

普通高等院校计算机类专业“十二五”规划教材

数据结构实验教程 (C语言版)

SHUJU JIEGOU
SHIYAN
JIAOCHENG



王青海 编 著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

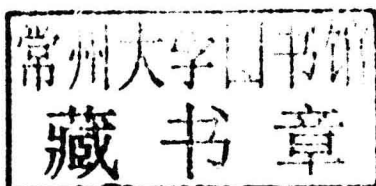


普通高等院校计算机类专业“十二五”规划教材

数据结构实验教程

(C 语言版)

王青海 编著



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

《数据结构实验教程(C语言版)》是为“数据结构”课程编写的辅助教材,是主教材《数据结构(C语言版)》的配套实验用书,是集作者“数据结构”课程及指导学生实验的教学经验编写而成。

本书内容由浅入深、循序渐进,注重培养学生的实践技能。书中自始至终使用NS图来描述主要算法,对程序结构进行了整体分析,并给出了实验的完整的实现细节,全部程序都在Visual C++ 6.0中调试通过。

本书可配合主教材使用,也具有相对的独立性,可作为高等院校计算机类专业及相关专业本科生的教材,也可作为专科生和成人教育学生的辅助教材,还可供从事计算机应用相关工作的科技人员参考。

图书在版编目(CIP)数据

数据结构实验教程/王青海主编. —北京:中国铁道出版社, 2012. 9
普通高等院校计算机类专业“十二五”规划教材
ISBN 978-7-113-15231-4

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

中国版本图书馆CIP数据核字(2012)第195027号

书 名: 数据结构实验教程(C语言版)
作 者: 王青海 编著

策 划: 张永生 读者热线: 400-668-0820
责任编辑: 张永生
编辑助理: 赵 迎
封面设计: 付 巍
封面制作: 刘 颖
责任印制: 李 佳

出版发行: 中国铁道出版社(100054,北京市西城区右安门西街8号)
网 址: <http://www.51eds.com>
印 刷: 三河市华业印装厂
版 次: 2012年9月第1版 2012年9月第1次印刷
开 本: 787mm×1092mm 1/16 印张: 10.5 字数: 253千
印 数: 1~2 000册
书 号: ISBN 978-7-113-15231-4
定 价: 23.00元

版权所有 侵权必究

凡购买铁道版图书,如有印制质量问题,请与本社教材图书营销部联系调换。电话:(010) 63550836

打击盗版举报电话:(010) 63549504

“数据结构”是计算机专业的一门核心课程，主要研究信息的逻辑结构及其基本操作在计算机中的表示与实现，是计算机科学的算法理论基础和软件开发的技术基础。在计算机专业中，数据结构是“操作系统”、“编译原理”及“数据库原理”等专业课程的基础，起着承上启下的作用，在计算机专业课程的学习中具有极其重要的地位。

数据结构的学习内容具有较强的抽象性，所涉及的具体结构的抽象数据类型、数据的组织、存储及操作对于初学者往往不容易入门，课程实验效果不理想，不利于学生对于逻辑结构与物理结构的理解和掌握。本书将数据结构课程所涉及的主要内容以最详细的实现方式来展示，从最基本的实验常规知识入手，逐步引入数据结构算法的相关知识，自然构建实现的框架，采用统一的实现界面来加以实现。本书将数据结构课程的基本内容分为 22 个实验，每个实验力求完整而细致，并对算法的原理、算法的思想、算法的流程及最终的实现细节等进行了详尽的描述，使得学生能够尽快掌握各种数据结构的基本实现和基本应用，为深入理解数据的组织（逻辑结构）和实现（物理结构）提供丰富的物质资料，为进一步学习算法理论打下坚实的基础。

本书是在青海师范大学精品课程——“数据结构”课程建设期间编写的。本书凝聚着课题组全体成员的心血，尤其是耿生玲教授、胡枫教授、刘丹青老师以及彭少扬同学对于本书的写作给予了很多帮助，在此，编者表示衷心的感谢。另外，在本书的编写过程中，参考了一些国内外优秀教材，对这些参考书目的作者表示诚挚谢意。

本书在编写过程中力求概念清晰，通俗易懂，便于自学。希望读者通过本书的学习，能够快速入门，从而能够更全面、更透彻地理解和掌握常用的数据结构知识。

由于编者水平有限，书中难免存在疏漏和错误，恳请读者批评指正。

编 者

2012 年 6 月

实验一 顺序表	1
实验二 单链表	12
实验三 顺序栈	23
实验四 链栈	33
实验五 栈的应用	43
实验六 链队列	47
实验七 循环队列	57
实验八 队列的应用	66
实验九 串	70
实验十 压缩矩阵及三元组实验	76
实验十一 二叉树基本操作实验	84
实验十二 中序线索二叉树实验	93
实验十三 二叉树应用实验	99
实验十四 图（网）的创建实验	104
实验十五 无向图广度优先遍历实验	112
实验十六 网的单源点最短路径实验	118
实验十七 查找实验（一）	123
实验十八 查找实验（二）	129
实验十九 查找实验（三）	134
实验二十 排序实验（一）	140
实验二十一 排序实验（二）	148
实验二十二 排序实验（三）	155
附录 实验报告模板	161
参考文献	162

实验一 顺序表

一、实验目的

(1) 掌握顺序表的相关概念、基本方法。

(2) 掌握顺序表的基本操作，包括顺序表初始化、顺序表创建、顺序表插入、顺序表删除、判断顺序表是否为空、顺序表按值查找元素、求顺序表长度、顺序表输出的编程实现。

二、实验学时

2 学时。

三、实验类型

验证型。

四、实验内容

顺序表初始化、顺序表创建、顺序表插入元素、顺序表删除元素、判断顺序表是否为空、顺序表查找元素、求顺序表长度、顺序表输出操作。

五、实验环境

(1) 硬件：学生每人一台计算机。

(2) 软件：Windows 操作系统，Visual C++ 6.0。

六、实验原理

1. 顺序表概述

线性表是 n 个类型相同的数据元素的有限序列，对于非空线性表 ($n > 0$)，除了第一元素无直接前驱、最后一个元素无直接后继外，其余每个元素只有唯一的直接前驱和直接后继，数据元素之间具有一对一的关系。线性表数据元素之间的关系在计算机中具有两种不同的表示方法：顺序映像（顺序存储结构）和非顺序映像（非顺序存储结构）。

顺序表是采用顺序存储结构存放的线性表。线性表的顺序存储是指用一组地址连续的存储单元依次存储线性表中的各个数据元素，使得线性表中在逻辑结构上相邻的数据元素存储在连续的物理存储单元中，即通过数据元素物理存储的连续性来反映数据元素之间逻辑上的相邻关系。采用顺序存储结构即指顺序表逻辑位序与物理位序完全相同，建表时通过一个循环赋值即可，且其表长为最后一个元素下标加 1；当在线性表顺序存储结构的第 i 个位置上插入一个元素时，必须先将线性表中第 i 个元素之后的所有元素依次后移一个位置，以便空出一个位置，再把新元素插入到该位置。若是想要删除第 i 个元素时，也必须把第 i 个元素之后的所有元素前移一个位置；判断顺序表是否为空只需要检验其表长是否大于 0；输出顺序表只需要计数器

i 在 0 到表长间循环输出其值。

2. 顺序表基本算法

(1) InitSeqList(L): 初始化顺序表操作, 其结果是将顺序表 L 初始化为空表。

(2) CreatSeqList(L): 创建顺序表操作, 其结果是创建一个长度为 n 的顺序表 L。

(3) InsertSeqList(L,i,e): 为插入元素操作, 其结果是在顺序表 L 中第 i 个位置前插入数据元素 e , 并使顺序表长度加 1。

(4) DeleteSeqList(L,i,e): 删除元素操作, 其结果是把顺序表 L 中第 i 个元素删除, 并用 e 返回其值, 并使顺序表长度减 1。

(5) EmptySeqList(L): 判断顺序表是否为空操作, 其结果是, 若顺序表 L 为空, 则返回 TRUE, 否则, 返回 FALSE。

(6) LocateSeqList(L): 按值查找。

(7) SeqListLength(L): 求顺序表长度。

(8) DisSeqList(L): 顺序表输出。

3. 顺序表程序模块图

图 1-1 是顺序表程序模块图。

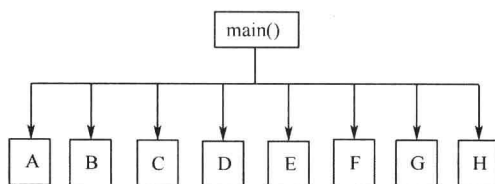


图 1-1 顺序表程序模块图

图中: A 代表 InitSeqList(L), B 代表 CreatSeqList(L), C 代表 InsertSeqList(L,i,e), D 代表 DeleteSeqList(L,i,e), E 代表 EmptySeqList(L), F 代表 SeqListLength(L), G 代表 LocateSeqList(L,e), H 代表 DisSeqList(L)。

4. 顺序表关键算法 NS 图

图 1-2、图 1-3 分别是顺序表的插入算法和删除算法的 NS 图。

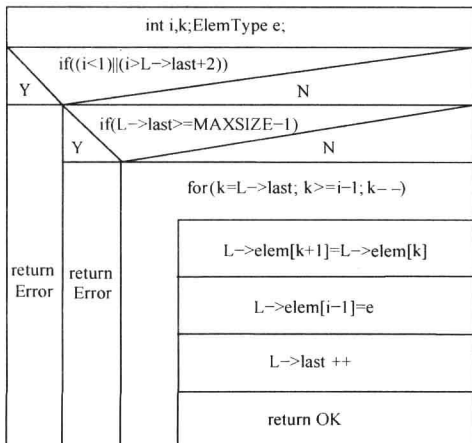


图 1-2 顺序表插入算法

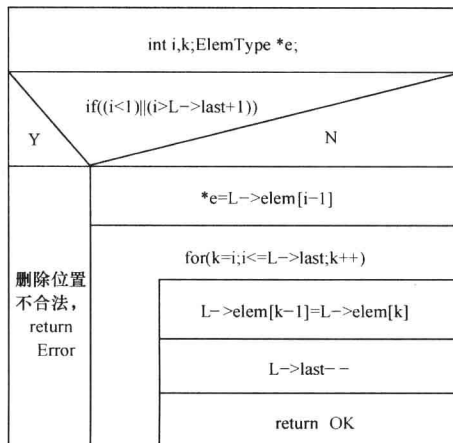


图 1-3 顺序表删除算法

七、测试数据

- (1) 顺序表创建。顺序表：23,41,6,1,890,64,32,91。
- (2) 求线性表长度。顺序表长度：8。
- (3) 插入元素。在第六个元素之前插入数据 333。
- (4) 判断顺序表是否为空。顺序表非空。
- (5) 删除元素。把第二个元素删除。
- (6) 查找元素。查找元素 890，位置是第四个。
- (7) 输出线性表。此时的顺序表是：

23 6 1 890 333 64 32 91

八、实验步骤及要求

- (1) 创建顺序表。
- (2) 输出此顺序表。
- (3) 求此时顺序表长度。
- (4) 指定一个元素，在此元素之前插入一个新元素。
- (5) 求此时顺序表长度。
- (6) 输出此线性表。
- (7) 判断此顺序表是否为空。
- (8) 指定一个元素，删除此元素。
- (9) 查询某个数据元素。
- (10) 求此时顺序表长度。
- (11) 输出此顺序表。
- (12) 求此时顺序表长度。
- (13) 程序完成。

九、程序运行

顺序表创建操作程序运行图如图 1-4 所示。

顺序表插入操作程序运行图如图 1-5 所示。

顺序表删除操作程序运行图如图 1-6 所示。

求顺序表长度操作程序运行图如图 1-7 所示。

顺序表是否为空操作程序运行图如图 1-8 所示。

顺序表查找操作程序运行图如图 1-9 所示。

顺序表输出操作程序运行图如图 1-10 所示。

```

顺序表基本操作实验
*****
* 请选择命令序号 (0-7): *
* 1 创建顺序表; *
* 2 顺序表插入; *
* 3 顺序表删除; *
* 4 求顺序表长度; *
* 5 判断顺序表是否为空; *
* 6 顺序表元素查找; *
* 7 顺序表输出; *
* 0 程序结束; *
*****
请输入命令号(0---7):
1
请输入顺序表的长度:
8
请输入顺序表的元素 (整型数):
23
41
6
1
890
64
32
91
*****
顺序表基本操作实验
*****
* 请选择命令序号 (0-7): *
* 1 创建顺序表; *
* 2 顺序表插入; *
* 3 顺序表删除; *
* 4 求顺序表长度; *
* 5 判断顺序表是否为空; *
* 6 顺序表元素查找; *
* 7 顺序表输出; *
* 0 程序结束; *
*****
请输入命令号(0---7):
7
此时的线性表是:
23 41 6 1 890 64 32 91

```

图 1-4 顺序表创建操作运行图

```

顺序表基本操作实验
*****
* 请选择命令序号 (0-7): *
* 1 创建顺序表; *
* 2 顺序表插入; *
* 3 顺序表删除; *
* 4 求顺序表长度; *
* 5 判断顺序表是否为空; *
* 6 顺序表元素查找; *
* 7 顺序表输出; *
* 0 程序结束; *
*****
请输入命令号(0---7):
2
请输入要插入的位置:
6
请输入要插入的值:
333
插入成功!
*****
顺序表基本操作实验
*****
* 请选择命令序号 (0-7): *
* 1 创建顺序表; *
* 2 顺序表插入; *
* 3 顺序表删除; *
* 4 求顺序表长度; *
* 5 判断顺序表是否为空; *
* 6 顺序表元素查找; *
* 7 顺序表输出; *
* 0 程序结束; *
*****
请输入命令号(0---7):
7
此时的线性表是:
23 41 6 1 890 333 64 32 91

```

图 1-5 顺序表插入操作程序运行图

```
===== 顺序表基本操作实验 =====
*请选操作之序号 (0-7):
*1  创建顺序表;
*2  顺序表插入;
*3  顺序表删除;
*4  求顺序表长度;
*5  判断顺序表是否为空;
*6  顺序表元素查找;
*7  顺序表输出;
*8  程序结束;
=====
请输入操作号(0-7):
3
请输入要删除位置:
2
删除成功!
===== 顺序表基本操作实验 =====
*请选操作之序号 (0-7):
*1  创建顺序表;
*2  顺序表插入;
*3  顺序表删除;
*4  求顺序表长度;
*5  判断顺序表是否为空;
*6  顺序表元素查找;
*7  顺序表输出;
*8  程序结束;
=====
请输入操作号(0-7):
7
同时的线性表是:
23  6  1  894  333  64  32  91
```

图 1-6 顺序表删除操作运行图

```
===== 顺序表基本操作实验 =====
*请选操作之序号 (0-7):
*1  创建顺序表;
*2  顺序表插入;
*3  顺序表删除;
*4  求顺序表长度;
*5  判断顺序表是否为空;
*6  顺序表元素查找;
*7  顺序表输出;
*8  程序结束;
=====
请输入操作号(0-7):
4
此时线性表长度为8
===== 顺序表基本操作实验 =====
*请选操作之序号 (0-7):
*1  创建顺序表;
*2  顺序表插入;
*3  顺序表删除;
*4  求顺序表长度;
*5  判断顺序表是否为空;
*6  顺序表元素查找;
*7  顺序表输出;
*8  程序结束;
=====
请输入操作号(0-7):
```

图 1-7 求顺序表长度运行图

```
===== 顺序表基本操作实验 =====
*请选操作之序号 (0-7):
*1  创建顺序表;
*2  顺序表插入;
*3  顺序表删除;
*4  求顺序表长度;
*5  判断顺序表是否为空;
*6  顺序表元素查找;
*7  顺序表输出;
*8  程序结束;
=====
请输入操作号(0-7):
5
此时顺序表为非空表!
===== 顺序表基本操作实验 =====
*请选操作之序号 (0-7):
*1  创建顺序表;
*2  顺序表插入;
*3  顺序表删除;
*4  求顺序表长度;
*5  判断顺序表是否为空;
*6  顺序表元素查找;
*7  顺序表输出;
*8  程序结束;
=====
请输入操作号(0-7):
```

图 1-8 顺序表是否为空操作程序运行图

```

顺序表基本操作实验
*****
* 请选择命令序号 (0-7):
* 1 创建顺序表;
* 2 顺序表插入;
* 3 顺序表删除;
* 4 求顺序表长度;
* 5 判断顺序表是否为空;
* 6 顺序表元素查找;
* 7 顺序表输出;
* 8 程序结束.
*****
请输入命令序号 (0-7):
6
请输入要查找的元素:
899
所查找到的元素位置是第4个
顺序表基本操作实验
*****
* 请选择命令序号 (0-7):
* 1 创建顺序表;
* 2 顺序表插入;
* 3 顺序表删除;
* 4 求顺序表长度;
* 5 判断顺序表是否为空;
* 6 顺序表元素查找;
* 7 顺序表输出;
* 8 程序结束.
*****
请输入命令序号 (0-7):

```

图 1-9 顺序表查找程序运行图

```

顺序表基本操作实验
*****
* 请选择命令序号 (0-7):
* 1 创建顺序表;
* 2 顺序表插入;
* 3 顺序表删除;
* 4 求顺序表长度;
* 5 判断顺序表是否为空;
* 6 顺序表元素查找;
* 7 顺序表输出;
* 8 程序结束.
*****
请输入命令序号 (0-7):
7
此时的线性表是:
23    6    1    899    333    64    32    91
顺序表基本操作实验
*****
* 请选择命令序号 (0-7):
* 1 创建顺序表;
* 2 顺序表插入;
* 3 顺序表删除;
* 4 求顺序表长度;
* 5 判断顺序表是否为空;
* 6 顺序表元素查找;
* 7 顺序表输出;
* 8 程序结束.
*****
请输入命令序号 (0-7):
8
Press any key to continue.

```

图 1-10 顺序表输出操作程序运行图

十、思考问题

结合实验过程, 回答问题:

何时采用顺序表处理线性结构问题为最佳选择?

十一、实验报告要求

- (1) 根据对顺序表的理解, 说明如何创建顺序表。
- (2) 写出实现顺序表插入和删除操作的程序设计思路。
- (3) 总结调试程序过程中遇到的问题及解决方案。
- (4) 写出本次实验的结论与体会。

十二、程序附录

```
#include "stdio.h"
#include "malloc.h"
#include "stdlib.h"
#define MAXSIZE 100
#define OK 1
#define Error 0
int yes=0;
typedef struct      /*顺序表结点结构*/
{
    int elem[MAXSIZE];
    int length;
}Seqlist;
void menu() /*系统界面设计与显示*/
{
    printf("\t\t\t 顺序表基本操作实验\t\t\t\n");
    printf("*****\n");
    printf("**请选择命令序号 (0-7):\t\t\t\t*\n");
    printf("**1 创建顺序表;\t\t\t\t*\n");
    printf("**2 顺序表插入;\t\t\t\t*\n");
    printf("**3 顺序表删除;\t\t\t\t*\n");
    printf("**4 求顺序表长度;\t\t\t\t*\n");
    printf("**5 判断顺序表是否为空;\t\t\t\t*\n");
    printf("**6 顺序表元素查找;\t\t\t\t*\n");
    printf("**7 顺序表输出;\t\t\t\t*\n");
    printf("**0 程序结束.\t\t\t\t*\n");
    printf("*****\n");
}
void Initlist(Seqlist *l)/*顺序表初始化*/
{
    l->length=0;
}
int Creatlist(Seqlist *l)/*创建顺序表*/
{
    int i,n;
    if(l->length>0)
    {
        printf("线性表已经存在,不需要再创建!\n");
        return Error;
    }
    printf("请输入顺序表的长度:\n");
    scanf("%d",&n);
    printf("请输入顺序表的元素(整型数):\n");
    for(i=0;i<n;i++)
        scanf("%d",&(l->elem[i]));
    l->length=n;
}
```

```
        yes=1;
        return OK;
    }
int Inslist(SeqList *l,int i,int e)/*顺序表插入元素操作*/
{
    int k;
    if(l->length>MAXSIZE)
        printf("空间不足,请补充空间!\n");
    else
    {
        for(k=l->length-1;k>=i-1;k--)
            l->elem[k+1]=l->elem[k];
        l->elem[i-1]=e;
        l->length++;
        printf("插入成功!\n");
    }
    return OK;
}
int Dellist(SeqList *l,int i)/*顺序表删除元素操作*/
{
    int k;
    for(k=i-1;k<l->length;k++)
        l->elem[k]=l->elem[k+1];
    l->length--;
    printf("删除成功!\n");
    return OK;
}
int Lengthlist(SeqList *l)/*求顺序表长度*/
{
    if(yes==0)
    {
        printf("此时表未创建! 因而没有长度!\n");
        return Error;
    }
    else
        printf("此时线性表长度是%d\n",l->length);
    return OK;
}
void Emptylist(SeqList *l)/*判断顺序表是否为空表*/
{
    if(yes==0)
        printf("此时表未创建! 不能判断是否为空!\n");
    else
    {
        if(l->length==0)
            printf("此时顺序表为空表!\n");
    }
}
```

```

        else
            printf("此时顺序表为非空表!\n",l->length);
    }
}

void Findlist(Seqlist *l,int e)/*按值查找元素,若找到,返回顺序表中序号;否则返回错误信息*/
{
    int i=0;
    while((l->elem[i]!=e)&&(i<l->length))
    {
        i++;
    }
    if(i==l->length)
        printf("顺序表中不存在这个元素!\n");
    else
        printf("所查找到的元素位置是第%d个!\n",i+1);
}

void Dislist(Seqlist *l)/*输出顺序表*/
{
    int i;
    if(l->length==0)
        printf("此时线性表为空!\n");
    else
    {
        printf("此时的线性表是:\n");
        for(i=0;i<l->length;i++)
            printf("%d\t",l->elem[i]);
        printf("\n");
    }
}

int main(void)                /*程序入口*/
{
    Seqlist L;
    int i,sel,e,n;
    Initlist(&L);
    while(1)
    {
        menu();
        printf("请输入命令号(0---7):\n");
        scanf("%d",&sel);
        switch(sel)
        {
            case 1:Creatlist(&L);break;
            case 2:if(yes==0)
                {
                    printf("此时表未创建!不能插入元素!\n");

```

```
        break;
    }
    else
    {
        printf("请输入要插入的位置:\n");
        scanf("%d", &i);
        printf("请输入要插入的值:\n");
        scanf("%d", &e);
        Inslist(&L, i, e);
        break;
    }
case 3: if(yes==0)
    {
        printf("此时表未创建!不能删除元素!\n");
        break;
    }
    else
    {
        printf("请输入要删除位置:\n");
        scanf("%d", &i);
        Dellist(&L, i);
        break;
    }
case 4: Lengthlist(&L); break;
case 5: Emptylist(&L); break;
case 6: if(yes==0)
    {
        printf("此时表未创建!不能查找元素!\n");
        break;
    }
    else
    {
        printf("请输入要查找的元素:\n");
        scanf("%d", &n);
        Findlist(&L, n);
        break;
    }
case 7: if(yes==0)
    {
        printf("此时表未创建!不能输出线性表!\n");
        break;
    }
    else
    {
        Dislist(&L);
        break;
    }
```

```
        }  
    case 0:exit(1);break;  
    default:  
        printf("输入命令错误!请重新输入:\n");  
        break;  
    }  
}  
return OK;  
}
```

实验二 单链表

一、实验目的

- (1) 掌握单链表相关概念。
- (2) 掌握单链表的基本操作, 包括单链表创建、单链表插入元素、单链表删除元素、单链表查找元素、判断单链表是否为空、单链表输出及单链表销毁编程实现。

二、实验学时

2 学时。

三、实验类型

验证型。

四、实验内容

单链表初始化、单链表创建、单链表插入、单链表删除、判断单链表是否为空、单链表输出、求单链表长度、单链表销毁操作。

五、实验环境

- (1) 硬件: 学生每人一台计算机。
- (2) 软件: Windows 操作系统, Visual C++ 6.0。

六、实验原理

1. 单链表概述

单链表是采用链式存储结构存放的线性表。与顺序表的顺序存放不同, 单链表是用一组任意的存储单元来存放线性表的结点, 这组存储单元可以是连续的, 也可以是不连续的, 甚至是零散分布在内存的任何位置上。因此, 链表中结点的逻辑次序和物理顺序不一定相同, 为了正确表示结点之间的逻辑关系, 必须在存储线性表的每个数据元素值的同时, 存储指示其后继元素的地址 (或位置) 信息, 因此, 每个结点包括数据域与指针域两个部分, 由于含有一个指针域, 故称为单链表。为了操作方便, 可在单链表的第一个结点之前附设一个头结点。当在线性表的链式存储结构上的第 i 个位置上插入一个元素时, 只需要先确定第 i 个元素前一个元素位置, 然后修改相应指针将新元素插入即可。若想要删除第 i 个元素, 也必须先确定第 i 个元素前一个元素位置, 然后修改相应指针将该元素删除。

2. 单链表基本算法

- (1) InitList(L): 初始化单链表操作, 其结果是将单链表 L 初始化为空表。
- (2) CreatList(L): 创建单链表操作, 其结果是创建一个长度为 n 的单链表 L。