**第2章 线性表**

**注：所有未特别说明的链表，均为带头结点**

1. 设线性表有n个元素，以下操作中，\_\_A\_\_\_在顺序表上实现比在链表上实现效率更高

A 输出第i个元素值（i在1-n之间）

B 交换第1个元素与第2个元素的值

C 顺序输出这n个元素的值

D 输出与给定值x相等的元素在线性表中的序号

1. 设线性表中有2n个元素，以下操作中，\_\_A\_\_\_在单链表上实现要比在顺序表上实现效率更高

A 删除指定的元素

B 在最后一个元素的后面插入一个新元素

C 顺序输出前k个元素

D 交换第i个元素和第2n-i-1个元素的值(i在0 - n-1间)

1. 如果最常用的操作是取第i个结点及其前驱，则采用\_\_D\_存储方式最节省时间

A 单链表

B 双链表

C 单循环链表

D 顺序表

1. 将两个各有n个元素的有序顺序表(某个表中的元素，两个表之间的元素，值均有可能相同)归并成一个有序顺序表，其最少比较次数是\_\_A\_\_\_

A n

B 2n-1

C 2n

D n-1

1. 一个长度为n(n>1)的带头结点单链表h上,另设有尾指针r(指向尾结点),执行\_\_B\_\_的操作与链表的长度有关

A 删除单链表中的第一个元素

B 删除单链表的最后一个元素

C 在单链表的第一个元素前插入一个新元素

D 在单链表的最后一个元素后插入一个新元素

1. 双向循环链表中,在p结点之前插入q结点的操作是\_D\_\_\_

A p->prior = q;

q->next = p;

p->prior->next = q;

q->prior = p->prior;

B p->prior = q;

p->prior->next = q;

q->next = p;

q->prior = p->prior;

C q->next = p;

q->prior = p->prior;

p->prior = q;

p->prior->next = q;

D q->next = p;

q->prior = p->prior;

p->prior->next = q;

p->prior = q;

1. 在一个单链表中删除p结点(假设p不是尾结点)时，应执行如下操作:

(1) q = p->next；

(2) p->data = p->next->data；

(3) p->next = \_q->next\_\_\_\_\_\_\_\_\_；

(4) free(q)；

1. 在一个单链表中的p结点之前插入一个s结点，可执行如下操作：

(1) s->next = \_\_\_\_p->next\_\_\_\_\_\_\_

(2) p->next = s;

(3) t = p->data;

(4) p->data = \_\_s->data\_\_\_\_\_\_\_\_\_

(5) s->data = \_\_\_\_t\_\_\_\_\_\_\_

1. 在一个双向循环链表中删除p结点时，应执行如下操作:

(1) \_\_\_\_\_p->next->prior\_\_\_\_\_\_\_\_\_ = p->prior；

(2) p->prior->next = \_\_\_\_\_p->next\_\_\_\_\_\_\_\_；

(3) free(p)；

1. 在单链表、双向链表和单循环链表中，若仅知道指针p指向某结点，不知道头指针，能否将p从相应的链表中删除(不允许进行结点之间数据域的复制)？若可以，时间复杂度各为多少？

单链表不能

双向链表可以O（1）

循环链表可以O（n）

1. 设计一个高效算法，将顺序表的所有元素逆置，要求算法的空间复杂度为O(1)

for i in range（1,len（list）/2 ）

swap（list[i],list[len(list)-i]

1. 设计一个高效算法，从顺序表中删除所有元素值为x的元素，要求空间复杂度为O(1)

ctr = 0

for i in range(0,len(list))

if list[i] ! = x :

list[i-ctr] = list[i];

else

ctr ++;

1. 用顺序表表示集合，设计一个求集合交集的算法

ctr = 0

for I in range(0,len(list1)):

if list1[i] in list2:

list[i-ctr] = list[i]

else

ctr ++

1. 从带头结点的循环单链表中删除值为x的第一个结点

p = head

while(p->next ! = x)

p = p->next

q = p->next

p->next = q->next

free(q)

q = NULL

1. 假定有一个带头结点的链接表，头指针为HL，每个结点含三个域:data，next和range，其中data为值域，next和range均为指针域，现在所有结点已经由next域链接起来，试编一算法，利用range域(此域的初始值均为NULL)把所有结点按照其值从小到大的顺序链接起来（next域不变）

a = b = HL->next->next;

for(p = (\*HL)->next;p;p = p->next)

if(p->data<a->data)

p->range = a

a = p

else if(p->data>b->data)

b->range = p

b = p;

else

for(q = (\*HL)->next;q! = p;q = q->next)

If(p->data>q->data && p->data<q->range->data)

break;

p->range = q->range;

q->range = p;

1. 已知带头结点的单链表L是一个递增有序表，设计一个高效算法，删除表中data值在[min .. max]之间的所有结点，并分析算法的时间复杂度

时间复杂度 O(n)

同理先找到最小值，然后统计个数直到超过最大值为止，把最大值之后的数值赋给 第k-ctr的位置即可

1. 有一个值按非递减有序排列的单链表，设计一个算法删除值域重复的结点，并分析算法的时间复杂度

遍历，每当访问一个节点，判断下一个节点是否相等，如果相等，删除下一个节点

时间复杂度O(n)

1. 用单链表表示集合，设计一个算法表示集合的交

便利a链表，对每一个元素遍历一遍b表，如果存在则保留，否则删除该节点..

1. 写出将带头结点的双向循环链表倒置的算法

p = \*L;

do{

q = p->next;

p->next = p->prior;

p->prior = q;

p = q;

}while(p! = \*L)

1. 设有一个双向链表h,设计一个算法查找第一个元素值为x的结点，将其与后继结点进行交换

p = \*h;

do {

if(p->next->data == x)

break;

p = p->next;

}while(p! = \*h)

if(p->next->data == x)

q = p->next

if(p = \*h)

p = p->prior

e = q->data

q->data = p->data

p->data = e

**【作业要求：】**

1、**5月15日前**网上提交本次作业（直接在本文件中作答，转换为PDF后提交即可）

2、每题所占平时成绩的具体分值见网页

3、超过截止时间提交作业会自动扣除相应的分数，具体见网页上的说明

4、**答案用蓝色标注(选择题将正确选项直接设置为蓝色文字即可)**