# 单链表（结构体）

```c
#include <stdio.h>

#include <stdlib.h>

typedef struct Node{
    int date;
    struct Node *next;
}Node, *pNode;

void CreatList (pNode head){
    pNode p, pre =head;
    int num;
    printf("输入-1时停止\n");
    while(scanf("%d", &num),num != -1){
        p = (Node*)malloc(sizeof(Node));
        p->date = num;
        pre->next = p;
        pre = p;
    }
    pre->next = NULL;
}

void ListInsert (pNode head, int b, int c){
    pNode p, pre=head;
    int i=0;
    while (pre && i < b - 1){
        pre = pre->next;
        i++;
    }
    p=(Node*)malloc(sizeof(Node));
    p->date = c;
    p->next = pre->next;
    pre->next = p;
}

void DeletList (pNode head, int x){
    int i;
    pNode p = head, pre;
    while (p && i < x - 1){
        p = p->next;
        i++;
    }
    pre = p->next;
    p->next = pre->next;
    free(pre);
}

int LengthList (pNode head){
    pNode p = head;
    int len = 0;
    while (p->next){
        len++;
```

```
        p = p->next;
    }
    return len;
}

int SearchList (pNode head, int k){
    pNode p = head;
    int i = 0;
    while (i <= k - 1){
        p = p->next;
        i++;
    }
    return p->date;
}

void ShowList (pNode head){
    pNode p = head->next;
    while (p){
        printf("%d ", p->date);
        p = p->next;
    }
    printf("\n");
}

void AddList (pNode head, pNode head1, pNode Lc){
    pNode p = head->next, p1 = head1->next, pre = Lc, p2;
    while (p && p1){
        p2 = (Node*)malloc(sizeof(Node));
        if (p->date >= p1->date){
            p2->date = p1->date;
            pre->next = p2;
            pre = p2;
            p1 = p1->next;
        }
        else{
            p2->date = p->date;
            pre->next = p2;
            pre = p2;
            p = p->next;
        }
    }
    if (p){
        while (p){
            p2 = (Node*)malloc(sizeof(Node));
            p2->date = p->date;
            pre->next = p2;
            pre = p2;
            p = p->next;
        }
    }
    if (p1){
        while (p1){
            p2 = (Node*)malloc(sizeof(Node));
            p2->date = p1->date;
            pre->next = p2;
            pre = p2;
            p1 = p1->next;
        }
```

```
    }
    pre->next = NULL;
}

int main(){
    pNode head, head1, Lc;
    head = (Node*)malloc(sizeof(Node));
    head->next  = NULL;
    head1 = (Node*)malloc(sizeof(Node));
    head1->next  = NULL;
    Lc = (Node*)malloc(sizeof(Node));
    Lc->next  = NULL;
    int a;
    while(1){
        printf("输入要选择的操作\n");
        printf("1：创建初始链表\n2:在链表中插入元素\n3:删除元素\n4:输出链表的长度\n5:查找
链表中的元素\n6:输出链表中的元素\n7：创建新链表\n8：将两个链表进行归并排序\n9：退出系统\n");
        scanf("%d", &a);
        switch(a){
            case 1:
                CreatList (head);
                break;
            case 2:
                int b, c;
                printf("输入您要添加的点的位置与数值\n");
                scanf("%d%d", &b, &c);
                ListInsert(head, b, c);
                break;
            case 3:
                int x;
                printf("输入您要删除的点的位置\n");
                scanf("%d", &x);
                DeletList(head, x);
                break;
            case 4:
                printf("链表长度为%d\n", LengthList(head));
                break;
            case 5:
                int k;
                printf("输入您要查找的链表的位置\n");
                scanf("%d", &k);
                printf("%d\n", SearchList(head, k));
                break;
            case 6:
                ShowList (head);
                break;
            case 7:
                CreatList (head1);
                break;
            case 8:
                AddList (head, head1, Lc);
                ShowList (Lc);
                break;
            case 9:
                exit(0);
        }
    }
    return 0;
```

```
}
```