江涛　钟宏 编著

C 语言版

数据结构

科学出版社

数 据 结 构

C 语言版

江涛 钟宏 编著

（京）新登字 092 号

内 容 简 介

数据结构教科书已出版多年，但直至目前为止其内容仍采用类 Pascal 语言，而作为教学语言的 Pascal 正在或已被流行的 C 语言所替代。本书正是用 C 语言描述数据结构及其算法。本书以数据元素之间的“1 对 1”、“1 对多”及“多对多”关系为主要线索，用 C 语言由简到繁地描述了：线性表、树及图等几种基本数据结构，以及查找、排序等常用算法和文件组织。该书重点介绍基本数据结构的存贮结构及其基本运算的实现。

本书虽为高等院校计算机专业必修课的教科书，但经过适当删节即可作为大学专科、成人教育用书，亦可作为自学参考书。

数据结构

C 语言版

江涛　钟宏　编著

责任编辑　徐津津

科学出版社出版

北京东黄城根北街 16 号

邮政编码：100717

地矿部航空物探遥感中心　印刷

新华书店北京发行所发行　各地新华书店经销

*

1995 年 2 月第　一　版　　开本：787×1092 1/16

1995 年 2 月第一次印刷　　印张：16 ½

印数：1—5000　　字数：373 000

ISBN 7-03-00 4338-3/TP・392

定价：12.80 元

前 言

数据结构是计算机科学技术专业的基础。计算机科学技术的各个领域都要用到多种数据结构。在我国它不仅是各级各类计算机专业教学计划中的一门核心课程，而且是一些非计算机专业的主要先修课程之一。数据结构主要的先修课程有：离散数学、C 语言程序设计以及计算机存贮器的存、取机理。它既是程序设计，特别是非数值性程序设计的基础，又是设计和实现编译程序、操作系统、数据库管理系统以及其它系统程序和大型应用程序的重要基础。

在 Pascal 语言作为教学语言时期，**数据结构**几乎都是用 Pascal 语言描述其算法。C 语言诞生后，以其功能完备、及其可移植性强等特征，逐渐为计算机领域中的广大科技工作者所偏爱。因此，目前高等学校不仅在计算机专业开设了 C 语言课程，而且在一些非计算机专业也开始设置了 C 语言课程。作为教学用的语言，C 语言已基本代替了 Pascal 语言。C 语言版的数据结构一书，正是由这一背景的驱使而产生。

本书充分利用了 C 语言这一强大工具，采用这种“高级汇编语言”对各种数据结构及其基本算法进行描述和实现。不仅使读者掌握了数据结构的高层次应用，而且加深了读者对数据结构在更接近硬件机构上实现的理解。为了使读者能顺利学习本书的其它章节，在第一章中，对 C 语言做了简明扼要的介绍，以期使初学者在较短的时间内掌握对小型 C 语言程序的阅读和编程能力，或为今后更进一步学习 C 语言提供一个入门。

本书其它章节的安排，是以数据元素之间的关系：“一对一”关系（即线性表）、“一对多”关系（即树型结构）和“多对多”关系（即图结构）为主线索，由简到繁地，按照它们的定义、逻辑结构、存贮结构、基本运算、算法分析、应用举例以及习题等基本环节为顺序，由浅入深，循序渐进地组织了各章节的内容。只要沿着上述的主要线索，并抓住每一基本环节加深理解，我们就能掌握该课程的最基本、也是最重要的内容。对线性表的特例——栈和队列，由于它们在很多场合占有重要地位，因而在线性表一章中重点介绍了它们的应用。链接表——作为动态存贮分配的基础，书中专门开辟了章节。在第五章中，重点介绍了二叉树的性质、遍历算法以及一般树和二叉树之间的转换规则。这样一来，使得千姿百态的树型结构的研究规范化。二叉树的遍历算法即成为树型结构中最基础的算法。同样，拓扑排序则是第六章 AOV 网和 AOE 网在大型工程网络结构应用中算法的基础。此外，以数据结构和算法相结合，从解决实际问题出发，重点介绍了在数据处理领域中占有举足轻重份量的查找和排序的各种算法，并对每种算法进行了粗略分析。随着进入计算机系统中的数据量日益增大，本书对外存上的文件组织形式及外排序方法也做了梗概地介绍。

上述各章节内容的安排及有关内容所处的地位和作用，是**数据结构**课程的主要内容及重要环境。也是作者历经多年教学后所获得的一点认识，以求读者共识。

本书可作为大专院校计算机及其应用专业的教材，也可供从事计算机工程及其应用

工作的广大科技工作者参考。

本书由北京理工大学计算机系江涛教授和钟宏副教授合作编写，(其中第三章由钟宏执笔，其余各章由江涛执笔)。全书由李书涛副教授审阅。在整个编写过程中，作者参阅了各种数据结构的教材及专著，并引用了众多作者的精彩论述，在书后所提供的参考文献只列出了其中的主要部分。在此谨向他们表示衷心的感谢。太原机械学院王敏光同志在攻读硕士期间为本书的第一章和第四章提供了修改稿。91级张云泉、李树学等同学在算法描述和文稿录入方面做了大量工作。借此机会向上述同志致以衷心感谢！

受作者水平所限，差错难免，敬请读者及同行们批评指正。

作者 1994年8月 北京

目　录

第一章　C语言简介

第二章 数据结构引论

第三章 线性表与数组

第四章 线性链表

第五章 树

第六章 图

第七章 查找

第八章　排序

第九章　文件

第一章　C语言简介

C语言是一种通用的结构程序设计语言，其表达简洁，具有先进的控制结构和数据结构及丰富的操作符。C语言紧紧地把握住了现代计算机系统结构的公共特征，使其适用于从微型机，小型机直至大型机等各类处理机。它使小巧、快速的程序不做任何改动即可运行在所有这些计算机上。C语言的特点和它强有力的功能，使它得以流行并广泛应用于计算机的各个领域。

本书充分利用了C语言这一强大工具，做为程序描述语言。使用这种“高级汇编语言”对各种数据结构进行描述和实现，不仅使读者掌握了数据结构的高层次应用，而且加深了读者对数据结构在更接近硬件的较低层机构上实现的理解。因此，为使读者能顺利学习本书的其它章节，在此专门开辟一章，对C语言进行简要明了的介绍，使读者在较短的时间内掌握对小型C语言程序的阅读和编写能力，或为读者今后更进一步学习C语言提供了一个入门。

1.1　C语言的结构

1.1.1　C语言的特点

C语言在许多方面优于其它程序设计语言，它具有很强的语言表达能力，并提供了丰富的运算符，使得不论是用来编写操作系统、编译程序这样的系统软件，还是用来编写用于统计、计算的应用程序都游刃有余。以下简要列举C语言特点的几个方面：

1. C语言可移植性好。C语言程序可以从某一环境不加或稍加改动就可搬到另一个完全不同的环境上运行，这是最为人们称赞的一点，C语言真正实现了一个程序仅写一次的理想。

2. C语言执行速度快。C语言可以直接存取内存中的数据以及完成通常由硬件实现的算术、逻辑运算。在多数情况下，一个优秀的C语言程序在速度上可与汇编程序媲美，它足以有效地取代汇编语言来编写各种软件。

3. C语言提供了丰富的数据构造类型，如数组、结构、联合等，用以实现对复杂的数据类型的描述。

4. C语言的另一个优点是具有高级程序结构性，它包含了完整的控制流程及很好的函数连接关系，使你能把一段程序看成是一个函数，并执行其特定工作；而C程序本身只不过是如何安排这些函数以组合成我们所要的程序功能而已，这种模块化的设计方式使得程序在调试和维护上更容易。

5. C语言所提供的指针是一有力工具，它能直接的接触到存贮器的每一点，并且能和各种数据类型交互作用，创高级语言之先例。

6. C语言易学易懂且具有高度的灵活性、通用性，你可用C语言来写各种各样的程序，它不仅可以编写操作系统、编译程序等系统软件，而且可以编写CAD、数据处理、过程控制、信息通讯以及人工智能、解计算机病毒等各种应用软件。例如，许多传真机、复

印机上的控制软件是用C语言所编写的。

7. C语言提供的预处理器对使用者尤为方便，它所提供的 #include 和 #define，#ifdef，……等均可使你制作的程序更简洁、更灵活、易懂、易测试。

C语言的设计原则是提供最大的功能而仅用最少的系统资源，它是一种内在很低级的程序语言，但实质外在上它的能力却丝毫不逊于目前一些最高级的程序语言。所谓"内在低级"是指C语言能很经济很有效地将程序、数据转换成几乎一一对应的机器语言，提供了系统观点上和使用者观点上方便的折衷。因此不管是在程序员所花费的时间、精力上还是在占用系统存储空间和执行时间上，C都给人以很好的印象。

1. 1. 2 一个简单的C语言程序

为了能更快地进入角色，我们通过一个简单的C语言例题，给出一些有关C语言的概念。以下为这个例题的源程序代码：

```
/* Program to print 1776 in Roman numerals */
main ( )
  {
  int a=1776;
  a=romanize (a, 1000,'m');
  a=romanize (a, 500,'d');
  a=romanize (a, 100,'c');
  a=romanize (a, 50,'l');
  a=romanize (a, 10,'x');
  a=romanize (a, 5,'v');
  a=romanize (a, 1,'i');
  putchar ('\n');          /* end with a new line */
  }
romanize (i, j, c)
  char c;
  int i, j;
   {
  while (i>=j)
       {
       putchar (c);
       i=i-j;
       }
  return (i);
  }
```

此程序执行后，将在屏幕上显示如下结果：

```
mdcclxxvi
```

读者也许不知道程序所显示的结果是什么意义，mdcclxxvi 其实是罗马数字的1776，程序的目的就是将阿拉伯数字转换成罗马数字。

现在我们来说明一下这个程序是如何工作的。基本上它是由多次的减法运算所组成。开始它试着从 1776 中减去 1000。只要够减的话，程序每减一次，就显示出一个罗马数字 m。由于 1776 减去 1000 后，只剩下 776，所以程序执行只显出一个 m。接着程序就试着由剩下的数值内减去 500，而且每减一次就印出一个罗马数字 d。以后程序从剩下的数值内减去 100，50，10，1 并显示出相对的 c，l，x，v 和 i 这些罗马数字来。这就是整个程序的工作过程。

1. 函数

整个程序的工作是由两个函数完成的。函数是 C 语言中运算处理的基本单位。每个 C 语言的程序都是由一个以上的函数所构成的。在本例中，构成程序的两个函数分别是 main 和 romanize。在 main 中，主要是安排减法运算的次序和程序的起始和终止。在 romanize 这个函数中做实际的减法运算，并利用一个库函数 putchar 将罗马数字印出。

所有的函数，对 C 而言，都是彼此独立不相关的。但 C 会从这些函数中，选出 main 这个函数来执行。

2. 函数的定义

在 C 语言中，提供了很多类似 putchar 的库函数供我们使用。但是，在此我们所要讨论的是有关使用者自行定义的函数。

所有库函数是 C 语言提供的，使用者可以直接调用，而所有使用者自行定义的函数，都必须编写之后才能使用。因此，我们将函数的编写称之为“函数的定义”。

一个函数可以分为函数的首部和主体二部分来看。函数的首部定义函数的名称和相关的参数，而函数的主体则定义了函数的工作和处理内容。

现在，我们来看看本例中 romanize 这个函数所构成的内容。romanize 这个函数首部如下所示：

```
romanize (i, j, c)
 char c;
 int i, j;
```

在第一行语句中，定义了函数的名字 romanize 和此函数的参数 i，j，c。我们先暂时将这些参数视为其它函数所传送过来的数据。其后的两行语句的作用是告诉 C 编译器有关参数占存储空间的大小。

读者可以想到，main 这个函数是不需要任何参数的。实际上，也没有人规定一个函数一定要有参数存在。如果一个函数没有参数部分的话，数据通常来自外部——如键盘输入，从磁盘读入，或是在函数内部自己定义。

函数的主体部分，也称函数体，函数体必须用一对花括号括起来。花括号内是一组可执行语句。函数体也可以是个空体，但花括号不能缺少。

如下是 romanize 函数的主体部分：

```
{
  while (i>=j)
   {
  putchar (c);
  i=i-j;
```

```
    }
return (i);
}
```

各语句都以一个分号来加以分隔。

许多行语句也可组合成一个复合语句，并用括号来加以区别，如：

```
 {
putchar (c);
i=i-j;
}
```

如此，编译过程中就会把这些语句视为一个单元以便将来程序执行时，能一起执行。在本例中，只要（i>=j）这个条件满足，则位于括号内的语句即会不断的执行。

3. 数据

无论哪种程序，都必须有数据的存在。而数据可能有许多种形态，如常数：1776，1000及'm'等，如变量i，j等。一般情况下，变量中所存的值是影响程序执行的最重要因素，因此，变量的起始值应当在程序执行前给定。下面就是在main中，给定变量起始值的语句：

```
int a=1776;
```

这样的语句，在C语言中仍然视之为一项说明，就如同参数的说明：int i，j一样。不过前者多出一项起始值给定的作用。

4. 参数传递

我们前面曾经提过：函数之间是独立不相关的。那么读者要问，main和romanize是如何传递数据的？答案就在于函数的参数和返回值。

现在，我们再一步一步地看这个程序的执行。首先，计算机会去执行main中的第一个语句：

```
int a=1776;
```

这是一个说明语句，它将变量a的初始值定为1776。其次语句：

```
a=romanize (a, 1000,'m');
```

其中1000和'm'分别是数值和字符常数，a是变量。由于函数romanize的参数定义为(i，j，c)，根据排列的次序，我们可以得到下列对应关系：

```
a ⟶i
1000 ⟶j
'm'⟶c
```

romanize (a，1000,'m') 这个表达式是函数调用的一个例子。当程序执行到此处时，就会由main转到romanize去执行。在这同时，a的值1776，1000和'm'会被送到i，j，和c这三个参数里，以便函数romanize的执行。

romanize会由1776内减去1000并显示出'm'；此步骤将重复做到不能减为止。最后，romanize将会将减剩下的值（亦即1776－1000＝776）返回到main内。并赋给变量a即：

```
a=776;
```

romanize 的返回值是通过执行 return (i) 实现的。接下来的程序执行过程，这里将不多做说明；读者可以依据先前的解说做下去。

5. 括号

括号的使用是帮助编译器决定语句执行次序的一个工具。例如：

```
while (i>=j)
    {
    putchar (c);
    i=i-j;
    }
```

和

```
while (i>=j)
     {
    putchar (c);
    }
i=i-j;
```

二者的意义和执行的结果就截然不同。前者以变量 i 的变化为下次是否继续循环的判断依据；而后者循环过程中 i 和 j 的值均不发生变化，很可能导致程序死循环。

6. 注释

我们看到注释在程序中有如下形式：

```
/* Program to print 1776 in Roman numerals */
```

程序的注释只是帮助我们了解程序意义的一项工具，所以 C 语言编译器并不会加以处理而直接跳过。注释部分必须放在/*和*/二组符号之间。注释的行数并无限制。虽然注释并无实际上的意义，但是一般来说，一个函数内如果没有注释的说明，则并不能算是完整。

另外，main 执行的最后一个语句是调用 putchar，负责输出一个特别的符号 '\n'。这个符号的作用是产生一个换行的控制信号。

7. 函数 getchar ( ) 和 putchar ( )

getchar ( ) 的作用是从标准输入设备上输入一个字符。标准输入设备一般是用户使用的键盘。

putchar (c) 的作用是将字符型变量 c 代表的字符输向标准输出设备。标准输出设备一般是用户使用的显示器。

8. 格式化输入输出

标准 C 语言库中包括两个对基本数据类型进行格式化输入输出的函数。它们是 printf ( ) 和 scanf ( )。函数 printf ( ) 把数据输出在显示屏幕上，而 scanf ( ) 是从键盘读入数据。所谓格式化是指这两个函数可以在你的控制下读写各种格式的数据。

printf ( ) 的一般格式为：

```
printf ("控制串", 参数表);
```

其中参数表为由要输出的变量和常数组成。控制串中包括两类内容。第一类是直接在屏幕上显示出来的字符，第二类是规定变量显示方式的格式命令。格式命令用%开头，再

加上一个格式代码。下表给出常用的格式代码：

%c	单个字符
%s	字符串
%d	带符号十进制整数
%i	带符号十进制整数
%f	浮点数（十进制表示形式）
%e	浮点数（幂指数表示形式）
%g	浮点数（%f 或%e，无论那个更短）
%u	无符号十进制整数
%x	无符号十六进制整数
%o	无符号八进制整数

实际变量的数目要和格式命令中的完全对应。例如，下面的调用将显示" Hi c 10 there!"

```
printf ( "Hi %c%d%s",'c', 10, "there!");
```

scanf ( ) 的一般格式为：

```
scanf ("控制串", 参数表)
```

其中控制字符串中包括三类字符：

. 格式说明符
. 空白字符
. 非空白字符

格式说明符用%开头，指明下一个要读入的数据类型。这些内容在下面的表中给出。例如，%s 表示读入字符串，而%d 表示读入整数。

%c	读一个字符
%d	读一个十进制整数
%i	读一个十进制整数
%l	读一个浮点数
%f	读一个浮点数
%s	读一个字符串
%p	读一个指针
%x	读一个十六进制数

控制串中的空白字符使 scanf ( ) 跳过输入流中的一个或多个空白字符。所谓空白字符是空格，制表符或换行符。实际上，控制串中的一个空白符会让 scanf ( ) 跳过一系列空白字符，直到遇到一个非空白字符为止。

非空白字符使 scanf ( ) 读入并抛弃匹配的字符。例如，"%d,%d" 首先读入一个整数，然后读入并抛弃一个逗号，最后再读入一个整数。

所有在 scanf ( ) 中接收值的变量都必须用地址引用。也就是说，所有变量都必须用指针形式。指针我们将在第六节讨论。这里只要求读者搞清 scanf ( ) 的格式。例如下面调用：

```
scanf ("%d%d", &r, &c);
```

其中 &r 和 &c 是变量 r 和 c 的指针形式，上面的调用可使 r 和 c 分别接收一个整数。

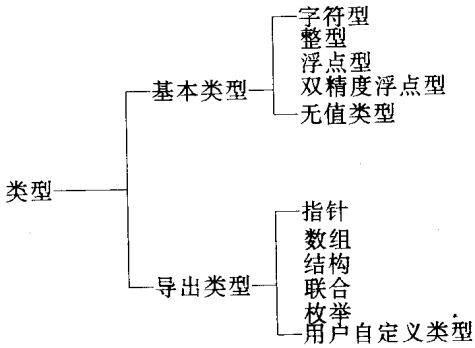
1. 2 数据类型、运算符和表达式

1. 2. 1 数据类型

在一个 C 语言程序中所用的每个常数、变量和函数是程序中的基本操作对象，它们都以隐式或显式与一种数据类型相联系。每种数据类型表明它的可能取值和能在其上进行哪些运算。

常数的类型是由该常数本身隐含的，如 3 的类型是 int（整型）；3.14159 的类型是 float（浮点型）。变量的类型或在程序的说明部分显式地说明；或隐式地不予说明，此时隐含该变量类型为 int 类型。函数的类型也是显式或隐式地说明的。

C 语言中的类型可划分为：基本的和导出的，具体如下：



本节将介绍五种基本类型，其它类型将在以后几节中叙述。

C 语言提供的五种基本类型为：

char ：字符型

int ：整型

float ：浮点型

double：双精度浮点型

void ：无值类型

无值类型 void 有两个用处：一是表示一个函数没有返回值，二是用来指明一个通用型指针。

除了 void 型外，各种基本类型都可以在前面加上不同的修饰符，用来改变基本类型的含义，使之适合更细致的场合。下面列出一些修饰符：

signed

unsigned

long

short

各种数据类型所占位数和取值范围与所使用机器和 C 语言的版本有关。下面我们列出 ANSI（美国国家标准）中所允许的各种组合，这里假定处理器是 16 位字长。

类型	位数	取值范围
char	8	ASCII 字符
unsigned char	8	0～255