XXX

February 28, 2020

Contents

Chapter 1

前言

为什么要学习程序设计?

大学生不能满足于只会用办公软件，应当有更高的要求。

只有懂得程序设计，才能进一步懂得计算机，真正了解计算机是怎样工作的。通过学习程序设计，学会进一步了解计算机的工作原理，更好地理解和应用计算机；掌握计算机处理问题的方法；培养分析问题和解决问题的能力；具有编制程序的初步能力。即使将来不是计算机专业人员，由于学过程序设计，理解软件生产的过程，就能与程序开发人员更好地沟通与合作，开展本领域中的计算机应用，开发与本领域有关的应用程序。

因此，都应当学习程序设计知识，并且把它作为进一步学习和应用计算机的基础。

关于计算机工作原理的一个特点是，当代计算机以高速的顺序指令执行。

怎样学习 C 程序设计

1. 要着眼于培养能力。应当注意培养分析问题的能力、构造算法的能力、编程的能力和调试程序的能力。
2. 要把重点放在解题的思路上，通过大量的例题学习怎样设计一个算法，构造一个程序。
3. 掌握基本要求，注意打好基础。
4. 要十分重视实践环节。
5. 要举一反三。
6. 要提倡和培养创新精神。
7. 如果对学生有较高的程序设计要求，应当在学习本课程后，安排一次集中的课程设计环节，要求学生独立完成一个有一定规模的程序。

Chapter 2

FOREWORD

为什么要学习程序设计?

大学生不能满足于只会用办公软件，应当有更高的要求。只有懂得程序设计，才能进一步懂得计算机，真正了解计算机是怎样工作的。通过学习程序设计，学会进一步了解计算机的工作原理，更好地理解和应用计算机；掌握计算机处理问题的方法；培养分析问题和解决问题的能力；具有编制程序的初步能力。即使将来不是计算机专业人员，由于学过程序设计，理解软件生产的过程，就能与程序开发人员更好地沟通与合作，开展本领域中的计算机应用，开发与本领域有关的应用程序。因此，都应当学习程序设计知识，并且把它作为进一步学习和应用计算机的基础。

关于计算机工作原理的一个特点是，当代计算机以高速的顺序指令执行。

怎样学习 C 程序设计

1. 要着眼于培养能力。应当注意培养分析问题的能力、构造算法的能力、编程的能力和调试程序的能力。
2. 要把重点放在解题的思路上，通过大量的例题学习怎样设计一个算法，构造一个程序。
3. 掌握基本要求，注意打好基础。
4. 要十分重视实践环节。
5. 要举一反三。
6. 要提倡和培养创新精神。
7. 如果对学生有较高的程序设计要求，应当在学习本课程后，安排一次集中的课程设计环节，要求学生独立完成一个有一定规模的程序。

Chapter 3

程序设计和 C 语言

3.1 什么是计算机程序

第一个要明确的概念是计算机程序（程序）。程序是用来告诉计算机对数据如何进行处理的指令集合。

3.2 什么是计算机语言

这种计算机能直接识别和而接受的二进制代码称为机器指令（machine instruction）。机器指令的集合就是该计算机的机器语言（machine language）。

符号语言

人们创造了符号语言，它用一些英文字母和数字表示一个指令，符号语言又称为符号汇编语言或汇编语言（assembler language）。

高级语言的发展历程高级语言经历了不同的发展阶段:

1. 非结构化的语言。初期的语言属于非结构化的语言，编程风格比较随意，只要符合语法规则即可，没有严格的规范要求，程序中的流程可以随意跳转。人们为为最修程序执行的效率而采用许多“小技巧”，使程序变得难于阅读和维护。

2. 结构化的语言。为了解决以上问题，提出了“结构化程序设计方法”，规定程序必须由具有良好特性的基本结构（顺序结构、分支结构、顺环结构）构成，程序中的流程不允许随意跳转，程序总是由上而下顺序执行各个基本结构。这种程序结构清晰，易于编写、阅读和维护。QBASIC,FORTRAN 77 和 C 语言等属于结构化的语言，这些语言的特点是支持结构化程序设计方法。

   以上两种语言都是基于过程的语言，在编写程序时需要具体指定每一个过程的细节。在编写规模较小的程序时，还能得心应手，但在处理规模较大的程序时，就显得捉襟见肘、力不从心了。在实践的发展中，人们又提出额面向对象的程序设计方法。程序面对的不是过程的细节，而是一个对象，对象是有数据以及对数据进行的操作组成的。

3. 面向对象的语言。近十多年来，在处理规模较大的问题时，开始使用面向对象的语言。C++,C# ,Visual Basic 和 Java 等语言是支持面向对象程序设计方法的语言。有关面向对象的程序设计方法和面向对象的语言在本书不做详细介绍，有兴趣的可参考有关专门书籍。

3.3 C 语言的发展及其特点

本书是介绍怎样利用 C 语言作为工具进行程序设计的。为什么要选择 C 语言呢? 这里有必要对 C 语言的发展和特点有一定的了解。

C 语言是国际上广泛流行的计算机高级语言。

C 语言的祖先是 BCPL 语言。1967 年英国剑桥大学的 Martin Richards 推出没有类型的 BCPL (Basic Combined Programming Languade) 语言。1970 年美国 AT&T 贝尔实验室的 Ken Thompson

以 BCPL 语言为基础，设计除了很简单且很接近硬件的 B 语言（取 BCPL 的第一个字母）。但 B 语言过于简单，功能有限。1972-1973 年间，美国贝尔实验室的 D.M.Ritchie 在 B 语言的基础上设计出了 C 语言。C 语言既保持了 BCPL 和 B 语言的优点（精练，接近硬件），又克服了它们的缺点（过于简单、无数据类型等），C 语言的新特点主要表现在具有多种数据类型（如字符、数值、数组、结构体和指针等）。开发 C 语言的目的在于尽可能降低用它写的软件对硬件平台的依赖程度，使之具有可移植性。

最初的 C 语言只是为描述和实现 UNIX 操作系统提供一种工作语言而设计。1973 年，Ken Thompson 和 D.M.Ritchie 合作把 UNIX 的 90% 以上用 C 语言改写，即 UNIX 第 5 版。随着 UNIX 的日益广泛使用，C 语言也迅速得到推广。1978 年以后，C 语言先后移植到大、中、小和微型计算机上。C 语言便很快风靡全世界，成为世界上应用最广泛的程序设计高级语言。

以 UNIX 第 7 版中的 C 语言编译程序为基础, 1978 年, Brian W.Kernighan 和 Dennis M.Ritchie 合著了影响深远的名著 The C Programming Language，本书中介绍的 C 语言成为广泛使用的 C 语言版本的基础，它是实际上第一个 C 语言标准。

C 语言是一种用途广泛、功能强大、使用灵活的过程性（procedural）编程语言，即可用于编写应用软件，又能用于编写系统软件。因此 C 语言问世以来得到迅速推广。自 20 世纪 90 年大初，C 语言在我国开始推广以来，学习和使用 C 语言的人越来越多，成了学习和使用人数最多的一种计算机语言，绝大多数理工科大学都开设了 C 语言程序设计课程。掌握 C 语言成为计算机开发人员的一项基本功。

C 语言有以下一些主要特点。

1. 语言简洁、紧凑，使用方便、灵活。C 语言一共只有 37 个关键字，9 种控制语句，程序书写形式自由。C 语言程序比其他许多高级语言简练，源程序短，因此输入程序时工作量少。

2. 运算符丰富。C 语言的运算符包含的范围很广泛，共有 34 种运算符。C 语言把括号、赋值和强制类型转换等都作为运算符处理，从而使 C 语言的运算类型极其丰富，表达式类型多样化。灵活使用各种运算符可以实现在其他高级语言种难以实现的运算。

3. 数据类型丰富。C 语言提供的数据类型包括：整型、浮点型、字符型、数组类型、指针类型、结构体类型和共用体类型等，C99 又扩充了复数浮点类型、超长整型和布尔类型等。尤其时指针类型数据，使用十分灵活和多样化，能用来事项各种复杂的数据结构（如链表、树、栈等）的运算。

4. 具有结构化的控制语句。用函数作为程序的模块单位，便于事项程序的模块化。C 语言时完全模块化和结构化的语言。

5. 语法限制不太严格，程序设计自由度大。

6. C 语言允许直接访问物理地址，能进行位（bit）操作，能实现汇编语言的大部分功能，可以直接对硬件进行操作。

7. 用 C 语言编写的程序可移植性好。

8. 生成目标代码质量高，程序执行效率高。

C 原来是专门为编写系统软件而设计的，许多大的软件都用 C 语言编写，这是因为 C 语言的可移植性好和硬件控制能力高，表达和运算能力强。许多以前只能用汇编语言处理的问题，后来可以改用 C 语言来处理了。目前 C 的主要用途之一是编写“嵌入式系统程序”。由于具有上述优点，使 C 语言应用十分广泛，许多应用软件也用 C 语言编写。

对 C 语言以上的特点，待学完 C 语言以后再回顾以下，就会有比较深的体会。

3.4 最简单的 C 语言程序

为了使用 C 语言编程序，必须了解 C 语言，并且能熟练地使用 C 语言。本书将由浅到深地介绍怎样阅读 C 语言程序和使用 C 语言编写程序。

3.4.1 最简单的 C 语言程序举例

3.4.2 C 语言程序的结构

3.5 运行 C 程序的步骤与方法

计算机不能直接识别和执行用高级语言写的指令。必须用编译程序把 C 源程序翻译成二进制形式的目标程序，然后再将该目标程序与系统的函数库以及其他目标程序连接起来，形成可执行的目标程序。

在编写好一个 C 源程序后，怎样上级进行编译和运行呢？一般要经过以下几个步骤：

1. 上机输入和编辑源程序。
2. 对源程序进行编译，先用 C 编译系统提供的“预处理器”对程序中的预处理指令进行编译预处理。由预处理得到的信息与程序其他部分一起，组成一个完整的、可以用来进行正是编译的源程序，然后由编译系统对该源程序进行编译。

   编译的作用首先是对源程序进行检查，判定它有无语法方面的错误，如有，则发出“出错信息”，告诉编程人员认真检查改正。修改程序后重新进行编译，如有错，在发出“出错信息”。如此反复进行，知道没有语法错误为止。这是，编译程序自动把源程序转换为二进制形式的目标程序。如果不特别指定，此目标程序一般也存放在用户当前目录下，此时源文件没有消失。

   在用编译系统对源程序进行编译时，自动包括了预编译和正式编译两个阶段，一气呵成。用户不必分别发出二次指令。
3. 进行链接处理。经过编译所得到的二进制文件还不能共计算机直接执行。前面已说明：一个程序可能包含若干个源程序文件，而编译是以源程序文件为对象的，一次编译只能得到与一个源程序文件相对应的目标文件，它只是整个程序的一部分。必须把所有的编译后得到的目标模块链接装配起来，在于函数库相连接成为一个整体，生成一个可供计算机执行的目标程序，成为可执行程序（executive program）。

   即使一个程序只包含一个源程序文件，编译后得到的目标程序也不能直接运行，也要经过连接阶段，因为要与寒暑假进行链接，才能生成可执行程序。
4. 运行可执行程序，得到运行结果。

一个程序从编写到运行成功，并不是一次成功的，往往要经过多次反复。编写好的程序并不一定能保证正确无误，除了用人工方式检查外，还必须借助编译系统来检查有无语法错误。

为了编译、链接和运行 C 程序，必须要有相应的编译系统。

写出源程序后可以用任何一种编译系统对程序进行编译和连接工作，只要用户感到方便、有效即可。

对编译系统的态度是，不应当只会一种编译系统，无论用哪一种编译系统，都应当能举一反三，在需要时会用其他编译系统进行工作。

3.6 程序设计的任务

如果只是编写和运行一个很简单的程序，上面介绍的步骤就够了。但是实际上要处理的问题比上面见到的例子复杂得多，需要考虑和处理的问题也复杂得多。程序设计就是指从确定任务到得到结果、写出文档的全过程。从确定问题到最后完成任务，一般经历以下几个工作阶段：

1. 问题分析。对于接手的任务要进行认真的分析，研究所给定的条件，分析最后应达到的目标，找出解决问题的规律，选择解体的方法。在此过程种可以忽略一些次要的因素而使问题抽象化。这就是建立模型。
2. 设计算法。即设计出解体的方法和具体步骤。
3. 编写程序。根据得到的算法，用一种高级语言编写出源程序。
4. 对源程序进行编辑、编译和链接，得到可执行程序。

5. 行程序，分析结果。运行可执行程序，得到运行结果。能得到运行结果并不意味着程序正确，要对结果进行分析，看它是否合理。

6. 编写程序文档。许多程序使提供给别人使用的，如同正式的产品应当提供产品说明书一样，正式提供给用户使用的程序，必须向用户提供程序说明书。内容包括：程序名称、程序功能、运行环境、程序的装入和启动给、需要输入的数据，以及使用的注意事项等。

Chapter 4

算法——程序的灵魂

通过第 1 章的学习，了解了 C 语言的特点，看到了简单得 C 语言程序。现在从程序的内容方面进行讨论，也就是一个程序中应该包含什么信息。或者说，为了实现解题的要求，程序应当向计算机发送什么信息。

一个程序主要包括以下两发面的信息：

- 对数据的描述。在程序中要指定那些数据以及这些数据的类型和数据的组织方式。这就是数据结构（data structure）。
- 对操作的描述。即要求计算机进行操作的步骤，也就是算法（alorithm）。

实际上，一个过程化的程序除了以上两个主要要素以外，要应当采用结构化程序设计方法进行程序设计，并且用某一种计算机语言表示。因此，算法、数据结构、程序设计方法和语言工具 4 个方面是一个程序设计人员所具备的知识，在设计一个程序时要综合运用这几方面的知识。本书中不可能全面介绍这些内容，它们都属于有关的专门课程范畴。在这 4 个方面中，算法是灵魂，数据结构是加工对象，语言是工具，编程需要采用合适的方法。

算法解决“做什么”和“怎么做”的问题。程序中的操作语句，实际上就是算法的体现。显然，不了解算法就谈不上程序设计。本书不是一本专门介绍算法的教材，也不是一本只介绍 C 语言语法规则的使用说明。本书将通过一些实例把以上 4 个方面的知识结合起来，使读者学会考虑解题的思路，并且能正确地编写出 C 程序。

由于算法的重要性，本章先介绍有关算法的初步知识，以便后面各章的学习建立一些基础。

4.1 什么是算法

算法是一种逐步解决问题或完成任务的方法。更正式的定义：算法是一组明确步骤的有序集合，它产生结果并在有限时间内终止。

计算机算法可分为两大类别：数值运算算法和非数值运算算法。数值运算的目的是求数值解，例如求方程的根、求一个函数的定积分等，都属于数值运算范畴。非数值运算包括的面十分广泛，最常见的是用于事务管理领域，例如对一批职工按姓名排序、图书检索、认识管理和行车调度管理等。目前，计算机在非数值运算方面的运用远远超过了在数值运算方面的应用。

由于数值运算往往有现成的模型，可以运用数值分析方法，因此对数值运算的算法的研究比较深入，算法比较成熟。对各种数值运算都有比较成熟的算法可供选用。人们常常把这些算法汇编成册，或者将这些程序存放在磁盘或光盘上，供用户调用。例如有的计算机系统提供“数学程序库”，使用起来十分方面。

非数值运算的种类繁多，要求各异，难以做到全部都有现成的答案，因此已有一些典型的非数值运算算法有现成的、成熟的算法可供使用。许多问题往往需要使用者参考已有的类似算法的思路，重新设计解决特定问题的专门算法。本书不可能罗列所有算法，只是向通过一些典型算法的介绍，帮助读者了解什么是算法，怎样设计一个算法，帮助读者与一反三。希望读者通过本章介绍的例子了解怎样提出问题，怎样思考问题，怎样表示一个算法。

4.2 简单的算法举例

例 2.1 求 1 x 2 x 3 x 4 x 5。
例 2.2 有 50 名学生，要求输出成绩在 80 分以上的学生的学号和成绩。
S1: 1 => i
S2: 如果 gi>=80，则输出 ni 和 gi，否则不输出
S3: i + 1 => i
S4: 如果 i <= 50，返回到步骤 S2，继续执行，否则，算法结束。
例 2.3 判断 2000-2500 年中的每一年是否为闰年。

4.3 算法的特性

- 有穷性
- 确定性
- 有零个或多个输入
- 有一个或多个输出
- 有效性

4.4 这样表示一个算法

4.4.1 用自然语言表示算法

4.4.2 用流程图表示算法

4.4.3 三种结构和改进的流程图

4.4.4 用 N-S 流程图表示算法

4.4.5 用伪代码表示算法

伪代码使用于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。
例 2.16 求 5!，用伪代码表示的算法如下:

```
begin
t := 1
i := 2
while i <= 5
{
        t := t * i
        i := i + 1
}
print t
end
```

4.4.6 用计算机语言表示算法

4.5 结构化程序设计方法

前面介绍了结构化的算法和 3 种基本结构。一个结构化程序就是用计算机语言表示的结构化算法，用 3 种基本结构组成的程序必然是结构化的程序。这种程序便于编写、阅读、修改和维护，这就减少了程序出错的机会，提高了程序的可靠性，保证了程序的质量。结构化程序设计强调程序设计风格和程序结构的规范化，提倡清晰的结构。怎样才能得到一个结构化的程序呢？如果面临一个

复杂的问题，是难以一下子写出一个层次分明、结构清晰、算法正确的程序的。结构化程序设计方法的基本思路是：把一个复杂问题的求解过程分阶段进行，每个阶段处理的问题都控制在人们容易理解和处理的范围内。

具体的说，采取以下方法来保证得到结构化的程序：

- 自顶而下；
- 逐步细化；
- 模块化设计；
- 结构化编码。

提倡用这种方法设计程序，这就是用工程的方法设计程序。

应当掌握自顶而下、逐步细化的程序设计方法。这种设计方法的过程是将问题求解由抽象逐步具体化的过程。

用这种方法便于验证算法的正确性，在向下一层展开之前应仔细检查本层设计是否正确，只有上一层是正确的才能向下细化。如果每一层设计都没有问题，则成哥算法就是正确的。由于每一层向下细化时都不太复杂，因此容易保证整个算法的正确性。检查时也是由上而下逐层检查，这样做，思路清楚，有条不紊地一步一步地进行，既严谨由方便。

在程序设计种长采用模块化的设计方法，尤其当程序比较复杂时，更有必要。子啊拿到一个程序模块以后，根据程序模块的功能将它划分为若干个子模块，入宫这些子模块的规模还嫌大，可以再划分为更小的模块。这个过程采用自顶而下的方法来实现。

程序中的子模块在 C 语言中通常用函数来实现。

程序中的子模块一般不超过 50 行，即把它打印输出时不超过一页，这样的规模便于组织，也便于阅读。划分子模块时应注意模块的独立性，即使用一个模块完成一项弄能，耦合性越少越好。模块化设计的思想实际上时一种“分而治之”的思想，把一个大人物分为若干个子任务，每一个子任务就相对简单了。

结构化程序设计方法用来解决人脑思维能力的局限性和被处理问题的复杂 ing 之间的矛盾。

在设计好一个结构化算法之后，还要善于进行结构化编码。所谓编码就是将已设计好的算法用计算机语言来表示，即根据已经细化的算法正确地写出计算机程序。结构化语言都有与 3 中基本结构对应的语句，进行结构化编程序是不困难的。

习题

4. 用传统流程图表示求解以下问题的算法

1. 有两个瓶子 A 和 B，分别盛放醋和酱油，要求将它们互换。
2. 一次将 10 个数输入，要求输出其中最大的数。
3. 有 3 个数 a，b，c，要求按大小顺序把它们输出。
4. 求 $1+2+3+\cdots+100$
5. 判断一个数 n 能否同时被 3 和 5 整除。
6. 将 100 200 之间的素数输出。
7. 求两个数 m 和 n 的最大公约数。
8. 求方程式 $ax^2+bx+c=0$ 的根。分别考虑：
   - 有两个不等的实根
   - 有两个相等的实根。

8. 用子等而下、逐步细化的方法进行以下算法的设计：

1. 输出 100 2000 年中是闰年的年份，复合 i 安眠两个条件之一的年份是闰年。
2. 输入 10 个数，输出其中最大的一个数。

Chapter 5

最简单的 C 程序设计——顺序程序设计

有了前两章的基础，现在可以开始由浅入深地学习 C 语言程序设计了。
为了能编写 C 语言程序，必须具备以下地知识和能力：

1. 要有正确地解题思路，即学会设计算法，否则无从下手。
2. 掌握 C 语言的语法，知道怎样使用 C 语言所提供的功能编写出一个完整的、正确的程序。也就是在设计好算法之后，能用 C 语言正确表示此算法。
3. 在写算法和编写程序时，要采用结构化程序设计方法，编写出机构化的程序。

算法的种类很多，可以说是无底洞，不可能等到吧所有的算法都学透以后再来学习编程序。C 语言的语法规定很多、很烦琐，独立地学习语法不但枯燥乏味，而且即使倒背如流，也不一定能写出一个好的程序。必须找到一种有效的学习方法。

本书的做法是：以程序设计为主线，把算法和语法紧密结合起来，引导读者由易到及难地学会编写 C 程序。对于简单的程序，算法比较简单，程序中涉及的语法现象也比较简单。对于比较复杂的算法，程序中用到的语法现象也比较复杂（例如要使用数组、指针和结构体等）。本章先从简单的程序开始，介绍简单的算法，同时介绍最基本的语法现象，使读者具有编写简单的程序的能力。在此基础上，逐步介绍复杂一些的程序，介绍比较复杂的算法，同时介绍深入的语法现象，把算法与语法有机地结合起来，不不深入，由简单到复杂，使读者很自然地、循序渐进地学会编写程序

5.1 顺序程序设计举例

例 3.1 有人用温度计测量出用华氏法表示的温度，今要求把它转化为以摄氏法表示的温度。

例 3.2 计算存款利息。有 1000 元，想存一年。有 3 种方法：（1）火气，年利率为 $r1$；（2）一年期定期，年利率为 $r2$；（3）存两次半年定期，年利率为 $r3$。请分别计算出一年后按 3 中方法所的到的本息和。

5.2 数据的表现形式及其运算

有了以上写程序的基础，本章对程序中最基本的成分作必要的介绍。

5.2.1 常量和变量

在计算机高级语言中，数据有两种表现形式：常量和变量。

1. 常量

   在程序运行过程中，其值不能被改变的两称为常量。

   常用的常量有以下几类：

   (a) 整型常量。

(b) 实型常量。
(c) 字符常量。
(d) 字符串常量。
(e) 符号常量。

2. 变量

变量代表一个有名字、具有特定属性的一个存储单元。它用来存放数据，也就是存放变量的值。在程序运行期间，变量的值是可以改变的。

变量必须先定义，后使用。在定义时制定改变量的名字和类型。一个变量应该有一个名字，以便被引用。

3. 常变量

C99 允许使用常变量，如

```
const int a = 3;
```

表示 a 被定义为一个整形变量，指定其值为 3, 而且在变量存在期间其值不能改变。

4. 标志符

在计算机高级语言中，用来对变量、符号常量名、函数、数组、类型等命名的有效字符序列统称为标识符（identifier)。

5.2.2 数据类型

C 语言要求在定义所有变量的时候要制定变量的类型。

所谓的类型，就是对数据分配存储单元的安排，包括存储单元的长度以及数据的存储形式。不同的类型分配不同的长度和存储形式。

5.2.3 整形类型

1. 整型数据的分类

本节介绍基本的整形类型。

(a) int
(b) short int
(c) long int
(d) long long int

2. 整型变量的符号属性

5.2.4 字符型数据

由于字符是按其代码形式存储的，因此 C99 把字符型数据作为整数类型的一种。但是，字符型数据在使用上有自己的特点，因此把它单独列为一节来介绍。

1. 字符与字符代码

2. 字符变量

5.2.5 浮点型数据

浮点型数据用来表示具有小数点的实数。

1. float

2. double

3. long double

5.2.6 怎样确定常量的类型

怎样确定常量的类型呢？从常量的表示形式即可以判断其类型。
整型常量
浮点型常量

5.2.7 运算符和表达式

几乎没一个程序都需要进行计算，对数据进行加工处理，否则程序就没有意义了。要进行运算，就需规定可以是使用的运算符。C 语言的运算符范围很宽，把除了控制语句和输入输出以外的几乎所有的基本操作都作为运算符处理，如方括号，点号。

1. 基本的算术运算符
2. 自增、自减运算符
3. 算术表达式和运算符的优先级与结合性

   用运算符和括号将运算对象（操作数）链接起来、符合 C 语法规则的式子，成为 C 算术表达式。
4. 不同类型数据间的混合运算

   这些转换是编译系统自动完成的，用户不必过问。
5. 强制类型转换运算符
6. C 运算符

   - 算术运算符 `+- * / ++ --`
   - 关系运算符 `> < == >= <= !=`
   - 逻辑运算符 `! &&`
   - 位运算符 `<<>>~`
   - 赋值运算符 = 及其扩展赋值运算符
   - 条件运算符 `?:`
   - 都好运算符 `,`
   - 指针运算符 `* &`
   - 求字节数运算符 sizeof
   - 强制类型转换运算符 (类型)
   - 成员运算符 `. ->`
   - 下表运算符 `[]`
   - 其他 ()

例给定一个大写字母，要求用小写字母输出。

5.3 C 语句

5.3.1 C 语句的作用和分类

在前面的例子可以看到：一个函数包含声明部分和执行部分，执行部分是由语句组成的，语句的作用是想计算机系统发出操作指令，要求执行相应的操作。一个 C 语句经过编译后产生若干条机器指令。声明部分不是语句，它不产生机器指令，只是对有关数据的声明。

C 程序的结构可以用图表示。即一个 C 程序可以由若干个源程序文件组成，一个源文件可以由若干个函数和预处理指令以及全局变量声明部分组成。一个函数由数据声明部分和执行语句组成。

C 语句分为以下 5 类。

1. 控制语句
   - if()...else
   - 2)for()...
   - while()...
   - do...while
   - continue
   - break
   - switch
   - return
   - goto
2. 函数调用语句。函数调用语句由一个函数调用加上一个分号构成。
3. 表达式语句。
4. 空语句。
5. 复合语句。

5.3.2 最基本的语句——赋值语句

在 C 程序中，最常用的语句是：赋值语句和输入输出语句。其中最基本的是赋值语句。程序中的计算功能大部分是由赋值语句实现的。

例给出三角形的长，求三角形的面积。

5.4 数据的输入输出

5.4.1 输入输出举例

前面已经看到了利用 printf 函数进行数据输出的程序，现在再介绍一个包含输入和输出的程序。

例求 $ax^2 + bx + c = 0$ 方程的根。a, b, c 由键盘输入，设 $b^2 - 4ac > 0$。

5.4.2 有关数据输入输出的概念

5.4.3 用 printf 函数输出数据

5.4.4 用 scanf 函数输入数据

5.4.5 字符数据的输入输出

除了可以用 printf 函数和 scanf 函数输出和输入字符外，C 函数库还提供了一些专门用于输入和输出字符的函数。他们是很容易理解和使用的。

1. putchar
2. getchar

5.5 习题

1. 写两个函数，分别求两个整数的最大公约数和最小公倍数，用主函数调用这两个函数，并输出结果。两个整数由键盘输入。
2. 求方程 $ax^2 + bx + c = 0$ 的根，用 3 个函数分别求当：$b^2 - 4ac$ 大于 0、等于 0 和小于 0 时的根并输出结果。
3. 写一个判素数的函数，在主函数输入一个整数，输出是否为素数的信息。
4. 写一个函数，使给定一个 3×3 的二维整型数组转置，即行列互换。
5. 写一个函数，使输入的一个字符串按反序存放，在主函数中输入和输出字符串。

Chapter 6

选择结构程序设计

第 3 张介绍了顺序结构程序设计。在顺序结构中，各语句是按自上而下的顺序执行的，执行上一个语句就自动执行下一个语句，是无条件的。实际上……需要根据某个条件是否满足来决定是否执行指定的操作任务，或者从给定的两种或多种操作选择其一。

6.1 选择结构和条件判断

6.2 用 if 语句实现选择结构

6.3 关系运算符和关系表达式

6.3.1 关系表达式

6.4 逻辑运算符和逻辑表达式

6.4.1 逻辑运算符机器优先次序

6.4.2 逻辑表达式

6.4.3 逻辑型变量

如果源文件中用`#include<stdbool.h>`，那么上面的程序段可以写成……

6.5 条件运算符和条件表达式

6.6 选择结构的嵌套

6.7 用 switch 语句实现多分支选择结构

6.8 选择结构程序综合举例

Chapter 7

循环结构程序设计

7.1 用 while 语句实现循环

7.2 用 do whiel 语句实现循环

7.3 用 for 语句实现循环

7.4 循环的嵌套

7.5 几种循环的比较

7.6 改变循环执行的状态

7.6.1 用 break 语句提前终止循环

7.6.2 用 continue 语句提前结束本次循环

7.6.3 break 语句和 continue 语句的区别

7.7 循环程序举例

例用 $\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots$ 公式求 π 的近似值，直到发现某一项的绝对值小于 10^-6 为止。

Chapter 8

利用数组处理批量数据

8.1 怎样定义和引用一维数组

一维数组是数组中最简单的，它的元素只需要用数组名加一个下标……

8.1.1 这样定义一维数组

8.1.2 怎样引用一维数组元素

8.1.3 一维数组的初始化

8.1.4 一维数组程序举例

例用数组处理求 Fibonacci 数列问题。

8.2 怎样定义和引用二维数组

8.3 字符数组

Chapter 9

用函数实现模块化程序设计

Chapter 10

善于利用指针

指针是 C 语言中一个重要的概念，也是 C 语言的一个重要特色。正确而灵活地运用它，可以使程序简洁、紧凑、高效。没一个学习和使用 C 语言的人，都应该深入地学习和掌握指针。

10.1 指针是什么

10.2 指针变量

10.3 通过指针引用数组

10.4 通过指针引用字符串

10.5 指向函数的指针

10.6 返回指针值的函数

10.7 指针数组和多重指针

10.8 动态内存分配与指向它的指针变量

Chapter 11

用户自己建立数据类型

C 语言提供了一些由系统已定义好的数据类型，如：int，float，char 等，用户可以在程序中用它们定义变量，解决一般的问题，但是人们要处理的问题往往比较复杂，只有系统提供的类型还不能满足应用的要求，C 语言允许用户根据需要自己建立一些数据类型，用它来定义变量。

11.1 定义和使用结构体变量

11.1.1 自己建立结构体类型

在前面所见到的程序中，所有的变量大多数是相互独立、无内在联系的……C 语言允许用户自己建立由不同数据组成的组合型的数据结构，它成为结构体（structre)。在其他一些高级语言中称为“记录”(record)。

11.1.2 定义结构提类型变量

前面只是建立了一个结构体类型，它相当于一个模型，并没有定义变量，其中并无具体数据，系统对之也不分配存储单元。为了能在程序中使用结构体类型的数据，应当定义结构体类型的变量，并在其中存放具体的数据。

1. 先声明结构提类型，再定义该类型的变量

   前面已经声明了一个结构体类型 struct Student，可以用它来定义变量。

2. 在声明类型的同时定义变量

```
struct Student
{int num;
char name[20];
char sex;
int age;
float score;
char addr[30];
} student1, student2;
```

3. 不指定类型名而直接定义结构提类型变量

```
struct
{
        成员列表
        }变量名列表;
```

显然不能再以此结构体类型去定义其他变量。这种方式用得不多。

11.1.3 结构体变量的初始化和引用

在定义结构提变量是，可以对它初始化，即赋予初始值。然后可以引用这个变量。

```
struct Student
{long itn num;
} a = {1};
```

例输入两个学生的学好、姓名和成绩，输出成绩较高的学好的学号、姓名和成绩。

```
struct Person
{char name[20];
int count;
}leader[3] = {"Li", 0, "Zhang", 0, "Sun", 0};
int main(void)
{
        ...
}
```

11.2 使用结构提数组

一个结构提变量中可以存放一组有关联的数据。如果有 10 个学生的数据需要参加运算，显然应该用数组，这就是结构体数组。结构提数组与以前介绍过的数值型数组的不同之处在于每个数组元素都是一个结构体类型的数据。

11.2.1 定义结构提数组

下面举一个简单的例子来说明定义和引用结构体数组。

例有 3 个候选人，每个选民只能投票选一个人，要求编一个统计选票的程序，先后输入被选人的名字，最后输出各人得票结果。

11.2.2 结构体数组的应用举例

例有 n 个学生的信息，要求按照成绩的高低顺序输出各学生的信息。

11.3 结构体指针

所谓结构提指针就是指向结构体变量的指针，一个结构提变量的起始地址就是这个结构提变量的指针。

11.3.1 指向结构提的指针

指向结构体对象的指针即可指向结构体变量，也可指向结构体数组中的元素。指针变量的基类型必须与结构体变量的类型相同。

11.3.2 指向结构提数组的指针

可以用指针变量指向结构体数组的元素。

11.3.3 用结构提变量和结构提变量的指针作函数参数

将一个结构提变量的值传递给另一个函数，有 3 个方法:

1. 用结构体变量的成员作参数。
2. 用结构体变量作实际参数。
3. 用指向结构体变量的指针作实际参数。

11.4 用指针处理链表

11.4.1 什么是链表

链表是一种常见的重要的数据结构。它是动态地进行分配的一种结构。

11.4.2 建立简单的静态链表

11.4.3 建立动态链表

11.5 共用体类型

11.5.1 什么是共用体类型

11.6 使用枚举类型

11.7 用 typedef 声明新类型名

除了可以直接使用 C 提供的标准类型名和程序编写者自己声明的结构体、共用体、枚举类型外，还可以用 typedef 指定新的类型名来代替已有的类型名。有以下两种情况：

1. 简单地用一个新的类型名代替原有的类型名。
2. 命名一个简单的类型名代替复杂的类型表示方法。

11.8 习题

1. 定义一个结构提变量（包括年、月、日）。计算该日在本年中是第几天，注意润年问题。
2. 写一个函数 days，是想第 1 题的计算。
3. 编写一个函数 print，打印一个学生的成绩数组，该数组中有 5 个学生的数据记录，每个记录每个记录包括 num, name, `score[3]`，用主函数输入这些记录，用 print 函数输出这些记录。
4. 在第 3 提的基础上，编写一个函数 input，用来输入 5 个学生的数据记录。
5. 有 10 个学生，每个学生的数据包括学好、姓名、3 门课程的成绩，从键盘输入 10 个学生数据，要求输出 3 们课程总平均成绩，以及最高分的学生的成绩（包括学号、姓名、3 门课程成绩、平均分数）。
6. 13 个人围成一圈，从第 1 个人开始顺序报号 1, 2, 3. 凡报到 3 者退出圈子。找出最后留在圈子中的人原来的序号。要求用链表实现。
7. 在第 9 章例 9.9 和例 9.10 的基础上，写一个函数 del，用来删除动态链表中的制定节点。
8. 写一个函数 insert，用来向一个动态链表插入结点。
9. 综合本章例 9.9 和本章习题第 7 题，在写一个主函数，先后调用这个函数。用以上 5 个函数组成一个程序，实现链表的建立、输出、删除和插入，在主函数中制定需要删除和插入的结点的数据。
10. 已有 a, b 两个链表，每个链表中的结点包括学好、成绩。要求把两个链表合并，按学号升序排序。
11. 有两个链表 a 和 b，设结点中包含学号、姓名。从 a 链表中删去与 b 链表中有相同学号的那些结点。
12. 建立一个链表，每个结点包括：学好、姓名、性别、年龄。输入一个年龄，如果链表中的结点所包含的年龄等于此年龄，则将此节点删去。

Chapter 12

对文件的输入输出

12.1 C 文件的有关基本知识

凡是用过计算机的人都不会对“文件”感到模式，大多数人都接触过或使用过文件。

12.1.1 什么是文件

文件有不同的类型，在程序设计中，主要用到的两种文件：

1. 程序文件。
2. 数据文件。