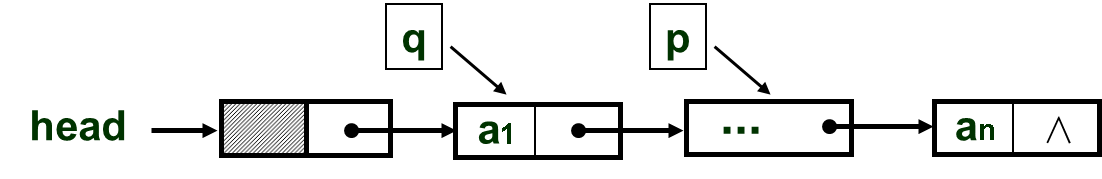
**第二章作业**

二、算法设计

3.设有一线性表E={e1,e2,…,en1,en)，试设计一个算法，将线性表逆置，即使元素排列次序颠倒过来，成为逆线性表E={en,en1,…,e2,e1)，要求逆线性表占用原线性表空间，并且用顺序和单链表两种方法表示，写出不同的处理过程。

顺序表逆置：

|  |
| --- |
| 1、函数接口描述： |
| **输入：E的地址；元素个数n；** |
| **输出：无** |
| **2、数据结构描述：** |
| **typedef struct** |
| **{ datatype data[maxsize]; //线性表空间** |
| **int last; // 指向最后结点的位置** |
| **} SequenList;** |
| **int n；//元素个数** |
| **3、算法描述：** |
| **(1)顶部伪代码描述：** |
| **以n/2为对称中心，将E中镜像元素交换位置** |
| **(2)细化描述：** |
| **int i=0;** |
| **当 (i<n/2)** |
| **将E中 i 与 n-1- i 单元内容互换;** |
| **i++;** |
| **4、测试：** |
| **n为奇数、偶数的情形** |

链表逆置：

|  |
| --- |
| **1、函数接口描述：** |
| **输入：E的地址head；** |
| **输出：无** |
| **2、数据结构描述：** |
| **typedef struct node** |
| **{** |
| **datatype data;** |
| **struct node \*next;** |
| **} LinkList; //链表结点描述** |
| **3、算法描述：** |
| **(1)顶部伪代码描述：** |
| **从链表头开始逐个摘下原链结点p，再用p结点，以头插法，建立新链表。** |
| **(2)细化描述：** |
| **p=head->next; //跳过头结点** |
| **当 ( p ≠ NULL ) // 当p指向结点非空** |
| **q=p; // 用q记录当前结点** |
| **p指向下一结点;** |
| **摘下q并插入head后** |
| **4、测试：** |
| **head=NULL**  **head->next=NULL** |

4、已知带头结点的动态单链表L中的结点是按整数值递增排列的，试写一算法将值为X结点插入表L，使L仍然有序。

|  |  |
| --- | --- |
| **1、函数接口描述：** | |
| **输入：递增整数列以及X** | |
| **输出：排序后L序列** | |
| **2、数据结构描述：** | |
| **typedef struct list{** | |
| **int data；** | |
| **struct list \*next；** | |
| **}Linklist；** | |
| **3、算法描述：** |
| **(1)顶部伪代码描述：** |
| **单链表递增，从表头开始对比结点值，若大于则指向下一个结点。否则插入X后建立新链表。** | |
| **(2)算法实现** | |
| **p=head->next;**  **q=head;**  **while(p!=NULL&&p->data<X){**  **q=p;**  **p=p->next;//指向下一节点**  **}**  **p=(ListNode\*)malloc(sizeof(ListNode));**  **p->data=X;**  **p->next=q->next;**  **q->next=p;**  **}** | |

**思考题P97**

1.数据在计算机存储器内表示时，物理地址连续，数据间的逻辑关系依靠其物理地址间的连续性来表达，称之为( )

(A)存储结构 (B)逻辑结构 (C)顺序存储结构 (D)链式存储结构

2.在n个结点的顺序表中，算法的时间复杂度是O（1）的操作是：

(A)访问第i个结点（1≤i≤n）和求第i个结点的直接前驱（2≤i≤n）

(B)在第i个结点后插入一个新结点（1≤i≤n）

(C)删除第i个结点（1≤i≤n）

(D)将n个结点从小到大排序

3.向一个有127个元素的顺序表中插入一个新元素并保持原来顺序不变，平均要移动几个个元素( )

(A)8 (B)63.5 (C)63 (D)7

4.链接存储的存储结构所占存储空间( )

(A)分两部分，一部分存放结点值，另一部分存放表示结点间关系的指针

(B)只有一部分，存放结点值

(C)只有一部分，存储表示结点间关系的指针

(D)分两部分，一部分存放结点值，另一部分存放结点所占单元数

5.链表是一种采用哪种存储结构存储的线性表( )

(A)顺序 (B)链式 (C)星式 (D)网状

6.线性表若采用链式存储结构时，要求内存中可用存储单元的地址( )

(A)必须是连续的 (B)部分地址必须是连续的

(C)一定是不连续的 (D)连续或不连续都可以

7.线性表Ｌ在哪种情况下适用于使用链式结构实现( )

(A)需经常修改Ｌ中的结点值 (B)需不断对Ｌ进行删除插入

(C)Ｌ中含有大量的结点 (D)Ｌ中结点结构复杂

8.单链表的存储密度( )

(A)大于1； (B)等于1； (C)小于1； (D)不能确定

9.下述哪一条是顺序存储结构的优点？（ ）

(A)存储密度大 (B)插入运算方便

(C)删除运算方便 (D)可方便地用于各种逻辑结构的存储表示

10.下面关于线性表的叙述中，错误的是哪一个？

(A)线性表采用顺序存储，必须占用一片连续的存储单元。

(B)线性表采用顺序存储，便于进行插入和删除操作。

(C)线性表采用链接存储，不必占用一片连续的存储单元。

(D)线性表采用链接存储，便于插入和删除操作。

11.若某线性表最常用的操作是存取任一指定序号的元素和在最后进行插入和删除运算，则利用（ ）存储方式最节省时间。

(A)顺序表 (B)双链表 (C)带头结点的双循环链表 (D)单循环链表

12．某线性表中最常用的操作是在最后一个元素之后插入一个元素和删除第一个元素，则采用（ ）存储方式最节省运算时间。

(A)单链表 (B)仅有头指针的单循环链表 (C)双链表 (D)仅有尾指针的单循环链表

13. 链表不具有的特点是（ ）

(A)插入、删除不需要移动元素 (B)可随机访问任一元素

(C)不必事先估计存储空间 (D)所需空间与线性长度成正比

14. 下面的叙述不正确的是（ ）

(A) 线性表在链式存储时，查找第i个元素的时间同i的值成正比

(B) 线性表在链式存储时，查找第i个元素的时间同此元素的值无关

(C) 线性表在顺序存储时，查找第i个元素的时间同i 的值成正比

(D) 线性表在顺序存储时，查找第i个元素的时间同i的值无关

15．在一个以 h 为头的单循环链中，p 指针指向链尾的条件是（）

(A) P->next=h (B) p->next=NIL

(C) p->next->next=h (D) p->data=-1

16．完成在双循环链表结点p之后插入s的操作是（ ）；

(A)p->next=s ; s->priou=p; p->next->priou=s ; s->next=p->next;

(B)p->next->priou=s; p->next=s; s->priou=p; s->next=p->next;

(C)s->priou=p; s->next=p->next; p->next=s; p->next->priou=s ;

(D)s->priou=p; s->next=p->next; p->next->priou=s ; p->next=s;

17．在单链表指针为p的结点之后插入指针为s的结点，正确的操作是：（ ）。

(A)p->next=s;s->next=p->next;

(B) s->next=p->next;p->next=s;

(C)p->next=s;p->next=s.next;

(D) p->next=s->next;p->next=s;

18．对于一个头指针为head的带头结点的单链表，判定该表为空表的条件是（ ）

(A)head==NULL (B)head→next==NULL

(C)head→next==head (D)head!=NULL

**上机题目**

1、利用凯撒密码本，将密码：

“jkhammotm oy znk vxuikyy ul xksubotm yulzcgxk hamy”进行译码。

**代码实现：**

#include<stdio.h>

#include<string.h>

#define N 1000

void main()

{

int i,j; /\* 将凯撒密码本以两行27列的数组存放\*/

char code[2][27]={"beafkilcjgdmqzhyoptxvrnwus","abcdefghijklmnopqretuvwxyz"},trans[N]={1000};

printf("please enter the code\n");

gets(trans);

for(i=0;i<strlen(trans);i++)/\*将输入得到的新数组trans中的元素与凯撒的密码比较\*/

{

for(j=0;j<27;j++)

{

if(trans[i]==code[0][j])

printf("%c",code[1][j]);

if(trans[i]==' ')

{

printf(" ");

break;

}

}

}

printf("\n");

}

2、Josephus环

**代码实现：**

#include<stdio.h>

#include<malloc.h>

//数据结构的定义

typedef struct list

{

int a;

list \*next;

}LinkNode;

LinkNode \*init\_LKlist(int n); //循环链表的建立

void Delete\_i\_Output(LinkNode \*head,int i,int n); //删除及打印

void main()

{

int n,k;

LinkNode \*head;

printf("请输入环内元素总数n及计数k：\n");

printf("n=");

scanf("%d",&n);

printf("k=",&k);

scanf("%d",&k);

head=init\_LKlist(n); //建立链表

printf("输出为：");

Delete\_i\_Output(head,k,n); //正向每隔k个次输出节点数据及释放该节点

printf("\n");

}

/\*===========================================

函数功能：单向循环链表的建立及初始化

函数输入：节点个数

函数输出：第一个带有数据的节点的地址

============================================\*/

LinkNode \*init\_LKlist(int n)

{

int i;

LinkNode \*head,\*p,\*q;

head=(LinkNode \*)malloc(sizeof(LinkNode));//创建头节点

q=head;

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

{

p=(LinkNode \*)malloc(sizeof(LinkNode));

p->a=i+1; //从1到n依次赋值

q->next=p;

q=p; //工作指针后移

}

p->next=head->next; //跳过初始化时生成的头结点，让尾指针与第一个节点相连

return p->next;

}

/\*=============================================

函数功能：每前移两次输出该节点的数据项，并释

放该节点，直到所有元素输出

函数输入：起始节点地址，计数跨度k，节点个数n

函数输出：无

==============================================\*/

void Delete\_i\_Output(LinkNode \*head,int i,int n)

{

int j,x,k;

LinkNode \*p,\*q;

for(k=0;k<n;k++)

{

p=head;

for(j=2;j<i;j++)

p=p->next; //指针前移

x=p->next->a;

printf("%d ",x); //输出目标节点数据

q=p->next;

p->next=q->next; //删除该节点

free(q); //释放

p=p->next;

head=p; //重定义计数起始位

}

}