



高等院校计算机系列规划教材

C语言程序设计教程

冉崇善 主编



机械工业出版社
CHINA MACHINE PRESS

免费提供电子教案

下载网址 <http://www.cmpedu.com>



高等院校计算机系列规划教材

C 语言程序设计教程

主编 冉崇善

参编 白涛 侯涛 杨悦欣 吉涛



机械工业出版社

本书是为高等院校程序设计课程编写的教材。主要内容包括：程序设计与算法；数据类型、运算符与表达式；顺序结构程序设计；分支结构程序设计；循环结构程序设计；数组；函数；预处理命令与位运算；指针；结构体与共用体；文件。书后有附录，每章后都有习题。

本教材在结构上本着以程序设计为中心，以语言知识为工具对 C 语言的语法规则进行了整理和提炼，深入浅出地介绍了 C 语言在程序设计中的应用；在内容上注重知识的完整性，以满足初学者的需求；在写法上遵循循序渐进的原则，实例讲解，通俗易懂。

本书可供计算机专业的本科生、大专生使用，也可作为各大学计算机教材或全国计算机等级考试参考书。

图书在版编目(CIP)数据

C 语言程序设计教程/冉崇善主编. —北京:机械工业出版社,2009. 11
(高等院校计算机系列规划教材)

ISBN 978-7-111-29007-0

I. C… II. 冉… 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(2009)第 202846 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

责任编辑:赵 轩

责任印制:洪汉军

三河市宏达印刷有限公司印刷

2009 年 12 月第 1 版·第 1 次印刷

184mm×260mm·21.5 印张·530 千字

0001-3000 册

标准书号: ISBN 978-7-111-29007-0

定价: 35.00 元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

电话服务

网络服务

社服务中心:(010)88361066

门户网:<http://www.cmpbook.com>

销售一部:(010)68326294

教材网:<http://www.cmpedu.com>

销售二部:(010)88379649

读者服务部:(010)68993821

封面无防伪标均为盗版

前 言

C 语言是一种通用的程序设计语言。它的结构简单，数据类型丰富，运算灵活方便。用它编写的程序，具有速度快、效率高、代码紧凑和可移植性好等优点，适用于开发各种系统软件和应用软件，是当今广为流行的一种计算机语言。

针对许多人想学 C 语言而又感到 C 语言难学的实际情况，作者在总结多年从事高校 C 语言教学经验的基础上，参照全国计算机等级考试二级 C 考试大纲，编写了本书。本书以实际应用为出发点，通过理论与范例相结合，系统地讲述了 C 语言程序设计应掌握的基本编程技能和必要的理论知识。将程序设计语言与最基本的算法相结合，是有效地避免由于重语法规则、轻算法概念，而导致学生由于基本技能缺乏而在后续课程的学习中产生困难等情况的发生。

本书对关键知识点进行了详细的说明，并附有大量的图表使读者能正确、直观地理解问题。实例程序由浅入深，强化知识点、算法、编程方法与技巧，并给出了详细的解释。

本书的编写思路及写作特色是：对每个重要知识点从相关知识、程序范例、知识归纳和实例验证四个方面进行编写，便于读者边学边练。

本教材由陕西科技大学电气与信息工程学院冉崇善教授拟定编写大纲和框架结构，并编写了第 3、7、9 和 11 章，白涛编写了第 2、6 章、侯涛编写了第 4、10 章、杨悦欣编写了第 5 章和附录、吉涛编写了第 1、8 章，全书由冉崇善统一定稿。

本书的作者都是长期从事高校计算机软件教学的一线教师，有丰富的教学经验和科研开发能力，书中文字流畅、通俗易懂、概念清楚、深入浅出、例题丰富、实用性强。本书可供计算机专业的本科生、大专生、学生使用，也可以作为大学计算机教材或全国计算机等级考试参考书。

欢迎读者对本书内容提出批评和修改意见。

编 者
2009 年 8 月

目 录

前言

第1章 程序设计与算法	1
1.1 计算机语言和程序	1
1.1.1 计算机语言的概念	1
1.1.2 C语言的特点	1
1.1.3 程序的概念	2
1.2 算法	2
1.2.1 算法的概念	3
1.2.2 算法的特性	3
1.2.3 算法的设计	3
1.2.4 算法的描述方法	5
1.3 数据结构	11
1.3.1 概述	11
1.3.2 数学模型	11
1.3.3 数据结构的概念	11
1.4 程序设计	12
1.4.1 程序设计的概念	12
1.4.2 养成良好的编程习惯	12
1.4.3 程序设计简介	12
1.4.4 程序设计的基本过程	13
1.5 C语言的程序结构	14
1.5.1 基本程序结构	14
1.5.2 源程序的基本结构特点与书写规则	16
1.5.3 函数库和链接	16
1.5.4 开发一个C程序的步骤	17
1.6 C语言上机环境 (Visual C++6.0)	18
习题一	21
第2章 数据类型、运算符与表达式	23
2.1 基本字符、标识符和关键字	23
2.1.1 基本字符	23
2.1.2 标识符 (名字) 的构成	24
2.1.3 关键字	24
2.2 数据类型与变量	25
2.2.1 数据类型	25

2.2.2 常量	27
2.2.3 变量	29
2.3 基本类型与数据表示	31
2.3.1 整数类型和整数的表示	31
2.3.2 实数类型和实数的表示	33
2.3.3 字符类型和字符的表示	35
2.4 运算符与表达式	38
2.4.1 算术运算	38
2.4.2 关系运算与逻辑运算	43
2.4.3 赋值运算	46
2.4.4 逗号运算	47
2.4.5 条件运算	48
2.5 计算和类型转换	50
2.5.1 类型对计算的限制	50
2.5.2 混合类型运算和类型自动转换（隐式转换）	51
2.5.3 强制类型转换	52
2.5.4 赋值转换	53
习题二	53
第3章 顺序结构程序设计	55
3.1 C 语句概述	55
3.1.1 C 程序的结构	55
3.1.2 赋值语句	56
3.2 数据的输入/输出	57
3.2.1 数据输入/输出的概念及其在 C 语言中的实现	57
3.2.2 数据的格式化输入	58
3.2.3 数据的格式化输出	66
3.2.4 字符数据的输入/输出	70
3.2.5 字符串数据的输入/输出	71
3.3 顺序结构程序设计举例	73
习题三	75
第4章 分支结构程序设计	79
4.1 if 语句	79
4.2 if 语句的嵌套	83
4.3 switch 语句	86
4.4 分支程序设计举例	89
习题四	93
第5章 循环结构程序设计	97
5.1 while 语句	97
5.2 do-while 语句	99

5.3	for 语句	102
5.3.1	for 语句的一般形式及其执行过程	102
5.3.2	for 语句的应用形式	102
5.4	循环的嵌套	105
5.5	goto 语句及使用 goto 语句构成循环	108
5.6	循环控制	109
5.6.1	break 语句	109
5.6.2	continue 语句	110
5.7	典型算法举例	111
5.7.1	递推法	111
5.7.2	迭代法	114
5.7.3	枚举法	116
5.7.4	取整数的各位数字	116
5.7.5	其他问题	118
5.8	循环程序设计举例	121
	习题五	125
第6章	数组	130
6.1	一维数组的定义和引用	130
6.1.1	一维数组的定义	130
6.1.2	一维数组元素的引用	131
6.1.3	一维数组的初始化	133
6.1.4	一维数组应用	134
6.2	二维数组的定义和引用	142
6.2.1	二维数组的定义	142
6.2.2	二维数组元素的引用	143
6.2.3	二维数组的初始化	144
6.2.4	二维数组的应用	146
6.3	字符数组	154
6.3.1	字符数组的定义	154
6.3.2	字符数组的初始化	155
6.3.3	字符数组的引用	155
6.3.4	字符串和字符串结束标志	156
6.3.5	字符数组的输入/输出	156
6.3.6	字符串处理函数	158
6.3.7	字符数组应用	161
6.4	数组程序设计举例	164
	习题六	170
第7章	函数	175
7.1	概述	175

7.2	函数定义的一般形式	176
7.3	函数的参数和函数的值	178
7.3.1	形式参数和实际参数	178
7.3.2	函数的返回值	179
7.4	函数的调用	181
7.4.1	函数调用的一般形式	181
7.4.2	赋值调用与引用调用	181
7.4.3	函数调用的方式	182
7.4.4	被调用函数的声明和函数原型	183
7.5	函数的嵌套调用	184
7.6	函数的递归调用	185
7.7	数组作为函数参数	190
7.7.1	数组元素作为函数实参	190
7.7.2	数组名作为函数参数	190
7.8	局部变量和全局变量	193
7.8.1	局部变量	193
7.8.2	全局变量	195
7.8.3	变量的存储类别	197
7.9	函数应用程序设计	201
	习题七	207
第8章	预处理命令与位运算	211
8.1	宏定义	211
8.1.1	无参宏定义	211
8.1.2	带参宏定义	214
8.2	文件包含	217
8.3	条件编译	218
8.4	位运算	220
8.4.1	位运算符与位运算	221
8.4.2	位域(位段)	223
	习题八	226
第9章	指针	229
9.1	地址指针的基本概念	229
9.2	变量的指针和指向变量的指针变量	230
9.2.1	指针变量的定义	230
9.2.2	指针变量的引用	230
9.2.3	指针变量作为函数的参数	234
9.2.4	对指针变量的进一步说明	237
9.3	数组指针和指向数组的指针变量	240
9.3.1	指向数组元素的指针	240

9.3.2 通过指针引用数组元素	241
9.3.3 数组名作为函数的参数	244
9.3.4 指向多维数组的指针和指针变量	249
9.4 字符串指针和指向字符串的指针变量	252
9.4.1 字符串的表示形式	252
9.4.2 字符串指针变量与字符数组的区别	254
9.5 函数指针变量	255
9.5.1 函数指针变量定义	255
9.5.2 指针型函数	257
9.6 指针数组和指向指针的指针	258
9.6.1 指针数组的概念	258
9.6.2 指向指针的指针	261
9.6.3 main 函数的参数	263
9.7 有关指针的数据类型和指针运算的小结	264
9.7.1 有关指针的数据类型的小结	264
9.7.2 指针运算的小结	265
9.8 指针应用程序设计举例	265
习题九	269
第 10 章 结构体与共用体	273
10.1 结构体	273
10.1.1 结构体的定义	273
10.1.2 结构类型变量的说明	274
10.1.3 结构变量成员的表示方法	275
10.1.4 结构变量的赋值	276
10.1.5 结构变量的初始化	277
10.1.6 结构数组的定义与使用	277
10.1.7 结构指针变量的说明和使用	279
10.1.8 结构指针变量作为函数的参数	282
10.1.9 动态存储分配	283
10.2 链表	286
10.2.1 链表的概念	286
10.2.2 单链表的基本操作	287
10.3 共用体	293
10.3.1 共用体的定义	293
10.3.2 共用体变量的引用	294
10.4 枚举类型	297
10.4.1 枚举类型的定义和枚举变量的说明	297
10.4.2 枚举类型变量的赋值和使用	298
10.5 类型定义符 typedef	299

10.5.1 类型定义符 typedef 的概念与定义	300
10.5.2 typedef 定义类型的使用方法	301
10.6 结构体程序设计	303
习题十	308
第 11 章 文件	311
11.1 C 文件概述	311
11.2 文件的打开与关闭	312
11.2.1 文件指针	312
11.2.2 文件的打开与关闭具体操作方法	312
11.3 文件的读写	315
11.3.1 字符读写函数 fgetc() 和 fputc()	315
11.3.2 字符串读写函数 fgets() 和 fputs()	318
11.3.3 数据块读写函数 fread() 和 fwrite()	320
11.3.4 格式化读写函数 fscanf() 和 fprintf()	321
11.4 文件的随机读写	323
11.4.1 文件定位	323
11.4.2 文件的随机读写具体操作方法	324
11.4.3 文件检测函数	324
11.5 C 语言系统库文件	325
11.6 C 语言文件系统程序设计举例	326
习题十一	330
附录 ASCII 码表	332

第 1 章 程序设计与算法

计算机程序是由人事先规定的计算机完成某项工作所需的操作步骤（算法）。每一步骤的具体内容用计算机能够理解的指令来描述，这些指令告诉计算机“做什么”和“怎样做”。

程序设计则是用计算机语言（程序设计语言）实现算法的过程。所以说程序设计离不开算法，算法用来指导程序设计，是程序的灵魂。

1.1 计算机语言和程序

1.1.1 计算机语言的概念

计算机语言是计算机能够理解和识别的语言，是人与计算机进行信息交流的工具。它通过一定的方式向计算机传送操作指令，从而使计算机能够按照人们的意愿执行各种操作。任何一种计算机语言都有一定的使用规则，通常称之为语法规则。

要学习计算机语言，必须先学习它的语法规则，就像学汉语要学汉语语法一样。而学习计算机语言的目的是为了设计计算机程序。

计算机语言的种类有很多种，大体上经过了由低级语言到高级语言的发展过程，目前广泛使用的有 C、Visual Basic、Visual FoxPro、Delphi、C++、C#和 Java 等。

1.1.2 C 语言的特点

1. C 语言是中级语言

C 语言通常称为中级计算机语言。中级语言并没有贬义，不意味着它功能差、难以使用，或者比 Basic、Pascal 那样的高级语言原始，也不意味着它与汇编语言相似，会给用户带来类似的麻烦。C 语言之所以被称为中级语言，是因为它把高级语言的成分同汇编语言的功能结合了起来。表 1-1 表明了 C 语言在计算机语言中所处的地位。

表 1-1 C 语言在计算机语言中的地位

级 别	语 言
高级	C++、Java、Delphi、C#、Visual Basic、Visual FoxPro
中级	C、Forth、Macro – assembler
低级	Assembler

作为中级语言，C 语言允许对位、字节和地址这些计算机功能中的基本成分进行操作。C 语言程序非常容易移植。可移植性表现为为某种计算机编写的软件可以用到另一种计算机上去。例如，如果为苹果机编写的一个程序能够方便地改为可以在 IBM PC 上运行的程序，则称该程序是可移植的。

所有的高级语言都支持数据类型的概念。一个数据类型定义了一个变量的取值范围和可在其上的一组运算。常见的数据类型是整型、字符型和实数型。虽然 C 语言有 5 种基本数据类型，但与 Pascal 或 Ada 相比，它却不是强类型语言。C 程序允许几乎所有的类型转换。例如，字符型和整型数据能够自由地混合在大多数表达式中进行运算。这在强类型高级语言中是不允许的。

C 语言的另一个重要特点是它仅有 32 个关键字，这些关键字就是构成 C 语言的命令。

2. C 语言是结构化语言

虽然从严格的学术观点上看，C 语言是块结构（block-structured）语言，但是它还是被称为结构化语言。这是因为它在结构上类似于 Algol、Pascal 和 Modula-2（从技术上讲，块结构语言允许在过程和函数中定义过程或函数。用这种方法，全局或局部的概念可以通过“作用域”规则加以扩展，用“作用域”管理变量和过程的“可见性”。因为 C 语言不允许在函数中定义函数，所以不能称之为通常意义上的块结构语言）。

结构化语言的显著特征是代码和数据的分离。这种语言能够把执行某项特殊任务的指令和数据从程序的其余部分分离出去、隐藏起来。获得隔离的一个方法是调用使用局部（临时）变量的子程序。通过使用局部变量，我们能够写出对程序其他部分没有副作用的子程序。这使得编写共享代码段的程序变得十分简单。如果开发了一些分离性很好的函数，在引用时我们仅需要知道函数做什么，不必知道它如何做。切记：过多使用全局变量（可以被全部程序访问的变量）会由于副作用而在程序中引入错误。

结构化语言比非结构化语言更有利于程序设计，用结构化语言编写的程序清晰性好，这使得它们更易于维护。这已是人们普遍接受的观点。C 语言的主要结构成分是函数 C 的独立子程序。

在 C 语言中，函数是完成程序功能的基本构件（程序块）。函数允许一个程序的多个任务被分别定义和编码，使程序模块化。可以确信，一个好的函数不仅能正确工作，且不会对程序的其他部分产生副作用。此外，C 语言的特点还包括：数据结构类型丰富，具有结构化的控制语句，语法限制不太严格，程序设计自由度大，生成目标代码质量高，程序执行效率高，与汇编语言相比，用 C 语言编写的程序可移植性好。

但是，C 语言对程序员要求较高，虽然程序员用 C 编写程序会感到限制少、灵活性大、功能强，但较其他高级语言在学习上要困难一些。

1.1.3 程序的概念

从生活的角度来说，程序通常是指完成某些事务的一种既定方式或过程。

从计算机的角度来说，程序就是按照计算机语言规则组织起来的一组指令，是为完成某项功能所需要执行的命令序列。这些命令序列按照一定的结构合理地、有机地组合在一起，并以文件的形式存储在磁盘上，故称为命令文件。

1.2 算法

程序的实质是数据结构与算法的结合。对于面向对象的程序设计语言，强调的是数据结构，而对于面向过程的程序设计语言（如 C、Pascal 和 FORTRAN 等语言），主要关注的是

算法。掌握算法，也是为面向对象的程序设计打下一个扎实的基础。

1.2.1 算法的概念

使用计算机，就是要利用计算机解决各种问题。而要达到此目的，就必须事先对各类问题进行分析，制定出解决问题的具体方法和步骤，再具体编写一组指令（即程序），让计算机按照人们事先指定的步骤运行这些指令。这些具体的方法和步骤，其实就是解决问题的算法。根据算法，依据某种规则编写计算机执行的命令序列，就是编写程序，而书写时所应遵守的规则，即为某种语言的语法。

由此可见，程序设计的关键在于解题的方法与步骤，即算法。学习高级语言的重点，就是掌握分析问题、解决问题的方法，就是培养读者分析、分解，最终归纳整理出算法的能力。如 C 语言的语法是工具，是算法的一个具体实现。所以在高级语言的学习中，一方面应熟练掌握该语言的语法，因为它是算法实现的基础，另一方面必须认识到算法的重要性，加强思维训练，以写出高质量的程序。

综上所述，人们为解决某一个具体问题而采取的有效的、科学的方法与步骤称为“算法”。不要认为只有“计算”的问题才有算法。

1.2.2 算法的特性

对同一个问题，可以有不同的解题方法和步骤。方法有优劣之分，有的方法只需很少步骤就能解决问题，而有些方法则刚好相反。一般来说，建议采用简单和运算步骤少的方法。因此，为了有效地进行解题，不仅需要保证算法正确，还要考虑算法的质量，选择合适的算法。

算法应当具备以下特性：

- (1) 有穷性。一个算法必须保证执行有限步之后结束。
- (2) 确定性。算法的每一个步骤必须具有确切的定义。
- (3) 输入性。应对算法给出初始量。
- (4) 输出性。算法具有一个或多个输出。
- (5) 可行性。算法的每一步都必须是计算机能执行的有效操作。

1.2.3 算法的设计

下面通过实例来介绍如何设计算法。

【例 1-1】 输入 3 个数，然后输出其中最大的数。

首先，定义 3 个变量 A、B 和 C 来存放这 3 个数，也就是将 3 个数依次输入到 A、B 和 C 中，然后，再准备一个变量 Max 存最大数。由于计算机一次只能比较两个数，我们先比较 A 与 B，把大的数放入 Max 中，再比较 Max 与 C，再把大的数放入 Max 中。最后输出 Max，此时 Max 中存的就是 A、B 和 C 3 个数中最大的数。算法表示如下：

- (1) 输入 A、B、C。
- (2) 把 A 与 B 中较大的一个放入 Max 中。
- (3) 把 C 与 Max 中较大的一个放入 Max 中。
- (4) 输出 Max，Max 即为最大数。

其中的第(2)、(3)两步仍不明确,无法直接转化为程序语句,可以继续细化。

(2) 把 A 与 B 中较大的一个放入 Max 中,若 $A > B$, 则 $\text{Max} \leftarrow A$; 否则 $\text{Max} \leftarrow B$ 。

(3) 把 C 与 Max 中较大的一个放入 Max 中,若 $C > \text{Max}$, 则 $\text{Max} \leftarrow C$ 。

最后算法可以写成:

(1) 输入 A、B、C。

(2) 若 $A > B$, 则 $\text{Max} \leftarrow A$; 否则 $\text{Max} \leftarrow B$ 。

(3) 若 $C > \text{Max}$, 则 $\text{Max} \leftarrow C$ 。

(4) 输出 Max, Max 即为最大数。

这样的算法已经可以很方便地转化为相应的程序语句了。

[例 1-2] 猴子吃桃问题。有一堆桃子不知数目,猴子第 1 天吃掉一半,觉得不过瘾,又多吃了一个,第 2 天仍吃掉剩下桃子的一半多一个,天天如此。直到第 10 天早上,猴子发现只剩一个桃子了,问这堆桃子原来有多少个?

此题应从何处着手呢?假设第 1 天开始时有 a_1 个桃子,第 2 天有 a_2 个, ..., 第 9 天有 a_9 个,第 10 天是 a_{10} 个,在 $a_1, a_2, \dots, a_{10}$ 中,只有 $a_{10} = 1$ 是已知的,现要求 a_1 ,由题干可知, $a_1, a_2, \dots, a_{10}$ 之间存在一个简单的关系:

$$a_9 = 2 * (a_{10} + 1)$$

$$a_8 = 2 * (a_9 + 1)$$

...

$$a_1 = 2 * (a_2 + 1)$$

也就是 $a_i = 2 * (a_{i+1} + 1), i = 9, 8, 7, 6, \dots, 1$ 。

这就是此题的数学模型。

再观察上面从 $a_9 \sim a_1$ 的计算过程,发现它其实是一个递推过程,这种递推的方法在计算机解题中经常用到。另一方面,这 9 步运算从形式上完全一样,不同的只是 i 的值而已。由此,我们引入循环的处理方法,并统一用 a_0 表示前一天的桃子数, a_1 表示后一天的桃子数,将算法改写如下:

(1) $a_1 = 1$ (第 10 天的桃子数, a_1 的初值); $i = 9$ (计数器初值为 9)。

(2) $a_0 = 2 * (a_1 + 1)$ (计算当天的桃子数)。

(3) $a_1 = a_0$ (将当天的桃子数作为下一次计算的初值)。

(4) $i = i - 1$ 。

(5) 若 $i > 1$, 执行步骤 (2)。

(6) 输出 a_0 的值。

其中,步骤 (2) ~ (5) 为循环。

这就是一个从具体到抽象的过程,具体实现过程如下:

(1) 如果由人来做,应该通过哪些步骤来实现。

(2) 对这些步骤进行归纳整理,抽象出数学模型。

(3) 对其中的重复步骤,通过使用相同变量等方式求得形式的统一,然后简练地用循环实现。

算法的描述方法有自然语言描述、伪代码、流程图、N-S 图和 PAD 图等。

1.2.4 算法的描述方法

算法是实现某一个问题求解的框架流程，而程序设计则是根据这一求解的框架流程进行语言细化，并实现这一问题求解的具体过程。

为了使算法表达更清晰，编写更容易，在程序设计时通常使用专门的工具对算法进行描述。如流程图，N-S图，PAD图、伪代码等。对于复杂的问题，可以先对算法进行描述，再进行编程，算法的最终实现应该是通过计算机程序。算法的评价标准涉及很多方面，但正确性和清晰性永远是评价一个算法好坏的基本标准。

描述算法的常用工具有以下几种。

1. 自然语言

使用人们日常进行交流的语言来描述算法。

例如，从 a, b 中找出一个大的数存入 Max。

- (1) 从键盘输入两个数并赋值给 a 和 b。
- (2) 如果 a 比 b 大，则把 a 的值存入 Max，否则把 b 的值存入 Max。
- (3) 输出 Max 的值。

2. 专用工具

借助于有关图形工具或代码符号来描述算法。常用的工具有流程图、N-S 图等。

(1) 流程图

流程图是采用图形符号、数学语言和自然语言结合的方法表示算法。常用的图形符号如图 1-1 所示。

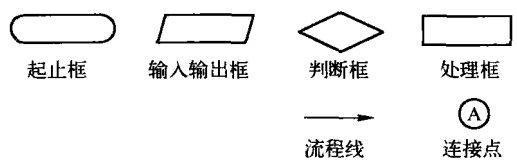


图 1-1 图形符号

算法可用顺序结构、选择结构和循环结构这 3 种基本程序结构来描述。C 语言是结构化程序设计语言，具有与这 3 种基本程序结构相对应的语句。

1) 顺序结构。顺序结构是一种最简单的结构。它以图 1-2 所示的步骤顺序执行，即先执行 A，再执行 B。在 C 语言中对应的语句是表达式语句和函数语句。

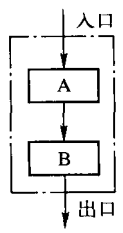


图 1-2 顺序结构

[例 1-3] 交换整数 a 和 b 的值。

交换整数 a 和 b 就像交换一杯白酒和一杯红酒一样，必须有一个空酒杯，其算法可用自然语言和流程图（图 1-3）两种方法描述。

第一步 读入数据存入 a, b。

第二步 交换 a, b。可用如图 1-3 所示的 $temp = a, a = b, b = temp$ 等描述实现交换，其一般格式：

变量名 = 表达式

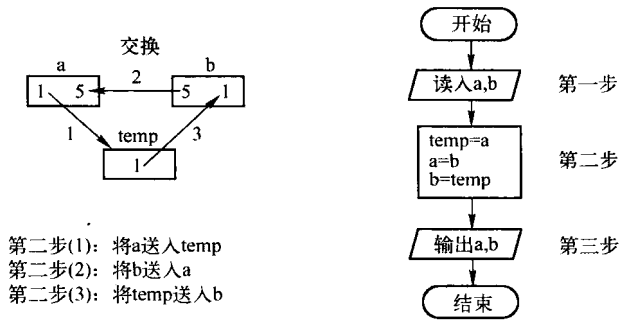


图 1-3 交换整数 *a* 和 *b* 的值

符号“=”的含义是将右边的表达式值赋给左边的变量名，不是等号。

第三步 输出 *a*, *b*。

2) 选择结构。选择结构是通过判断条件 *p* (表达式值) 来决定执行哪一种或哪一组操作的结构。选择结构有一选、二选、多选 3 种基本形式。

图 1-4 为一选一的逻辑。若条件 *p* (表达式值为 1 或非 0) 成立时执行 *A*，否则不执行任何操作。退出选择结构。

[例 1-4] 当 *a* > *b* 时，交换整数 *a* 和 *b* 的值。

该算法描述如图 1-5 所示。

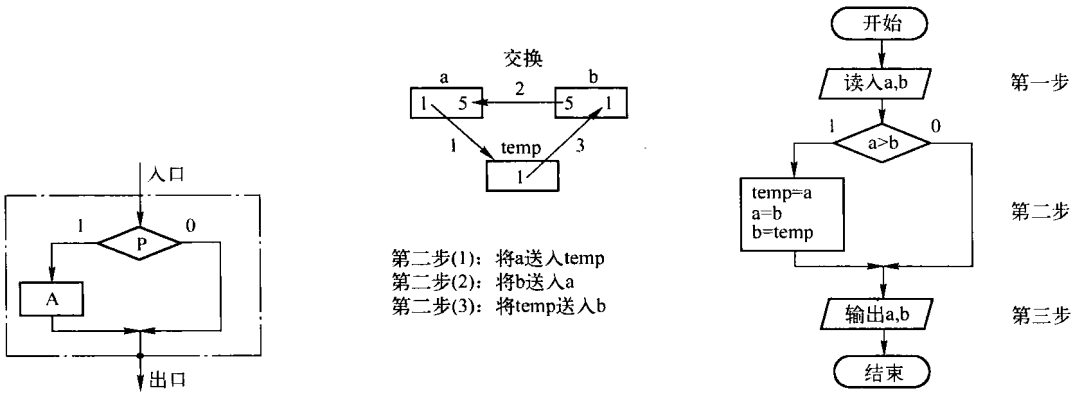


图 1-4 一选一的逻辑

图 1-5 一选一 当 *a* > *b* 时，交换整数 *a* 和 *b* 的值

第一步 读入数据存入 *a*, *b*。

第二步 判定，如果 *a* > *b*，交换 *a* 和 *b*；否则不交换 *a* 和 *b*。

第三步 输出 *a*, *b*。

二选一

图 1-6 所示为二选一的逻辑。若条件 *p* (表达式值为 1 或非 0) 成立时执行 *A*；否则执行 *B*。结束选择结构。

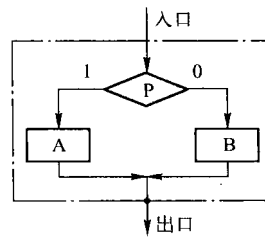


图 1-6 二选一的逻辑

[例 1-5] 当 $x \geq 0$ 时, $y = 1$; 否则, $y = -1$ 。

该题算法描述如图 1-7 所示。

第一步 读入数据存入 x 。

第二步 判断, 如果 $x \geq 0$, 将 y 置为 1; 否则, 将 y 置为 -1。

第三步 输出 y 。

多选一

图 1-8 所示为多选一的逻辑。它依次判断条件 p_1 、 p_2 、 $\dots$ 、 $p(n-1)$, 如果 p_i 成立时执行 i ($i = 1 \sim n-1$)。否则, 不执行任何操作。结束选择结构。

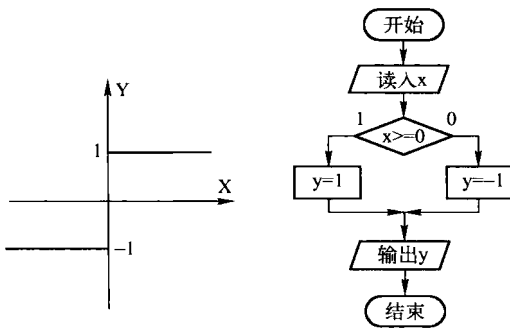


图 1-7 二选一 当 $x \geq 0$ 时, $y = 1$; 否则, $y = -1$

第一步

第二步

第三步

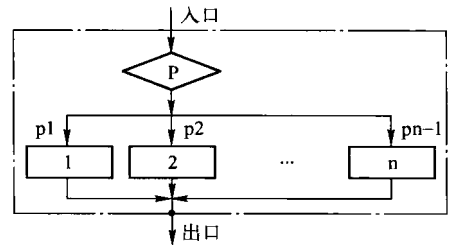
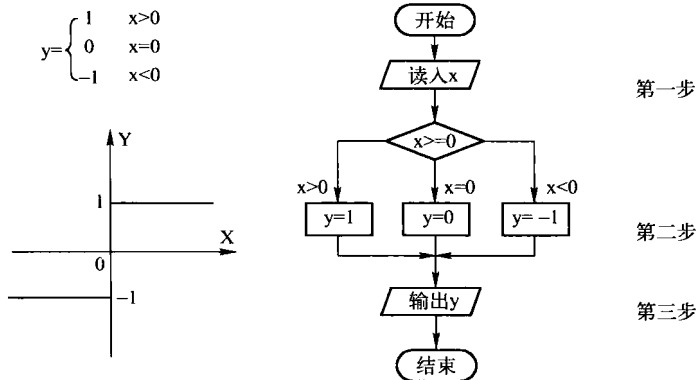


图 1-8 多选一的逻辑

[例 1-6] 如果 $x > 0$, $y = 1$; $x = 0$, $y = 0$; $x < 0$, $y = -1$ 。

该题算法描述如图 1-9 所示。



第一步

第二步

第三步

图 1-9 多选一 如果 $x > 0$, $y = 1$; $x = 0$, $y = 0$; $x < 0$, $y = -1$

第一步 读入数据存入 x 。

第二步 判断 x 的值。

$x > 0$, 将 y 置为 1;

$x = 0$, 将 y 置为 0;

$x < 0$, 将 y 置为 -1。