@TOC

线性表的基本操作

库函数应用以及结构申明

```
#include<stdio.h>
#include<stdlib.h>
#define MaxSize 50
#define InitSize 10

//静态表示数组，无法动态申请内存
typedef struct
{
   int data[MaxSize]; //主要包括数组以及表长
   int length;
}SqList;
/*

// 可以动态申请内存的表示方法
typedef struct
{
    int *data;              //采用指针方式定义
    int maxSize;            //为了区分MaxSize,这里写成小写
    int length;
}SeqList;

void InitList(SeqList &L){
    //用malloc申请连续的地址空间
    L.data = (int *)malloc(Sizeof(int)*InitSize);
    L.length = 0;
    L.maxSize = InitSize;
}

//增加动态数组内存
void IncreaseSize(SeqList &L, int len){
    int *p = L.data;
    L.data = (int *)malloc(sizeof(int)*(L.maxSize+len));
    for(int i = 0; i < L.length; i++){
        L.data[i] = p[i];       //给新申请的空间覆盖原来的值
    }
    L.maxSize =  L.maxSize+len;
    free(p);                    //释放原来的空间
}

*/
```

初始化

```
// 初始化线性表，表长置为0，数据元素也置为0
void InitList(SqList &L){
    for(int i=0; i < MaxSize; i++){
        L.data[i] = 0;
        L.length = 0;
    }
}
```

求表长

```
//求表长
int Length(SqList L){
    return L.length;
}
```

按值查找

```
//按值查找操作，根据值返回位序
int LocateElem(SqList L, int e){
    for(int i = 0; i < L.length; i++){
        if （e = L.data[i]）
        return i+1;                          //找到直接返回位序，注意这里返回的是位序，不
是下标
    }
    return 0;                                //未找到直接返回0
}
```

按位查找

```
//按位查找 获取第i个位置的元素
int GetElem(SqList L, int i){
    if(i < 1 || i > L.length){
        return 0;                            //下标范围不对返回0
    }
    else{
        return L.data[i-1];                  //返回该位置的值
    }
}
```

插入元素

```
//插入元素，在第i个位置插入元素e
bool ListInsert(SqList &L, int i, int e){
    if(i < 1 || i > L.length+1){
        return false;                        //位置出错，返回false
    }
    if(L.length >= MaxSize){
```

```
        return false;                           //当前空间已满，不能插入
    }
    for(int j = L.length; j >= i; j--){      //将第i个元素以及之后的元素后移
        L.data[j] = L.data[j-1];
    }
    L.data[i-1] = e;
    L.length++;                                 //长度加1
    return true;                                //插入成功
}
```

删除元素

```
//删除元素，删除表中的第i个元素，并返回给e
bool ListDelete(SqList &L, int i,int &e){
    if(i < 1 || i > L.length+1){
        return false;                           //位置出错，返回false
    }
    e = L.data[i-1];
    for(int j = i; j < L.length; j++){  //将第i个元素以及之后的元素前移
        L.data[j-1] = L.data[j];
    }
    L.length--;                                 //长度减1
    return true;                                //插入成功
}
```

输出线性表

```
//输出线性表
void PrintList(SqList L){
    for(int i = 0; i < L.length; i++){
        printf("线性表第%d个元素为%d\n",i+1,L.data[i]);
    }
}
```

判断是否为空

```
//判读是否为空
bool Empty(SqList L){
    if(L.length == 0){
        return true;
    }
    else{
        return false;
    }
}
```

销毁线性表

```
//销毁线性表
bool DestoryList(SqList &L){
    free(L.data);
    L.length = 0;
    return true;
}
```