普通的编辑器编写、修改。组成程序的字符序列通常按照人们习惯的阅读方式分行书写,因每一行的功能不同长度也各不相同。C 语言是一种“自由格式”程序设计语言,除了一些简单的限制,如预定义要单独定义一行等以外,程序员完全可以根据自己的想法和需要选择程序书写的

格式,确定在哪里分行,在哪里插入或增加空格等,格式的不同并不影响程序的功能和作用。但是并不是没有规定和约束,格式就不重要。

程序的一个重要作用是给人读,首先是编写者自己要读。对于阅读者而言,程序的格式非常重要。好的编码习惯可以大大提高程序代码的可读性,帮助编写者理清编程思路,提高程序的执行效率,从根本上提高系统的质量。初学者初始时就应该养成良好的编码习惯。经过多年的经验和程序设计实践,程序设计人员在编码风格上有了基本统一的认识:程序可能很长,结构可能很复杂,因此必须具有良好的习惯,所用格式应很好地表现程序的层次结构,反应程序中各个部分之间的结构;程序是由特定意义的字符序列构成的,在定义字符序列时应采用接近自然语言的字符序列,尽可能做到见名知意。具体规则如下。

(1) 排版。

① 程序文件的组成应按照如下的顺序排列:

```
#include <标准函数头文件>
#include <自定义函数头文件>
#define 预定义
全局变量定义
函数原型说明
main()函数定义
自定义函数的定义
```

② {}括号的排版方式。{}的前后两个语句一般不应出现在同一行上,同一个函数体的{},应在同一列上。

③ 采用锯齿状组织。函数、结构的定义以及 if、for、do、while、switch 及 case 等语句中的程序代码都应采用锯齿样的缩进格式排版,排版缩进一般为 4 个空格位。其中{}应各占一行并且位于同一列,同时引用它们的语句应缩进左侧对齐。例如,如下结构不符合规范:

```
for (...) {
...//程序代码
}
```

应用如下格式书写:

```
for (...)
{
...//程序代码
}
```

④ 在两个以上的关键字、变量、常量进行对等操作时,它们之间的操作符之前、之后或者前后要加空格;进行非对等操作时,如果是关系密切的立即操作符(如 →、++ 等),后不应加空格。用这种松散的方式编写代码的目的是使代码更清晰。例如,逗号、分号只在后面加空格:

```
printf(" %d% d% d", a, b, c);
```

比较操作符(>、<等)、赋值操作符(=、+=等)、算术运算符(+、%等)、逻辑运算符(&&、||等)、位运算符(<<、^等)等双目运算符的前后都应加空格,例如:

```
a = b + c;
```

单目运算符(!、~、++、--、&、*等)前后均不加空格,例如:

```
p = &num;
```

⑤ 语句行应一行只写一条语句,并且应在功能相对独立的语句块之间,以及 for、while、if 等语句前后适当加空行。例如:

```
for(i = 1, s = 0; i <= 100; i++)
    s += i;
```

```
printf(" %d", s);
```

⑥ 不同类型的运算符混合使用时,应用()给出优先级。例如:

```
if(year % 4 == 0 && year % 100 != 0 || year % 400 == 0)
```

是正确的代码,但如果加上小括号,则更清晰:

```
if((year % 4) == 0 && (year % 100) != 0 || (year % 400) == 0)
```

⑦ 若语句过长(一般超过 80 个字符),可分多行书写。分出的新行可以采用适当的缩进格式,以使排版整齐,可读性好。例如:

```
scanf(" %s% ld% d% lf% lf% lf", p -> stu_name, p -> stu_id, p -> course, p -> num,
      p -> stu_score1, p -> stu_score2, p -> stu_score3);
```

(2) 命名。

① 标识符的命名。标识符命名的基本原则是见名知意,要清晰、明了、有明确的含义,可以统一选用完整的英文单词、全拼或大家基本可以理解的缩写来命名。标识符的长度一般不应超过 32 个字符,当名字较长时可以采用缩写形式。缩写的基本规则是:去掉所有不在词头的元音字母,如 temp 可缩写为 tmp;使用每个单词的第一个或头几个字母,多个单词时通常使用下划线(_)来连接,如 student number 可以写为 stu_num。

② 变量的命名。变量命名时尽量不要用单个字母(一般循环变量除外,可以用 i、j、k 等作为循环变量),因为这样在程序中很容易输错,而且当程序比较长时,容易重复定义,同时编译时又不容易检查,引起不必要的错误。对于变量名的定义,习惯上除了表明含义外,还能表明作用域类型、数据类型等。例如,全局的基本整型变量用 g_int 开头,静态存储的字符型变量用 s_char 开头,指针变量一般用“p”或“pointer”开头,结构体变量用“struct”开头,数组用“array”开头等,便于识别变量的类型或意义,提高程序的可读性。

③ 宏和常量的命名。宏和常量的命名规则是使用下划线分隔的大写字母表示的单词或惯用字符。例如,声明圆周率 π , 可以用 PI; 声明最大值可以用 MAX。

④ 自定义函数的命名。自定义函数应用英文表示,比较常用的是两种命名习惯:一种是用下划线分隔的小写字母的合成词方式;另一种是用大写字母开头不带下划线的合

成词方式,两种方式一般均用动词开头,表明函数的功能。例如,函数的功能是根据姓名排序,则可以命名为sort_name 或 Sortname。

一般在一个程序系统中,命名规则应统一。

(3) 注释。注释的原则是有助于阅读理解程序,注释不宜过多也不应太少,太多会对阅读产生干扰,太少不利于对程序代码的理解,因此在程序中应只在必要的地方才加注释。注释要准确、易懂、尽可能简洁。

① 程序的头部应加注释。注释内容为对程序的基本说明,对于初学者可以列出程序的功能、内容等,在后续学习或实践实习,需要多人合作时,应列出作者名和日期等信息。

② 函数头部应进行注释,列出函数的目的/功能、输入参数、输出参数、返回值、调用关系(函数、表)等。例如:

```
/* 完成对所有学生的总分的降序排序,其中 N 为学生的总人数 */
void sort_score(float score[], int)
```

③ 注释应与其描述的代码相近,对代码的注释应放在其上(对一段代码的注释)或右方(对单条语句的注释)相邻位置,不可放在下面。若放在上方,应与其上面的代码用空行隔开。

④ 对全局变量的含义、作用、取值范围及使用方法等应有明确的注释。例如:

```
/* 全局变量符号、含义如下、变量的作用、意义 */
/* 0 - 退出 1 - 成功 2 - 错误 */
```

⑤ 对程序中相对重要的代码段的功能、意图进行注释,提供有用的、额外的信息,并在该代码段的结束处加一行注释表示该段代码结束。

另外,在编写程序时,尽量不要用难懂的技巧性较高的语句,除非很有必要。高技巧语句不等于高效率的程序,实际上程序的效率关键在于算法。

例如:

```
* point_num ++ += 1;
```

应写为

```
* point_num += 1;
```

```
* point_num ++;
```

后者更容易理解。

下面的代码

```
while( ( scanf( "%d", &n) ) != EOF )
{
    ... //程序代码
}
```

可以用

```
while(1)
{
    k = scanf( "%d", &n );
```