

可下载教学资料

<http://www.tup.tsinghua.edu.cn>

21 世纪普通高校计算机公共课程规划教材

C 程序设计基础

李瑞 戚海英 徐克圣 刘月凡 编著

清华大学出版社



21世纪普通高校计算机公共课程规划教材

TP312/2927

2008

C 程序设计基础

李瑞 戚海英 徐克圣 刘月凡 编著

清华大学出版社
北京

内 容 简 介

全书分为 8 章,包括程序设计思想、数值处理、逻辑处理、数据组织、编程模块化思想、指针、文件以及综合设计。本书知识覆盖面广,例题多而丰富。

通过大量知识点明确的例题,让读者更好地掌握程序设计方法,强调实践中学习,每章均配有上机实践训练。

本书可以作为高等院校计算机专业本科、专科低年级学生学习计算机语言的入门教材,还可以作为科技人员自学 C 语言的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C 程序设计基础/李瑞等编著. —北京:清华大学出版社,2008.7

(21 世纪普通高校计算机公共课程规划教材)

ISBN 978-7-302-17326-7

I. C… II. 李… III. C 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(2008)第 049199 号

责任编辑:梁 颖 赵晓宁

责任校对:梁 毅

责任印制:李红英

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

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

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京国马印刷厂

装 订 者:三河市李旗庄少明装订厂

经 销:全国新华书店

开 本:185×260 印 张:16.5 字 数:401 千字

版 次:2008 年 7 月第 1 版 印 次:2008 年 7 月第 1 次印刷

印 数:1~4000

定 价:25.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770177 转 3103 产品编号:028159-01

出版说明

随着我国改革开放的进一步深化,高等教育也得到了快速发展,各地高校紧密结合地方经济建设发展需要,科学运用市场调节机制,加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度,通过教育改革合理调整和配置了教育资源,优化了传统学科专业,积极为地方经济建设输送人才,为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是,高等教育质量还需要进一步提高,以适应经济社会发展的需要,不少高校的专业设置和结构不尽合理,教师队伍整体素质亟待提高,人才培养模式、教学内容和方法需要进一步转变,学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007年1月,教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》,计划实施“高等学校本科教学质量与教学改革工程(简称‘质量工程’)”,通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容,进一步深化高等学校教学改革,提高人才培养的能力和水平,更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中,各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势,对其特色专业及特色课程(群)加以规划、整理和总结,更新教学内容、改革课程体系,建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上,经教育部相关教学指导委员会专家的指导和建议,清华大学出版社在多个领域精选各高校的特色课程,分别规划出版系列教材,以配合“质量工程”的实施,满足各高校教学质量和教学改革的需要。

本系列教材立足于计算机公共课程领域,以公共基础课为主、专业基础课为辅,横向满足高校多层次教学的需要。在规划过程中体现了如下一些基本原则和特点。

(1) 面向多层次、多学科专业,强调计算机在各专业中的应用。教材内容坚持基本理论适度,反映各层次对基本理论和原理的需求,同时加强实践和应用环节。

(2) 反映教学需要,促进教学发展。教材要适应多样化的教学需要,正确把握教学内容和课程体系的改革方向,在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养,为学生的知识、能力、素质协调发展创造条件。

(3) 实施精品战略,突出重点,保证质量。规划教材把重点放在公共基础课和专业基础课的教材建设上;特别注意选择并安排一部分原来基础比较好的优秀教材或讲义修订再版,逐步形成精品教材;提倡并鼓励编写体现教学质量和教学改革成果的教材。

(4) 主张一纲多本,合理配套。基础课和专业基础课教材配套,同一门课程有针对不同层次、面向不同专业的多本具有各自内容特点的教材。处理好教材统一性与多样化,基本教材与辅助教材、教学参考书,文字教材与软件教材的关系,实现教材系列资源配套。

(5) 依靠专家, 择优选用。在制定教材规划时要依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时, 要引入竞争机制, 通过申报、评审确定主题。书稿完成后要认真实行审稿程序, 确保出书质量。

繁荣教材出版事业, 提高教材质量的关键是教师。只有建立一支高水平教材编写梯队才能保证教材的编写质量和建设力度, 希望有志于教材建设的教师能够加入到我们的编写队伍中来。

21 世纪普通高校计算机公共课程规划教材编委会

联系人: 梁颖 liangying@tup.tsinghua.edu.cn

前言

计算机程序设计基础是一门十分重要的基础课程，是大学生学习计算机程序设计的入门课程，由于该课程开设历史较长，在开设之初，一直沿用一种传统的理论研究式的教学模式，过于注重计算机语言的语法、语句格式的讲解，没有把计算机语言本身的目标是编程的逻辑思想放在主体地位上；对学生编程的能力训练力度不够，这样给后续课程的学习和研究留下了隐患。很多学生在学习这门课程时感到枯燥难学，学过之后又不能用来解决实际问题。

作为从事计算机基础教学多年的教学团队，我们在教学中越来越感到原有的教学模式和方法已经不能适用于今天的计算机语言课程的教学要求，通过一线教学工作者长期的教学研究和总结经验，通过参加有关计算机基础教学研究会议，以及与其他高校从事计算机基础教学的同行们的交流，大家都感到有必要改变我们的课程教学模式，用新的教学理念和方法培养新时代人才。目前，C 语言程序设计的课程建设工作，学校给予高度重视。我们通过学习研究清华大学等院校的改革经验，在精品课的课程建设中，开始对 C 语言程序设计课程的教学模式进行改革，以强调将实践上机为切入点。以任务驱动方式，通过实例讲授程序设计的基本概念和基本方法，重点放在学习编程思路上。要求学生养成良好的编程习惯；在教学过程中注重培养学生的计算机语言的思维能力和编程动手能力，鼓励学生探索、研究和创新。在指导思想上，强调转变观念，以学生为中心，将学生视为教学的主体，安排教学首先要考虑培养目标、学生的认知规律和学习特点。具体的教学改革措施主要为以下两点：教学模式和方法的改革；学生学习评价体系的改革。

对教学模式的改革：主要是从软件环境上进行改革，包括教学方法，思路，手段的改革。转变观念，把强化实践思想提到一定的高度上予以重视。

对学生评价体系的改革：考试是检验学生学习成果的重要环节。考试作为指挥棒对教学目标和教学过程都有着重大的影响。对于 C 语言课程建设来说，考试改革是调动和激发学生积极性、创造性的重要环节。如果对学生的考核是采取上机考核的话，对学生学习方式方法的影响是很大的，也是积极的。作为计算机语言课的学习，只有动手，动脑去实践，才能学到真本事。这就要求从硬件环境上以及软件的配置上，都要加大投入。因此，C 语言程序设计课程建设不是一朝一夕的事情，它是个系统工程，需要逐步完成。

而 C 语言程序设计的教材，目前绝大多数都是为了适应早期的教学模式和教学方法而编写的，如果要进行教学改革，首先教材编写风格必须改革。为了减少风险，我们查阅了许多相关的资料和教材，也借鉴了许多专家的成功经验和思想，经过广泛讨论和研究才编写了本书。

本书由大连交通大学的李瑞、威海英、徐克圣和刘月凡 4 位同志参加编写工作, 其中第 1 和第 2 章由李瑞编写, 第 3 和第 4 章由刘月凡编写, 第 5、第 7 和第 8 章由威海英编写, 第 6 章由徐克圣编写, 附录由徐克圣整理, 全书由李瑞统稿和审定, 威海英在排版, 整理过程中做了许多工作, 潘宇和康良玉等同学也参与了本书的一些编写工作, 在此表示感谢。

为方便阅读, 本书中的变量均为正体。

由于时间仓促以及作者的水平有限, 书中难免有纰漏, 欢迎广大读者多提宝贵意见。

编 者

2008 年 5 月

目 录

第1章 程序设计思想	1
1.1 程序设计	1
1.1.1 程序设计的基本步骤	1
1.1.2 程序设计的学习方法	1
1.2 算法	2
1.2.1 算法概念	2
1.2.2 算法的特性	3
1.2.3 算法的表示	3
1.2.4 算法的复杂度	6
1.2.5 结构化程序设计方法	6
1.2.6 算法举例	8
1.3 编程准备	9
1.3.1 Turbo C 编程开发环境	9
1.3.2 VC++编程开发环境	13
1.3.3 实例运行过程	13
1.4 上机实践	16
第2章 数值处理	18
2.1 程序的基本结构	18
2.2 数据类型	19
2.3 标识符、常量和变量	20
2.3.1 标识符	20
2.3.2 常量和变量	21
2.3.3 整型数据	22
2.3.4 实型数据	24
2.3.5 字符型数据	26
2.4 变量赋值	28
2.4.1 变量赋初值	28
2.4.2 赋值运算符和赋值表达式	28
2.4.3 运算符的优先级和结合性	29
2.5 算术运算符和算术表达式	30

2.5.1	算术运算符	30
2.5.2	算术表达式	31
2.5.3	算术运算符优先级和结合性	31
2.5.4	算术运算中的类型转换	32
2.6	位运算符、逗号运算符和求字节运算符	33
2.6.1	位运算符	33
2.6.2	逗号运算符	35
2.6.3	求字节运算符	36
2.7	基本输入输出函数	36
2.7.1	格式化输出函数——printf	37
2.7.2	格式化输入函数——scanf	40
2.7.3	字符输出函数——putchar	41
2.7.4	字符输入函数——getchar	41
2.8	编译预处理	42
2.8.1	宏定义	42
2.8.2	文件包含	44
2.8.3	条件编译	46
2.9	程序设计举例	47
2.10	上机实践	48
习题 2		49
第 3 章	逻辑处理	54
3.1	关系运算符和关系表达式	54
3.1.1	关系运算符	54
3.1.2	关系表达式	54
3.2	逻辑运算符和逻辑表达式	55
3.2.1	逻辑运算符	55
3.2.2	逻辑表达式	55
3.3	语句和复合语句	56
3.4	分支结构	56
3.4.1	if 语句	56
3.4.2	switch 语句	59
3.4.3	条件运算符	61
3.5	循环结构	61
3.5.1	while 语句	62
3.5.2	do-while 语句	63
3.5.3	for 语句	65
3.5.4	循环语句的嵌套	65
3.5.5	break 语句和 continue 语句	66

3.6	程序设计举例	67
3.7	上机实践	69
	习题 3	74
第 4 章	数据组织	80
4.1	数组	80
4.1.1	一维数组	80
4.1.2	二维数组	83
4.1.3	字符数组	85
4.2	结构体	89
4.2.1	结构体变量	89
4.2.2	结构体数组	92
4.3	共用体	92
4.4	枚举类型	94
4.5	typedef 自定义类型	96
4.6	程序设计举例	97
4.7	上机实践	99
	习题 4	104
第 5 章	编程模块化思想	110
5.1	函数	110
5.1.1	函数概述	110
5.1.2	函数的定义	111
5.1.3	函数的调用与参数	112
5.1.4	对被调用函数的声明	114
5.1.5	函数的返回值与函数类型	116
5.1.6	函数的参数传递	117
5.1.7	函数的嵌套调用和递归调用	121
5.2	局部变量和全局变量	124
5.2.1	局部变量	124
5.2.2	全局变量	125
5.3	变量的存储类别	128
5.3.1	局部变量的存储	129
5.3.2	全局变量的存储	131
5.4	内部函数和外部函数	133
5.4.1	内部函数	133
5.4.2	外部函数	133
5.4.3	多文件编译	134
5.5	程序设计举例	135

5.6 上机实践	138
习题 5	140
第 6 章 指针	148
6.1 指针的概念	148
6.2 变量与指针	149
6.2.1 指针变量的引用	150
6.2.2 指针变量作为函数参数	151
6.3 数组与指针	153
6.3.1 指向数组元素的指针	153
6.3.2 通过指针引用数组元素	153
6.3.3 用数组名作函数参数	154
6.3.4 二维数组与指针	156
6.4 字符串与指针	158
6.4.1 字符串的表示形式	158
6.4.2 字符指针作函数参数	160
6.5 函数与指针	161
6.5.1 用函数指针变量调用函数	161
6.5.2 用指向函数的指针作函数参数	162
6.5.3 返回指针值的函数	163
6.6 指针数组与二级指针	164
6.6.1 指针数组的概念	164
6.6.2 二级指针	166
6.6.3 主函数与命令行参数	166
6.7 结构与指针	167
6.7.1 指向结构体变量的指针	167
6.7.2 指向结构体数组的指针	168
6.7.3 用指向结构体的指针作函数参数	169
6.8 链表	170
6.8.1 动态分配和释放空间的函数	170
6.8.2 建立和输出链表	171
6.8.3 链表的基本操作	173
6.9 程序设计举例	176
6.10 上机实践	180
习题 6	182
第 7 章 文件	190
7.1 文件的概念	190
7.2 文件的打开和关闭	192

7.2.1	文件的打开	192
7.2.2	文件的关闭	193
7.3	文件的读写	193
7.3.1	读写一个字符的函数	194
7.3.2	读写字符串函数	196
7.3.3	读写一个数据块函数	197
7.3.4	文件格式化读写函数	199
7.4	文件的定位	200
7.5	程序设计举例	203
7.6	上机实践	205
	习题 7	206
第 8 章	综合设计	210
8.1	学生成绩管理系统	210
8.1.1	需求分析	210
8.1.2	总体设计	211
8.1.3	详细设计与实现	212
8.1.4	参考程序	215
8.2	通信录系统	223
8.2.1	需求分析	223
8.2.2	总体设计	224
8.2.3	详细设计	225
8.2.4	参考程序	227
8.3	贪吃蛇游戏	234
8.3.1	需求分析	234
8.3.2	总体设计	234
8.3.3	详细设计与实现	235
8.3.4	参考程序	237
附录 A	常用字符与 ASCII 代码对照表	242
附录 B	运算符的优先级和结合性	243
附录 C	库函数	244
	参考文献	250

第1章

程序设计思想

程序设计通俗地说就是完成一件事情时对步骤的安排。人们平时每做一件事情，其实都存在程序设计的思想在里面。比如要举行一次会议，也要筹划、安排会议的步骤。这就是程序设计。程序设计思想就是这样的。而计算机程序设计则是指在计算机上完成一件事情的过程。通常说完成一件事情，就是解决问题。这里所说的问题，不是平时所说的问题，而是要解决的一个任务，要完成的一件事情。也就是说，计算机程序设计：就是通过计算机解决问题的过程。这里面实际上有两个层面的问题，首先是解决问题的方法和步骤；其次是如何把解决问题的方法和步骤通过计算机来实现。要想在计算机上完成这个任务，得用计算机语言来完成，就如同和英国人说话要用英语，和日本人说话要用日语一样，和计算机说话要用计算机语言。

有一个著名的计算机程序设计（以后简称“程序设计”）的公式：

程序设计=算法+数据结构+计算机语言

其实，初学者要想更容易上手的话，可以从算法和计算机语言上掌握程序设计。也就是说，初学者了解程序设计，只要了解算法和计算机语言，就可以进行程序设计工作了。

1.1 程 序 设 计

程序设计（programming）是指设计、编制、调试程序的方法和过程。上面已经说过，对于初学者，了解程序设计可以把解决问题的方法与步骤和在计算机上实现这个过程分开来考虑。解决问题的方法与步骤，就是所谓的算法。把算法在计算机上实现，也就完成了程序设计的过程。从这个过程来看，算法是程序的核心，是程序设计要完成的任务的灵魂。初学者可以只考虑这样的公式：

程序设计=算法+计算机语言

1.1.1 程序设计的基本步骤

程序设计其实最终就是利用计算机来解决问题，完成任务。

（1）把解决问题的方法与步骤设计完成，即算法设计完成。

（2）把算法在计算机上用计算机语言实现。

（3）调试编辑好的程序。这也是程序设计思想之一，人们完成的程序设计不可能一次就成功，就是再天才的人，思维再缜密的人，也不可能保证自己编的程序没有错误。

1.1.2 程序设计的学习方法

从程序设计的基本步骤上可以看出，要想学好程序设计，首先要了解和掌握算法的概

念,然后再学习一门计算机语言,这样,才可以初步完成在计算机上进行程序设计的工作。本章主要介绍算法的概念和思想。从第2章开始详细学习C语言(计算机语言),通过学习并使用C语言来完成计算机程序设计工作,学习计算机语言的目的最终是要进行程序设计,学习计算机语言的语法规则的目的是为了更好地掌握计算机语言。

目前的计算机语言已经从低级语言发展成为高级语言了,高级语言更方便于用户使用,它的源代码都是文本型的。但是,计算机本身只能接受二进制编码的程序,它不能直接运行这种文本型的代码,需要通过一个翻译把高级语言源程序代码转换成计算机能识别的二进制代码,这样计算机才能执行。而这个翻译,在这里把它叫做“编译系统”,也可以看成是计算机语言的编程界面。

在这章,先介绍一下算法的概念和思想,然后再介绍计算机语言的上机环境,也就是C语言的编译系统。目前大家比较喜欢使用的C语言编译系统有Turbo C和VC++环境。Turbo C简单灵活,适合初学者掌握,VC++是Windows系统下的编程环境,界面友好。

1.2 算 法

算法是解决问题的方法与步骤,比人们平时理解的数学中算法的概念要广义一些。算法是程序的核心,是程序设计要完成的任务的灵魂。不论是简单还是复杂的程序,都是由算法组成。算法不仅构成了程序运行的要素,更是推动程序正确运行,实现程序设计目的的关键。

1.2.1 算法概念

当要买东西时,就会先有目标,然后到合适的商店挑选想要的物品,然后结账,拿发票(收据)、离开商店;当要理发时,就会先到一家理发店,与理发师商量好发型、理发、结账;当要使用计算机时,就会先打开屏幕、开机、输入密码,然后使用。不论什么事情,都要有一定的步骤。算法(algorithm)简单来说就是解题的步骤,算法是程序设计的“灵魂”,它独立于任何具体的程序设计语言,一个算法可以用多种编程语言来实现。算法是一组有穷的规则,它们规定了解决某一特定类型问题的一系列运算,是对解题方案的准确与完整的描述。在程序设计中,算法要用计算机算法语言描述出来,算法代表用计算机解决一类问题的精确、有效的方法。

【例 1.1】 输入三个互不相同的数,求其中的最小值(min)。

首先设置一个变量min,用于存放最小值。当输入a、b、c三个不相同的数后,先将a与b进行比较,把相对小的数放入min,再把c与min进行比较,若c小于min,则将c的数值放入min替换min中的原值,若c大于min,则min值保持不变,最后min中就是三个数中的最小值。详细步骤如下:

- ① 先将a与b进行比较,若 $a < b$,则 $a \rightarrow \min$,否则 $b \rightarrow \min$;
- ② 再将c与min进行比较,若 $c < \min$,则 $c \rightarrow \min$ 。

这样,min中存放的就是三个数中的最小数。

求解一个给定的可计算或可解的问题,不同的人可以编写出不同的算法,来解决同一个问题。例如计算 $1999+2999+3999+\cdots+9999$,也许有的人会选择一个一个加起来,当然

也有人会选择 $(2000-1) + (3000-1) + \cdots + (10000-1)$ 的算法。理论上,不论有几种算法,只要逻辑正确并能够得出正确的结论就可以,但是,为了节约时间、运算资源等,当然提倡简单易行的算法。制定一个算法,一般要经过设计、确认、分析、编码、测试、调试、计时等阶段。

对算法的研究主要应包括5方面的内容:

(1) 设计算法。算法设计工作的完全自动化是不现实的,算法的设计最终还是要自己来完成,应学习和了解已经被实践证明可行的一些基本的算法设计方法,这些基本的设计方法不仅适用于计算机科学,而且适用于电气工程、运筹学等任何相关的领域。

(2) 表示算法。算法的类型不同,解决的问题不同,解决问题的步骤不同,表示算法的方法也自然有很多种形式,例如自然语言、图形、算法语言等。这些表示方式,各有特色,也分别有适用的环境和特点。

(3) 认证算法。算法认证其实就是确认这种算法能够正确的工作,达到解决问题的目的,即确认该算法具有可行性。正确的算法用计算机算法语言描述,构成计算机程序,计算机程序在计算机上运行,得到算法运算的结果。

(4) 分析算法。对算法进行分析,确认这个算法解决问题所需要的计算时间和存储空间,并对其进行定量分析。对一个算法的分析可以很好地预测一种算法适合的运行环境,从而判断出其适合解决的问题。

(5) 验证算法。用计算机语言将算法描述出来,进行运行、测试、调试,客观地判断算法的实际应用性、合理性。

1.2.2 算法的特性

一个算法应当具有以下5方面性质:

(1) 确定性。与日常的行为不同,算法绝对不能有含糊其辞的步骤,像“请把那天的书带来!”,这种无法明确哪一天、哪一本书、带到哪里的语句是不能够出现在算法中的,否则,算法的运行将变得无所适从。算法的每一步都应当是意义明确,毫不模糊的。

(2) 可行性。算法的基本目的是解决问题,所以要求算法至少是可以运行并能够得到确定的结果的,不能存在违反基本逻辑的步骤。

(3) 输入。一个算法有0个或多个输入,在算法运算开始之前给出算法所需数据的初值,这些输入取自特定的对象集合。

(4) 输出。作为算法运算的结果,一个算法产生一个或多个输出,输出是同输入有某种特定关系的量。

(5) 有穷性。一个算法应包含有限个操作步骤,而不能是无限的。一个算法总是在执行了有限步骤的运算后终止,即该算法是可达的。

满足前4个特性的一组规则不能称为“算法”,只能称为“计算过程”,操作系统是计算过程的一个例子,操作系统用来管理计算机资源,控制作业的运行,没有作业运行时,计算过程并不停止,而是处于等待状态。

1.2.3 算法的表示

由于算法的步骤繁简不同,解决的问题不同,算法的表示方法也有许多种,一般可以

归纳为自然语言、图形和伪代码三种。

1. 自然语言表示

自然语言，简单来说就是通常日常生活中应用的语言。相对于计算机语言来说，自然语言更容易被接受，也更容易学习和表达，但是自然语言往往冗长烦琐，而且容易产生歧义。例如：“他看到我很高兴。”便不清楚是他高兴，还是我高兴。尤其是在描述分支、循环算法时，用自然语言十分不方便。所以，除了一些十分简单的算法外，一般不采用自然语言来表示算法。

2. 图形表示

用图形表示算法即用一些有特殊意义的几何图形来表示算法的各个步骤和功能。使用图形表示算法的思路是一种很好的方法，因为千言万语不如一张图明了易懂，图形的表示方法比较直观、清晰，易于掌握，有利于检查程序错误，在表达上也克服了产生歧义的可能。一般我们使用的比较多的有传统流程图、N-S 流程图、PAD 流程图等。本书只介绍传统流程图，其他流程图请参看其他程序设计书籍。

传统的流程图一般由如图 1.1 所示的几种基本图形组成。

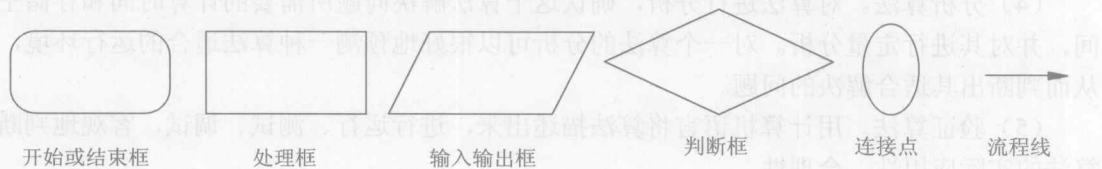


图 1.1

【例 1.2】 输入两个整数给变量 x 和 y，交换 x 和 y 的值后再输出 x 和 y 的值。

分析：完成本题需要 3 个步骤，首先输入两个整数给 x 和 y；之后交换 x 和 y 的值；最后输出 x 和 y。

根据以上分析，容易画出程序的流程图（如图 1.2 所示），并根据流程图写出程序：

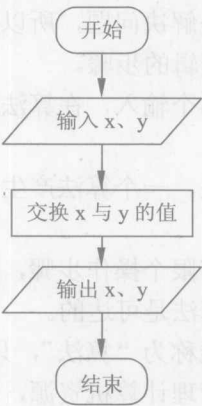


图 1.2

main()

```

int w,x,y;
printf("请输入两个整数: ");
scanf("%d%d",&x,&y);
w=x;
x=y;
y=w;
printf("交换后: x=%dy=%d\n",x,y);
}

```

说明:

- ① scanf 函数为输入函数，可以用来输入数据；输出数据可以使用 printf 函数。
- ② 引入第三个变量 w，先把变量 x 的值赋给变量 w，再把变量 y 的值赋给 x，最后把变量 w 的值赋给 y，最终达到交换变量 x 和 y 的目的。引入 w 的作用是交换变量 x 和 y 的值。交换 x 和 y 的值不能简单地用 “x=y;” 和 “y=x;” 这两个语句，如果没有把 x 的值保存到其他变量，就执行 “x=y;” 语句，把 y 的值赋给 x，将使 x 和 y 具有相同的值，丢失 x 原来的值，也就无法实现两个值的交换。

假如输入 3 和 9，运行程序时，屏幕可能显示如下信息：

请输入两个整数：3 9

交换后：x=9 y=3

【例 1.3】输入 a、b、c 三个数，把最小的值输出，流程图如图 1.3 所示。

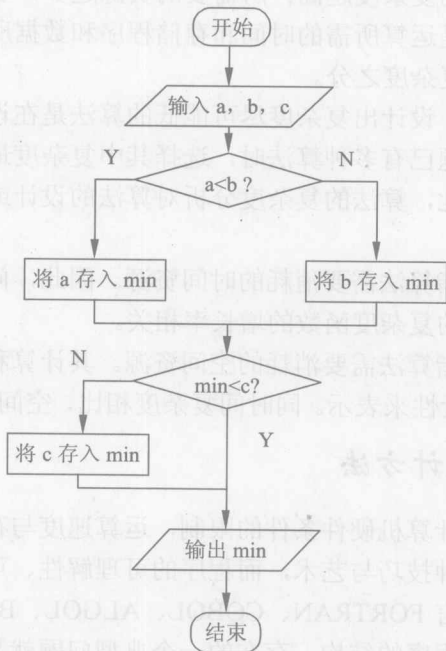


图 1.3

先将 a 与 b 进行比较，若 $a < b$ ，则将 a 存入 min，否则将 b 存入 min；再将 c 与 min 进行比较，若 $c < \min$ ，则将 c 存入 min，然后输出 min，否则直接输出 min。