第二章 线性表

1、已知长度为 n 的线性表 A 采用顺序存储结构，请写一时间复杂度为 O(n)、空间复杂度为 O(1)的算法，该算法删除线性表中所有值为 item 的数据元素。

[参考答案 1-教材答案]

https://blog.csdn.net/l_jd_gululu/article/details/113487343

在顺序存储的线性表上删除元素，通常要涉及到一系列元素的移动（删第 i 个元素，第 i+1 至第 n 个元素要依次前移）。本题要求删除线性表中所有值为 item 的数据元素，并未要求元素间的相对位置不变。因此可以考虑设头尾两个指针（i=1，j=n），从两端向中间移动，凡遇到值 item 的数据元素时，直接将右端元素左移至值为 item 的数据元素位置。

[算法描述]

```
void  Delete (ElemType A[ ], int  n)
//A 是有 n 个元素的一维数组，本算法删除 A 中所有值为 item 的元素。
{i=1; j=n; //设置数组低、高端指针（下标）。
 while (i<j)
   {
     while (i<j && A[i]!=item) i++;  //若值不为 item，左移指针。
     if (i<j) while (i<j && A[j]==item) j--; //若右端元素为 item，指针左移
     if (i<j) A[i++]=A[j--];
}
```

[参考答案 2- https://www.cnblogs.com/wchenfeng/p/16136765.html]

核心思路：把不等于 e 的值往前移动

```
void delSeqList(SeqList *L, int e)
{
  int j = 0,
  int i = 0;
  for(; i<L->length; i++)
  {
    if(L->data[i] != e)
    {
      L->data[j] = L->data[i];
      j++;
    }
  }
  L->length = j;
}
```

2、设计一个算法，通过一趟遍历在单链表中确定值最大的结点。

[题目分析]

假定第一个结点中数据具有最大值，依次与下一个元素比较，若其小于下一个元素，则设其下一个元素为最大值，反复进行比较，直到遍历完该链表。

[算法描述]

```
ElemType Max (LinkList L ){
if(L->next==NULL) return NULL;
pmax=L->next; //假定第一个结点中数据具有最大值
p=L->next->next;
while(p != NULL ){//如果下一个结点存在
    if(p->data > pmax->data) pmax=p;//如果 p 的值大于 pmax 的值，则重新赋值
    p=p->next;//遍历链表
}
    return pmax->data;
```

3、设计一个算法，通过遍历一趟，将链表中所有结点的链接方向逆转，仍利用原表的存储空间。

[题目分析]

从首元结点开始，逐个地把链表 L 的当前结点 p 插入新的链表头部。

[算法描述]

```
void  inverse(LinkList &L)
{// 逆置带头结点的单链表 L
    p=L->next;  L->next=NULL;
    while ( p) {
        q=p->next;     // q 指向*p 的后继
        p->next=L->next;
        L->next=p;        // *p 插入在头结点之后
        p = q;
    }
}
```

4、设计一个算法，删除递增有序链表中值大于 mink 且小于 maxk 的所有元素（mink 和 maxk 是给定的两个参数，其值可以和表中的元素相同，也可以不同 ）。

[题目分析]

分别查找第一个值>mink 的结点和第一个值 ≥maxk 的结点，再修改指针，删除值大于 mink 且小于 maxk 的所有元素。

[算法描述]

```
void delete(LinkList &L, int mink, int maxk) {
   p=L->next; //首元结点
   while (p && p->data<=mink)
      { pre=p;  p=p->next; } //查找第一个值>mink 的结点
   if (p)
    {while (p && p->data<maxk)  p=p->next;
                    // 查找第一个值 ≥maxk 的结点
      q=pre->next;   pre->next=p;  // 修改指针
      while (q!=p)
         { s=q->next;  delete q;  q=s; } // 释放结点空间
   }//if
}
```

```cpp
//删除范围元素
int Delete(LinkList& L)
{
    cout << "请输入你要删除的范围(例: 5 7 (大于5小于7) ): ";
    int mink, maxk;
    cin >>mink>>maxk ;
    LinkList r = L;
    LinkList p = L->next;
    while (p)
    {
        if (p->data <= mink || p->data>=maxk)
        {
            p=p->next;
            r=r->next;
        }
        else
        {
            LinkList d = p;
            r->next = p->next;
            delete d;
            p = r->next;
        }
    }
    return 0;
}
```