普通高等教育“十一五”国家级规划教材 计算机系列教材

教育部—微软精品课程建设立项项目

C/C++与数据结构（第3版）（下册）

王立柱 编著

清华大学出版社

普通高等教育“十一五”国家级规划教材　计算机系列教材

教育部—微软精品课程建设立项项目

王立柱 编著

C/C++与数据结构(第3版)(下册)

清华大学出版社

北京

内 容 简 介

本书(下册)共9章,从第26～第34章,涵盖了二叉树、堆、树、图、二叉搜索树、平衡二叉搜索树、B树、散列和排序等主要内容。基于上册已经包含了C++基础,模拟的C++新标准中的Vector、List、String等数据结构线性部分,通用算法和迭代器等内容,本书集中讨论了数据结构的非线性部分,并利用C++实现了全部算法。

本书可以作为C语言和C++语言的本科或专科教材,也可以作为计算机爱好者和程序员的自学教材或参考书。

图书在版编目(CIP)数据

C/C++与数据结构.下册 / 王立柱编著. —3版. —北京:清华大学出版社,2009.9
(计算机系列教材)
ISBN 978-7-302-20067-3

Ⅰ.C… Ⅱ.王… Ⅲ.①C语言－程序设计－高等学校－教材 ②数据结构－高等学校－教材 Ⅳ.①TP312 ②TP311.12

中国版本图书馆CIP数据核字(2009)第065999号

责任编辑:战晓雷　赵晓宁
责任校对:时翠兰
责任印制:杨　艳

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:北京市清华园胶印厂
装 订 者:三河市溧源装订厂
经　　销:全国新华书店
开　　本:185×260　　印　张:10.25　　字　数:242千字
版　　次:2009年9月第3版　　印　次:2009年9月第1次印刷
印　　数:1～3000
定　　价:17.00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:028363-01

读者意见反馈

亲爱的读者：

感谢您一直以来对清华版计算机教材的支持和爱护。为了今后为您提供更优秀的教材，请您抽出宝贵的时间来填写下面的意见反馈表，以便我们更好地对本教材做进一步改进。同时如果您在使用本教材的过程中遇到了什么问题，或者有什么好的建议，也请您来信告诉我们。

地址：北京市海淀区双清路学研大厦 A 座 602　　计算机与信息分社营销室 收

邮编：100084　　电子邮件：jsjjc@tup.tsinghua.edu.cn

电话：010-62770175-4608/4409　　邮购电话：010-62786544

教材名称：C/C++与数据结构（第 3 版）（下册）

ISBN：978-7-302-20067-3

个人资料

姓名：_____ 年龄：_____ 所在院校/专业：_____

文化程度：_____ 通信地址：_____

联系电话：_____ 电子信箱：_____

您使用本书是作为：□指定教材 □选用教材 □辅导教材 □自学教材

您对本书封面设计的满意度：

□很满意 □满意 □一般 □不满意　改进建议_____

您对本书印刷质量的满意度：

□很满意 □满意 □一般 □不满意　改进建议_____

您对本书的总体满意度：

从语言质量角度看 □很满意 □满意 □一般 □不满意

从科技含量角度看 □很满意 □满意 □一般 □不满意

本书最令您满意的是：

□指导明确 □内容充实 □讲解详尽 □实例丰富

您认为本书在哪些地方应进行修改？（可附页）

_____

_____

您希望本书在哪些方面进行改进？（可附页）

_____

_____

电子教案支持

敬爱的教师：

为了配合本课程的教学需要，本教材配有配套的电子教案（素材），有需求的教师可以与我们联系，我们将向使用本教材进行教学的教师免费赠送电子教案（素材），希望有助于教学活动的开展。相关信息请拨打电话 010-62776969 或发送电子邮件至 jsjjc@tup.tsinghua.edu.cn 咨询，也可以到清华大学出版社主页（http://www.tup.com.cn 或 http://www.tup.tsinghua.edu.cn）上查询。

计算机系列教材

主编：周立柱　副主编：王志英、李晓明

计算机组织与体系结构(第4版 立体化教材)　白中英
计算机操作系统教程(第3版)　张尧学、史美林、张高
计算机操作系统教程(第3版)习题解答与实验指导　张尧学
计算机组成与设计　薛宏熙
数字逻辑设计　薛宏熙、胡秀珠
数据库系统设计与原理(第2版)　冯建华、周立柱
数据库专题训练　冯建华、周立柱
C/C++与数据结构(第3版)上册　王立柱
C/C++与数据结构(第3版)(上册)习题解答与实验指导　刘志红、薛其华
C/C++与数据结构(第3版)(下册)　王立柱
计算机组成原理与汇编语言　易小琳
高档微机原理与技术　毛国君、方娟
微型机原理及技术(第2版)　戴梅萼、史嘉权
微型机原理及技术——习题、实验和综合训练题(第2版)　戴梅萼、史嘉权

本书（下册）内容是在上册已经包含了C++基础，模拟的C++新标准中的Vector、List、String等数据结构线性部分，通用算法和迭代器内容的基础上组织的，主要讨论数据结构非线性部分，共9章，从第26～第34章，涵盖了二叉树、堆、树、图、二叉搜索树、平衡二叉搜索树、B树、散列和排序，并利用C++实现了全部算法。这是与第2版教材的主要区别。

本书将一些经典的算法纳入相应的经典结构中进行描述。例如，快速排序和幂集归入二叉树前序遍历，汉诺塔问题并入二叉树中序遍历，哈夫曼树是堆类的直接应用，八皇后和迷宫分别在树和图的深度优先遍历中解决。本书中的算法描述清晰紧凑，代码简洁流畅，既保留了第2版的特点，又因为改用C++语言描述而有所不同。

算法性能分析历来是学习数据结构的难点之一，本书将它放在了最后一章，和排序算法一起讨论，使读者可以首先集中精力熟悉和理解各种算法设计和代码实现，然后在充分的感

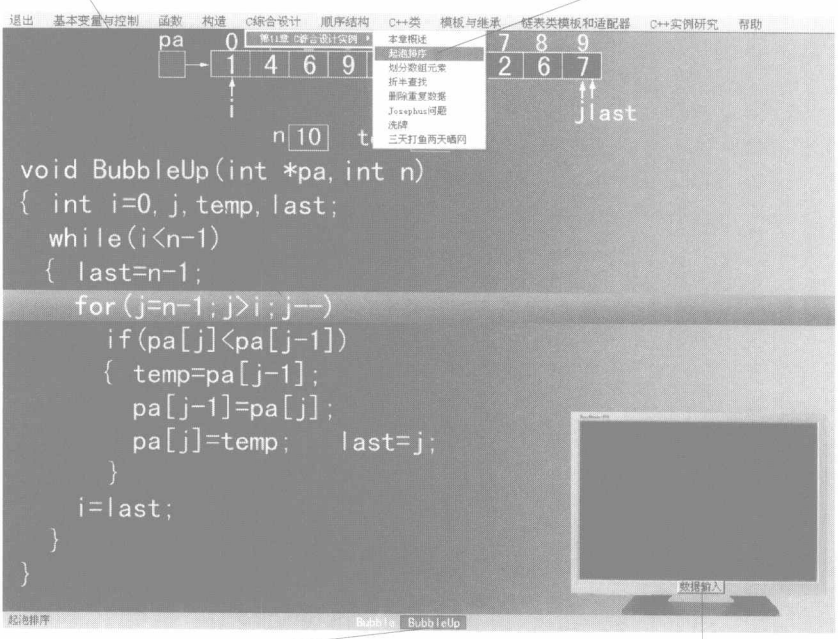


图1 多媒体软件示例

性认识基础上，以综合提高的姿态，以排序为典型内容，学习和掌握算法的时间和空间复杂度分析技术。这是第2版教材没有的内容。

本书的图类综合了Vector、List、适配器、堆方法和C++文件读写方法等多项内容，增强了教材内容的连贯性和整体性。

和上册一样，本书配备了多媒体课件(如图1所示)，既可助教又可助学，使复杂和抽象的算法变得直观而简单，尤其对平衡二叉树、图的最短路径、单元最短路径等难度很大的算法，效果更加明显，对初学者更是意义非凡。

本书的源代码和课件可以从精品课网站上直接下载，网址：http://59.67.71.237:8080/dsc。

读者如有问题或发现错误，欢迎直接与作者联系，作者将不胜感激。

E-mail：tjwanglizhu@163.com。

王立柱

2009年3月

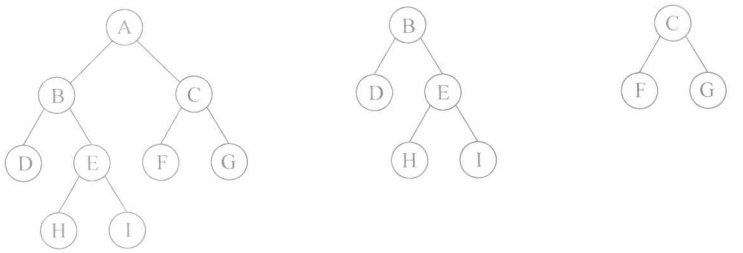
第 26 章　二　叉　树

利用计算机处理问题的算法 90%以上都是用非线性结构描述的。非线性结构不仅是许多现实问题的真实的描述，也是提高算法性能的最有效的手段。

与线性结构不同，非线性结构中的元素可以有多个前驱和后继。二叉树是最简单的非线性结构，它从思想和方法两个方面提供了进一步研究其他非线性结构的基础。

26.1　二叉树的基本概念

二叉树是具有 $n(n\geqslant 0)$ 个结点(node)的集合，每个结点最多有一个前驱、两个后继。只有一个结点无前趋，这个结点称为**根**。没有后继的结点称为**叶子**(leaf)或**终端结点**。叶子以外的结点称为**分支结点**(branch)。一个结点的后继称为该结点的**孩子**(child)，而且有左右之分，分别称它们为该结点的**左孩子**(left)和**右孩子**(right)，即使只有一个后继，也要区分是左孩子还是右孩子。一个结点的前驱称为该结点的**双亲**(parent)。一般用倒垂的树形(即根在上)来表示二叉树的抽象结构。如图 26.1 所示，其中 A 是根。



(a) 以 A 为根的二叉树　　(b) A 的左子树　　(c) A 的右子树

图 26.1　二叉树

一棵二叉树，以根结点以外的任何一个结点为根也是一棵二叉树，这棵二叉树称为真子树。对二叉树的一个结点，分别以其左孩子或右孩子为根的子树称为该结点的**左子树**或**右子树**(见图 26.1(b)和图 26.1(c))。

若树中存在一个结点序列$<K_0,K_1,K_2,\cdots,K_n>$，且 K_i 是 K_{i+1} 的双亲，则称该序列是从 K_0 到 K_n 的一条**路径**或**道路**，路径长度为 n。若从结点 K_i 到 K_j 存在一条路径，则称 K_i 是 K_j 的**祖先**，K_j 是 K_i 的**子孙**。其中，<A,B,E,I>和<B,E,I>分别是长度为 3 和 2 的路径，A 和 B 都是 I 的祖先。

从根到一结点的路径长度称为**该结点层数**，树中最大的结点层数称为**树的高度**或**深度**。结点 E 的层数是 2，树的深度是 3。

26.2 二叉树的性质

性质 1 二叉树第 i 层上的结点数目最多为 2^i。

当 $i=0$ 时，$2^i=2^0=1$，结果正确，因为第 0 层只有一个根结点。

假设第 i 层上的结点最多为 2^i，结论正确，那么第 $i+1$ 层上的结点最多为 $2*2^i=2^{i+1}$，结论也正确。

性质 2 深度为 K 的二叉树至多有 $2^{k+1}-1$ 个结点。

按每一层最多的结点数求和，便是树的最多的结点数：

$$2^0+2^1+\cdots+2^k=2^{k+1}-1$$

如果深度为 K 的二叉树有 $2^{k+1}-1$ 个结点，则称为**满二叉树**，如图 26.2(a)所示。

对二叉树的结点从上层到下层，从左到右排列（序号从 0 开始），称为二叉树的**层次序列**。

设一棵满二叉树的层次序列为

$$a_0,a_1,\cdots,a_n$$

它满足下列性质：

(1) a_0 是根。

(2) 若 $i>0$，则序号为 $(i-1)/2$ 的结点是 a_i 的双亲。

(3) 若 $2i+1\leqslant n$，则 a_{2i+1} 是 a_i 的左孩子；若 $2i+1>n$，则 a_i 是叶子。

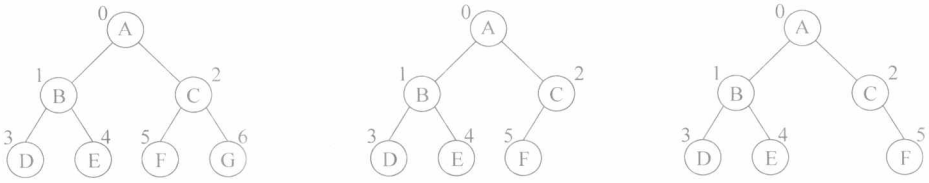
(4) 若 $2i+2\leqslant n$，则 a_{2i+2} 是 a_i 的右孩子；若 $2i+2>n$，则 a_i 无右孩子。

(5) 若 $i\leqslant n$，且 i 是大于 1 的偶数，则 a_{i-1} 是 a_i 的左兄弟。

(6) 若 $i+1\leqslant n$，且 i 是小于 n 的奇数，则 a_{i+1} 是 a_i 的右兄弟。

一棵二叉树，如果其层次序列与满二叉树的层次序列有相同的性质，则称为**完全二叉树**(complete binary tree)，如图 26.2(b)所示。满二叉树是完全二叉树。

图 26.2(c)不是一棵完全二叉树，因为按完全二叉树的要求，序号为 5 的结点 F 应该是 C 的左孩子而不是右孩子。



(a) 满二叉树　(b) 完全二叉树　(c) 非完全二叉树

图 26.2 满二叉树与完全二叉树

性质 3 具有 n 个结点的完全二叉树其深度为$\lfloor \mathrm{lb}n \rfloor$。

设完全二叉树的深度为 k，它从 0(1)层到 $k-1$ 层应该是一棵满二叉树，一共有 2^k-1 个结点，因此 $n>2^k-1$。又因为深度为 k 的满二叉树一共有 $2^{k+1}-1$ 个结点，所以 $n\leqslant 2^{k+1}-1$，综合起来得到不等式

$$2^k-1<n\leqslant 2^{k+1}-1$$

不等式两边加 1

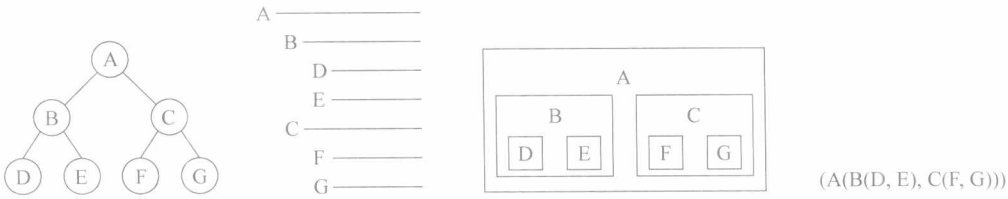
$$2^k \leqslant n < 2^{k+1}$$

取对数

$$k \leqslant \mathrm{lb}n < k+1$$

因为 k 是整数，所以应该是 $k=\lfloor \mathrm{lb}n \rfloor$。

图 26.1 仅仅是树的一种表示法，称为树型表示法，它比较直观，有利于认识树的基本概念。但是，树有各方面的应用，也有各种其他表示法(如图 26.3 所示)。根据不同的应用，选择有针对性的表示法，不仅有助于深入理解和灵活运用基本概念，而且有助于分析和解决实际问题。



(a) 树型表示法　(b) 目录表示法　(c) 集合表示法　(d) 广义表表示法

图 26.3　二叉树的各种表示法

26.3　二叉树的存储结构

26.3.1　二叉树顺序存储

线性结构用数组表示：线性结构中的元素最多只有一个前趋和一个后继，这种逻辑关系可以直接由存储单元的相邻关系来表示。二叉树是非线性结构，一个元素虽然有一个前驱，但是可以有两个后继，把这种关系存储到数组中不是一件容易的事，需要利用完全二叉树的性质。

可以把数组元素看作一棵完全二叉树按层次顺序排列的结点。对于长度为 N 的数组 a，$a[0]$是根结点，$a[i]$的左孩子是 $a[2*i+1]$(如果 $2*i+1<N$)，右孩子是 $a[2*i+2]$(如果 $2*i+2<N$)，$a[i]$($0<i<N$)的双亲是 $a[(i-1)/2]$。具体方法：首先用零元素把二叉树填充为完全二叉树，再将填充后的二叉树结点按层次序列的次序存储到数组中，如图 26.4 所示。

26.3.2　二叉树链式存储

用链式存储表示二叉树是很自然的选择，其中结点结构中除数据域之外，有两个指针域 left 和 right，分别存放该结点的左孩子和右孩子的地址。

二叉树的链式存储称为**二叉链表**，如图 26.5 所示。

二叉链表的结点类定义如下：

```
template<class T>
struct BTNode//Binary Tree Node
```

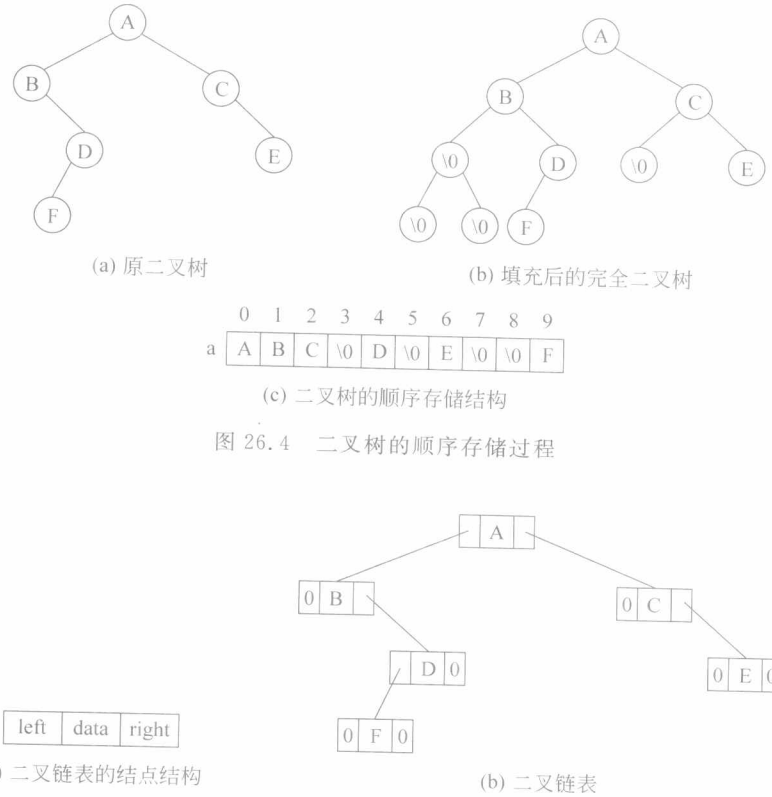


图 26.4　二叉树的顺序存储过程

图 26.5　二叉树的链式存储结构

```
{
    T data;
    BTNode * left, * right;        //指向结点左右孩子的指针
    BTNode(const T& item=T(),BTNode * lptr=NULL,BTNode * rptr=NULL):
            data(item),left(lptr),right(rptr){}
};
template<class T>                  //建立一个二叉树结点的函数
BTNode<T> * GetBTNode(const T& item,BTNode<T> * lp=NULL,BTNode<T> * rp=NULL)
{
    BTNode<T> * p;
    p=new BTNode<T>(item,lp,rp);
    if(p==NULL)
    {
        cerr<<"Memory allocation failure!"<<endl; exit(1);
    }
    return(p);
}
```

应用举例，生成图 26.5(a)。

```
BTNode<char> * nullp=NULL;
```

```
BTNode<char> * fp=GetBTNode('F');
BTNode<char> * dp=GetBTNode('D',fp);
BTNode<char> * bp=GetBTNode('B',nullp,dp);
BTNode<char> * ep=GetBTNode('E');
BTNode<char> * cp=GetBTNode('C',nullp,ep);
BTNode<char> * ap=GetBTNode('A',bp,cp);
```

与模板类的成员函数不同，对模板函数的参量指针不能直接传递指针 0 值(NULL)，除非使用默认值，否则要表明具体类型(上面的举例中，是通过一个值为 0 的指针 nullp)，以便系统解析。例如：

```
BTNode<char> * dp=GetBTNode('D');                  //合法。取缺省 NULL 值
BTNode<char> * dp=GetBTNode('D',nullp,nullp);      //合法。
BTNode<char> * dp=GetBTNode('D',NULL,NULL);        //非法。无法确定值 NULL 类型
```

但是下面句子合法，因为它们都调用构造函数(属于模板类的成员函数)。

```
BTNode<char>d('D');                                //合法。生成一个对象
BTNode<char>d('D',NULL,NULL);                      //合法。生成一个对象
BTNode<char> * dp=new BTNode<char>('D');           //合法。生成一个对象指针
BTNode<char> * dp=new BTNode<char>('D',NULL,NULL); //合法。生成一个对象指针
```

对模板函数和非模板函数的指针参量传递指针 0 值(NULL)的情况对比，如表 26.1 所示。

表 26.1 对模板函数和非模板函数的指针参量赋 0 值的比较

非模板函数的指针参量赋 0 值	模板函数的指针参量赋 0 值

void f1(int * p=NULL)
{
cout<<"f1 being called"<<endl;
}
f1(NULL); //合法
f1(); //合法
	template<class T>
	void g1(T * p=NULL)
	{
	cout<<"g1 being called"<<endl;
	}
	g1(NULL); //非法
	g1(); //非法

void f2(int item,int * p=NULL)
{
cout<<"f2 being called:"<<item<<endl;
}
f2(9,NULL); //合法
f2(9); //合法
	template<class T>
	void g2(T item,T * p=NULL)
	{
	cout<<"g2 being called:"<<item<<endl;
	}
	g2(9,NULL); //非法
	g2(9); //合法

26.4 二叉树层次遍历

所谓**遍历**，指按照某种次序、一次性地访问所有结点。遍历是算法的基础，算法都是在一种遍历的基础上完成的，遍历服务于某一类算法。在线性结构中，遍历的顺序与逻辑顺序