高等院校“十二五”核心课程辅导丛书

数据结构（C语言版）

答疑解惑与典型题解

单忆南 孙 涵 唐军军 编著

北京邮电大学出版社
www.buptpress.com

高等院校“十二五”核心课程辅导丛书

数据结构答疑解惑与典型题解

（C 语言版）

单忆南　孙涵　唐军军　编　著

北京邮电大学出版社
• 北京 •

内 容 简 介

本书是为熟悉C语言编程的读者学习数据结构而编写的教学辅导书，可帮助读者复习课程的基本内容，并学会用C语言使用相应的数据结构实现一定的算法和解决一些实际应用问题，力争使读者在学完本书之后，在课程的理解和掌握方面达到一个新的高度，本书也可供从事本课程教学的教师作参考书。

本书共分十章，包括数据结构概述、线性表、栈和队列、串和字符串、数组和广义表、树和二叉树、图、查找、排序，在全书最后给出了一套模拟试题和一套考研真题及参考答案。

本书每章内容均包括各基本知识点的要点归纳，并精选一些经典数据结构书中的经典例题(包括课程考试试题、主流教材课后难题以及考研真题)，给出了解题思路和分析方法，题后提示了解题中应注意的问题。力争使读者在尽可能短的时间内，巩固课程基本概念，加深理解数据结构的基本知识并融会贯通，熟练掌握基本的编程方法并举一反三，不断提高读者的C语言编程能力和利用各种数据结构解决实际问题的能力。

本书可供学习数据结构课程的读者以及考研读者和从事课程教学的教师参考。

图书在版编目(CIP)数据

数据结构(C语言版)答疑解惑与典型题解/单忆南，孙涵，唐军军编著．--北京：北京邮电大学出版社，2010.9
ISBN 978-7-5635-2285-9

Ⅰ.①数… Ⅱ.①单…②孙…③唐… Ⅲ.①数据结构②C语言—程序设计 Ⅳ.①TP311.12②TP312

中国版本图书馆CIP数据核字(2010)第175817号

书　　名：数据结构(C语言版)答疑解惑与典型题解
作　　者：单忆南　孙涵　唐军军
责任编辑：满志文
出版发行：北京邮电大学出版社
社　　址：北京市海淀区西土城路10号(邮编：100876)
发 行 部：电话：010-62282185　传真：010-62283578
E-mail：publish@bupt.edu.cn
经　　销：各地新华书店
印　　刷：北京忠信诚胶印厂
开　　本：787 mm×1 092 mm　1/16
印　　张：18.75
字　　数：621千字
版　　次：2010年9月第1版　2010年9月第1次印刷

ISBN 978-7-5635-2285-9　　定　价：35.00元

前　言

本书是为熟悉C语言的读者学习数据结构而编写的教学辅导书，可帮助读者复习课程的基本内容，检验各种数据结构形式和算法的掌握程度，培养和提高用C语言解决实际问题的能力，力争使读者在学完本书之后，在运用数据结构解决实际问题和C语言编程方面都达到一个新的高度。

1. 本书阅读指南

全书共分10章。

第1章主要介绍数据结构的基本概念，包括数据结构的应用、意义、时间复杂度、空间复杂度等内容。

第2章主要介绍线性表这个数据结构类型，包括线性表的定义和基本概念、线性表的存储、循环链表和双向链表等内容。

第3章主要介绍栈和队列的数据结构类型，包括栈的定义和基本操作、栈的存储、队列的定义和基本操作、队列的存储结构等内容。

第4章主要介绍串的数据结构类型，包括串的操作、串的模式匹配、以及KMP算法的理解等内容。

第5章主要介绍数组和广义表的数据结构类型，包括多维数组、特殊矩阵的存储、稀疏矩阵的存储、广义表的性质和操作等内容。

第6章主要介绍树和二叉树的数据结构类型，包括树和二叉树的性质和定义，二叉树的遍历，表达式的二叉树表示，线索化二叉树，树和森林，霍夫曼树等内容。

第7章主要介绍图的数据结构类型，包括图的基本性质和概念、图的存储、图的遍历、图的连同性、最小生成树算法、最短路径问题、拓扑排序等内容。

第8章主要介绍查找操作及相应的数据结构，包括查找的定义、顺序查找、折半查找、分块查找、二叉排序树、平衡二叉树、B一树、哈希表和散列表等内容。

第9章主要介绍排序操作及应用的数据结构，包括排序的概念、插入排序、冒泡排序、选择排序、归并排序、基数排序、各种内部排序时间复杂度空间复杂度的比较和外部排序等内容。

第10章给出一套课程测试和一套研究生入学考试全真预测试题及参考答案。

2. 本书的特色与优点

本书编写的指导思想是：在内容上重视数据结构的基本理论，利用C语言作为程序语言进行描述，覆盖课程全部基本教学要求；书中习题主要来自于经典数据结构教材中的经典习题，全书习题经过编者精挑细选，难度适中，适合各专业学习本课程的学生；在形式上根据教学实践经验和对相关内容的思考理解，简明描述课程的基本知识点、重点和难点内容，使学生迅速把握重点。

本书每章内容均包括各基本知识点的要点归纳，并精选一些具有代表性的例题，给出了解题思路和分析方法，部分编程题给出了实现代码，题后提示了解题中应注意的问题。这样编写

的目的在于：力争使读者在尽可能短的时间内，巩固课程基本概念，加深理解数据结构并融会贯通，熟练掌握编程的基本方法并举一反三，不断提高读者用各种数据结构解决实际应用问题的能力。在全书最后给出了一套研究生入学考试全真预测及参考答案，同时本书也可用作考研的辅导书。

3. 本书读者定位

本书可供熟悉C语言的读者和考研读者学习数据结构以及从事数据结构课程教学的教师参考。

本书由单忆南、孙涵、唐军军编著，全书框架结构由何光明、吴婷拟定。另外，感谢王珊珊、陈智、陈海燕、吴涛涛、李海、张凌云、陈芳、李勇智、许娟、史春联等同志的关心与帮助。

由于编者水平和经验有限，加之编写时间仓促，本书难免会有不妥或错误之处，敬请广大读者批评指正。如遇到疑难问题可通过以下方式与我们联系：bjbaba@263.net。

作　者

目　录

第1章 数据结构概念理解

1.1 答疑解惑

1.1.1 为什么要用数据类型来描述数据结构?

采用数据类型来描述数据结构是基于以下考虑:

(1) 数据类型(Data Type)是一个值的集合和定义在这个值集上的一组操作的总称。解决现实问题就必须进行数据处理,而数据处理包括对数据进行查找、插入、删除、合并、排序、统计以及简单计算等的操作过程。

(2) 数据类型是高级程序设计语言中的一个基本概念,它和数据结构的概念密切相关。一方面,在程序设计语言中,每一个数据都属于某种数据类型。类型明显或隐含地规定了数据的取值范围、存储方式以及允许进行的运算。可以认为,数据类型是在程序设计中已经实现了的数据结构。另一方面,在程序设计过程中,当需要引入某种新的数据结构时,总是借助编程语言所提供的数据类型来描述数据的存储结构。

(3) 用高级语言数据类型来描述数据结构,更避免了低级语言的复杂性,增加了可读性和简洁性,又有利于算法的实现。

数据结构正逐步用抽象数据类型描述,其更能紧密地和面向对象程序设计联系起来。抽象数据类型(Abstract Data Type,ADT)是指一个数学模型以及定义在该模型上的一组操作。

1.1.2 算法和程序有何区别?

数据结构中的算法通常用高级语言来描述,但和高级语言有一定的区别。如:目前的数据结构算法常用类 C 和类 Pascal 来描述。类 C 和标准 C 有一定的区别,同时,算法着重思想的描述,可能省略了很多细节。因而算法往往需要进行适当的修改才能变成程序在机器上实现。

算法必须满足 5 个特征,但程序未必能满足有穷性。

可见,算法不同于程序。不要把算法看成程序,切忌将算法中的相应函数和数据类型直接照搬到程序中。

1.1.3 怎样理解数据的逻辑结构和存储结构?

很多同学在学习数据结构时,难以深刻领会数据的逻辑结构和数据的存储结构,使得基本

概念模糊。

众所周知,数据的逻辑结构是指结构定义中的“关系”,描述的是数据元素之间的逻辑关系。而数据的物理结构(又称存储结构)是数据结构在计算机中的表示(又称映象)。它包括数据元素的表示和关系的表示。这样使得逻辑上相邻的数据元素,物理上未必相邻。例如:父子关系可以看成是逻辑关系,逻辑上是相邻;但他们未必生活在同一个地方,在物理上可能生活在不同的地方,甚至在不同的国家。

从数学的角度来看,数据的物理结构是数据的逻辑结构的函数映射,是由逻辑地址到物理地址的映射。逻辑地址相邻的地址可能映射到物理上不相邻的存储地址,但数据元素的内在关系保持不变。

1.1.4 怎样理解数据结构在计算机课程中的核心地位?

为什么数据结构在计算机课程中起核心作用?主要表现在以下两个方面:(1)数据结构课程的性质;(2)数据结构课程和其他课程之间的关系。

解决现实问题的方法一般分为这样几个步骤:(1)分析现实问题并得到相应的数学模型。(2)设计出解决该数学模型的算法。(3)编程实现该算法得到问题的解。而数据结构几乎体现在问题求解的各个步骤,尤其是步骤(2)。可见,学好数据结构具有十分重要的意义。

同时,数据结构依赖于前续课程的学习,使得前续课程得到应用和巩固;也为后续课程的学习打下扎实的基础,尤其是“编译原理”及“操作系统”等课程。

1.1.5 如何计算算法的时间复杂度?

用不同的机器,算法的执行时间不同,不能用执行时间长短来衡量算法的时间复杂度。算法的时间复杂度通常用算法中的主要操作步来度量。例如,求两个 n 阶方阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的乘积 $\boldsymbol{C}$ 的算法如下:

```
for(i = 0;i<n;i ++ )
    for(j = 0;j<n;j ++ )
       { C[i][j] = 0;
          for(k = 0;k<n;k ++ )
             C[i][j] + = A[i][k] * B[k][j];
       }
```

由于加法相对乘法来说可忽略不计,因而总的乘法操作步为 $f(n)=n^3=O(n^3)$。

本质上,一个算法的时间复杂度就是得到该算法的主要操作步的执行次数,而算法之间的比较就是相应的极限之间的比较。

1.1.6 如何评价算法的好坏?

求解同一计算问题可能有许多不同的算法,究竟如何来评价这些算法的好坏以便从中选出较好的算法呢?

选用的算法首先应该是“正确”的。此外,主要考虑如下三点:

(1) 执行算法所耗费的时间。

(2) 执行算法所耗费的存储空间,其中主要考虑辅助存储空间。

(3) 算法应易于理解,易于编码,易于调试,等等。

一个占存储空间小、运行时间短、其他性能也好的算法是很难做到的。原因是上述要求有时相互抵触：要节约算法的执行时间往往要以牺牲更多的空间为代价，而为了节省空间可能要耗费更多的计算时间。因此只能根据具体情况有所侧重：

(1) 若该程序使用次数较少，则力求算法简明易懂。

(2) 对于反复多次使用的程序，应尽可能选用快速的算法。

(3) 若待解决的问题数据量极大，机器的存储空间较小，则相应算法主要考虑如何节省空间。

1.2 典型题解

题型 1 数据结构基础知识

【例 1-1★】 (西南交通大学)数据的_____包括集合、线性结构、树和图结构这 4 种基本类型。

A. 存储结构　　B. 逻辑结构　　C. 基本运算　　D. 算法描述

解答：B

【例 1-2】 在数据结构中，从存储结构上可以把数据结构分成_____。

A. 顺序存储和链式存储　　B. 紧凑结构和非紧凑结构

C. 线性结构和非线性结构　　D. 动态结构和静态结构

解答：A

【例 1-3★】 (中南大学)数据结构在计算机内存储器中的表示是指_____。

A. 数据结构　　B. 数据元素之间的关系

C. 数据的逻辑结构　　D. 数据的物理存储结构

分析：本题考查数据结构的基本概念。

解答：D

【例 1-4★】 (中山大学)一个完整的算法应该具有有穷性、确定性和可行性等。其中有穷性是指_____。

A. 在有穷时间内终止　　B. 输入是有穷的

C. 输出是有穷的　　D. 描述是有穷的

解答：A

【例 1-5★】 (北京师范大学)简述数据的 4 种存储方式及其特点。

解答：数据元素之间通常有 4 种存储结构。

(1) 集合：结构中的数据元素之间除了“同属于一个集合”无其他关系。

(2) 线性结构：结构中的数据元素之间存在一个对一个的关系。

(3) 树形结构：结构中的数据元素之间存在多个对多个的关系。

(4) 突状结构：结构中的数据元素之间存在多个对多个的关系。

【例 1-6】 请叙述数据结构的逻辑描述及物理描述的方法。

解答：逻辑结构有 4 种基本类型：集合结构、线性结构、树状结构和网络结构。表和图是最常用的两种高效数据结构，许多高效的算法可以用这两种数据结构来设计实现。表是线性结构的(全序关系)，树(偏序或层次关系)和图[局部有序(weak/local orders)]是非线性结构。

数据结构的物理结构是指逻辑结构的存储镜像(image)。数据结构 DS 的物理结构 P 对应于从 DS 的数据元素到存储区 M(维护着逻辑结构 S)的一个映射：P:(D,S)→M。

题型2 时间与空间复杂度的计算

【例1-7】 某算法的时间复杂度为$O(n^2)$,表明该算法的_____。

A. 问题规模是n^2　　B. 执行时间等于n^2

C. 执行时间与n^2成正比　　D. 问题规模与n^2成正比

分析:考查时间复杂度的定义。

解答:C

【例1-8】 下面算法的时间复杂度是_____。

```
void suanfa3(int n)
{
int i=1,s=1;
    while (s<n) s+= ++i;
return i;
}
```

A. $O(n)$　　B. $O(2^n)$　　C. $O(\log_2 n)$　　D. $O(\sqrt{n})$

分析:假设while循环执行了k次,即$s=1+2+3+\cdots+k+1=(k+1)(k+2)/2$,而循环结束的条件为$s\geqslant n$,所以$(k+1)(k+2)/2\geqslant n$,所以时间复杂度为$O(\sqrt{n})$。

解答:D

【例1-9】 (南京邮电大学)可以使用大O记号表示一个算法的时间复杂度。下列表示中不正确的是_____。

A. $n^2+2n=O(n^3)$　　B. $n\log_2 n+2n=O(n^2)$

C. $n^2+n\log n=O(n^2\log_2 n)$　　D. $n^2+n\log_2 n=O(n\log_2 n)$

解答:ABCD

【例1-10】 (西南交通大学)假设n为2的乘幂,且$n>2$,指出下面算法的时间复杂度及变量count的值。

```
int Time(int n){
    count =0;x=2;
    while(x<n/2)
    {    x*=2;count++;}
    return count;
}//Time
```

时间复杂度为_____;count=____。

解答:$O(\log n)$,$\log_2^{n/2}-1$

【例1-11】 (西南交通大学)下面算法的时间复杂度为_____。

```
BTree SearchBST(BTree T,keytype key){
  if((!T)||EQ(key,T->data.key))
      return (T);
  else if LT(key,T->data.key)
      return(SearchBST(T->lchild,key));
  else
      return(SearchBST(T->rchild.key));
}
```

解答:$O(\log n)$

【例1-12】 (北京交通大学)判断题:有算法如下,若k值位于$a[j]$中的概率为$2^{-(j+1)}$,k不在a中的概率为2^{-a},那么该算法的时间复杂度为$O(n)$。

```
int locate(int a[],int n,int k)
{
   int i;
   for(i=0;i<n;i++) if(a[i]=k) return i;
   return -1;
}
```

解答:该段程序的时间复杂度主要看for循环中,与k值是否在数组a中无关。因此,根据for循环中的$i<n$可见,它的时间复杂度应该是$O(n)$。

【例1-13】 (清华大学)斐波那契数列F_n定义如下:

$$F_0=0,F_1=1,F_n=F_{n-1}+F_{n-2},n=2,3,\cdots$$

请就此斐波那契数列回答下列问题:

① 在递归计算 F_n 的时候，需要对较小的 F_{n-1}，F_{n-2}，…，F_1，F_0 精确计算多少次？

② 若干有关大 O 表示法，试给出递归计算 F_n 时递归函数的时间复杂度是多少？

分析：本题考查斐波那契数列。①可以考虑先用一些小数值代入 n，看看情况，例如 $n=6$ 时，发现 F_5，F_4，F_3，F_2，F_1，F_0 的值分别为：1,2,3,5,8,5。$F_0=F_2$，除去 F_0 外，发现其余值正好构成一个斐波那契数列，发现该规律之后，可以用数学归纳法来证明。

解答：①当 $n=2$ 时，F_1 计算 1 次，F_0 计算 1 次。

$n>3$ 时，有如下命题：F_{n-1}，F_{n-2}，…，F_1 的计算次数为 F_1，F_2，…，F_{n-1}，F_0 计算次数 $=F_2$ 计算次数。下用数学归纳法来证明。

当 $n=3$ 时，F_2 计算 1 次，F_1 计算 2 次，F_0 计算 1 次，满足。

当 $n=4$ 时，F_3 计算 1 次，F_2 计算 2 次，F_1 计算 3 次，F_0 计算 2 次，满足。

假设 $n\leqslant k$ 时，F_{k-1}，F_{k-2}，…，F_1 的计算次数为 F_1，F_2，…，F_{k-1}，F_0 计算次数 $=F_2$ 计算次数。

当 $n=k+1$ 时，$F_{k+1}=F_k+F_{k-1}$，故 F_k 计算一次，$F_i(1\leqslant i\leqslant k-1)$ 的计算次数为 $F_{k-i}+F_{k-i-1}=F_{k-i+1}$，故满足。

综上，命题成立。

② 设 F_n 的计算时间为 $T(n)$，则有

$$T(n)=T(n-1)+T(n-2)$$

通过画出该函数的递归树，可以发现，该递归树的高度约为 n，结点总数为 $O(2^n)$，所以时间复杂度为 $O(2^n)$。

【例 1-14】 回文是指正反读均相同的字符序列，如“abba”和“abdba”均是回文，但“good”不是回文，试写一个算法判断给定的字符向量是否为回文。

分析：判断回文只需要对字符向量从第一个位置开始，比较第 i 和第 $n-1-i$ 个位置的元素是否相等（$i=0,1,\cdots,n/2-1$）。

解答：

```
int test1(char * a,int l){ //判断长度为 l 的字符数组 a 的内容是否为回文
for(int i = 0;i<n/2;i ++ ){
      if(a[i]! = a[n - 1 - i])return 0;
}
return 1;
}
```

第2章

线性表

2.1 答疑解惑

2.1.1 如何理解线性表数据结构？

线性表的基本特征清晰地反映了线性表数据结构的特点，其基本特征为在一个非空线性结构中，有且只有一个称为第一个的元素；有且只有一个称为最后一个的元素；第一个元素无前驱，最后一个元素无后继；其余每个元素均有唯一前驱和唯一后继。

从数学的角度来看

$$\text{Linear\_list}=(D,R)$$

其中，D 是数据元素的集合：$D=\{a_i \mid a_i\in D_0, i=1,2,\cdots,n, n\geqslant 0\}$；$R$ 是数据元素间关系的集合：$R=\{N\}$，$N=\{<a_{i-1},a_i> \mid a_{i-1},a_i\in D_0, i=1,2,\cdots,n, n\geqslant 0\}$。

D_0 是具有某种性质的数据元素的集合。

2.1.2 线性表的顺序存储结构和链式存储结构的区别是什么？

若线性表的总数基本稳定，且很少进行插入和删除，但要求以最快的速度存取线性表中的元素，那么应采用哪种存取结构？

由于顺序存储结构一旦确定了起始位置，线性表中的任何一个元素都可以进行随机存取，即存取速度较高；并且，由于线性表的总数基本稳定，且很少进行插入和删除，故这一特点恰好避开了顺序存储结构的特点。因此，应用顺序存储结构。

可见，若需要在线性表的任意位置进行频繁地插入和删除，采用链式存储结构比较合适。

链式存储结构中的结点内部存储空间是连续的，但结点之间存储空间不一定连续。特别注意的是：尽管链式存储结构的物理地址未必连续，但结点之间仍然保持其逻辑次序，逻辑次序关系隐含在结点中。

顺序表和链表各有短长。在实际应用中究竟选用哪一种存储结构呢？这要根据具体问题的要求和性质来决定。通常有以下几方面的考虑，如表 2.1 所示。

表 2.1

		顺序表	链表
基于空间考虑	分配方式	静态分配。程序执行之前必须明确规定存储规模。若线性表长度 n 变化较大，则存储规模难于预先确定，估计过大将造成空间浪费，估计太小又将使空间溢出机会增多	动态分配。只要内存空间尚有空闲，就不会产生溢出。因此，当线性表的长度变化较大，难以估计其存储规模时，以采用动态链表作为存储结构为好
	存储密度	存储密度为 1。当线性表的长度变化不大，易于事先确定其大小时，为了节约存储空间，宜采用顺序表作为存储结构	小于等于 1
基于时间考虑	存储方式	随机存取结构，对表中任一结点都可在 $O(1)$ 时间内直接取得。线性表的操作主要是进行查找，很少做插入和删除操作时，采用顺序表做存储结构为宜	顺序存取结构，链表中的结点，需从头指针起顺着链扫描才能取得
	插入删除操作	在顺序表中进行插入和删除，平均要移动表中近一半的结点，尤其是当每个结点的信息量较大时，移动结点的时间开销就相当可观	在链表中的任何位置上进行插入和删除，都只需要修改指针。对于频繁进行插入和删除的线性表，宜采用链表做存储结构。若表的插入和删除主要发生在表的首尾两端，则采用尾指针表示的单循环链表为宜

存储密度(Storage Density)是指结点数据本身所占的存储量和整个结点结构所占的存储量之比，即：

存储密度 =(结点数据本身所占的存储量)/(结点结构所占的存储总量)

2.1.3 带头结点的单链表和不带头结点的单链表的区别是什么？

带头结点的单链表和不带头结点的单链表的区别主要体现在其结构上和算法操作上。

在结构上，带头结点的单链表不管链表是否为空，均含有一个头结点，而不带头结点的单链表不含头结点。

在操作上，带头结点的单链表的初始化为申请一个头结点，且在任何结点位置进行的操作算法一致；而不带头结点的单链表让头指针为空，同时其他操作要特别注意空表和第一个结点的处理。下面列举带头结点的单链表插入操作和不带头结点的插入操作的区别。

2.1.4 链表的指针修改的次序对结果的影响是什么？

链表的指针修改必须保持其逻辑结构的次序，否则将违背线性表的特征，尤其是进行插入和删除操作。下面通过双向链表的插入操作来说明，若在图 2.1 所示的 P 所指向的结点之前插入一个 S 所指向的结点，则需进行指针的修改，修改指针的策略有图 2.2 和图 2.3 两种，指针的修改次序为 1,2,3,4。根据线性表的性质知，图 2.2 可保证指针修改成功；而图 2.3 中指针修改不成功，主要原因是其首先将 P 的前趋指向 S，这样 P 结点的原前趋结点就不能找到了，因而指针修改步骤 3 和 4 不成立。

可见，指针的修改次序是链表插入成功与否的关键因素之一；同理，在进行结点的删除时

也同样需要主要指针的次序。

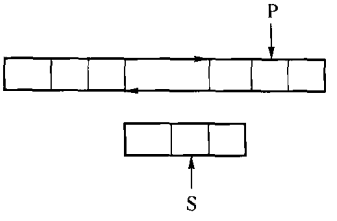

图 2.1　双向链表的结点插入前

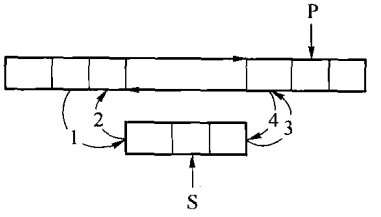

图 2.2　指针的修改策略 1

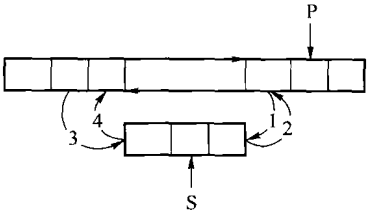

图 2.3　指针的修改策略 2

2.1.5　各种链表存储结构的特点是什么?

带头结点的链表和不带头结点的链表的特点已在 2.1.3 问题中进行了说明。单链表结构简单,而双向链表的结构略复杂些,但可方便地找到任何一个结点的前趋和后继结点,常作为编辑器的数据结构。

循环链表和循环双向链表可通过尾结点找到头结点,也常作为编辑器的数据结构,尤其是循环双向链表。

2.1.6　顺序存储结构上的算法如何移植到链式存储结构上?

很多优秀的算法都是建立在顺序存储结构上,如何在链式存储结构上实现这些优秀算法,是考生应注意的问题。近年来,不少重点大学的研究生入学试题上都出现了这样的题目。这里通过在顺序存储结构和链式存储结构两种存储结构上实现选择排序来说明。依据顺序存储结构下的算法 sort1 可以拓展到链式存储结构下的算法 sort2。

```
sort1(int R[],int n)
/*对 n 个元素的数组 R 进行由小到大排序*/
{
 int i,j,k,temp;
 for(i=0; i<n-1; i++)
     { k=i;
         for(j=i+1; j<n; j++) /*在 R[i..n-1]中找最小的元素*/
           if(R[j]<R[k])k=j;
     if(k!=i){ temp=R[i];R[i]=R[k];R[k]=temp;}
     }
}/*sort1*/
```

下面将算法 sort1 拓展到链式存储结构下的算法 sort2。

```
typedef struct node {
            int data;                /*结点的数据域*/
            struct node *next;       /*结点的指针域或链域*/
} Slink;
sort2(Slink *head)
/*将带头结点的单链表 head 进行由小到大排序*/
{
 Slink *p=head->next, *q, *r;
 int temp;
 while(p){
     q=p;
     r=p->next;
     while(r){ if(r->data<q->data)q=r; /*在以 p 为首结点的单链表中找最小
                                          的元素*/
               r=r->next;
              }
     if(q!=p){ temp=p->data; p->data=q->data;q->data=temp;}
         /*找到的结点和 p 结点数据交换*/
       p=p->next;
  }
 }/*sort2*/
```

可见，只要充分领会顺序存储结构下的算法思想，熟悉链表存储结构就可通过掌握顺序存储结构下的算法得到链表存储结构下的相应算法。

2.1.7 如何利用循环单链表实现队列的操作？

队列操作是先进先出，可用让指针指向循环单链表的表尾位置。进队操作在表尾进行，而出队操作在表头进行。由于在循环链表中很容易由表尾找到表头，因而很方便地进行队列操作。

2.1.8 如何应用线性表？

线性表的应用非常广泛，参考文献[1]和大多数教科书中都用一元多项式相加、减及乘等。也可用于实现多个整数的相加等。

2.1.9 顺序表的基本运算用 C 语言如何描述？

(1) 表的初始化。

```
void InitList(SeqList *L)                    L->length=0;
{//顺序表的初始化即将表的长度置为 0      }
```

(2) 求表长。

```
int ListLength(SeqList *L)                   return L->length;
{//求表长只需返回 L->length              }
```