链表代码题

1. 在带头结点的单链表工中，删除所有值为x的结点，并释放其空间，假设值为x的结点不唯一，试编写算法以实现上述操作。

思路

遍历，删除

代码

```
void Del_x_l(LinkList &L,ElemType x)
{
	LNode *p = L->next;=,*pre=L,*q;
	while(p!=NULL)
	{
		if(p->data==x)
		{
			q=p;
			p=p->next;
			pre->next=p;
			free(q);
		}
		else{
			pre=p;
			p=p->next;
		}
	}
}
```

1. 试编写在带头结点的单链表L中删除一个最小值结点的高效算法（假设该结点唯一）。

思路

遍历，记录，删除

代码

```
LinkList Delete_Min(LinkList &L)
{
	LNode *pre=L,*p=pre->next;
```

```
        LNode *minpre=pre,*minp=p;
        while(p!=NULL)
        {
                if(p->data<minp->data)
                {
                        minp=p;
                        minpre=pre;
                }
                pre=p;
                p=p->next;
        }
        minpre->next=minpre->next;
        free(minpre);
        return L;
}
```

1. 试编写算法将带头结点的单链表就地逆置，所谓“就地”是指辅助空间复杂度为 $O(1)$

思路

头节点摘下，依次使用头插法

代码

```
LinkList Reverse_1(LinkList L)
{
        LNode *p,*r;
        p=L->next;
        L->next=NULL;
        while(p!=NULL)
        {
                r=p->next;
                p->next=L->next;
                L->next=p;
                p=r;
        }
        return L;
}
```

1. 设在一个带表头结点的单链表中，所有结点的元素值无序，试编写一个函数，删除表中所有介于给定的两个值（作为函数参数给出）之间的元素（若存在），

思路

遍历，删除

代码

```
void Del_from_to(LinkList &L,int s,int t)
{
	LNode *pre = L,*p=L->next,*q;
	while(p!=NULL)
	{
		if(p->data>s&&p->data<t)
		{
			q=p;
			p=p->next;
			pre->next=p;
			free(q);
		}
		else
		{
			pre=p;
			p=p->next;
		}
	}
}
```

1. 给定两个单链表，试分析找出两个链表的公共结点的思想（不用写代码）。

思路

有公共节点，尾节点一定重合，直接遍历尾节点是否一样

1. 设C={a1,b1,a2,b2,…,anbn}为线性表，采用带头结点的单链表存放，设计一个就地算法，将其拆分为两个线性表，使得A={a1,a2,…,an},B={bn…,b2,b1}。

思路

算法思想：循环遍历链表C,采用尾插法将一个结点插入表A,这个结点为奇数号结点，这样建立的表A与原来的结点顺序相同：采用头插法将下一结点插入表B,这个结点为偶数号结点，
这样建立的表B与原来的结点顺序正好相反。

代码

```
LinkList DisCreat_2(LinkList &A)
{
	LinkList B = (LinkList)malloc(sizeof(LNode));
	B->next =NULL;
	LNode *p=A->next,*q;
	LNode *ra = A;
	while(p!=NULL)
	{
		ra->next=p;
		ra=p;
		p=p->next;
		if(p!=NULL)
		{
			q=p->next;
			p->next =B->next;
			B->next=p;
			p=q;
		}
	}
	ra->next=NULL;
	return B;
}
```

1. 在一个递增有序的单链表中，存在重复的元素。设计算法删除重复的元素，例如(7,10，10,21.30,42.42.42.51.70)将变为(7、10,21.30.42.51.70).

思路

遍历，如果想等保留，不相等删除

代码

```
void Del_same(LinkList &L)
{
	LNode *p=L->next,*q;
	if(P==NULL)
		return;
	while(p->next=NULL){
		q=p->next;
		if(p->data==q->data)
		{
			p->next=q->next;
			free(q);
		}
```

```
            else
                p=p->next;
            }
}
```

1. 设A和B是两个单链表（带头结点），其中元素递增有序。设计一个算法从A和B中的公共元素产生单链表C,要求不破坏A、B的结点，

思路

```
void Get-Common(LinkList A,LinkList B)
{
    LNode *p=A->next,*q=b->next,*r,*s;
    LinkList C =(LinkList)malloc(sizeof(LNode));
    r=C;
    while(p!=NULL&&q!=NULL)
    {
        if(p->data>q->data){
                q=q->data;
        }
        else if(p->data<q->data)
            p=p->next;
        else
        {
            s = (LNode *)malloc(sizeof(LNode));
            s->data=p->data;
            r->next=s;
            r=s;
            p=p->next;
            q=q->next;
        }
    }
    r->next=NULL;
}
```

1. 已知两个链表A和B分别表示两个集合，其元素递增排列。编制函数，求A与B的交集，并存放于A链表中。

思路

归并

代码

```
LinkList Union(LinkList *la,LinkList *lb)
{
	LNode *pa=la->next;
	LNode *pb=lb->next;
	LNode *u ,*pc=la;
	while(pa&&pb)
	{
		if(pa->data==pb->data)
		{
			pc->next=pa;
			pc=pa;
			pa=pa->next;
			u=pb;
			pb=pb->next;
			free(u);
		}
		if(pa->data<pb->data)
		{
			u=pa;
			pa=pa->next;
			free(u);
		}
		if(pa->data>pb->data)
		{
			u=pb;
			pb=pb->next;
			free(u);
		}
	}
	while(pa)
	{
		u=pa;
		pa=pa->next;
		free(u);
	}
	while(pb)
	{
		u=pb;
		pb=pb->next;
		free(u);
	}
	pc->next=NULL;
	free(lb);
	return la;
}
```

1. 两个整数序列A=a1,a2,a3,…,am和B=b1,b2,b,…,bn已经存入两个单链表中，设计一个算法，判断序列B是否是序列A的连续子序列。

思路

遍历A，用b的第一个值比较再依次比较

代码

```
int pattern(LinkList A,LinkList)
{
	LNode *p=A;
	Lnode *q=B;
	LNode *pre = p;
	while(p&q)
	{
		if(p->data=q->data){
			p=p->next;
			q=q->next;
		}
		else
		{
			pre=pre->next;
			p=pre;
			q=B;
		}
		if(q=NULL){
			return 1;
		}
		else
			return 0;
	}
}
```

1. 设计一个算法用于判断带头结点的循环双链表是否对称

思路

双向遍历

代码

```
int Symmetry(DLinkList L)
{
	DNode *p=L->next,*q=L->prior;
	while(p!=q&&p->next!=q){
		if(p->data=q->data)
		{
			p=p->next;
			q=q->prior;
		}
		else
			return 0;
	}
	return 1;
}
```

1. 有两个循环单链表，链表头指针分别为h1和h2,编写一个函数将链表h2链接到链表h1之后，要求链接后的链表仍保持循环链表形式。

思路

找为指针

代码

```
LinkList Link(LinkList &h1,LinkList &h2)
{
	LNode *p,*q;
	p=h1;
	while(p->next!=h1)
	{
		p=p->next;
	}
	q=h2;
	while(q->next!=h2)
	{
		q=q->next;
	}
	p->next=h2;
	q->next=h1;
	return h1;
}
```

1. 设有一个带头结点的非循环双链表L,其每个结点中除有pre、data和next城外，还有一个访问频度城freq,其值均初始化为零.每当在链表中进行一次Locate(L,x)运算

时，令值为x的结点中freq域的值增1，并使此链表中的结点保持按访问频度递减的顺序排列，且最近访问的结点排在频度相同的结点之前，以便使频繁访问的结点总是靠近表头。试编写符合上述要求的工ocate(L,)函数，返回找到结，点的地址，类型为指针型

思路

算法思想：首先在双向链表中查找数据值为x的结点，查到后，将结点从链表上摘下，然后
顺着结点的前驱链查找该结点的插入位置（频度递减，且排在同频度的第一个，即向前找到第一
个比它的频度大的结点，插入位置为该结点之后），并插入到该位置。

代码

```
DLinkList Locate(DLinkist &L,ElemType x)
{
	DNode *p =L->next,*q;
	while(p&&p->data!=x)
	{
		p=p->next;
	}
	if(!p)
		exit(0);
	else
	{
		p->freq++;
		if(p->prior==L||p->prioe->freq>p->freq)
			return p;
		if(p->next!=NULL)
			p->next->pre=p->pre;
		p->pre->next=p->next;
		q=p->pre;
		while(q!=L&&q->q->freq<=p->freq)
			q=q->pre;
		p->next=q->next;
		if(q->next!=NULL)
			q->next->pre=p;
		p->pre=q;
		q->next=p;
	}
	return p;
}
```

1. 没将n(n>1)个整数存数到不带头结点的单链表L中，没计算法将L中保存的序列循环右移k(0<k<n)个位置。例如，若k=1,则将链表{0,1,2,3}变为{3,0,1,2}。要求：1)给出算法的基本设计思想。2)根据设计思想，采用C或C++语言描述算法，关键之处给出注释。3)说明你所设计算法的时间复杂度和空间复杂度。

思路

首先，遍历链表计算表长，并找到链表的尾结点，将其与首结点相连，得到一个循环单链
表。然后，找到新链表的尾结点，它为原链表的第一k个结点，令工指向新链表尾结点的下一
个结点，并将环断开，得到新链表。

代码

```
LNode *Converse(LNode *L,int k)
{
	int n=1;
	LNode *p=L;
	while(p->next!=NULL)
	{
		p=p->next;
		n++;
	}
	p->next=L;
	while(int i=1;i<=n-k;i++)
	{
		p=p->next;
	}
	L=p->nextl
	p->next =NULL;
	return L;
}
```

时间复杂度

时间复杂度 $O(n)$,空间复杂度 $O(1)$

1. 单链表有环，是指单链表的最后一个结点的指针指向了链表中的某个结点（通常单链表的最后一个结点的指针域是空的)。试编写算法判断单链表是否存在环。1)给出算法的基本设计思想。2)根据设计思想，采用C或C++语言描述算法，关键之处给出注释。3)说明你所设计算法的时间复杂度和空间复杂度。

思路

快慢指针

代码

```
LNode* FindLoopStart(LNode *head){
        LNode *fast =head,*slow = fast;
        while(fast!=NULL&&fast->next!=NULL){
                slow=slow->next;
                fast=fast->next->next;
                if(slow==fast)
                        break;
        }
        if(fast==NULL&&fast->next==NULL)
                return NULL;
        LNode *p1=head,*p2=slow;
        while(p1!=p2){
                p1=p1->next;
                p2=p2->next;
        }
        return p1;
}
```

时间复杂度

时间复杂度 $O(n)$，空间复杂度 $O(1)$

1. 设有一个长度n(n为偶数)的不带头结点的单链表，且结点值都大于0，设计算法求这个单链表的最大孪生和。孪生和定义为一个结点值与其孪生结点值之和，对于第i个结点（从0开始），其孪生结点为第n-i-1个结点。要求：1)给出算法的基本设计思想。2)根据设计思想，采用C或C+语言描述算法，关键之处给出注释。3)说明你的算法的时间复杂度和空间复杂度。

思路

1)算法的基本设计思想：
设置快、慢两个指针分别为fast和slow,初始时s1ow指向工（第一个结点），fast指向L->next(第二个结点)，之后slow每次走一步，fast每次走两步。当fast指向表尾（第n个结点)时，s1ow正好指向链表的中间点（第n/2个结点），即s1ow正好指向链表前半部分的最后一个结点。将链表的后半部分逆置，然后设置两个指针分别指向链表前半部分和后半部分的首结点，在遍历过程中计算两个指针所指结点的元素之和，并维护最大值。

代码

```
int PairSum(LinkList L)
{
	LNode *fast =L->next,*slow=L;
	while(fast!=NULL&&fast->next!=NULL)
	{
		fast=fast->next->next;
		slow=slow->next;
	}
	LNode *newHead=NULL,*p=slow->next,*tmp;
	while(p!=NULL)
	{
		tmp=p->next;
		p->next=newHead;
		newHead=p;
		p=tmp;
	}
	int mx=0;
	p=L;
	LNode *q=newHead;
	while(p!=NULL)
	{
		if((p->data+q->data)>mx)
			mx=p->data+q->data;
		p=p->next;
		q=q->next;
	}
	return mx;
}
```

时间复杂度

时间复杂度 $O(n)$，空间复杂度 $O(1)$

1. 【2009统考真题】已知一个带有表头结点的单链表，结点结构为

data	link

假设该链表只给出了头指针List。在不改变链表的前提下，请设计一个尽可能高效的算法，查找链表中倒数第k个位置上的结点(k为正整数)，.若查找成功，算法输出该结点的data域的值，并返回1：否则，只返回0。要求：
1)描述算法的基本设计思想。
2)描述算法的详细实现步骤

3)根据设计思想和实现步骤，采用程序设计语言描述算法（使用C、C+或Java语言实现），关健之处请给出简要注释。

思路

双指针步频差k

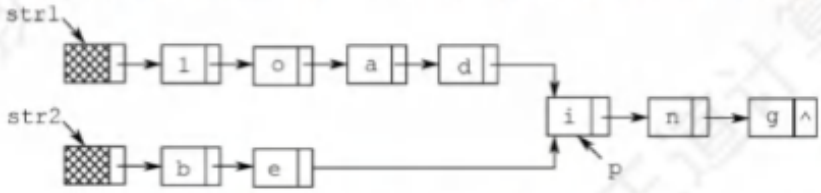
代码

```
typdef int ElemTypre;
typedef struct LNode
{
	ElemType data;
	struct LNode *link;
}LNode *LinkList;
int Search_k(LinkList list,int k)
{
	LNode *p =list->link,*q=list->link;
	int count=0;
	while(p!=NULL)
	{
		if(count<k)
			count++;
			else
				q=q->link;
			p=p->link;
	}
	if(count<k)
		return 0;
	else {
		printf("%d",q->data);
		return 1;
	}
}
```

时间复杂度

时间复杂度 $O(n)$，空间复杂度 $O(1)$

【2012 统考真题】假定采用带头结点的单链表保存单词，当两个单词有相同的后缀时，可共享相同的后缀存储空间，例如，loading 和 being 的存储映像如下图所示。



设 str1 和 str2 分别指向两个单词所在单链表的头结点，链表结点结构为 data next，请设计一个时间上尽可能高效的算法，找出由 str1 和 str2 所指向两个链表共同后缀的起始位置（如图中字符 i 所在结点的位置 p）。要求：

1）给出算法的基本设计思想。

2）根据设计思想，采用 C 或 C++或 Java 语言描述算法，关键之处给出注释。

3）说明你所设计算法的时间复杂度。

思路

先遍历长链表比短链表长的部分，再同步遍历

代码

```
typedef struct Node{
        char data;
        struct Node *next;
}SNode;
int listlen(SNode *head){
        int len =0;
        while(jead->next!=NULL)
        {
                len++;
                head=head->next;
        }
        return len;
}
SNode* find_list(SNode *str1,SNode *str2)
{
        int m,n;
        SNode *p,*q;
        m=listlen(str1);
        n=listlen(str2);
        for(p=str1;m>n;m--)
        {
                p=p->next;
        }
        for(q=str2;n>m;n--)
```

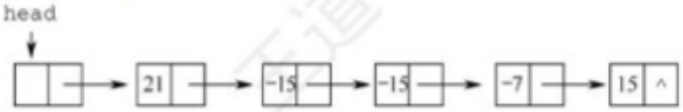
```
        {
                q=q->next;
        }
        while(p->next!=NULL&&p->next!=q->next){
                p=p->next;
                q=q->next;
        }
        return p->next;
}
```

时间复杂度

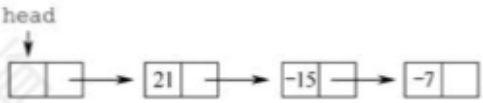
$o(max(len1, len2))$

19. 【2015 统考真题】用单链表保存 m 个整数，结点的结构为[data][link]，且|data|≤n（n 为正整数）。现要求设计一个时间复杂度尽可能高效的算法，对于链表中 data 的

绝对值相等的结点，仅保留第一次出现的结点而删除其余绝对值相等的结点。例如，若给定的单链表 head 如下：



则删除结点后的 head 为



要求：

1）给出算法的基本设计思想。

2）使用 C 或 C++语言，给出单链表结点的数据类型定义。

3）根据设计思想，采用 C 或 C++语言描述算法，关键之处给出注释。

4）说明你所设计算法的时间复杂度和空间复杂度。

思路

辅助数组，空间换时间

代码

```
typedef struct node{
        int data;
        struct node *Link;
}NODE;
Typedef NODE *PNODE;
void func(PNODE H, int n)
{
        PNODE p=h,r;
        int *q,m;
        q=(int *)malloc(sizeof(int)*(n+1));
        for(int i=0;i<n+1;i++)
                *(q+i)=0;
        while(p->link!=NULL){
                m=p->link->data>0?p->link->data:-p->link->data;
                if(*(q+m==0))
                {
                        *(q+m)=1;
                        p=p->Link;
                }
                else
                {
                r=p->link;
                p->link=r->link;
                free(r);
                }
        }
        free(q);
}
```

【2019 统考真题】设线性表 $L=(a_1,a_2,a_3,\cdots,a_{n-2},a_{n-1},a_n)$ 采用带头结点的单链表保存，链表中的结点定义如下：

```
typedef struct node
{   int data;
    struct node*next;
}NODE;
```

请设计一个空间复杂度为 $O(1)$且时间上尽可能高效的算法，重新排列 L 中的各结点，得到线性表 $L'=(a_1,a_n,a_2,a_{n-1},a_3,a_{n-2},\cdots)$。要求：

1）给出算法的基本设计思想。

2）根据设计思想，采用 C 或 C++语言描述算法，关键之处给出注释。

3）说明你所设计的算法的时间复杂度。

思路

后半部分逆置重排序

代码

```
void change_list(NODE *h){
	NODE *p,*q,*r,*s;
	p=q=h;
	while(p->next!=NULL){
		p=p->next;
		q=q->next;
		if(q->next!=NULL)
			q=q->next;
	}
	q=p->next;
	p->next=NULL;
	while(q!=NULL)
	{
		r=q->next;
		q->next=p->next;
		p->next=q;
		q=r;
	}
	s=h->next;
	q=p->next;
	p->next=NULL;
	while(q!=NULL)
	{
		r=q->next;
		q->next=s->next;
		s->next=q;
		s=q->next;
		q=r;
	}
}
```

时间复杂度

$O(n)$