

## [实验题目二] 单链表的基本操作

### 一、实验目的

- 1、理解单链表的结构特点。
- 2、掌握单链表的基本操作。

### 二、预习内容

- 1、教材 2.3.1 线性链表；
- 2、教材关于单链表的基本操作。

### 三、实验内容

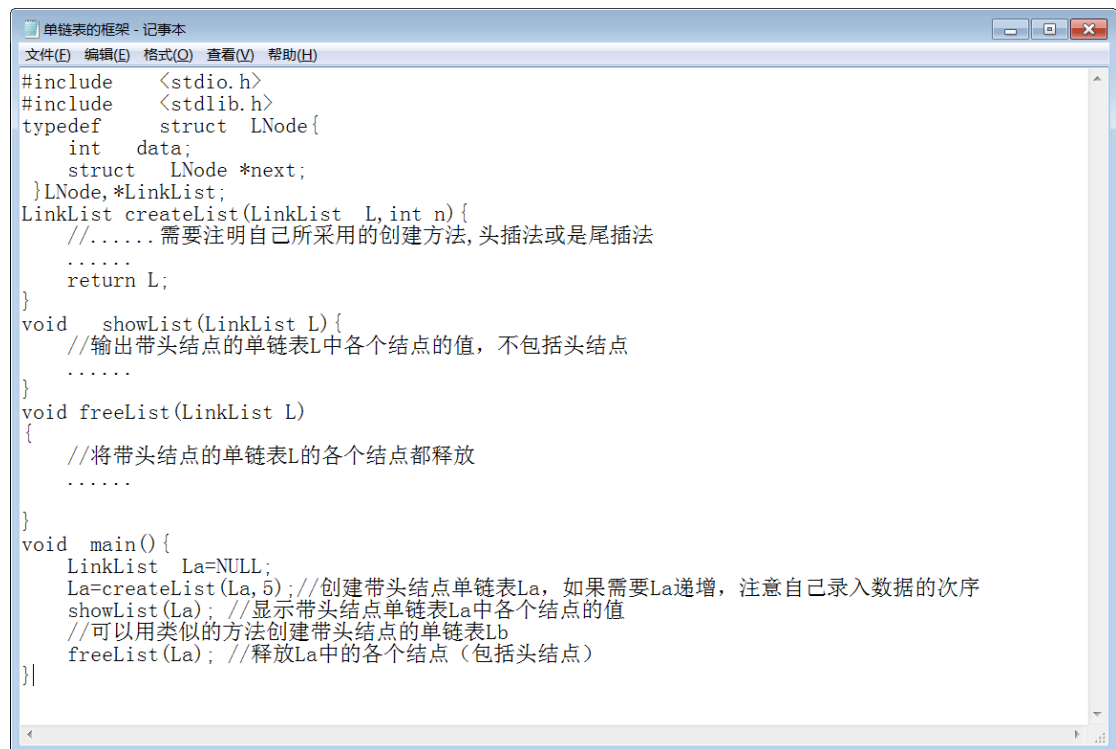
单链表的基本操作有以下 2 个实验内容：

#### 1、建立并输出一个单链表。

- A、首先要定义单链表的数据结构；
- B、编写一个函数使用“头插法”或“尾插法”创建一个单链表，例如：

```
LinkedList createList(LinkedList L,int n){  
    .....  
}
```

- C、编写一个函数（showList）能够显示一个单链表中各结点的值。
- D、编写一个函数（freeList）能够将单链表中的各个结点都释放。
- E、编写主函数（main）。
- F、以下的部分代码仅供参考。



```
#include <stdio.h>
#include <stdlib.h>
typedef struct LNode{
    int data;
    struct LNode *next;
}LNode,*LinkList;
LinkList createList(LinkList L,int n){
    //..... 需要注明自己所采用的创建方法, 头插法或是尾插法
    .....
    return L;
}
void showList(LinkList L){
    //输出带头结点的单链表L中各个结点的值, 不包括头结点
    .....
}
void freeList(LinkList L)
{
    //将带头结点的单链表L的各个结点都释放
    .....
}
void main(){
    LinkList La=NULL;
    La=createList(La,5); //创建带头结点单链表La, 如果需要La递增, 注意自己录入数据的次序
    showList(La); //显示带头结点单链表La中各个结点的值
    //可以用类似的方法创建带头结点的单链表Lb
    freeList(La); //释放La中的各个结点 (包括头结点)
}
```

注：如果做过了内容 1，那么可以做下面这个题目：

已知非递减单链表 L（录入数据时可以手动保证 L 是非递减的），插入一个元素 e 后，L 仍然是非递减的。

## 2、将一个带头结点单链表就地逆置

程序编制上的提示，仅供参考（其中很多代码可以借鉴上边实验内容 1）：

```
#include <stdio.h>
#include <stdlib.h>
typedef struct LNode{
    int data;
    struct LNode *next;
}LNode,*LinkList;
LinkList createList (LinkList L,int n){
    //创建一个含有 n 个结点的带头结点的单链表
    .....
    return L;
}
void showList (LinkList L){
```

```

//能够输出带头结点的单链表 L 中的各个结点的值
.....
}

void reverse_L(LinkList L) {
    //将一个带头结点单链表 L 就地逆置
    .....
}

main() {
    LinkList L=NULL;
    L=createList(L,6);
    showList (L);
    reverse_L (L);
    printf("\n 逆置后的结果是:\n");
    showList (L);
}

```

注：程序运行时如输入(假设采用头插法创建的链表)：6 5 4 3 2 1，

输出结果如下： 1 2 3 4 5 6

逆置后的结果是：6 5 4 3 2 1

- 3、一个值不重复带头结点单链表 L (L 中各个结点的值是无序的)，写一个函数实现将 L 中的值最小结点移到 L 的最前面，使其成为第一个结点(选做)。

#### 四、实验报告要求

- 1、写好实验题目、学号、姓名和实验日期；
- 2、写上实验的内容及程序源代码；
- 3、写上程序测试的结果。
- 4、写个实验小结。