第二章 答案

本章学习要点

- 熟练掌握线性表在顺序存储结构上实现基本操作的算法
- 熟练掌握在各种链表结构中实现线性表操作的基本方法，能在实际应用中选用适当的链表结构
- 能够从时间和空间复杂度的角度综合比较线性表两种存储结构的不同特点及其适用场合

编写链表相关复杂代码的小要点

- 理解指针和引用的含义
- 警惕指针丢失和内存泄漏
- 留意边界条件和特殊情况
- 举例画图，辅助思考
- 多写多练，没有捷径

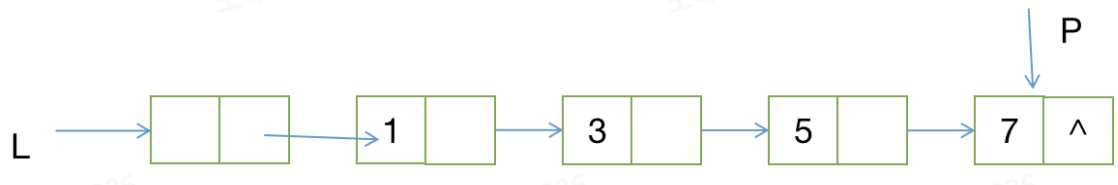
一. 基础题

1. 画出执行下列各行语句后最终的各指针及链表的示意图

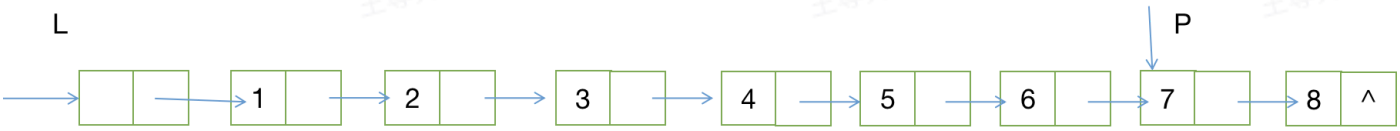
C

```c
L = (LinkList)malloc(sizeof(LNode));
p = L;
for(i=1; i<=4; i++){
    P->next = (LinkList)malloc(sizeof(LNode));
    P = P->next; P->data = i*2-1;
}
P->next = NULL;
for(i=4; i>=1; i--)
    Ins_LinkList(L,i+1, i*2);
for(i=1; i<=3; i++) Del_LinkList(L,i);
```

1-7行执行完



8-9行执行完



10行执行完

2. 已知P结点是某双向链表的中间结点，写出一下操作的语句序列
   a. 在P结点后插入S结点的语句序列；

      s->prior=p; s->next=p->next; p->next->prior=s; p->next=s

   b. 在P结点前插入S结点的语句序列；

      s->next=p; s->prior=p->prior; p->prior->next=s; p->prior=s

   c. 删除P结点的直接后继结点的语句序列；

      q=p->next; q->next->prior = p; p->next=q->next; free(q)

   d. 删除P结点的直接前驱结点的语句序列；

      q=p->prior; q->prior->next = p; p->prior=q->prior; free(q)

   e. 删除P结点的语句序列。

      p->next->prior=p->prior; p->prior->nxet=p->next; free(p)

3. 简述以下算法的功能（2.9）
   a. 第1小题

   L长度不小于2，将L第一个结点挪动到表尾。

C

```c
Status A(LinkedList &L){// L是无头结点的单链表
    if(L && L->next){
        Q = L; L = L->next; P = L;
        while(P->next) P = P->next;
        P->next = Q; Q->next = NULL;
    }
    return OK;
}
```

   b. 第2小题

   将1个单循环链表拆成两个单循环链表

C

```c
void BB(LNode *s, LNode *q){
    p = s;
    while(p->next !=q) p = p->next;
    p->next = s;
}//BB

void AA(LNode *pa, LNode *pb){
    //pa和pb分别指向不带头结点单循环链表中的两个结点
    BB(pa, pb);
    BB (pb, pa);
}//AA
```

4. 算法设计题，从一个给定的顺序表L中删除值在x~y （x<=y）之间的所有元素，要求以较高的效率来实现，写出算法伪代码并分析你的算法时间复杂度。（提示：移动位置一步到位）

C

```c
void delete(Sqlist &L, int x,int y){
    i=0;
    n=0;
    while(i<L.length){
        if(L.data[i]>=x && L.data[i]<=y){
            n++; //n用来记录当前发现了几个需要删除的数据
        }
        else
            L.data[i-n]=L.data[i] //留下的数据向前移动n个位置
        i++;
    }
    L.length-=n; //修改长度
}
//算法复杂度O(n)
```

5. 算法设计题，设计一个空间复杂度为0(1)的算法shift(SqList L, int k)将顺序表L中的元素整体循环左移k位，要求以较高的效率来实现，写出算法伪代码并分析你的算法时间复杂度。

示例数据：顺序表L初始数据为：1，2，3，4，5，6，整体循环左移2位后为：3，4，5，6，1，2

算法思路：

a. 将前k位原地倒置，如将 1，2 倒置为 2　1

b. 将后面的位原地倒置，如将 3，4，5，6倒置为 6，5，4，3

c. 将倒置的数据整体再倒置一次：即将 2 1 6 5 4 3倒置为 3 4 5 6 1 2

累计操作次数 3n，复杂度O(n)

6. 算法设计题：设计一个算法删除一个单链表倒数第k个结点

算法思路：要想删除倒数第k个结点，应先找到倒数第k+1个结点，定义两个指针，pFast和pSlow，都指向头结点，pFast先后移指向第k+1个结点，此时仍指向头结点的pSlow就是相对pFast的倒数第k+1个结点。然后同时后移，当pFast走到表尾时，pSlow正好指向倒数第k+1个结点。然后将pSlow指向的结点后面的结点删除。

7. 算法设计题，设计一个算法检测一个单链表L上是否因为某种错误操作而出现了环。

   快慢指针。慢指针每次只走一步，快指针每次走两步，如果有环那两个指针必定会重合。

二. 编程作业（提交项目源代码）

设计一个一元稀疏多项式简单计算器

基本要求：

a. 输入并建立多项式
b. 输出多项式，输出形式为整数序列：n, c1, e1, c2, e2,...cn, en,其中n为多项式的项数，ci和ei分别为第i项的系数和指数
c. 多项式a和b相加
d. 多项式a和b相减

选做内容：

e. 求多项式a的导函数a'