HZ BOOKS
华章教育

高等院校计算机专业人才能力培养规划教材

数据结构与算法

王立柱 编著

DATA S
AND A

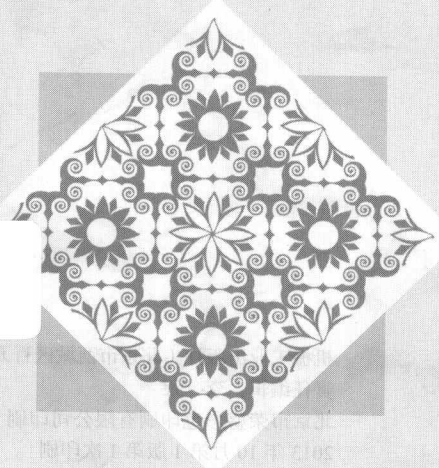
机械工业出版社
China Machine Press

高等院校计算机专业人才能力培养规划教材

数据结构与算法

王立柱 编著

DATA STRUCTURES AND ALGORITHMS



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

数据结构与算法 / 王立柱编著. —北京：机械工业出版社，2013.8
（高等院校计算机专业人才能力培养规划教材）

ISBN 978-7-111-43497-9

Ⅰ. 数… Ⅱ. 王… Ⅲ. ①数据结构－高等学校－教材 ②算法分析－高等学校－教材 Ⅳ. TP311.12

中国版本图书馆 CIP 数据核字（2013）第 175513 号

本书共分 13 章。第 1 章：从 C 到 C++，帮助读者系统复习 C++，同时使本书也适用于只学过 C 语言的读者。第 2 章～第 4 章的内容包括：String 类、向量类模板 Vector、链表类模板和适配器，主要为在自定义 STL 框架下描述数据结构奠定基础。第 5 章～第 13 章的内容包括：二叉树、堆、树、图、二叉搜索树、平衡二叉搜索树、B 树、散列、排序和性能分析，主要是在定义 STL 框架下描述的数据结构。

本书既可以作为高等院校计算机及相关专业本科生的数据结构教材，也可以作为计算机编程爱好者和工程技术人员的自学教材和参考书。

机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码 100037）
责任编辑：李 荣
北京市荣盛彩色印刷有限公司印刷
2013 年 10 月第 1 版第 1 次印刷
185mm × 260mm · 16.75 印张
标准书号：ISBN 978-7-111-43497-9
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
客服热线：（010）88378991 88361066
购书热线：（010）68326294 88379649 68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

出版者的话

机械工业出版社华章公司多年来以“全球采集内容，服务中国教育”为己任，致力于引进国际知名大学广泛采用的计算机、电子工程和数学方面的经典教材，出版了一大批在计算机科学界享誉盛名的专家名著与名校教材，其中包括 Donald E.Knuth、Alfred V. Aho、Jim Gray、Jeffery D. Ullman 等名家的一批经典作品。这些作品为我国计算机教育及科研事业的发展起到了积极的推动作用。

近年来，我们一直关注国内计算机专业教育的发展和改革并大力支持、参与相关的教学研究活动。2006 年，教育部高等学校计算机科学与技术专业教学指导分委员会在对我国计算机专业教育现状和社会对人才的需求进行研究的基础上，发布了《高等学校计算机科学与技术专业发展战略研究报告暨专业规范（试行）》（以下简称《规范》）。为配合《规范》的实施和推广，我们出版了“面向计算机科学与技术专业规范系列教材”。这套教材的推出，对宣传《规范》提出的“按培养规格分类”的理念、推进高校学科建设起到了一定的促进作用。

2007 年，教育部下发了《关于进一步深化本科教学改革全面提高教学质量的若干意见》，强调高等教育以育人为本，以学生为主体，坚持以培养创新人才为重点，下大力气深化教育教学改革。在“质量工程”的思想指导下，各高校纷纷开展了相关的学科改革和教学研究活动。高等学校计算机科学与技术专业的教育开始从过去单纯注重知识的传授向注重学科能力的培养转型。2008 年年底，教育部高等学校计算机科学与技术专业教学指导分委员会成立了“高等学校计算机科学与技术专业人才专业能力构成与培养”项目研究小组，研究小组由蒋宗礼教授（组长）、王志英教授、岳丽华教授、陈明教授和张钢教授组成，研究计算机专业人才基本能力的构成和在计算机专业的主干课程中如何培养这些专业能力。

为配合“高等学校计算机科学与技术专业人才专业能力构成与培养”专项研究成果的推广，满足高校从知识传授向能力培养转型的需求，在教育部高等学校计算机科学与技术专业教学指导分委员会专家及国内众多知名高校专家的指导下，我们策划了这套“高等院校计算机专业人才能力培养规划教材”。这套教材以专项研究的成果为核心，围绕计算机专业本科生应具有的能力组织教材体系。本套教材的作者长期从事教学和科研工作，他们将自己在本科生能力培养方面的经验和心得融入教材的编写中，力图通过理论教学及实践训练，达到提升本科生专业能力的目标。希望这些有益的尝试能对推动国内计算机专业学生的能力培养起

到积极的促进作用。

华章作为专业的出版团队，长久以来遵循着“分享、专业、创新”的价值观，实践着“国际视野、专业出版、教育为本、科学管理”的出版方针。这套教材的出版，是我们以教学研究指导出版的成功范例，我们将以严谨的治学态度以及全面服务的专业出版精神，与高等院校的老师们携手，为中国的高等教育事业走向国际化而努力。

高等院校计算机专业人才能力培养规划教材

编 委 会

丛书序言

作为我国规模最大的理工科专业，计算机本科专业为国家的建设培养了大批人才。2006年，教育部计算机科学与技术专业教学指导委员会发布了《高等学校计算机科学与技术专业发展战略研究报告暨专业规范（试行）》（以下简称《规范》），提出了以“按培养规格分类”为核心思想的专业发展建议，把计算机专业人才划分为研究型、工程型、应用型3个类型。在《规范》的方针指导下，培养合格的计算机本科人才。

教育包括知识、能力、素质三个方面，专业教育不仅要重视知识的传授，更应突出专业能力的培养，实施能力导向的教育。如何以知识为载体实现能力的培养和素质的提高，特别是实现专业能力和素质的提高是非常重要的。对计算机专业本科教育而言，要想实现能力导向的教育，首先要分析专业能力的构成并考虑如何将其培养落实到教学实践中。为此，教育部高等学校计算机科学与技术专业教学指导分委员会开展了计算机科学与技术专业人才专业能力（简称计算机专业能力）的培养研究。该项研究明确了计算机专业本科人才应具有的4大基本能力—计算思维能力、算法设计与分析能力、程序设计与实现能力、系统能力，并将这四大基本能力分解为82个能力点，探讨如何面对不同类型学生的教育需求，在教学活动中进行落实。

为体现研究成果在教学活动中的实现，我们根据《高等学校计算机科学与技术专业人才专业能力构成与培养》，出版了这套教材。本套教材面向高等院校从知识传授向能力培养转型的需求，在内容的选择、体系安排和教学方法上按照专业能力培养的需要进行了探索。其主要特点有：

（1）以教学研究为先导。本套教材以计算机专业能力专项研究成果为基础，体现了先进的教育理念和教学方法，内容选择、知识深度、结构安排更加符合计算机专业教育的需求。

（2）落实能力培养的思想，同时满足课程的要求。本套教材不仅关注知识点的讲授，还凸显能力培养的要求，将能力的培养分解到各门课程的各个知识点的讲授中。

（3）力求贴近教学实际。作者均长期从事实际教学工作且对专业能力培养具有一定研究，教材编写注重科学组织内容、合理安排体系、便于教学实施，更具操作性。

（4）构建立体化教材。为了方便教师的教学活动，配合主教材开发配套的实验教材、教师参考书、学生辅导书、电子课件等教辅资源。

本套丛书的出版是在配合计算机专业能力的培养和落实方面的初步尝试，我们衷心希望本套教材的出版能起到抛砖引玉的作用，也希望广大教育工作者加入到能力培养的研究和实践中来，并对相关的教材建设提出自己的宝贵意见。

丛书编委会

前 言

本书适用于三种读者：只学过 C 的、只学过 C++ 的，或学过 C 和 C++ 的。

第 1 章比较详细地介绍从 C 到 C++ 的过程。通过这一章，只学过 C 的读者，可以很顺利地进入 C++；只学过 C++ 的读者，可以居高临下地学习 C；学过 C 和 C++ 的读者，可以从发展的角度深刻地领会 C++。

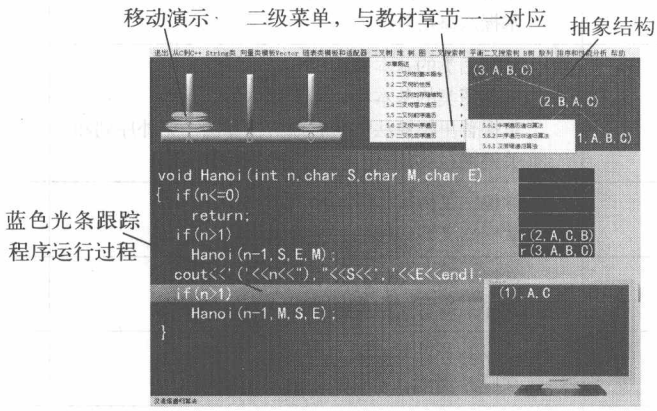
第 2 章～第 4 章实现了容器和算法标准库的 String、Vector、List 和适配器的简化版，使读者不仅学习了泛型程序设计，而且为实现数据结构提供了有力工具。

第 5 章～第 13 章主要是在定义 STL 框架下描述的数据结构。

本书将一些经典的算法纳入相应的数据结构。例如，将快速排序和幂集归入二叉树前序遍历，将汉诺塔问题并入二叉树中序遍历，把哈夫曼树作为堆的直接应用，把八皇后和迷宫问题分别放在树和图的深度优先遍历中解决，更全面地体现了学习数据结构的意义。

本书为了保持连贯性，特别将算法性能分析放在最后一章，先让读者集中精力学习算法设计和实现，在熟悉了大量经典算法的基础上，最后结合排序算法学习算法性能分析，即算法的时间和空间复杂度分析。

本书配有用 Authorware 开发的多媒体课件，既可助教又可助学，直观生动，使复杂抽象的算法和程序变得简单，对排序、堆、平衡二叉树、最短路径等难度很大的算法，效果尤其明显，对初学者更是意义非凡。



本书提供教学课件和源代码，读者可以登录华章网站（http://www.hzbook.com）下载。

读者如有问题或发现错误，欢迎直接与作者联系（E-mail:tjwanglizhu@163.com），作者将不胜感激。

王立柱

2013 年 4 月

教学建议

教学章节	教学要求	课时 （授课课时 + 上机机时）
第 1 章 从 C 到 C++	了解 C 语言的局限性 掌握 C++ 的基本内容 了解从 C 顺序表到 C++ 顺序表类的过程 掌握函数模板和类模板的基本概念 掌握继承和动态绑定的基本概念	4+4
第 2 章 String 类	掌握 String 类的基本构成和实现 了解模式匹配	2+2
第 3 章 向量类模板 Vector	掌握 Vector 的基本构成和实现 了解函数对象的概念	2+2
第 4 章 链表类模板和适配器	掌握链表类模板和适配器的基本构成和实现	2+2
第 5 章 二叉树	掌握二叉树的基本概念 掌握遍历的基本算法和应用	4+4
第 6 章 堆	掌握堆的基本构成和实现 掌握哈夫曼树的构成和实现	2+2
第 7 章 树	掌握树的基本概念 了解 Tree 类的基本构成和实现 掌握八皇后算法	2+2
第 8 章 图	掌握图类的基本概念 掌握图类的基本构成和实现 掌握普里姆算法、迪克斯特拉算法、拓扑序列和关键路径算法	4+4
第 9 章 二叉搜索树	掌握二叉搜索树的基本概念 掌握二叉搜索树的基本构成和实现	2+2
第 10 章 平衡二叉搜索树	掌握平衡二叉树的基本概念 掌握平衡二叉树的基本构成和实现	2+2
第 11 章 B 树	掌握 B 树的基本概念 掌握 B 树的基本构成和实现	2+2
第 12 章 散列	掌握散列的基本概念 掌握散列的基本构成和实现	2+2
第 13 章 排序和性能分析	掌握各种排序算法和性能分析	4+4
总课时		34+34

目　录

第1章

Chapter 1

从C到C++

C++是一个在C的文法上面增加了新的不同性能的语言（因此，它被认为是混合的面向对象的语言）。很多人学习走了弯路，因此，我们已经开始探索从C程序语言转移到C++语言的方法。

——Brace Eckel

和其他高级语言一样，C语言程序设计一开始是利用语言提供的类型来进行的。类型提供了对数据的基本运算，程序是在这个基础上形成的。随着问题越来越复杂，类型对基本数据的存储机制就不够用了，语言的局限性也就显示出来了，这时，用户从设计自定义数据类型开始扩展C语言。

1.1 从数组到顺序表

数组是最早的容器，它不仅使我们在使用中认识到C语言的局限性，而且也是坚实的基础，使我们稳健地走向C++。

1.1.1 数组的局限性

数组的插入、删除、查找等是经常使用的基本运算，但是数组类型和系统标准库都没有提供这些基本运算，它们只能由应用程序员自己设计。这就存在两点不足：一是不同的程序员设计的代码不同，很难重用；二是如果基本运算代码没有用函数封装，而是和应用程序代码混在一起，就会使代码难以阅读，而且不好控制。以下面的删除数组的重复数据为例，其中划线部分是对数组的基本运算：

```
void Purge(int *p,int *n)                    // 删除数组的重复数据
{
     int i,j,k;
     for(i=0;i<*n-1;i++)                     //*n 表示数据元素个数
     {
          j=i+1;
          while(j<*n)
              if(p[j]==p[i])                 // 数据元素比较
              {
```

```
            for(k=j+1;k<*n;k++)                    // 删除重复的数据元素
                p[k-1]=p[k];
            (*n)--;
        }
        else
            j++;
    }
}
```

这种将基本运算代码和应用程序代码混合在一起的程序设计被认为是低级程序设计。本节我们用自定义函数来封装数组的基本运算代码，并且将它们和数组“绑”在一起，组成模块，形成自定义类型即自定义存储模式，然后在这个基础上实现数组上的算法。

1.1.2 顺序表的声明

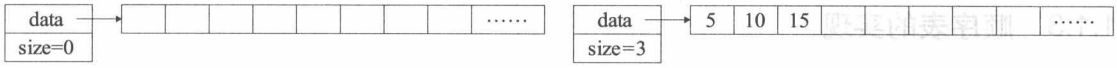
数组指针、数组容量（即数组长度）和数据元素个数是数组的三个基本数据特征，对数组的基本运算几乎都与它们有关。例如：往数组中插入一个数据，首先要判定数据元素个数小于数组容量；插入之后，数据元素个数还要加 1。从数组中删除一个数据，首先要判定数据元素个数不为 0；删除之后，数据元素个数要减 1。这表明，数组的三个数据特征和数组的基本运算是一个整体。现在我们用结构包含数组的三个数据特征，用一组函数表示数组的基本运算，然后把结构和函数组织在一起，称为顺序表。为了提高重用性，我们用宏常量表示顺序表容量即数组长度；用形式化类型 Type 表示顺序表类型，即顺序表元素类型，在使用顺序表之前，用 typedef 命令将其用一个 C 语言合法类型来替换，我们称之为实例化；用动态数组代替静态数组。顺序表定义如下（见图 1-1）：

```
//seqlist.h
#ifndef SEQLIST_H                          // 条件编译开始
#define SEQLIST_H

#define MAXSIZE 100                                // 宏常量 MAXSIZE 表示顺序表容量
struct SeqList                             // 顺序表结构
{
    Type *data;                            //Type 表示待定的顺序表元素类型
    int size;                              // 顺序表的数据元素个数
};
typedef struct SeqList SeqList;
void Error(const char *c);                 // 错误信息报告
// 准构造函数和准析构函数
void IniList(SeqList *l);                  // 申请动态数组，并令数据元素个数为 0
void FreeList(SeqList *l);                 // 释放动态数组
// 修改的方法
void Insert(SeqList *l,int id,Type item);// 定点插入。在下标为 id 处插入一个数据
void InsertRear(SeqList *l, Type item);  // 尾插。在表尾插入一个数据
void Erase(SeqList *l,int id);             // 定点删除。将下标为 id 的数据元素删除
```

```
    void Clear(SeqList *l);                    // 清表。令数据元素个数 size 为 0
// 读取的方法
    int Find(const SeqList *l,Type item);      // 查找。找到数据元素，返回其下标，否则返回 -1
    Type GetData(const SeqList *l,int id);     // 取值。取下标为 id 的数据元素
    int Size(const SeqList *l);                // 求数据元素个数
    bool Empty(const SeqList *l);              // 判空
    bool Full(const SeqList *l);               // 判满
    ……                                         // 基本运算函数实现代码
    #endif                                     // 条件编译结束
```

如果没有使 Type 实例化的 typedef 命令，那么这个顺序表是一个抽象顺序表。当把 Type 实例化为整型 int，这才是一个实在的顺序表——整型顺序表，它是抽象顺序表的一个实例。如果将 Type 实例化为字符型 char，这便是一个字符型顺序表，它也是抽象顺序表的一个实例。在一个程序中，抽象顺序表只能实例化一次。如果一个程序同时需要多种类型的顺序表，这种抽象顺序表就不适用了，而需要扩展为顺序表类模板，这是 1.2 节的内容。



a）空表　　b）非空表

图 1-1　整型顺序表示意图

程序 1.1　检验顺序表（见图 1-2）。

```
typedef int Type;                          // 将抽象顺序表实例化为整型顺序表
#include"seqlist.h"
#include<stdio.h>
int main( )
{
     int i,n;
     SeqList L;                            // 定义一个整型顺序表变量
     IniList(&L);                          // 给顺序表变量赋初值，形成空表
     nsertRear(&L,5);                      // 尾插 5
     InsertRear(&L,15);                    // 尾插 15
     Insert(&L,1,10);                      // 在下标为 1 的位置插入 10
     Erase(&L,0);                          // 删除下标为 0 的数据元素
     n=Size(&L);                           // 求数据元素个数
     for(i=0;i<n;i++)                      // 输出数据元素
          printf("%d\t",GetData(&L,i));
     FreeList(&L);                         // 释放动态数组空间
     return(0);
}
```

< 运行结果 >

```
10    15
```

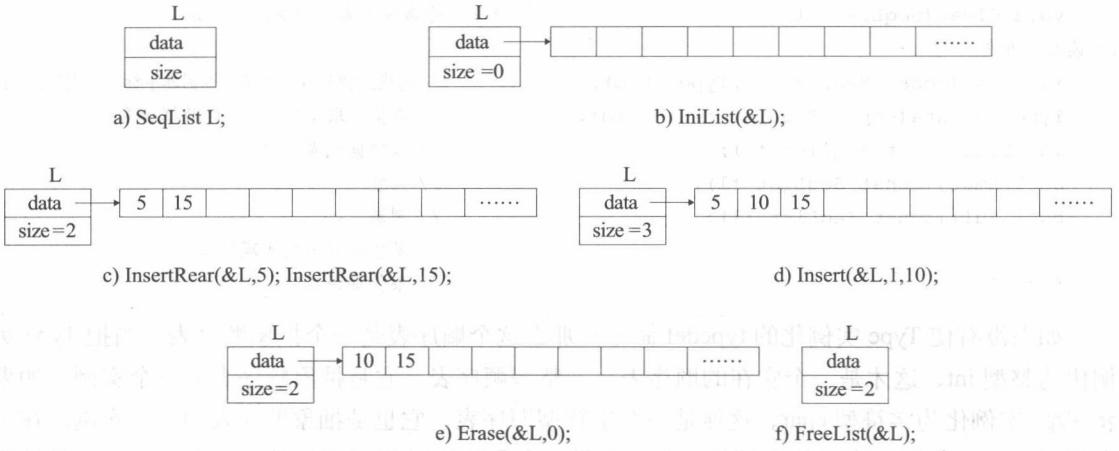


图 1-2 顺序表示意图

1.1.3 顺序表的实现

1．错误信息报告函数

每个基本运算函数都要对参数的合法性进行检验，如果不合法，就调用错误信息报告函数。

```
void Error(const char *c)
{
    printf("%s",c);
    exit(1);
}
```

因为几乎每一个基本运算函数都调用它，所以它的存在不仅简化了基本运算函数的代码，而且使它们更容易阅读，更便于修改和控制。

2．准构造函数和准析构函数

顺序表变量与任何变量一样，必须首先初始化或赋初值，才能进行其他表运算。只是它的初值首先需要申请动态数组，然后令 size 为 0，这项工作由准构造函数完成。

```
void IniList(SeqList *l)
{
    l->data=(Type*)malloc(MAXSIZE*sizeof(Type));
    if(l->data==NULL)
        Error("overflow");
    l->size=0;
}
```

因为顺序表含有动态数组，因此在不需要时，要撤销动态数组。这项工作由准析构函数完成。

```
void FreeList(SeqList *l)
{
    free(l->data);
}
```

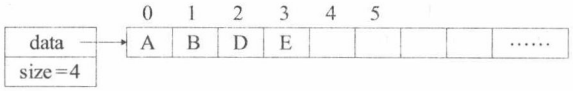
3．定点插入

定点插入是指将一个数据插入到某一指定位置，其步骤如下：

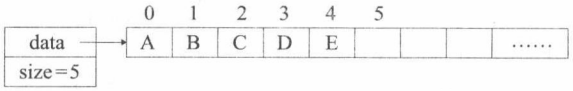
1）判断顺序表变量是否已满。如果满，就显示“表已满”的信息，并且终止程序。

2）判断数据插入位置是否合法。顺序表变量中的数据是连续存放的，因此，插入后的数据元素要与已有数据元素相邻。如下图所示，假设已有4个数据元素，现在要插入一个数据元素C，那么只能插在下标0~4的位置。

如果插入位置错了，就显示“插入位置不合法”的信息，并且终止程序。



3）假设插入位置为2，那么从顺序表变量最后一个数据元素到下标为2的数据元素都依次向后移动一个位置；然后把C插入到下标为2的位置；最后数据元素个数加1。结果如下图所示：




```
void Insert(SeqList *l,int id,Type item)
{
     int i;
     if(l->size==MAXSIZE)                                    // 步骤1)
          Error("Insert:SeqList is full!");
     if(id<0||id>l->size)                                    // 步骤2)
          Error("Insert:id is out of range!");
     for(i=l->size-1;i>=id;i--)                              // 步骤3)
          l->data[i+1]=l->data[i];
     l->data[id]=item;
     l->size++;
}
```

4．尾插

尾插是指将一数据元素插到顺序表变量最后一个数据元素的后面。

```
void InsertRear(SeqList *l, Type item)
{
     if(l->size==MAXSIZE)
          Error("InsertRear:SeqList is full!");
     l->data[l->size]=item;
     l->size++;
}
```

5．定点删除

定点删除是指将某一下标的数据元素删除，其步骤如下：

1）判断顺序表变量是否为空。如果为空，就显示“表已空”的信息，并且终止程序。