0924 第三次作业

顺序存储实现的线性表，体现了存储密度高、可按数据元素在线性表中的位置进行随机存取的优点；但是除了在表尾进行插入数据元素和删除表尾的数据元素外，其他位置的插入、删除数据元素均可能引起数据元素的移动。

可以结合顺序存储的特点，将单向的存储空间视为首尾相连的逻辑环，从而对插入、删除操作进行优化：

1. 在表头元素之前插入数据元素时进行，不移动数据元素，在逻辑环的表头的前一个存储位置进行存储；
2. 在表中插入数据元素时（非表头前，非表尾后）时，根据插入位置距离表头近还是表尾近，选择移动数据元素少的方向进行数据元素的移动；
3. 在表中删除元素时，分为删除表头数据元素、删除表尾数据元素、表中数据元素三种情况，对删除表中的数据元素，同样根据删除位置距离表头还是表尾近来选择移动数据元素少的方向进行移动。

在考虑进行上述优化后，给出优化后的实现，可参考教材第三章循环队列的相关内容：

1. 线性表的顺序存储的结构体定义，需要进行那些改动？请定义新的结构体。
2. 按新的结构体和上述优化思路实现如下线性表的ADT函数，注意正确定义参数、参数类型和返回值。
   1. 在插入数据元素时，如初始化分配的空间不够，需要扩充存储空间，按增加1倍来扩充。
   2. 删除数据元素时，无需考虑缩小存储空间。
3. 线性表初始化：InitList(&L)
4. 销毁线性表： DestroyList(&L)
5. 按位置取表中某个元素：GetElem(L,i,&e)
6. 在i位置之前插入值为e的数据元素操作：ListInsert (&L,i,e)

//i==1，表示在表头之前插入

//i >表长度，表示在表尾之后插入

1. 将i位置数据元素值复制到e并删除i位置数据元素操作：ListDelete (&L,i,&e)
2. 编写main函数对你实现的ADT函数进行测试。
3. 提交内容：
   1. 源程序；
   2. 运行截图；并对截图的测试结果加以说明。
   3. 在这个优化实现中，插入、删除还是O(n)的时间复杂度，同时又多了很多编程工作，这样做有意义吗？说说你的想法。
   4. 如果还想要进一步减少插入、删除时的数据的移动次数，还有什么实现方法吗？这些实现方法，对最基础的线性表的顺序存储实现的优点有多大的影响，这些实现方法还能不能称为是“顺序存储”实现？