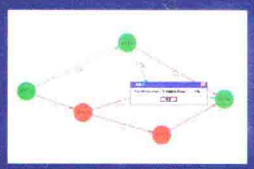
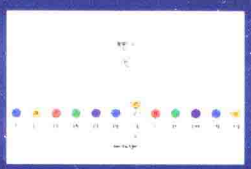
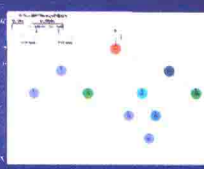
重点大学计算机专业系列教材

数据结构算法解析
（第2版）

高一凡 著

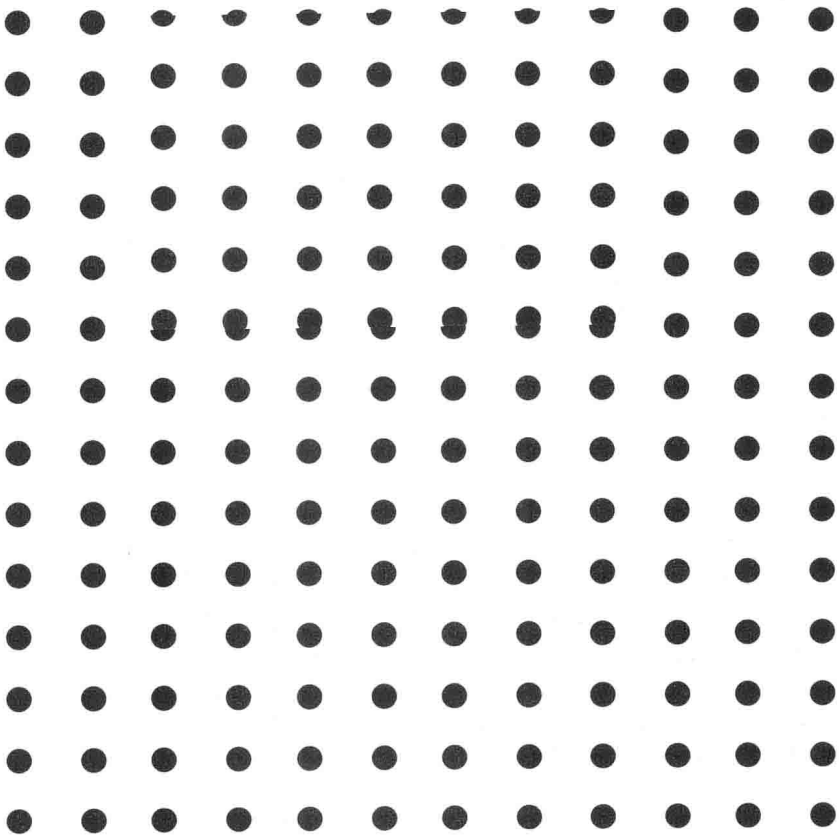




清华大学出版社

重点大学计算机专业系列教材

数据结构算法解析
（第2版）

高一凡 著



清华大学出版社

内 容 简 介

本书为严蔚敏、吴伟民编著的《数据结构(C 语言版)》(清华大学出版社出版,本书按惯例将其简称为严书)的学习辅导书,主要内容包括严书中各主要数据存储结构的基本操作函数、调用这些基本操作的主程序和程序运行结果以及严书中各主要算法的演示课件。

本书作者长期教授“数据结构”课程,有着独到的教学心得和先进的教学方法,教学效果显著,使“数据结构”的学习成为一件赏心乐事,深受学生喜爱。本书是作者多年教学经验的总结。

本书所有程序和算法演示课件均在计算机上运行通过,这些程序的源代码和算法演示课件可通过清华大学出版社的网站下载。

本书适用于使用严蔚敏、吴伟民编著的《数据结构(C 语言版)》作为教材的高等学校学生和自学者,也可供使用其他《数据结构》教材者和软件编程人员参考,同时也是很好的考研参考书。

图书在版编目(CIP)数据

数据结构算法解析/高一凡著. —2 版. —北京:清华大学出版社,2015
重点大学计算机专业系列教材
ISBN 978-7-302-40967-0

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

中国版本图书馆 CIP 数据核字(2015)第 167180 号

责任编辑:郑寅堃　薛　阳
封面设计:常雪影
责任校对:梁　毅
责任印制:王静怡

出版发行:清华大学出版社
网　　址:http://www.tup.com.cn, http://www.wqbook.com
地　　址:北京清华大学学研大厦 A 座　　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
课 件 下 载:http://www.tup.com.cn,010-62795954

印 装 者:河北新华第一印刷有限责任公司
经　　销:全国新华书店
开　　本:185mm×260mm　　**印　　张**:20.5　　**字　　数**:509 千字
版　　次:2008 年 2 月第 1 版　2015 年 8 月第 2 版　　**印　　次**:2015 年 8 月第1 次印刷
印　　数:1～1500
定　　价:39.00 元

产品编号:062703-01

INTRODUCTION

出版说明

随着国家信息化步伐的加快和高等教育规模的扩大，社会对计算机专业人才的需求不仅体现在数量的增加上，而且体现在质量要求的提高上，培养具有研究和实践能力的高层次的计算机专业人才已成为许多重点大学计算机专业教育的主要目标。目前，我国共有16个国家重点学科、20个博士点一级学科、28个博士点二级学科集中在教育部部属重点大学，这些高校在计算机教学和科研方面具有一定优势，并且大多以国际著名大学计算机教育为参照系，具有系统完善的教学课程体系、教学实验体系、教学质量保证体系和人才培养评估体系等综合体系，形成了培养一流人才的教学和科研环境。

重点大学计算机学科的教学与科研氛围是培养一流计算机人才的基础，其中专业教材的使用和建设则是这种氛围的重要组成部分，一批具有学科方向特色优势的计算机专业教材作为各重点大学的重点建设项目成果得到肯定。为了展示和发扬各重点大学在计算机专业教育上的优势，特别是专业教材建设上的优势，同时配合各重点大学的计算机学科建设和专业课程教学需要，在教育部相关教学指导委员会专家的建议和各重点大学的大力支持下，清华大学出版社规划并出版本系列教材。本系列教材的建设旨在“汇聚学科精英、引领学科建设、培育专业英才”，同时以教材示范各重点大学的优秀教学理念、教学方法、教学手段和教学内容等。

本系列教材在规划过程中体现了如下一些基本组织原则和特点。

1. 面向学科发展的前沿，适应当前社会对计算机专业高级人才的培养需求。教材内容以基本理论为基础，反映基本理论和原理的综合应用，重视实践和应用环节。

2. 反映教学需要，促进教学发展。教材要能适应多样化的教学需要，正确把握教学内容和课程体系的改革方向。在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养，为学生知识、能力、素质协调发展创造条件。

3. 实施精品战略，突出重点，保证质量。规划教材建设的重点依然是专业基础课和专业主干课；特别注意选择并安排了一部分原来基础比较好的优秀教材或讲义修订再版，逐步形成精品教材；提倡并鼓励编写体现重点大学计算机专业教学内容和课程体系改革成果的教材。

4. 主张一纲多本，合理配套。专业基础课和专业主干课教材要配套，同一门课程可以有多本具有不同内容特点的教材。处理好教材统一性与多样化的关系；基本教材与辅助教材以及教学参考书的关系；文字教材与软件教材的关系，实现教材系列资源配套。

5. 依靠专家,择优落实。在制订教材规划时要依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主编。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平的以老带新的教材编写队伍才能保证教材的编写质量,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

教材编委会

FOREWORD

第2版前言

本书第1版受到了读者的好评，这对作者是莫大的鼓励。时隔数年，作者在教学实践中进一步积累了经验、方法、手段和心得体会。值此第2版出版之际，作者将这些新的成果与读者分享。

作者在第1版的基础上做了如下修改。

(1) 增加了包括大多数算法的演示课件。该演示课件是与算法逐语句对应的，并在教育部举办的“第十届全国多媒体课件大赛”中获得高教工科组三等奖。

(2) 在第3章增加了“离散事件模拟”一节。因为算法较复杂在第1版没有收入，第2版增加了该算法的演示课件，降低了理解算法的难度。

(3) 增加了“动态存储管理”一章(放在最后)。同样增加了该章算法的演示课件，降低了理解的难度。

(4) 去掉了第5章中广义表的内容。

(5) 去掉了第6章中线索二叉树的内容。

(6) 增加了平衡二叉树删除结点的操作函数，这是我曾经的学生曾金龙在课程设计中做的，后来又几经修改。对于这组函数我特别满意，特地收在书中与读者分享。

(7) 去掉了树形选择排序的算法。

(8) 对基数排序的讲述做了较大修改，着重强调基数排序适用于不等长字符串排序，更换了输入数据文件，增加了一些图，使得基数排序算法一目了然。

(9) 对第1版内容做了仔细的修订，有些算法的讲述也做了修改，在此无法一一细数。

本书所有程序都在 Microsoft Visual C++ 6.0 和 Visual Studio C++ 2012 下运行通过，稍做修改也可以在 UNIX 下运行通过。这些程序和算法演示课件都可通过清华大学出版社网站(www.tup.tsinghua.edu.cn 或 www.tup.com.cn)下载。

算法演示课件的工作量巨大，非我一人能够完成。我的同事李鹏给了我很大的帮助，我的许多学生也参与了课件的编写工作，他们的名字都出现在所做课件中。借此机会，向他们表示衷心的感谢！

尽管作者尽了最大努力，但限于水平，书中疏漏之处在所难免，希望读者不吝赐教，以便借助清华大学出版社网站及时改正。读者可通过 505810175@qq.com 与作者取得联系。

作　者

2014年12月于长安大学

FOREWORD

第1版前言

作者多年讲授“数据结构”课程，所用教材为严蔚敏、吴伟民编著的《数据结构(C语言版)》(以下简称为严书)清华大学出版社出版。根据多年的授课经验，作者深知学习“数据结构”的关键点：

- 首先，要产生兴趣，兴趣是求知的动力。
- 其次，要加强形象思维训练，用形象思维帮助建立抽象思维。
- 最后，要使算法活起来，使算法不再是抽象的、枯燥的、孤立的、晦涩的，而是具体的、生动的、互相有联系的、易于理解的。

本书是作者多年来潜心研究的成果，其中有许多独到之处。

一、本书不仅包括严书中绝大多数算法的实现，对于许多主要的存储结构，也包括它们的基本操作的实现。这些基本操作构成了存储结构的完整体系，使得该存储结构可以直接使用在需要的地方。如在第7章的拓扑排序中就用到了第3章顺序栈的存储结构和基本操作。作者也经常直接将本书的存储结构和基本操作用在自己的科研课题程序中，效果都很好。读者如果需要了解少数严书中提及而本书中未提及的算法、存储结构和基本操作，可以参考阅读本书后面的参考文献[3]。

二、为了加强形象思维训练，作者绘制了各种数据存储结构、算法、程序运行过程的示意图，共计281幅(有些图本身又由一系列小图组成)。这些图清楚地说明了数据的存储结构和算法。

三、通过将算法编写到计算机可运行的程序中的方法，使算法活起来。对于可运行的算法，输出变量、单步执行、设置断点、修改算法、尝试各种输入数据等都是轻而易举的，这些做法都有助于深刻地理解算法。

四、对于较难理解的算法，都有详细的、图文并茂的解析，有些解析(如平衡二叉树)还包含作者自己的研究。较为简单的算法，也尽量利用程序中的空白处，多加注释。对于相应于严书算法中须说明的新增行与修改行，在注释行中也分别注以“新增”与“修改”。

五、本书第7章以后的许多程序中的数据来自文本格式的数据文件，避免了人工键盘输入的麻烦，也有利于掌握使用文件输入输出的方法(这是很多学生不熟悉，却又很重要的方法)。根据程序所用文件的格式，读者很容易编写出自己需要的数据文件。

六、本书除了实现严书中已有的算法，还实现了克鲁斯卡尔、2- 路插入排序(包括改进的2- 路插入排序)、树形选择排序等严书中没有写出的算法。

七、本书还包含许多编程的技巧和小窍门,这些是作者多年编程所积累的经验。

有严书和本书对算法和数据结构的详细讲解,又有可执行的程序,还有程序的运行结果,加上读者自己的思考和努力,还有什么学不会的呢?甚至会觉得,数据结构是简单的,又是有趣的,还是有用的。其中的许多算法是巧妙的、启迪心智的,徜徉其中,其乐无穷。

本书紧密配合严书,故在章节编排上尽量与严书保持一致,以便读者对照查找。严书的第8章和第12章由于内容较为独立,并应用较少,故没有收进本书。因此,严书的第9、10、11章在本书中相应地成为第8、9、10章。同样,严书中有些节的内容没有收进本书,后面节的编号随之减小。但各章节的名称与严书保持一致。

本书所有程序都在Borland C++ 3.1和Microsoft Visual C++ 6.0下运行通过。这些程序都可通过清华大学出版社网站(www.tup.tsinghua.edu.cn或www.tup.com.cn)下载。

本书虽然是以严蔚敏、吴伟民编著的《数据结构(C语言版)》为基础,但对其他数据结构教材也极具参考价值。

尽管作者尽了最大努力,但限于水平,书中疏漏之处在所难免,希望读者不吝赐教,以便将来改正。读者可通过gyfan@chd.edu.cn与作者取得联系。

作　者

2007年11月于长安大学

CONTENTS

目录

CHAPTER 1

第1章 绪　论

本书内容主要由实现基本操作和算法的程序构成，这些程序文件有以下6类。

(1) 数据存储结构。文件名第一个字母为c，以h为扩展名，如c1-1.h是第1章的第1种存储结构。

(2) 每种存储结构的一组基本操作函数。以bo(bo表示基本操作)开头，h为扩展名，如bo1-1.h是第1章第1种存储结构的一组基本操作函数。严书中涉及基本操作的算法也收到基本操作函数中，且在函数中注明算法的编号。

(3) 调用基本操作的主程序。以main(main表示主程序)开头，cpp为扩展名，如main1-1.cpp是调用bo1-1.h的主程序。

(4) 实现算法的程序。以algo(algo表示算法)开头，cpp为扩展名，如algo2-1.cpp是实现算法2.7的程序。

(5) 不属于基本操作又被多次调用的函数或程序段。以func开头，h为扩展名，如func2-2.h中的函数equal()、comp()、print()等被许多程序调用。为节约篇幅，将这些函数放到func2-2.h中。

(6) 数据文件。以txt为扩展名，如第7章的f7-1.txt等。

只有以main、algo开头的程序文件是可执行的程序。其他程序都是被调用的程序，通过#include命令包括在可执行程序中，它们可能被多次调用，为了节省篇幅，只出现在书中第一次调用前。

1.1 抽象数据类型的表示与实现

严书定义OK、ERROR等为函数的结果状态代码，Status为其类型，把这些信息放到头文件c1.h中。c1.h还包含一些常用的头文件，如string.h、stdio.h等。为了操作方便，本书几乎每一个程序都把c1.h包含进去，也就把这些结果状态代码的定义和头文件包含进去了。对于某一个程序来说，有些结果状态代码和头文件并没有用到，不过这不影响使用。头文件c1.h内容如下：

```
// c1.h(文件名)
#include<string.h> // 字符串函数头文件
#include<ctype.h> // 字符函数头文件
#include<malloc.h> // malloc()等
#include<limits.h> // INT_MAX 等
#include<stdio.h> // 标准输入输出头文件,包括 EOF( = ^Z 或 F6),NULL 等
#include<stdlib.h> // atoi(),exit()
#include<io.h> // eof()
#include<math.h> // 数学函数头文件,包括 floor(),ceil(),abs()等
#include<sys/timeb.h> // ftime()
#include<stdarg.h> // 提供宏 va_start,va_arg 和 va_end,用于存取变长参数表
// 函数结果状态代码。在严书第 10 页
#define TRUE 1
#define FALSE 0
#define OK 1
#define ERROR 0
// #define INFEASIBLE -1 没使用
// #define OVERFLOW -2 因为在 math.h 中已定义 OVERFLOW 的值为 3,故去掉此行
typedef int Status; // Status 是函数的类型,其值是函数结果状态代码,如 OK 等
typedef int Boolean; // Boolean 是布尔类型,其值是 TRUE 或 FALSE,第 7、8、9 章用到
```

严书中的例 1-7 定义了一个三元组的抽象数据类型 Triplet,通过例 1-7,给出了这个三元组的表示方法和所定义的基本操作函数的实现。

例 1-7 实际上是严书第 2～7 章中各种存储结构的一个范例:定义一种存储结构及建立在这种存储结构上的一组基本操作,并给出基本操作的实现。下面就是例 1-7 在程序中的具体实现。

```
// c1-1.h 采用动态分配的顺序存储结构。在严书第 12 页
typedef ElemType * Triplet; // 由 InitTriplet 分配 3 个元素存储空间
// Triplet 类型是 ElemType 类型的指针,存放 ElemType 类型的地址
```

头文件 c1-1.h 定义了三元组的抽象数据类型 Triplet。它是一个 ElemType 类型的指针。用一个始于 Triplet 框内的箭头表示 Triplet 是指针类型。用该箭头止于 ElemType 类型的边框表示 Triplet 是 ElemType 类型的指针。因为此处并不是说明 ElemType 类型,故用虚线表示 ElemType(见图 1-1)。

在 c1-1.h 中遇到 ElemType(元素类型),在后面还会遇到 SElemType(栈元素类型)、QElemType(队列元素类型)、TElemType(树元素类型)和 VertexType(图的顶点元素类型)等。在诸如 c1-1.h 这类头文件中,它们是抽象的数据类型,也称为多形数据类型。可以根据需要在主程序中设置为具体的类型。

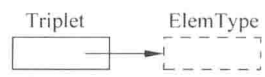


图 1-1 采用动态分配的顺序存储结构

bo1-1.h 是有关抽象数据类型 Triplet 和 ElemType 的 8 个基本操作函数。这 8 个函数返回值的类型都是 Status,即函数返回值只能是头文件 c1.h 中定义的 OK、ERROR 等。

```
// bo1-1.h 抽象数据类型 Triplet 和 ElemType(由 c1-1.h 定义)的基本操作(8 个)
Status InitTriplet(Triplet &T, ElemType v1, ElemType v2, ElemType v3)
```

```
{//操作结果：构造三元组 T,依次置 T 的三个元素的初值为 v1,v2 和 v3。在严书第 12 页(见图 1-2)
  T = (ElemType * )malloc(3 * sizeof(ElemType)); // 分配三个元素的存储空间
  if(!T)
    exit(OVERFLOW); // 分配失败则退出
  T[0] = v1, T[1] = v2, T[2] = v3;
  return OK;
}
Status DestroyTriplet(Triplet &T)
{ // 操作结果：三元组 T 被销毁。在严书第 12 页(见图 1-3)
```

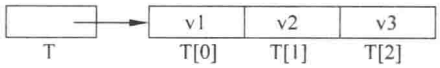


图 1-2 构造三元组 T

图 1-3 三元组 T 被销毁

```
  free(T); // 释放 T 所指的三元组存储空间
  T = NULL; // T 不再指向任何存储单元
  return OK;
}
Status Get(Triplet T, int i, ElemType &e)
{ // 初始条件：三元组 T 已存在,1≤i≤3。操作结果：用 e 返回 T 的第 i 元的值。在严书第 12 页
  if(i < 1 || i > 3) // i 不在三元组的范围之内
    return ERROR;
  e = T[i - 1]; // 将三元组 T 的第 i 个元素的值赋给 e
  return OK;
}
Status Put(Triplet T, int i, ElemType e)
{ // 初始条件：三元组 T 已存在,1≤i≤3。操作结果：改变 T 的第 i 元的值为 e。在严书第 12 页
  if(i < 1 || i > 3) // i 不在三元组的范围之内
    return ERROR;
  T[i - 1] = e; // 将 e 的值赋给三元组 T 的第 i 个元素
  return OK;
}
Status IsAscending(Triplet T) // 在严书第 13 页
{ // 初始条件：三元组 T 已存在。操作结果：如果 T 的三个元素按升序排列,则返回 1；否则返回 0
  return(T[0]<= T[1] && T[1]<= T[2]); // 只在 T[0]不大于 T[1]且 T[1]不大于 T[2]时返回真
}
Status IsDescending(Triplet T) // 在严书第 13 页
{ // 初始条件：三元组 T 已存在。操作结果：如果 T 的三个元素按降序排列,则返回 1；否则返回 0
  return(T[0]>= T[1] && T[1]>= T[2]); // 只在 T[0]不小于 T[1]且 T[1]不小于 T[2]时返回真
}
Status Max(Triplet T, ElemType &e)
{ // 初始条件：三元组 T 已存在。操作结果：用 e 返回指向 T 的最大元素的值。在严书第 13 页
  e = (T[0]>= T[1]) ? (T[0]>= T[2] ? T[0] : T[2]) : (T[1]>= T[2] ? T[1] : T[2]);
  return OK; // 上行是嵌套的条件运算符
}
Status Min(Triplet T, ElemType &e)
{ // 初始条件：三元组 T 已存在。操作结果：用 e 返回指向 T 的最小元素的值。在严书第 13 页
  e = (T[0]<= T[1]) ? (T[0]<= T[2] ? T[0] : T[2]) : (T[1]<= T[2] ? T[1] : T[2]);
  return OK; // 上行是嵌套的条件运算符
}
```

main1-1.cpp是检验bo1-1.h中的各基本操作函数是否正确的主函数。在main1-1.cpp中可根据需要定义抽象数据类型ElemType为int、double或其他数值型类型(只能是数值类型,因为其中有几个函数是比较大小的)而无须改变基本操作bo1-1.h,这使得bo1-1.h的通用性大为增强。func1-1.h是输出函数,根据ElemType在主函数中被定义的类型选择合适的语句。

```
// func1-1.h 输出函数
void PrintE(ElemType e) // 输出元素的值
{ printf("%d\n", e); // 定义 ElemType 为整型选此句。第 3 行
//printf("%f\n", e); // 定义 ElemType 为双精度型选此句。第 4 行
}
void PrintT(Triplet T) // 依次输出三元组的值
{ printf("%d,%d,%d\n", T[0], T[1], T[2]); // 定义 ElemType 为整型选此句。第 7 行
//printf("%f,%f,%f\n", T[0], T[1], T[2]); // 定义 ElemType 为双精度选此句。第 8 行
}

// main1-1.cpp 检验基本操作 bo1-1.h 的主函数
#include"c1.h" // 要将程序中所有 #include 命令所包含的文件复制到当前目录下
// 以下两行可根据需要选其一(且只能选其一),而无须改变基本操作 bo1-1.h
typedef int ElemType; // 定义抽象数据类型 ElemType 在本程序中为整型。第 4 行
//typedef double ElemType; // 定义抽象数据类型 ElemType 在本程序中为双精度型。第 5 行
#include"c1-1.h" // 在此命令之前要定义 ElemType 的类型
#include"bo1-1.h" // 在此命令之前要包括 c1-1.h 文件(因为其中定义了 Triplet)
#include"func1-1.h" // 输出函数,根据 ElemType 的类型选择不同的语句
void main()
{
  Triplet T;
  ElemType m;
  Status i;
  i = InitTriplet(T, 5, 7, 9); // 初始化三元组 T,其三个元素依次为 5,7,9。第 14 行
//i = InitTriplet(T, 5.0, 7.1, 9.3); // 当 ElemType 为双精度型时,可取代上句。第 15 行
  printf("调用初始化函数后,i = %d(1: 成功)。T 的 3 个值为",i);
  PrintT(T); // 输出 T 的 3 个值
  i = Get(T, 2, m); // 将三元组 T 的第二个值赋给 m
  if(i == OK) // 调用 Get()成功
  { printf("T 的第 2 个值为");
    PrintE(m); // 输出 m( = T[1])
  }
  i = Put(T, 2, 6); // 将三元组 T 的第二个值改为 6
  if(i == OK) // 调用 Put()成功
  { printf("将 T 的第 2 个值改为 6 后,T 的 3 个值为");
    PrintT(T); // 输出 T 的三个值
  }
  i = IsAscending(T); // 测试升序的函数
  printf("调用测试升序的函数后,i = %d(0: 否 1: 是)\n", i);
  i = IsDescending(T); // 测试降序的函数
  printf("调用测试降序的函数后,i = %d(0: 否 1: 是)\n", i);
  if((i = Max(T, m)) == OK) // 先赋值再比较
  { printf("T 中的最大值为");
    PrintE(m); // 输出最大值 m
```

```
  }
  if((i = Min(T, m)) == OK)
  { printf("T 中的最小值为");
    PrintE(m); // 输出最小值 m
  }
  DestroyTriplet(T); // 函数也可以不带回返回值
  printf("销毁 T 后,T = %u\n", T);
}
```

程序运行结果：

```
调用初始化函数后,i = 1(1: 成功)。T 的 3 个值为 5,7,9
T 的第 2 个值为 7
将 T 的第 2 个值改为 6 后,T 的 3 个值为 5,6,9
调用测试升序的函数后,i = 1(0: 否 1: 是)
调用测试降序的函数后,i = 0(0: 否 1: 是)
T 中的最大值为 9
T 中的最小值为 5
销毁 T 后,T = 0
```

把 main1-1.cpp 的第 4、14 行注释掉，启用第 5、15 行，定义 ElemType 为 double 类型。同时把 func1-1.h 的第 3、7 行的语句注释掉，启用第 4、8 行，定义 ElemType 为 double 类型的输出语句，则程序运行结果如下：

```
调用初始化函数后,i = 1(1: 成功)。T 的 3 个值为 5.000000,7.100000,9.300000
T 的第 2 个值为 7.100000
将 T 的第 2 个值改为 6 后,T 的 3 个值为 5.000000,6.000000,9.300000
调用测试升序的函数后,i = 1(0: 否 1: 是)
调用测试降序的函数后,i = 0(0: 否 1: 是)
T 中的最大值为 9.300000
T 中的最小值为 5.000000
销毁 T 后,T = 0
```

main1-1.cpp 是本书运行的第一个程序。通过运行 main1-1.cpp，要掌握这样几点：①main1-1.cpp 是提供本书程序风格的一个例子，它是严书中例 1-7 的完整实现，通过它，把数据的存储结构、建立于此结构上的基本操作函数以及调用这些函数的主程序结合到一起；②提供了将抽象的数据类型根据实际需要具体化的方法；③函数类型 Status 的应用：若函数类型为 Status，实际上它是 int 类型的(这由文件 c1.h 中的语句“typedef int Status;”可知)，但它的返回值只能是 OK、ERROR 等；④熟悉 C 语言中 malloc 函数的使用，在学习 C 语言时，malloc 函数使用得并不多，但在“数据结构”中它却几乎是使用最多的函数。

本书的所有程序可以在 Microsoft Visual C++6.0 环境下运行，也可在 Borland C++ 3.1、Visual Studio C++ 2012 和 Linux 等环境下运行。本书中所列的程序运行结果是 Microsoft Visual C++ 6.0 环境下的，它和其他运行环境下的基本一样，只是整型最大值、指针变量的值等有所不同。

下面以运行 main1-1.cpp 为例说明如何在几种不同的环境下运行程序。首先，要把被 main1-1.cpp 调用的所有程序(c1.h、c1-1.h、bo1-1.h 和 func1-1.h 等 4 个文件)复制到

main1-1.cpp 所在的目录下。

在 Microsoft Visual C++ 6.0 环境下运行的一种方法是：在“Windows 资源管理器”中双击 main1-1.cpp 的文件，进入 VC++ 6.0 的环境。按 F7 键编译，如果没错误则按 Ctrl+F5 键运行。另一种方法是：先进入 VC++ 6.0 的环境，单击主菜单 File，选择 Open 命令，在“打开”文件对话框中选择 main1-1.cpp 的程序，进入 VC++ 6.0 的环境。仍然按 F7 键编译，如果没错误则按 Ctrl+F5 键运行。如果是新编程序，单击主菜单 File 之后，选择 New 选项，在 Files 选项卡中选择 C++ Source File，给出 File name 和 Location(文件名和路径)后，单击 OK 按钮就可编写程序。

在 Visual Studio C++ 2012 等环境下运行的一种方法的步骤如下。

(1) 在 Visual Studio C++ 2012 等环境下建立一个空控制台程序，也可将本书程序中所提供的“空控制台程序”目录复制到一个合适的目录下；

(2) 进入其中的 d 目录(以本书程序中所提供的“空控制台程序”目录为例)；

(3) 将所有需要的文件(即 main1-1.cpp、bo1-1.h、c1.h、c1-1.h 和 func1-1.h 等 5 个文件)复制到 d 目录下；

(4) 删去原有的文件 d.cpp；

(5) 将 main1-1.cpp 改名为 d.cpp；

(6) 退到“空控制台程序”目录下；

(7) 双击 d.sln，进入 Visual Studio C++ 2012 的编译环境下；

(8) 按 Ctrl+Alt+F7 键(重新生成解决方案)，提示“全部重新生成：成功 1 个，……”；

(9) 按 Ctrl+F5 键(开始执行(不调试))，出现运行结果。

在 Linux 系统下编译运行，需要确保 Linux 系统中具有 C++ 的基本开发环境(需要安装 gcc/g++，安装配置方法请查阅相应 Linux 发行版文档)。编译运行步骤如下：

(1) 将 main1-1.cpp 中的主函数 void main()改为 int main()，同时在函数最后增加一条语句“return 0;”；

(2) 使用 g++ 编译，编译命令为“g++ main1-1.cpp -o main1-1”，如果没有错误，将会生成“main1-1”可执行文件；

(3) 程序运行命令为“./main1-1”。

注意：如果出现“error: io.h: No such file or directory”错误提示，请确保 io.h 头文件存在于“/usr/include/”目录中。如文件不存在，使用“find/-name io.h”命令进行查询，将 io.h 头文件拷贝到“/usr/include/”目录中。

本书中的 main 主程序是用于检验相应 bo 程序中的基本操作各个函数程序是否正确的，它往往很长。若读者对某个基本操作函数特别关注，就可以对相应的 main 主程序进行删改，以适应自己的需要，又不必花太多的时间去输入程序。

在 bo1-1.h 中，有些基本函数的形参带有 &，如第一个基本函数的声明：

```
Status InitTriplet(Triplet &T, ElemType v1, ElemType v2, ElemType v3);
```

其中，形参 T 前带有 &，说明形参 T 是引用类型的。引用类型是 C++语言特有的。引用类型的变量，其值若在函数中发生变化，则变化的值会带回主调函数中。程序 algo1-1.cpp 说明了函数中引用类型变量和非引用类型变量的区别。