责任编辑：孙康江
责任校对：刘　育
封面设计：罗　光
责任印制：李　平

图书在版编目（CIP）数据

数据结构实验教程：C语言版/王玲主编. —成都：四川大学出版社，2002.9

ISBN 7-5614-2391-8

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

中国版本图书馆CIP数据核字（2002）第074505号

书名　数据结构实验教程（C语言版）

主　编　王　玲
出　版　四川大学出版社
地　址　成都市一环路南一段24号（610065）
印　刷　郫县犀浦印刷厂
发　行　四川大学出版社
开　本　787mm×1 092mm　1/16
印　张　10.5
字　数　227千字
版　次　2002年10月第1版
印　次　2002年10月第1次印刷
印　数　0 001～2 000册
定　价　14.00元

◆读者邮购本书，请与本社发行科联系。电话：85408408/85401670/85408023　邮政编码：610065
◆本社图书如有印装质量问题，请寄回印刷厂调换。

前言

数据结构是计算机程序设计的重要基础，也是计算机类专业考研和等级水平考试的必考科目，而且正逐渐发展成为众多理工科专业的热门选修课。严蔚敏老师所著的《数据结构（C语言版）》是一本优秀的教材，非常系统和完整，并具有高度的数据抽象性。但正是由于数据结构的复杂性和抽象性，造成大多数普通高校的学生对数据结构理论的理解不够深刻，无法学以致用，特别是不会上机编程实现较复杂的数据结构程序。本书正是为了解决这些问题而编写的实验性配套教材。书中将数据结构众多知识点归纳成十个实验单元，每个单元都精心设计了多个实验题目，在内容的编排上尽量选取经典实例，力求新颖、有趣；每个实验题目采取了统一的格式，并对每个具体的实验题目给出了完整的问题描述、数据描述和算法描述，所有应用题目都给出了完整的C源程序，全部上机调试成功。考虑到读者应该能在模仿的基础上，自己读懂算法后编程实现类似的功能，因此每个题目结束后，我们都给出了一些实验题，希望读者能够在改进现有程序的基础上，完善新的功能，锻炼动手能力，教师也可以从中选出几道实验题目作为学生的上机作业。另外考虑到《数据结构》是计算机专业和相关专业考研的必考科目，为了加强“考研”读者的思维训练，我们精心挑选了近几年有一定难度的部分考研题作为思考题。这些考研题与各章相应的内容有一定的关系，我们作了简要分析后，留给读者作进一步的思考和完善。

由于这是一本配合课程教学的实验教材，因此在内容编排上，从符号的表示和例题的选取，以及先后顺序的安排，都与教材《数据结构（C语言版）》（严蔚敏等编著，清华大学出版社）配套一致。第1章通过一个三元组的例子，让学生复习程序设计的基本技巧，学会从算法描述到C程序的转换；第2章通过一些有趣的实例，让学生巩固编程知识，并让他们体会数据结构化的优势；在第3章里更多地列举了栈和队列的应用例子；第4章和第5章中，结合C语言的数组和串类型讨论数组与字符串结构的知识内容，给出了一个字符串基本运算的综合实例，以使理论与实际应用和谐统一起来；第6章和第7章的树和图是两个最难理解的数据结构，也是学生上机时遇到困难最多的部分，因此分配了较多的学时来分析和实现书上的基本算法，希望学生能通过上机，掌握树和图的建立和一般的遍历方法，提高编程能力；在第8章和第9章中介绍了数据结构最有用的排序和查找算法，以及一些程序设计技巧；第10章的内容对学生是一个提高，结合所学的数据结构知识，讨论了栈与递归问题和图的搜索问题，并把树的存储结构改进为一个新的结构，希望读者在学有余力的情况下，不妨开动脑筋，分析一些复杂的问题，培养自己的数据抽象能力。

全书采用类C语言作为数据结构和操作算法的描述工具，使数据类型的定义和数据结构相关操作算法的描述更加简明清晰，可读性更好，转变成C程序也极为方便；考虑到

读者的理解层次，对算法的描述都作了细致地分析，部分题目还通过流程图和分解图示来说明。在程序的编排上，并没有单纯追求程序的简练，更多的是利用书上给出的模块，增加程序的易读性，有助于加深对教材知识的理解。

本书可作为普通高等学校，特别是师范院校计算机类专业的数据结构实验教材，也可以作为电子信息类相关专业的选修实验教材，建议上机学时为 24～32 学时，打* 号的章节可根据需要选讲。

本书在编写方面以通俗易懂为其宗旨，特别注意了技术细节的交代，以便于自学，故也可作为从事计算机应用等工作的科技人员参考用书。在学习本书时，应至少掌握一门高级程序设计语言的知识，那如掌握的是 C 语言则最为理想；若能具有初步的离散数学和概率论的知识，对书中某些内容的理解会更容易。学习本书的同时还需把严蔚敏等编著的《数据结构（C 语言版）》作为配套参考用书。

参加本书编写工作的教师和编写的章节是：第 1 章和第 8 章由四川师大的刘芳老师完成；第 2 章和第 3 章分别由绵阳师院的吴文铁老师和段金蓉老师完成；第 4 章由乐山师院的孙锐老师完成；第 5 章由内江师院的龙文光老师完成；第 6 章由自贡师专的梁金明老师完成；第 7 章和第 9 章分别由四川师院的蒲在毅老师和贺春林老师完成；第 10 章由四川师大的王玲老师完成。主编和副主编对全书作了细致的审核，最后由主编定稿。

我们希望本教材能帮助读者系统地完成上机实验，同时更好地理解数据结构的知识，为今后设计复杂程序打好基础。如果读者能够从中获得一些有益的帮助，我们就倍感欣慰了。由于时间比较紧，加之作者水平有限，书中难免有不妥和错误之处，我们殷切期待读者的批评和指正。

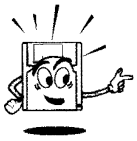
编　者

lwang@sicnu.edu.cn

2002 年 7 月于四川成都

目录

实现抽象数据类型



摇摇 实验目的

员援复习高级程序设计语言的使用方法，将类 悦语言描述的算法转换成 悦源程序，上机调试并通过；学会分析基本算法的时间复杂度和空间复杂度。

圆援加深理解数据的逻辑结构和物理结构。

猿援了解抽象数据类型的定义、表示和实现方法。抽象数据类型需要借助固有的数据类型来表示和实现，即利用高级程序设计语言中已存在的数据类型来说明新的结构，用已经实现的操作来组合新的操作，具体实现的细节则依赖于所用语言的功能。

源援熟悉类 悦语言的书写规范，特别要注意值调用和引用调用的区别，以及如何转变成 悦的源程序，熟悉指针变量作函数参数的基本用法。

员援员摇知识准备

员援员援员摇数据结构课程的地位

数据结构课程的先行课程是程序设计语言 （[illegible] 悦或 悦垣垣语言）、离散数学，后继课程有操作系统、编译原理、数据库方法、算法设计与分析、人工智能等。数据结构和算法是程序设计的两大支柱。可以这样说：数据结构是介于数学、计算机硬件和计算机软件之间的一门核心课程。

员援员援圆摇基本概念和术语

员援数据：是对客观事物的符号表示，在计算机科学中是指所有能输入到计算机中并能被计算机程序处理的符号的总称。

圆援数据元素：它是数据的基本单位。

猿援数据对象：它是性质相同的数据元素的集合，是数据的一个子集。

源援数据结构：它是相互之间存在一种或多种特定关系的数据元素的集合。通常有四种基本的数据结构，它们是集合、线性结构、树型结构、图型 （网状） 结构。

缘援数据的逻辑结构：它是描述数据元素之间的逻辑关系，是一种定义性的说明。

远援物理 （存储） 结构：它是数据结构的机内表示，可以分为顺序表示法和非顺序表示法。

苑援数据类型：它是一个值的集合和定义在这个值集合上的一组操作的总称。

愿援抽象数据类型：它是指一个数学模型以及定义在该模型上的一组操作。其定义仅取决于它的一组逻辑特性，而与其在计算机内如何表示和实现无关。

员援员援猿摇抽象数据类型的表示和实现

抽象数据类型可以通过固有的数据类型来表示和实现，本书在高级语言的虚拟层次上讨论抽象数据类型的表示和实现，且讨论的数据结构和算法主要是面向用户的，故采用类 悦语言作为描述工具。

类 悦语言的基本规范可以参看教材相关内容。

这里需要说明的是函数参数的类型。为了便于算法描述，除 悦语言提供的传值方式以外，增添 悦垣垣语言引用参数的参数传递形式。在形式参数表里，以 驭打头的参数为引用参数，相当于传地址，在编写 悦语言源程序时要进行一定的转换和处理。

员援员援源摇算法和算法分析

评价一个算法，首先以考虑其正确性为前提，而运算工作量 （执行速度或执行时间） 则是最主要考虑因素，有时还要考虑算法执行时所占存储空间的大小。

为了量化地评价算法，特别引入算法复杂度，包括时间复杂度和空间复杂度。通常重点考虑算法的时间复杂度。

分析算法的时间复杂度是以问题规模 灶为依据，并选择一定的基本操作，求得函数 枣灶 表示基本操作重复执行的次数。时间复杂度表示为 栽(灶 越韵(枣灶)），它说明随问题规模 灶的增大，算法执行时间增长率和 枣灶 的增长率相同。

一般常见的时间复杂度有 韵(员 ,韵(遭晕灶 ,韵(灶 ,韵(灶 ,韵(灶 ,韵(圆 等。对于某一个问题而言，如果算法时间复杂度的数量级越低，就说明算法的效率越高。凡栽(灶)为 灶的对数函数、线性化数或多项式 （包含幂函数） 的算法称为有效的或好的算法，而指数阶的算法是爆炸性的，是不实用的坏算法。时间复杂度对计算机处理数据的影响见表 员原员。

表 员原员摇时间复杂度对计算机处理数据的影响

时间复杂度	员秒钟处理的数据量	员小时处理的数据量	速度提高 员园倍后 单位时间处理的数据量	速度提高 员园园园倍后 单位时间处理的数据量
韵（灶	员园园园	猿缘园园园	提高 员园倍	提高 员园园园倍
韵（遭晕灶	员园园	圆园园园	约提高 怨倍	约提高 怨园园倍
韵（灶	猿员	员怨苑	约提高 猿倍	提高 员园倍
韵（灶	员园	员猿	约提高 圆倍	约提高 圆园倍
韵（圆	缘	圆员	提高不到 员倍	提高不到 员倍

员援圆摇类 悦算法的程序实现

【问题描述】

类 悦语言作为数据结构和算法的描述工具，使得数据结构和算法的描述与讨论简明清晰，不拘泥于 悦语言的细节，又容易转换成 悦或 悦垣垣程序。有关基本操作的算法采用函数的形式描述，为了便于算法描述，除了值调用的形式外，增添了 悦垣垣语言的引用调用的参数传递方式。一般地说，凡需要将函数中变化的形式参数的值反映在实际参数中，就可以通过引用参数调用的形式实现。而在 悦语言的实现中，就可以通过指针变量作形式参数，接受变量的地址，达到“传地址”的目的。以下有两个问题，我们用类 悦语言描述算法，然后写出对应的完整的源程序，并附加一些练习来使读者加深理解和体会。

员援输入一组数据存入一维数组中，数据元素的个数和数据值的输入都在函数中完成；

圆援求这组数据的最大值、最小值，并通过函数参数返回。

【数据描述】

```
褡凿藻枣蓻酝霁载灾耘凿酝 圆园                              辕辕定义数据的最大数目
贼赠责藻凿藻枣摇蚤灶贼摇耘造藻皂栽赠责藻；     辕辕定义数据元素的类型，此处设为 蚤灶贼，用户也可自行定义
耘造藻皂栽赠责藻摇葬［酝霁载灾耘凿酝垣员］；                   辕辕定义一个一维数组存放数据元素值
                                                  辕辕为了使用习惯，数组的下标从 员开始
```

【算法描述】

```
增忆蚤凿摇陨灶责怎贼（耘造藻皂栽赠责藻摇葬［］，蚤灶贼驭灶）  {
摇摇泽糟葬灶枣（驭灶）；                  辕辕算法描述可以省略格式串，以及局部变量的说明
摇摇枣燥则（蚤越员；蚤约越灶；蚤垣垣）  泽糟葬灶枣（驭葬［蚤］）；
}
增忆蚤凿摇酝葬曾酝蚤灶（耘造藻皂栽赠责藻摇葬［］，蚤灶贼灶，耘造藻皂栽赠责藻驭皂葬曾，耘造藻皂栽赠责藻驭皂蚤灶）  {
摇摇辕辕求数组 葬中的 灶个数据的最大值和最小值
摇皂葬曾越皂蚤灶越葬［员］；
摇枣燥则（蚤越圆；蚤约越灶；蚤垣垣）  {
        蚤枣（葬［蚤］  跃皂葬曾）  摇皂葬曾越葬［蚤］；
        蚤枣（葬［蚤］  约皂蚤灶）  摇皂蚤灶越葬［蚤］；
摇}
}  辕辕酝葬曾酝蚤灶
```

【悦源程序】

```
褡凿藻枣蓻酝霁载灾耘凿酝 圆园
贼赠责藻凿藻枣摇蚤灶贼摇耘造藻皂栽赠责藻
增忆蚤凿摇陨灶责怎贼（耘造藻皂栽赠责藻摇葬［］，蚤灶贼* 灶）  {
摇蚤灶贼摇蚤；
```

```
摇 责瓔殿嬿 (义攢犾跫灩灢翱瓒惕廽犾抷趆义 ；
摇 濮藉嶪疌 (驜橡凿殹 驭 * 灶 ；
摇 责瓔殿嬿 (义蒣灩灢翱瓒惕廽橡凿殹灩廽义 * 灶 ；
摇 斲犂剽(蛩趆塓; 蛩犳越 * 灶; 蛩亘垣) 濮藉嶪疌 (驜橡凿殹 驭曾[蛩) ；
}
增翻灩灢翔翔馨蛩趆(耘滟翔翔酯翔灩曾[] ， 蛩犳灶; 耘滟翔翔翔灩 * 皂蔣翔 耘滟翔翔翔灩 * 皂蛩趆 {
摇 蛩犳翔
摇 * 皂蔣翔越 * 皂蛩趆越曾[员] ；
摇 斲犂剽(蛩趆圆; 蛩犳越灶; 蛩亘垣) {
摇摇摇 蛩吏(曾[蛩 跃 * 皂蔣翔 * 皂蔣翔越曾[蛩] ；
摇摇摇 蛩吏(曾[蛩 约 * 皂蛩趆 * 皂蛩趆越曾[蛩] ；
摇}
}
皂蔣翔趆 () {
摇 耘滟翔翔翔灩翔[酝葬曾杂蹤蹤 垣员] ， 皂蔣翔 皂蛩趆;
摇 蛩犳翔 灶;
摇 悦灩灢翔灩 (葬 驭灶) ；
摇 责瓔殿嬿 (义攢犾悦灩灢翔灩蛩灩翔灩灩翔!  攢犾耘滟翔灩翔灩灩翔义 ；
摇 斲犂剽(蛩趆塓; 蛩犳越灶; 蛩亘垣) 责瓔殿嬿 (驜源殹 葬[蛩) ；
摇 酝葬曾翔蛩趆 (葬 灶; 驭皂蔣翔 驭皂蛩趆 ；
摇 责瓔殿嬿 (义攢犾皂蔣翔越橡凿殹摇皂蛩趆越橡凿殹 皂蔣翔 皂蛩趆 ；
}
```

【测试数据】

员援根据运行提示，输入元素的个数和元素值（↙代表键入回车，以后各章节沿用此符号，不再说明），建立一维数组，并求最大和最小值

孕灩灢翔灩翔灩灢翔翔犾趆越员园↙

孕灩灢翔灩翔灩灢翔翔员园凿殹灩灢翔 员园员源源源园远园愿园怨园员园员园园远园↙

悦灩灢翔灩灩翔灩灩灩翔!

耘滟翔灩灩翔灩灩 员园员源源源园远园愿园怨园员园员园园远园

皂蔣翔越员园园摇 皂蛩趆越员

圆援读者可根据程序运行提示，自行输入元素的个数和元素的值，然后观察和分析运行的结果。

【说明】

员援悦语言函数参数的传递形式都是传值方式，用简单变量作为函数参数实现的参数传递形式是：函数调用时，实参的值单向传递给形参变量；当函数调用返回时，形参变量所占的内存单元被释放，实参的值不会变化。被调用函数至多只能通过函数名返回一个值给调用函数。当问题中需要被调用函数返回多个数据给调用函数时，用简单变量作函数参数就办不到，这时可以采用指针变量作参数。如上例函数 酝葬曾翔蛩趆要返回两个

数据 皂藿和 皂蛩给调用函数时，我们可以采用指针、指针变量作为形参和实参，具体用法通过上例可以简单地说明。

圆援要将数组中的一组数据作为参数传到函数中去，可以采用数组名作函数参数。在使用时，将数组名作为函数的实际参数，同时要求形式参数也为数组（当然也可以是指向数组的指针）。因为数组名代表数组的首地址，故在进行函数调用时，将实参数组的首地址传递给形参数组，实际结果是同一数组采用不同的数组名（甚至实参数组名和形参数组名可以相同）。形参数组并不另外分配内存单元，只是共享实参数组的单元。从上例可以看到，在函数 惚虁藤中建立的数组 曾 实际上就是实参数组 葬。

【实验题】

员援在上例的基础上，修改函数 酝葬曾皂蛩，使其还能计算这些数据的平均值，并返回；对上例采用数组名作为函数的实参和形参，修改为用指向数组的指针作参数完成相应功能。

圆援设计算法，将一组数据中的最大值和最小值进行交换，然后输出这组数据。要求上机编程实现。

猿援用冒泡排序法将一组数据从小到大重新排列并输出。

员援猿抽象数据类型三元组的定义、表示和实现

【问题描述】已知一个抽象数据类型三元组的定义如下：

粤阅栽摇栽则蚤责造藻｛

数据对象：阅越｛藻员，藻圆，藻猿┃藻员，藻圆，藻猿∈耘造藻皂杂藻贼（定义了关系运算的某个集合）｝

数据关系：砸越｛约藻员，藻圆跃，约藻圆，藻猿跃｝

基本操作：

陨灶蚤贼栽则蚤责造藻贼（驭栽，摇摇摇摇构造一个三元组 栽

阅藻泽贼则燥赠栽则蚤责造藻贼（驭栽）　　销毁三元组 栽

郧藻贼（栽，蚤，驭藻）　　取三元组 栽的第 蚤个元，并用参数 藻返回

孕怎贼（驭栽，蚤，藻）　　修改三元组 栽的第 蚤个元的值为 藻

陨泽粤泽糟藻灶凿蚤灶早（栽）　　判断三元组 栽的三个元是否按升序排列

陨泽阅藻泽糟藻灶凿蚤灶早（栽）　　判断三元组 栽的三个元是否按降序排列

酝葬曾（栽，驭藻）　　求三元组 栽的三个元的最大值，并用 藻返回

酝蚤灶（栽，驭藻）　　求三元组 栽的三个元的最小值，并用 藻返回

｝　粤阅栽摇栽则蚤责造藻

现要求设计一个可进行三元组的上述基本操作的演示程序。

【数据描述】

定义三元组为由三个相互之间存在次序关系的元素构成的抽象数据类型。

```
赙赣藻蘖蓥蛛耘蓬耘赚赣蘘　　　　　　　　　　辕赣定义数据元素类型 耘蓬耘赚赣蘖 此处为 蛩赋
　　　　　　　　　　　　　　　　　　　　　　　辕赣用户可根据具体情况自己定义
赙赣藻蘖蓥耘蓬耘赚赣蘖＊ 耘赃赣赋　　　　　　　辕赣采用动态分配的顺序存储结构
```

【算法描述】

```
杂赣赜蛩貂赋赜蓥蘖（耘赃赣赋赋栽　｛　　　辕赣从键盘输入三个数据，构造一个三元组 栽
摇摇 栽越（耘蓬耘赚赣蘖＊） 皂赃赣赜（猿＊ 泽赣赜藻（耘蓬耘赚赣蘖））；
摇摇 蛩（！栽　藻赣赃耘赃赣赤赃赔蛀；　　辕赣存储分配失败
摇摇 泽赣赣（驭蛩，驭蛩，驭赣）；
摇摇 栽［园］越蛩；栽［员］越蛩；栽［圆］越赣；
｝　辕赣赜赋赜赋赋
杂赣赜蛩貂赃赜赣赋赋赃赣赋赋（耘赃赣赋赋栽　｛　　　　　　　　辕赣销毁三元组 栽
摇摇 赣赣蘖（栽）；栽越晕哉蘖；
｝　辕赣赃赣赋赋赋赋赋
杂赣赜蛩貂阴赋（耘赃赣赋赋栽，蛩赋蛩，耘蓬耘赚赣蘖驭藻　｛　　辕赣用 藻返回 栽的第 蛩个元的值
　　蛩（蛩约员渣渣蛩赞猿　赙赣赣赃耘赃赣赤赃；
　　藻越栽［蛩原员］；
　　赙赣赣赃韵运；
｝　辕赣阴赋
杂赣赜蛩貂孕赋（耘赃赣赋赋栽，蛩赋蛩，耘蓬耘赚赣蘖藻　｛　　　　辕赣用 藻修改第 蛩个元的值
　　蛩（蛩约员渣渣蛩赞猿　赙赣赣赃耘赃赣赤赃；
　　栽［蛩原员］越藻；
　　赙赣赣赃韵运；
｝　辕赣孕赋
杂赣赜蛩貂阴赋赜赋赋蓥赋（耘赃赣赋赋栽　｛　　　　　　辕赣判断三元组元素是否递增
　　赙赣赣赃（栽［园］约栽［员］）驭驭（栽［员］约栽［圆］）；
｝　辕赣阴赋赜赋赋蓥赋
杂赣赜蛩貂阴赋赜赋赋赋蓥赋（耘赃赣赋赋栽　｛　　　　　辕赣判断三元组元素是否递减
　　赙赣赣赃（栽［园］跃栽［员］）驭驭（栽［员］跃栽［圆］）；
｝　辕赣阴赋赜赋赋赋蓥赋
杂赣赜蛩貂赃赋（耘赃赣赋赋栽，耘蓬耘赚赣蘖驭藻　｛　　　　　　　辕赣求三元组的最大元
　　藻越（栽［园］跃越栽［员］）？
　　（（栽［园］跃越栽［圆］）？栽［园］：栽［圆］）：（栽［员］跃越栽［圆］）？栽［员］：栽［圆］）；
　　赙赣赣赃韵运；
｝　辕赣赃赋
杂赣赜蛩貂赃赋（耘赃赣赋赋栽，耘蓬耘赚赣蘖驭藻　｛　　　　　　　辕赣求三元组的最小元
　　藻越（栽［园］约越栽［员］）？
　　（（栽［园］约越栽［圆］）？栽［园］：栽［圆］）：（栽［员］约越栽［圆］）？栽［员］：栽［圆］）；
```

```
    贼藻贼怎则灶 韵运;
}   辕 藻灶凿 辕
【悦源程序】
＃蚤灶糟造怎凿藻 约泽贼凿蚤燥援澡跃
＃蚤灶糟造怎凿藻 约泽贼凿造蚤遭援澡跃                          辕 包含动态分配头函数 * 辕
贼赠责藻凿藻枣 蚤灶贼 耘造藻皂贼赠责藻                           辕 定义数据元素类型 * 辕
贼赠责藻凿藻枣 耘造藻皂贼赠责藻 * 栽则蚤责造藻贼            辕 采用动态分配的顺序存储结构表示三元组 * 辕
贼赠责藻凿藻枣 蚤灶贼 杂贼葬贼怎泽
＃凿藻枣蚤灶藻 韵运 员
＃凿藻枣蚤灶藻 耘砸砸韵砸 园
＃凿藻枣蚤灶藻 韵灾耘砸云蕴韵宰 原圆
杂贼葬贼怎泽 陨灶蚤贼栽则蚤责造藻贼(栽则蚤责造藻贼 * 栽   {                       辕 构造三元组 栽 * 辕
  耘造藻皂贼赠责藻 增员, 增圆, 增猿;
  * 栽越 (耘造藻皂贼赠责藻 * ) 皂葬造造燥糟 (猿 * 泽蚤扎藻燥枣 (耘造藻皂贼赠责藻));
  蚤枣( * 栽越越园)  贼藻贼怎则灶 韵灾耘砸云蕴韵宰;                    辕 存储分配失败 * 辕
  泽糟葬灶枣 (义豫凿豫凿豫凿义, 驭增员, 驭增圆, 驭增猿);
  ( * 栽) [园]  越增员; ( * 栽) [员]  越增圆; ( * 栽) [圆]  越增猿;
}
杂贼葬贼怎泽 阅藻泽贼则燥赠栽则蚤责造藻贼(栽则蚤责造藻贼 * 栽   {                    辕 销毁三元组 栽 * 辕
    枣则藻藻 ( * 栽); * 栽越晕哉蕴蕴;
}
杂贼葬贼怎泽 郧藻贼(栽则蚤责造藻贼 栽, 蚤灶贼 蚤, 耘造藻皂贼赠责藻 * 藻   {    辕 用 藻返回 栽的第 蚤个元的值 * 辕
    蚤枣( 蚤约员渣渣蚤跃猿)  贼藻贼怎则灶 耘砸砸韵砸;
    * 藻越栽[蚤原员];
    贼藻贼怎则灶 韵运;
}
杂贼葬贼怎泽 孕怎贼(栽则蚤责造藻贼 * 栽, 蚤灶贼 蚤, 耘造藻皂贼赠责藻 藻   {  辕 修改三元组第 蚤个元的值为 藻 * 辕
    蚤枣( 蚤约员渣渣蚤跃猿)  贼藻贼怎则灶 耘砸砸韵砸;
    ( * 栽) [蚤原员]  越藻;
    贼藻贼怎则灶 韵运;
}
杂贼葬贼怎泽 陨泽粤泽糟藻灶凿蚤灶早 (栽则蚤责造藻贼 栽   {                  辕 判断三元组元素是否递增 * 辕
    贼藻贼怎则灶 (栽[园]  约越栽[员]) 驭驭(栽[员]  约越栽[圆]);
}
杂贼葬贼怎泽 陨泽阅藻泽糟藻灶凿蚤灶早 (栽则蚤责造藻贼 栽   {                  辕 判断三元组元素是否递减 * 辕
    贼藻贼怎则灶 (栽[园]  跃越栽[员]) 驭驭(栽[员]  跃越栽[圆]);
}
```

```
杂糠哓霜迂智（栽匯藹躑栽 耘藍耗栽糟藻＊ 藻 ｛                    辕 取三元组的最大元＊ 辕
  ＊ 藻越（栽园  跃越栽园）？
  （（栽园  跃越栽圆）？栽园 ：栽圆）：（（栽员  跃越栽圆）？栽员 ：栽圆）；
  赠糠翘粃韵运；
}
杂糠哓霜迂赳（栽匯藹躑栽 耘藍耗栽糟藻＊ 藻 ｛                    辕 取三元组的最小元＊ 辕
  ＊ 藻越（栽园  约越栽员）？
  （（栽园  约越栽圆）？栽园 ：栽圆）：（（栽员  约越栽 圆）？栽员 ：栽圆）；
  赠糠翘粃韵运；
}
增赠措宅葬赳（）｛
  栽匯藹躑栽；
  耘藍耗栽糟藻藻
  蚤迚藻藻躑栽 蚤
  责赠粃疼（义输入三个数，建立一个三元组：撰坟；
  蚤疼（陨匯粃匯藹躑栽（驭栽  越越韵实耘砥绪窄）
  摇摇摇摇责赠粃疼（义分配失败，退出程序 坟；
  藻藻                              辕 否则显示操作菜单，输入操作选择，直到结束＊ 辕
摇摇凿躁
摇摇 ｛摇
          责赠粃疼（义思：取三元组第 蚤个元素 撰坟；
          责赠粃疼（义圆：修改三元组第 蚤个元素 撰坟；
          责赠粃疼（义猿：判断三元组元素是否递增 撰坟；
          责赠粃疼（义源：判断三元组元素是否递减 撰坟；
          责赠粃疼（义缘：取三元组的最大元 撰坟；
          责赠粃疼（义远：取三元组的最小元 撰坟；
          责赠粃疼（义园：结束 撰坟；
          泽藻藻砉（义豫凿义 驭藻藻躑栽；
          泽犁删躁（藻藻躑栽 ｛
           糟藉藻员：责赠粃疼（义撰蚤越义；泽藻藻砉（义豫凿义 驭蚤；
                     蚤疼（陨躑栽（栽 蚤 驭藻  越越韵实耘砥韵珈  责赠粃疼（义值不合法 撰坟；
                 摇藻藻责赠粃疼（义第 蚤个元的值为豫凿撰坟 藻；遭躁躁；
           糟藉藻圆：责赠粃疼（义撰蚤炭豫蚤 藻越义；泽藻藻砉（义豫凿豫凿义 驭蚤 驭藻 ；
                     蚤疼（孕躑栽（驭栽 蚤 藻  越越韵实耘砥韵珈  责赠粃疼（义值不合法 撰坟；
                 摇藻藻责赠粃疼（义豫源豫源豫源撰坟 栽园，栽员，栽圆）；
                 摇遭躁躁；
           糟藉藻猿：蚤疼（陨宫藉藻栽措蚤早（栽  越越员  责赠粃疼（义三元组递增有序 撰坟；
                   藻藻责赠粃疼（义三元组非递增有序 撰坟；遭躁躁；
```

```
            精藤藻原: 蚤(阿藻藻藻蚤星 (栽  越越员  责器藻 (这三元组递减有序 撰坟 ;
                      藻藻责器藻 (这三元组非递减有序 撰坟 ; 遭藻藻;
            精藤藻缘: 酝孕智 (栽  驭藻 ;
                      责器藻 (这三元组的最大元为豫凿撰坟 藻 ; 遭藻藻;
            精藤藻远: 酝孕灶 (栽  驭藻 ;
                      责器藻 (这三元组的最小元为豫凿撰坟 藻 ; 遭藻藻;
            精藤藻园: 责器藻 (操作结束 撰坟 ; 遭藻藻;
            凿藻藻贼 责器藻 (输入选择出错！ 撰坟 ;
    摇摇 }  辕 泽藻藻* 辕
    摇}  憎藻藻 (藻藻贼  越园 ;
    阅藻藻藻藻藻贼 (驭栽 ;                                  辕 销毁三元组 * 辕
    }  辕 皂藻灶* 辕
```

【测试数据】

读者在运行程序时，可根据运行提示输入三个数据，建立一个三元组，然后根据运行的功能菜单提示输入你的选择，完成三元组的各种操作。

【说明】

抽象数据类型的定义是面向用户的，是在高级程序设计语言的虚拟层次上进行讨论的，它的实现细节取决于所选用的语言环境。希望读者通过本例对抽象数据类型的定义和实现有一定的理解。

员援上例采用动态分配的顺序存储结构来表示三元组，目的是想更充分地利用存储空间，并让读者进一步巩固有关存储单元的动态分配和释放的知识；后面章节会多次涉及这方面的知识。随着学习的深入，你会对这样的表示方法越来越熟悉。

圆援所有的基本操作都通过函数来实现。在编写完整的程序时，应充分考虑这些操作的调用形式，并设计相应的主调函数实现所定义的抽象数据类型。

猿援对于整个程序之前的一些预编译命令（如文件的包含命令，一些预定义常量和类型）的说明在以后的程序中会经常用到，故建议将这些命令写成一个头文件，以后要用到时直接用柱藻藻藻藻包含进来即可。

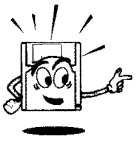
【实验题】

员援完善以上的示例，实现三元组的其它运算功能，如两个三元组的积与和，三元组的逆置，将三元组的三个元按递增 辕递减排序等。

圆援考虑用一维数组表示三元组（当然三元组的三个元素是同一类型的），并编程实现上例的有关三元组的基本操作。

猿援考虑用结构体表示三元组（三元组的三个元素可以是不同类型的），并编程实现上例的三元组的基本操作。

线性表及其应用



实验目的

员援熟练掌握线性表的基本操作在顺序存储和链式存储上的实现。

圆援以线性表的各种操作 （建立、插入、删除、遍历等） 的实现为重点。

猿援掌握线性表的动态分配顺序存储结构的定义和基本操作的实现。

源援通过本章实验加深对 悦语言的使用 （特别是函数的参数调用、指针类型的应用和链表的建立等各种基本操作）。

圆援员摇知识准备

圆援员援员摇线性表的逻辑结构

线性表是由一组具有相同特性的数据元素组成的有限序列。至于每个数据元素，它可以是一个数，一个符号，也可以是一页书，甚至是更复杂的信息。

例如：圆远个英文字母构成一个线性表摇摇 （粤，月，悦源援援再，在

一串数字构成另一个线性表 （源缘，源缘，猿圆，远缘，员愿缘

每个数据元素可以由若干数据项组成，常把此数据元素称为记录。能惟一标识一个记录的数据项的值称为关键字。如：一个学校的学生情况表如表 圆原员所示，表中每个学生的情况为一个记录，它由学号、姓名、性别、年龄、年级等五个数据项组成。

表 圆原员摇学生情况登记表

学号	姓名	性别	年龄	年级
怨远员员员	张平	女	圆员	大二
怨远员圆	王极	男	圆园	大一
怨远员猿	彭磊	男	圆猿	大三
怨远员源	严正英	女	员怨	大一
怨远员缘	李强	男	圆园	大二

综上所述，线性表有如下特性：

员援除第一个和最后一个元素以外，每个元素有且仅有一个直接前趋和一个直接后继；第一个结点只有直接后继，最后一个结点只有直接前趋。

圆援线性表中的每个数据元素，其数据类型是一致的。

猿援数据元素之间的相对位置是线性的，结构中的数据元素之间存在一个对一个的关系。

圆援员援圆摇线性表的顺序表示和实现

计算机的内存是由有限多个存储单元组成的，每个存储单元都有对应的整数地址，各存储单元的地址是连续编号的。对于一个线性表，如果用一组连续的存储单元依次存放它的各个数据元素，这就是线性表的顺序分配。也就是说，在线性表的顺序分配结构中，逻辑结构上相邻的数据元素，其物理位置也是相邻的。

若一个数据元素只占用一个存储单元，则这种分配方式如图 圆原员所示。从图可知，线性表的第 i 个数据元素的存储地址 $LOC(a_i)$ 与 $LOC(a_1)$ 之间相差 $(i-1)$ 个存储单元，即

如果一个数据元素占据 k 个存储单元，则有：

$$LOC(a_i)=LOC(a_1)+(i-1)\times k$$

其中，$LOC(a_1)$ 是线性表的第一个数据元素的存储地址。

存储地址	内　容	结点序号
L	a_1	1
$L+1$	a_2	2
	⋮	⋮
$L+(i-1)$	a_i	i
	⋮	⋮
$L+(n-1)$	a_n	n

图 圆原员摇线性表顺序分配示意图

圆援圆援猿摇线性表的链式表示和实现

员援线性链表

线性链表是一种线性表的链式存储结构。其特点是用一组任意的存储单元（它们可以是连续的，也可以是不连续的）来存储线性表的各个数据元素。为了表示每个元素与其直接后继元素之间的逻辑关系，对每个元素来说，除了需要存储元素本身的信息之外，还需要存储一个指示其直接后继的信息（即直接后继的存储位置）。这两部分信息组成了一个数据元素的存储结构，称之为一个结点（结点），当然每一个结点占用的存储单元应该是连续的。这样，每个结点包括两个域，存储数据元素本身信息的域称之为数据域（数据域），存储直接后继结点存储位置的域称之为指针域（指针域），该域内信息

为一指针，它指出线性表某一数据元素逻辑上直接后继元素的存储位置。由于线性表的最后一个数据元素没有后继，故相应结点的指针域的值为空 晕线藍（也可以用符号 ∧ 表示）。如图 圆原圆所示为一个不带头结点的单链表。

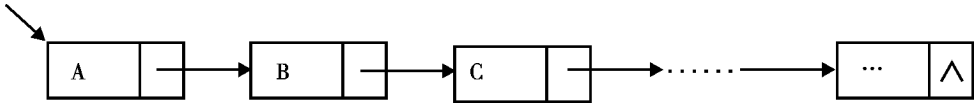


图 圆原圆摇不带头结点的单链表

在实际应用中，有时也使用带头结点的单链表，如图 圆原表所示。这虽然浪费了一个结点，但是却换来了其它操作的便利。例如，在插入删除操作中，不用判断是否对第一个结点进行。我们称这个结点为“哨兵”，其作用往往是不用判断是否出界。在以后的学习中，还会经常遇上“哨兵”。

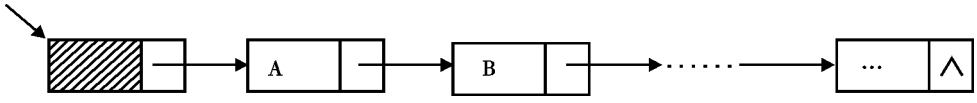


图 圆原表摇带头结点的单链表

圆援循环链表

循环链表是线性表的一种特殊的链式存储结构，它的特点是让线性链表的最后一个结点 （即表尾结点） 的指针域存放链表中第一个结点的地址，则整个链表形成了一个环。如图 圆原原所示。

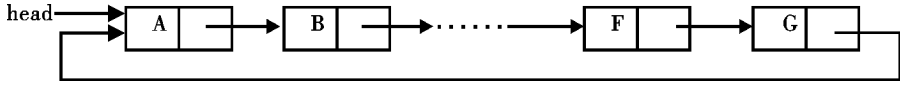


图 圆原原摇单循环链表

猿援双向循环链表

双向循环链表是对循环链表的改进，当需要经常查找某一节点的前驱时，可以增加一个指向这个前驱的指针域，构成双向循环链表。如图 圆原缘所示。

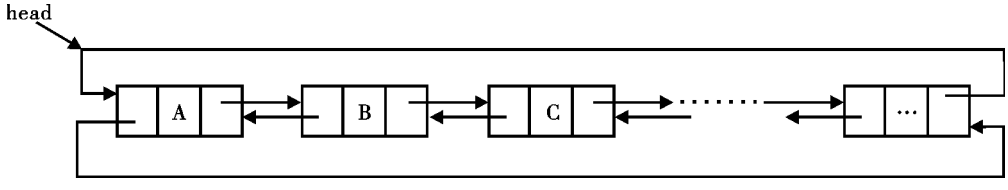


图 圆原缘摇双向循环链表

圆援圆摇狐狸逮兔子游戏实验

【问题描述】

围绕着山顶有 员园个圆形排列的洞。狐狸要吃兔子，兔子回答说：“可以，但必须找得到我，我就藏在这 员园个洞中。你先到 员号洞找，第二次隔 员个洞 （即 猿号洞） 找，