河南省高等职业教育规划教材

数据结构

主编　张红霞

河南大学出版社

河南省高等职业教育规划教材

数据结构

主　编　张红霞

河南大学出版社

图书在版编目(CIP)数据

数据结构/张红霞主编. —开封:河南大学出版社,2003.8
河南省高等职业教育规划教材
ISBN 7-81091-075-2

Ⅰ.数… Ⅱ.张… Ⅲ.数据结构—高等学校:技术学校—教材 Ⅳ.TP311.12

中国版本图书馆 CIP 数据核字(2003)第 060033 号

书　　名 数据结构
主　　编 张红霞

策　　划 史锡平　朱建伟
责任编辑 程　庆
责任校对 文　严
责任印制 苗　卉
封面设计 张　伟
版式设计 苗　卉

出　　版 河南大学出版社
地址:河南省开封市明伦街 85 号　　邮编:475001
电话:0378—2864669(行管部)　　0378—2825001(营销部)
网址:www.hupress.com　　E-mail:bangong@hupress.com
经　　销 河南省新华书店
排　　版 河南大学出版社印务公司
印　　刷 河南省诚和印制有限公司
版　　次 2003 年 8 月第 1 版　　**印　　次** 2003 年 8 月第 1 次印刷
开　　本 787mm×1092mm　1/16　　**印　　张** 11.75
字　　数 293 千字　　**印　　数** 1—4200 册

ISBN 7-81091-075-2/T·46　　**定　　价** 17.00 元

序

经河南省教育厅批准，由河南省高等职业教育研究会组织编写的河南省高等职业教育规划教材，就要付梓出版了。这是我省高教事业改革发展的一项重要成果，确实值得庆贺。

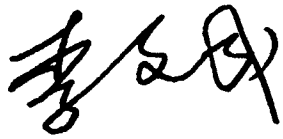
大力发展教育和科学事业，培养和造就数以亿计的高素质劳动者、数以千万计的专门人才和一大批拔尖创新人才，是党的十六大明确提出的新时期教育的任务。高等职业教育，作为高等教育的一种类型，其培养的是拥护党的基本路线，适应生产、建设、管理、服务第一线需要，德、智、体、美等方面全面发展的高等技术应用性专门人才。因而，是我国全面建设小康社会的一支重要力量。就其本质属性来说，高等职业教育具有鲜明的职业特征，这就要求我们在人才培养工作中，不能沿袭学科型教材，而是编写、出版和使用技术型教材，即要认真研究和改革高等职业教育的课程教学内容和教学方法，编写和出版体现高等职业教育规律和特点的优质教材，力求体现全面建设小康社会对高等技术应用性人才培养提出的新要求。从这个意义上来讲，河南省高等职业教育规划教材的编写出版，不仅非常必要，而且十分及时，它顺应了我国政治、经济、文化、科技发展的新形势，适应了高等教育尤其是高职高专教育改革发展的新趋势，对我省高职高专教育水平的提高将产生深远而积极的影响。

河南省高等职业教育研究会，作为省一级教育学会，在以赵金昭同志为会长的学会班子的组织和带领下，自2000年始，一直致力于高等职业教育理论与实践的研究工作，以专业建设为龙头，以教材建设为核心，以人才培养模式的建构为出发点，与时俱进，开拓创新，组织全省高职高专院校高水平的专家，研究并取得了一大批源自实践、富于特色、十分鲜活的教改成果。高等职业教育规划教材的编写、出版，正是这些研究成果的积淀和升华。

与全国其他同类教材相比，首批推出的计算机应用与维护、秘书、机电一体化等专业规划教材，有3个方面的显著特色：其一，适用性。教材编写人员，均是从事高职高专教育教学第一线的专家，全国知名的教授不乏其人。因此，规划教材体现了高职教育的特色，从而使教材的针对性和适应性得到完美的统一。其二，应用性。首批推出的高等职业教育规划教材有一个最显著的特色，就是强化和突出了应用性特征，每个专业的核心课程均配套编写了实训教材，如计算机应用与维护专业的《C语言程序设计实训教程》、秘书专业的《秘书实训》，机电一体化专业的《计算机工程制图实训教程》等，均将学生的实践能力培养纳入了教材建设体系。其三，新颖性。规划教材在内容的取舍上，遵循“基础理论必需、够用为度”的原则，适当精简验证性的原理阐述，大量充实新技术、新内容，及时反映本学科领域的最新科技成果，广泛吸收先进的教学经验，积极整合优秀教学成果，给人耳目一新的感觉。此外，在编写体例上，重视图表的运用，并在每章之后安排了思考题、实训题等供学习者练习，体现出编著者以人为本、注重技术应用能力培养的教育思想。

高等职业教育教材建设是一项十分重要的工作。因为，教材的基本作用，就是集人类先进的科学文化成果，传递给下一代，培养后继创新人才。优质的特色教材，在本质上是学校水平的体现。我们在肯定已编写的高等职业教育教材所取得成绩的同时，还要认识到我们在这方面改革探索的实践还不很充分，还需要继续进行广泛、扎实、深入的研究。并随着教育教学改革的深化，对出版的教材进行必要的充实、修改，使之日臻完善。

我相信，经过3～5年的努力，随着规划教材的陆续问世，随着系列统编教材在教育教学中的广泛使用，我们一定会迎来高职教育事业发展繁荣的新局面。



2003年8月20日

前　言

本教材由河南省高等职业教育研究会统一组织、征稿、编审，列为河南省高等职业教育规划教材。

计算机科学技术以惊人的速度迅猛发展，它的应用范围已渗入到社会和生活的各个领域。相应地，数据处理的对象也从简单的数值发展到字符、表格和图形等带有结构的数据。在这里要解决的关键问题是：针对每一种新的应用领域的处理对象，如何选择合适的数据表示(结构)，如何有效地组织数据、处理数据。数据结构就是研究数据以及数据之间关系的一门学科，主要研究数据之间的逻辑结构以及有关基本操作在计算机中的表示和实现。数据结构不仅是计算机专业重要的专业基础课，也是从事计算机软件开发必备的专业知识。

本书所选内容覆盖了数据结构的主要内容，主要培养学生分析数据、组织数据的能力，掌握编写效率高、结构好的程序的基本技巧。全书共分九章及附录(实训)。第1章介绍数据、数据结构、算法的性能分析等基本概念；第2至第4章介绍各种线性结构，包括线性表、栈、队列、串、数组；第5、6章介绍非线性结构，包括树形结构和图形结构；第7、8章介绍数据处理中广泛使用的两种技术：排序和查找；第9章介绍常用的文件组织结构。附录中对前面章节所介绍的每种数据结构都给出了典型的实训例题及精心挑选的实训习题。

作为高职高专的教材，本书注重突出高职高专的特点，理论够用即可，重点是应用。介绍每种数据结构都力求由浅入深，循序渐进，按三个层次来介绍：第一层是基本知识、基本运算，第二层介绍该数据结构的基本应用，第三层介绍该数据结构的深层应用。每种数据结构都给出了足够的例题，使学生加深对基本概念的理解，提高分析问题和解决问题的能力。

数据结构是实践性很强的课程，在“听”和“读”的基础上，必须大力加强对作业及上机实践环节的要求和管理。本书每章都给出了按上述三个层次精心挑选、难易搭配的题目，特别是在附录中给出了大量的实训例题和实训习题。通过习题与实训，使学生掌握所学知识，并能灵活运用所学知识解决实际问题。

书中的所有程序都运行通过。本书配有电子教案，使用本教材的教师可与编者联系。

本书由张红霞任主编，苏玉任副主编，朱乃立任主审。书中第2、3章由张红霞编写，第1、4、5章由苏玉编写，第6、7章由柏杏丽编写，第8、9章由王勤编写。张红霞统编全稿。

洛阳大学朱乃立教授在百忙中抽出宝贵时间对全书进行了审阅并提出宝贵意见，在此谨表衷心的感谢。

由于编者水平有限，书中疏漏和错误之处在所难免，殷切希望广大同行和读者不吝指正。

编者

2003年5月

目　录

第1章 绪 论

数据结构作为计算机领域里的一门学科形成于上世纪六七十年代。随着计算机产业的飞速发展和它的应用范围的迅速扩展，数据结构技术也得到了发展和完善。借助于新技术和新方法，数据结构技术更加易于使用，并逐渐巩固了它在计算机科学领域里独特的和不可替代的作用与地位。

本章将介绍数据结构的基本概念、几种典型的数据结构和常用的存储结构、算法的描述和算法分析。

1.1 引 言

数据结构是在整个计算机科学与技术领域里被广泛使用的术语。它用来反映一个数据的内部构成，即一个数据由哪些成分数据构成，以什么方式构成，呈什么结构。数据结构有逻辑上的数据结构和物理上的数据结构之分。逻辑上的数据结构反映成分数据之间的逻辑关系，而物理上的数据结构反映成分数据在计算机内部的存储安排。数据结构是数据存在的形式，是信息的一种组织方式，其目的是为了提高算法的效率。它通常与一组算法的集合相对应，通过这组算法集合可以对数据结构中的数据进行某种操作。下面通过例子来增加对数据结构的感性认识。

例如表 1-1 是一张学生成绩表，记录了一个班的学生各门课的成绩。这个表就是一个数据结构。

表 1-1 学生成绩表

学 号	姓 名	数 学	英 语	数据结构	平均成绩
20021001	张三	88	90	92	90
20021002	李四	86	84	85	85
20021003	王五	93	92	94	93
20021004	赵六	89	93	91	91
20021005	丁一	92	88	87	89
⋮	⋮	⋮	⋮	⋮	⋮

在表中，每一行记录了一个学生的学号、姓名和三门课的成绩及平均成绩，这每一行就称为该数据结构中的一个元素或一个结点。对于整个表来说，只有一个开始结点（它的前面无结点）和一个终端结点（它的后面无结点），其他的结点则各有一个也只有一个直接前驱和直接后继（它的前面和后面均有且只有一个结点）。这几个关系就确定了这个表的逻辑结构（为线性结构）。那么怎样把这个表中的数据存储到计算机里呢？用高级语言如何表示各结点之间的关系呢？是用一片连续的内存单元来存放这些结点（如用数组表示），还是随机存

放各结点数据再用指针进行链接呢？这就是存储结构的问题，我们将从高级语言的层次来讨论这个问题。最后，我们有了这个表(数据结构)，肯定要用它，就是要对这张表中的记录进行查询、修改、删除等操作，对这个表可以进行哪些操作以及如何实现这些操作就是数据的运算问题。

大多数计算机程序的主要目标与其说是完成运算，不如说是存储和检索数据。从存储空间和运行时间的实现角度看，这些程序必须组织数据，以支持高效的数据处理过程。因此，研究数据结构和算法以有效地支持程序的实现就成了计算机科学的核心问题。

1.2 基本概念和术语

● 数据(Data) 指能够被计算机识别、存储和加工处理的一切对象。

● 数据元素(Data Element) 数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。在某些情况下，数据元素也称为元素、结点、顶点、记录。一个数据元素有时可以由若干数据项组成。

● 数据项(Data Item) 数据项是具有独立含义的最小单位。如在表 1-1 中，一个学生的信息为一个数据元素，而这个元素由 6 个数据项组成，此时的数据元素通常称为记录。

● 数据结构(Data Structure) 指的是数据之间的相互关系，即数据的组织形式。一般包括三个方面的内容：数据的逻辑结构、存储结构和对数据的运算(或操作)。

如表 1-1 定义的学生成绩表描述的是数据元素(学生成绩)之间的逻辑关系，是从操作对象抽象出来的数学模型，称为数据的逻辑结构(Logical Structure)，在不引起混淆的情况下有时简称为数据结构。数据结构通常分为两大类：线性结构和非线性结构。

根据数据元素之间关系的不同特性，通常有下列四类基本数据结构：

(1) 集合 结构中的数据元素之间除了“同属于一个集合”的关系外，别无其他关系。

(2) 线性结构 结构中的数据元素之间存在一对一的线性关系。它的特征是若结构为非空集，则该结构有且只有一个开始结点和一个终端结点，并且所有结点都最多只有一个直接前驱(与该结点相邻且在其前面的结点)和一个直接后继(与该结点相邻且在其后面的结点)。

(3) 树形结构 结构中的元素之间存在一对多的层次关系。它的特征是若结构为非空集，则该结构有且只有一个结点没有双亲(前驱)结点，其余所有结点都只有一个双亲(前驱)结点、零个或多个孩子(后继)结点。

(4) 图状结构或网状结构 结构中的元素之间存在多对多的任意关系。它的特征是若结构为非空集，则该结构中的每个结点都可能有多个直接前驱和直接后继。

上述树形结构和图状结构是典型的非线性结构。

数据的存储结构(Storage Structure)是指数据的逻辑结构用计算机语言的实现，即数据元素及其关系在计算机存储器内的表示，也称为数据的物理结构。

数据的存储结构可以用以下四种基本的存储方法得到：顺序存储方法、链接存储方法、索引存储方法和散列存储方法。

(1) 顺序存储方法 它是把逻辑上相邻的结点存储在物理位置相邻的存储单元里，结点间的逻辑关系由存储单元的邻接关系来体现。由此得到的存储表示称为顺序存储结构。

(2) 链接存储方法 它不要求逻辑上相邻的结点在物理位置上也相邻,结点间的逻辑关系是由附加的指针字段表示的。由此得到的存储表示称为链式存储结构。

(3) 索引存储方法 除建立存储结点信息外,还建立附加的索引表来标识结点的地址。

(4) 散列存储方法 就是根据结点的关键字直接计算出该结点的存储地址。

上述四种基本的存储方法,既可以单独使用,也可以组合起来对数据结构进行存储映像。同一种逻辑结构采用不同的存储方法,可以得到不同的存储结构。选择何种存储结构来表示相应的逻辑结构,视具体要求而定,主要考虑运算方便及算法的时空要求。

数据的运算即是对数据施加的操作。比如上述学生成绩表,我们需要对学生成绩进行查找、增加、修改、删除等工作,怎样进行这些操作就是数据的运算。在数据结构中,这些运算常常涉及算法问题。

1.3 算法的描述和分析

1.3.1 算法的描述

算法(Algorithm)是一个能够解决问题的、有具体步骤的方法,是指令的有限序列,其中每一条指令表示一个或多个操作。算法具有下列5个性质:

(1) 输入 一个算法有0个或多个输入的外界量,它们是算法开始前对算法给出的最初量。

(2) 输出 一个算法至少产生一个输出,它们是同输入有某种关系的量。

(3) 有穷性 算法中的每一条指令的执行次数是有限的,且每条指令都可在有限时间内完成,即一个算法必须可以终止,不能进入死循环。

(4) 确定性 一个算法中每条指令的含义都必须明确,无二义性。

(5) 可行性 算法中的所有操作都必须足够基本,都可以通过已经实现的基本操作执行有限次来实现。

一个算法可以用自然语言、数学语言或约定的符号语言来描述。本书用类C语言来描述算法。

1.3.2 算法设计的要求

程序设计的实质是对确定的问题选择一种“好结构和好算法”。一个好的算法应符合以下算法设计要求:

(1) 正确性 一个好的算法应当满足具体问题的要求。正确性通常包含4层含义:不含语法错误,对几组输入数据运行正确,对精心选择的典型、苛刻而带有刁难性的几组输入数据运行正确,对一切合法的输入数据都能运行正确。

(2) 可读性 一个好的算法应易于人们阅读、理解、编码和调试。

(3) 健壮性 一个好的算法当输入数据非法时,算法也能做出适当反应或进行处理,而不会产生莫名其妙的输出结果。

(4) 高效率与低存储量需求 一个好的算法应能够有效利用计算机的资源,执行时间短,占用的存储空间小。

1.3.3 算法的分析

一个问题有多种解法，选择哪一种呢？计算机程序设计的核心有两个目标(有时它们互相冲突)：

(1) 设计一个容易理解、编码和调试的算法；

(2) 设计一个能有效利用计算机资源的算法。

在理想情况下，达到这两个目标的最终程序是正确的，有时也可以说是“完美的”。与目标(1)有关的问题主要涉及的是软件工程原理，而本书主要讲的是与目标(2)有关的问题。算法分析是对一种算法所消耗资源的估算，我们可以据此对解决同一问题的两种或两种以上算法的代价加以比较，算法设计者也可以据此判断一种算法在实现时是否会遇到资源限制的问题。通常我们需要分析一种算法(或者实现该算法的一个程序实例)所花费的时间以及所用数据结构所占用的空间。

1. 时间复杂度

算法执行时间需通过依据算法编制的程序在计算机上运行时所消耗的实际时间来度量。许多因素会影响程序的运行时间。一个算法用不同的语言实现，或者用不同的编译程序进行编译，或者在不同的计算机上运行时，效率均不同。这表明使用绝对的时间单位衡量算法的效率是不合适的。撇开这些与计算机硬件、软件有关的因素，可以认为一个特定算法“运行工作量”的大小，只依赖于问题的规模(通常用整数 n 表示)，或者说，它是问题规模 n 的函数。

为了便于比较同一问题的不同算法，通常的做法是：从算法中选出一种对于所研究的问题(或算法类型)来说是基本操作的原操作，以该基本操作重复执行的次数作为算法的时间度量。

【例 1-1】下面是查找大小为 n 的一维整数数组中最大元素的算法。该算法依次遍历数组中的元素，并保存当前的最大元素。

```
int largest(int array[ ], int n)
{ int currlarge = array[0];
    for ( i=0; i<n; i++ )
    if ( array[i] > currlarge )
    currlarge = array[i];
    retrun currlarge;
}
```

以上问题的规模为 n，这些整数存放在数组 array 中，基本操作是“检查”一个整数，即把一个存放现有最大整数的变量与它作比较。我们可以认为，检查数组中的某个整数所需要的时间是一定的，与该整数的大小或它在数组中的位置无关。若把检查一个元素所需要的时间记为 t，t 中包括变量 i 增值的时间和找到一个新的最大元素时要做的工作。在不考虑 t 的实际值和初始化时所需要的一小部分额外时间的情况下，可以得到该算法的一个合理的近似运行时间 tn(共需要进行 n 次检查，每执行一次检查需要时间 t)，记作 $T(n)=tn$。

一般情况下，由于算法转换为程序之后，每条语句执行一次所需要的时间取决于软硬件环境，这是很难确定的。我们假设执行每条语句所需要的时间均是单位时间 c，则算法的时间耗费就表示为：算法中所有语句重复执行的次数(也称为该语句的语句频度(Frequency

Count))之和，它是问题的规模 n 的函数 $f(n)$，记作 $T(n)=O(f(n))$。它表示随问题规模 n 的增大执行时间的增长率和 $f(n)$ 的增长率相同。我们把某个算法的时间耗费称做算法的时间复杂度，它是该算法所求解问题规模 n 的函数。而当问题规模趋向无穷大时，该算法时间复杂度的数量级表示称为该算法的渐进时间复杂度(Asymptotic Time Complexity)。当我们评价一个算法的时间性能时，主要标准就是算法的渐近时间复杂度，因此，在算法分析时，往往对两者不予区分，经常是将渐近时间复杂度 $T(n)=O(f(n))$ 简称为时间复杂度，其中的 $f(n)$ 一般是算法中频度最大的语句的语句频度。

此外，算法中语句的频度不仅与问题规模有关，还与输入实例中各元素的取值相关。但是我们总是考虑在最坏的情况下的时间复杂度，以保证算法的运行时间不会比它更长。常见的时间复杂度，按数量级递增排列依次为：常数阶 $O(1)$、对数阶 $O(\log_2 n)$、线性阶 $O(n)$、线性对数阶 $O(n\log_2 n)$、平方阶 $O(n^2)$、立方阶 $O(n^3)$、k 次方阶 $O(n^k)$、指数阶 $O(2^n)$。

【例 1-2】分析下面三个程序段的时间复杂度：

```
x=x+1;

for (i=0; i<n; i++) x=x+1;

for (i=0; i<n; i++)
    for (j=0; j<n; j++)
    x=x+1;
```

分析：上述三个程序段的问题规模均为 n，算法中基本操作“x=x+1”的频度(即基本操作的执行次数) $f(n)$ 分别为 1，n，n^2，故相应的时间复杂度 $T(n)$ 分别为 $O(1)$、$O(n)$ 和 $O(n^2)$，分别称为常数阶、线性阶和平方阶。

2. 空间复杂度

除了时间以外，空间代价也是程序员要经常考虑的计算机资源。近年来，在计算机运行速度提高的同时，其存储能力也大大增强了。虽然如此，可利用的磁盘或内存空间的大小仍是对算法设计者的重要限制。类似于时间复杂度，一个算法的空间复杂度(Space Complexity) $S(n)$ 定义为该算法所耗费的存储空间，记作 $S(n)=O(f(n))$。它也是问题规模(或大小) n 的函数。渐进空间复杂度也常常简称为空间复杂度。一个上机执行的程序除了需要存储空间来寄存本身所用指令、常数、变量以外，也需要一些对数据操作的工作单元和存储一些为实现计算所需辅助信息的辅助空间。程序本身所有指令所占空间对不同算法来说一般不会有数量级的差别，在比较算法时可以不加考虑。若输入数据所占空间只取决于问题本身，和算法无关，则在比较算法时也可以不加考虑，故对算法空间复杂度的分析通常只需要分析为实现计算所需存储辅助信息的辅助空间。

习 题 1

1. 简述下列概念：数据，数据元素，数据结构，逻辑结构，存储结构，线性结构，非线性结构。
2. 试举一个数据结构的例子，叙述其逻辑结构、存储结构、运算这三方面的内容。
3. 分析下列程序段，求其时间复杂度和空间复杂度。

```
(1) i=1; k=0;
    while (i<=n-1)
```

```
    {k=k*10*i;
     i++;
    }
(2) for (i=0; i<n; i++)
    for (j=0; j<i; j++)
    for (k=0; k<j; k++)
        x=x+1;
    i=1;
    do
    { j=1;
      do
      {printf ("%d\n", i*j);
       j++;
      } while (j>n);
    i++;
    }while ( i>n );
(3) x=n;     /* n>1 */
    y=0;
    while ((x>=(y+1)*(y+1))
        y=y+1;
```

4. 为下列问题设计算法，并分析所设计算法的时间和空间复杂度。

(1) 设计一个算法，判断一个数的奇偶性。

(2) 已知输入 x，y，z 三个不相等的整数，试设计一个算法，使这三个数按从小到大的顺序输出，并考虑此算法元素的比较次数和元素的移动次数。

第2章 线 性 表

线性表是最简单且最常用的一种线性数据结构。线性数据结构的特点是：在数据元素的非空有限集合中，存在一个惟一的没有前驱的(头)元素，存在一个惟一的没有后继的(尾)元素；集合中其他每个数据元素均有惟一的直接前驱和惟一的直接后继。本章将首先介绍线性表的定义，给出线性表的两种存储结构——顺序存储结构和链式存储结构，然后给出在相应存储结构上实现的线性表的运算。

2.1 线性表的定义和运算

2.1.1 线性表的逻辑结构

线性表(Linear List)是一种最简单且最常用的数据结构。简单地说，一个线性表是 n ($n\geqslant 0$)个类型相同的数据元素的有限序列($a_1,a_2,\cdots,a_n$)。其中数据元素的个数 n 定义为线性表的长度，当 $n=0$ 时称为空表，空表中没有任何元素。线性表中的数据元素的具体含义在不同情况下可以不同。例如英文字母表(A,B,…,Z)是一个线性表，表中的数据元素是单个字母。又如某公司 2002 年每月产值表(700,600,800,…,1000)(单位：万元)也是一个线性表，这里的数据元素是整数。在较为复杂的线性表中，数据元素可由若干数据项组成。如职员档案表中，每个职员档案是一个数据元素，它由职员号、姓名、性别、出生年月、所在单位、职务、职称等数据项组成。此时常把数据元素称为记录，含有大量记录的线性表又称文件。

综上所述可见，线性表中的数据元素虽然是各种各样的，但有以下 3 个共同点：

(1) 同一性　同一线性表中的数据元素具有相同的类型。

(2) 有穷性　线性表中数据元素的个数是有限的。

(3) 有序性　线性表中相邻数据元素之间存在着序偶关系$\langle a_i,a_i+1\rangle$，即对于非空的线性表($a_1,a_2,\cdots,a_{i-1},a_i,a_{i+1},\cdots,a_n$)，$a_{i-1}$领先于 a_i。称 a_{i-1}是 a_i 的直接前驱元素，而称 a_i 是 a_{i-1}的直接后继元素(i 称为 a_i 的序号)。除了第一个元素 a_1 外，每个元素 a_i 有且仅有一个被称为其直接前驱的结点 a_{i-1}；除了最后一个元素 a_n 外，每个元素 a_i 有且仅有一个被称为其直接后继的结点 a_{i+1}。由此也可知，线性表是一种线性结构。

2.1.2 线性表的基本运算

线性表是一种相当灵活且非常简便的数据结构，可以根据需要改变表的长度，即对线性表的数据元素不仅可以进行访问，还可进行插入和删除等。常见的线性表的基本运算(操作)有如下 6 种：

(1) InitList(L)　初始化：构造一个空的线性表 L。