# 静态顺序存储线性表的基本实现

**任务描述：**

        根据给出的存储结构，完成现在下列函数功能，并按照主函数的代码，输出规定的效果。

**预定义常量和类型：**

#define LIST\_INITSIZE 100

#define ElemType int

#define Status int

#define OK 1

#define ERROR 0

**存储结构：**

typedef struct

{

ElemType elem[LIST\_INITSIZE];

int length;

}SqList;

**函数操作：**

输出打印：void ListPrint(SqList L);

删除：void ListDelete(SqList\*L);

**注意：**

        对于初始化、插入和删除操作结果，见书上P19页，对于输出打印是指将表中**数据元素打印**出来；删除操作是**删除比第三个元素大的且比第六个元素小的数据元素,并保证元素的相对位置不变。**

**主函数如下：**

int main(void)

{

SqList L;

SqList B, C;

int i;

ElemType e;

ElemType data[9] = {10,-23,-32,-3,-25,88,77,0,-9};

printf("插入完成后L = : ");

ListPrint(L);

ListDelete(&L);

printf("**删除比第三个元素大的且比第六个元素小的数据元素：** ");

ListPrint(L);

return 0;

}

**编程要求：**

        1. 按照题目要求，编写实现每个函数，并按输出效果输出；

        2. 函数之间可以相互调用；

        3. 严格按照题目所给的存储结构实现；

        4. **如果该程序输出结果不对至少扣一半的分**。

# 动态顺序存储线性表的基本实现

根据给出的存储结构，完成现在下列函数功能，并按照主函数的代码，输出规定的效果。

**预定义常量和类型：**

#define LIST\_INIT\_SIZE 100

#define LISTINCREMENT 10

#define Status int

#define OVERFLOW -1

#define OK 1

#define ERROR 0

#define ElemType int

**存储结构：**

typedef struct

{

ElemType \* elem;

int length;

int listsize;

}SqList;

**函数操作：**

**打印：**void ListPrint(SqList \*L);

删除重复元素：void ListClassify (SqList \*L);

**注意：**

对于初始化、插入和删除操作结果，见书上P19页，对于输出打印是指将表中**数据元素打印**出来；分类是在线性表L中**将正数和负数分开，正数在前，负数在后，正数的相对位置保持不变，负数的相对位置也保持不变。**

**主函数如下：**

int main(void)

{

SqList L;

int i;

ElemType e;

ElemType data[9] = {10,-23,-32,-3,-25,88,77,0,-9};

printf("插入完成后 L = : ");

ListPrint(L);

ListClassify(&L);

printf("元素分类后的线性表：");

ListPrint(L);

return 0;

}

**编程要求：**

1. 按照题目要求，编写实现每个函数，并按输出效果输出；
2. 函数之间可以相互调用；
3. 严格按照题目所给的存储结构实现；
4. **如果该程序输出结果不对至少扣一半的分**。