数据结构与算法

文档版本：V0.0.1

修改日期：2021.09.04

1 线性表

1.1 概述

线性表（linear list）是具有相同数据类型的 n(n≥0)个数据元素的有限序列，其中数据元素的个数 n 称之为线性表的长度。当 n = 0 时称之为空表。

1.2 重要概念

指针：C 语言中基础语法功能，指针也是一个变量，存储的是地址值。

数组：C 语言中基础语法功能，n 个同一类型的数据的集合。

结构体：C 语言中基础语法功能，不同数据类型的数据的集合。

类型重定义：C 语言中基础语法功能，自定义数据结构体并自定义命名。

常用组和方式：

类型重定义+枚举，类型重定义+结构体，类型重定义+基础数据类型；

指针+结构体。

表（L 或 List）：按照连续地址排列的一连串数据，这一连串数据可称之为一个表，如 26 个英文字母。

数据(Iterm)：组成表的单元的单一数据，如整个英文字母表的第一结点的数据是 a，第二个节点的数据是 b。

表长（length）：表的长度。

结点或元素（Node）：组成表的单元，并不一定是最小单元，如英文字母 a 是整个英文字母表的第一个节点。

直接前趋：结点的上一个结点是该结点的直接前趋，除第一个元素以外，其他元素都有唯一的直接前趋。

直接后继：结点的下一个结点是该结点的直接后继，除最后一个元素以外，其他元素都有唯一的直接后继。

前趋结点：结点的上一结点称之为该结点的前趋结点。

后继结点：结点的后一结点称之为该结点的后继结点。

1.3 功能描述

初始化表（InitList）：将线性表清空操作。

求表长(ListLength)： 获取线性表的长度。

插入数据(Ins)：插入一个或多个结点到线性表中。

删除数据(Del)：删除线性表中一个或多个结点。

获取指定结点数据(GetNode)：指定线性表中的结点位置，获取该结点的数据值。

获取指定结点的后继节点数据(GetNext)：指定线性表中的结点位置，获取该结点的后继结点的数据值。

获取指定结点的前趋节点数据(GetPrior)：指定线性表中的结点位置，获取该结点的前趋结点的数据值。

定位(Loc)：根据数据获取该数据在线性表的位置。

1.4 API 参考

线性表提供的 API:

InitList:初始化线性表。
ClearList:清除线性表的内容。
ListLength:求线性表的长度。
GetNode:获取线性表中指定结点的数据。
GetNext: 获取线性表中指定结点的后继节点的数据。
GetPrior: 获取线性表中指定结点的前趋节点的数据。
Ins:指定一个位置，插入（后插入的方式）一项数据到线性表中。
Del: 指定一个位置，删除线性表的该位置中的结点。
Loc:定位（按值查找），根据数据返回该结点的位置。

InitList

【描述】

初始化线性表。

【语法】

```
InitList(sequenlist *L);
```

【参数】

参数名称	描述	输入/输出
L	线性表指针	输入

【返回值】

返回值	描述
0	成功
非 0	失败

【函数原型】

```
int InitList(sequenlist *L)
{
    sequenlist ->Length = 0;
    return 0;
}
```

【需求】

无

【注意】

无

【举例】
无
【相关主题】
无

ClearList
【描述】
清除线性表的内容。
【语法】
ClearList(sequenlist *L);
【参数】

参数名称	描述	输入/输出
L	线性表指针	输入

【返回值】

返回值	描述
0	成功
非 0	失败

【函数原型】

```
int ClearList (sequenlist *L)
{
    int i = 0;
    for(i = 0; i < sequenlist ->Length;i++)
    sequenlist ->data = 0;
    return 0;
}
```

【需求】
无
【注意】
无
【举例】
无
【相关主题】
无

ListLength
【描述】
求线性表的长度。
【语法】
ListLength(sequenlist *L,ElemType *length);

【参数】

参数名称	描述	输入/输出
L	线性表指针	输入
length	线性表长度	输出

【返回值】

返回值	描述
0	成功
非 0	失败

【函数原型】

```
int ListLength(sequenlist *L,ElemType *length)
{
   *length = sequenlist ->Length;
   return 0;
}
```

【需求】

无

【注意】

无

【举例】

无

【相关主题】

无

GetNode

【描述】

获取线性表中指定结点的数据。

【语法】

GetNode(sequenlist *L, unsigned int index, datatype *data);

【参数】

参数名称	描述	输入/输出
L	线性表指针	输入
index	结点位置	输入,[1,L->length]
data	结点数据	输出

【函数原型】

```
int GetNode(sequenlist *L, unsigned int index, datatype *data)
{
   if(( index > L->length )||( index < 1))
```

```
        return FUNC_ERR_UNFIX;

    *data = L->elem[index - 1];
    return FUNC_SUCCESS;
}
```

【返回值】

返回值	描述
FUNC_SUCCESS	成功
FUNC_ERR_UNFIX	输入结点位置不合适

【需求】

无

【注意】

无

【举例】

无

【相关主题】

无

GetNext

【描述】

获取线性表中指定结点的后继节点的数据。

【语法】

GetNext(sequenlist *L, unsigned int index, datatype *data);

【参数】

参数名称	描述	输入/输出
L	线性表指针	输入
index	结点位置	输入,[1,L->length)
data	结点数据	输出

【函数原型】

```
int GetNext(sequenlist *L, unsigned int index, datatype *data)
{
    if(( index > L->length  -  1 )||( index < 1))
        return FUNC_ERR_UNFIX;
    if(L->length < 2)
        return FUNC_ERR_SHORTLENGH;
    *data = L->elem[index];
    return FUNC_SUCCESS;
}
```

【返回值】

返回值	描述
FUNC_SUCCESS	成功
FUNC_ERR_UNFIX	输入结点位置不合适
FUNC_ERR_SHORTLENGH	线性表长度小于 2

【需求】

无

【注意】

无

【举例】

无

【相关主题】

无

GetPrior

【描述】

获取线性表中指定结点的前趋节点的数据。

【语法】

GetPrior(sequenlist *L, unsigned int index, datatype *data);

【参数】

参数名称	描述	输入/输出
L	线性表指针	输入
index	结点位置	输入，(1,L->Length]
data	结点数据	输出

【函数原型】

```
int GetPrior(sequenlist *L, unsigned int index, datatype *data)
{
    if(( index > L->length )||( index < 2))
        return FUNC_ERR_UNFIX;
    if(L->length < 2)
        return FUNC_ERR_SHORTLENGH;
    *data = L->elem[index - 2];
    return FUNC_SUCCESS;
}
```

【返回值】

返回值	描述
FUNC_SUCCESS	成功
FUNC_ERR_UNFIX	输入结点位置不合适
FUNC_ERR_SHORTLENGH	线性表长度小于 2

【需求】

无

【注意】

无

【举例】

无

【相关主题】

无

Ins

【描述】

Ins:指定一个位置，插入（后插入的方式）一项数据到线性表中。

【语法】

Ins (sequenlist *L, unsigned int index, datatype data);

【参数】

参数名称	描述	输入/输出
L	线性表指针	输入
index	结点位置	输入，[1,L->Length]
data	结点数据	输入

【函数原型】

```
int Ins (sequenlist *L, unsigned int index, datatype data)
{
    int temp;
    if(( index > L->length )||( index < 1))
        return FUNC_ERR_UNFIX;
    for(temp = L->length; temp > index; temp--)
            L->elem[temp] = L->elem[temp -1];
    L->elem[index - 1] = data;
    L->length++;
    return FUNC_SUCCESS;
}
```

【返回值】

返回值	描述
FUNC_SUCCESS	成功
FUNC_ERR_UNFIX	输入结点位置不合适

【需求】

无

【注意】

无

【举例】

无

【相关主题】

无

Del

【描述】

Del: 指定一个位置，删除线性表的该位置中的结点。

【语法】

Del (sequenlist *L, unsigned int index);

【参数】

参数名称	描述	输入/输出
L	线性表指针	输入
index	结点位置	输入，[1,L->Length]

【函数原型】

```
int Del (sequenlist *L, unsigned int index)
{
    int temp;
    if(( index > L->length )||( index < 1))
        return FUNC_ERR_UNFIX;
    if(L->length < 1)
        return FUNC_ERR_EMPTY;
    for(temp = index; temp < L->Length; temp++)
        L->elem[temp] = L->elem[temp+1];
    L->length--;

    return FUNC_SUCCESS;
}
```

【返回值】

返回值	描述
FUNC_SUCCESS	成功
FUNC_ERR_UNFIX	输入结点位置不合适
FUNC_ERR_EMPTY	线性表为空表

【需求】

无

【注意】

无

【举例】

无

【相关主题】

无

Loc

【描述】

Loc:定位（按值查找），根据数据返回该结点的位置。

【语法】

Loc(sequenlist *L, datatype Item);

【参数】

参数名称	描述	输入/输出

L	线性表指针	输入
Item	结点数据	输入
index	结点数据地址指针	输出

【函数原型】

```
int Loc (sequenlist *L, datatype Item, datatype *index)
{
    int i,j;
    if(L->length < 1)
        return FUNC_ERR_EMPTY;
    j = L->length;
    for(i = 0;i < L->length; i++)
    {
        if(L->elem[i] == Item)
            *index = i;
    }
    printf("no found %d!\n",Item);
    return FUNC_SUCCESS;
}
```

【返回值】

返回值	描述
FUNC_SUCCESS	成功
FUNC_ERR_EMPTY	线性表为空表

【需求】

无

【注意】

无

【举例】

无

【相关主题】

无

1.5 数据类型

Sequenlist

【说明】

线性表

【定义】

```
typedef struct
{
    datatype elem[maxsize];
```

```
        int length;
} sequenlist;
```

【成员】

成员名称	描述
elem	数据域，maxsize 为 1024
length	长度

datatype

【说明】

数据元素类型

【定义】

```
#define datatype int;
```

1.6 错误码

错误代码	宏定义	描述
-1	FUNC_ERR_UNFIX	输入结点位置不合适
-2	FUNC_ERR_EMPTY	线性表为空表
-3	FUNC_ERR_SHORTLENGH	线性表长度小于 2

2 链表

2.1 概述

2.2 重要概念

2.3 功能描述

2.4 API 参考

2.5 数据类型

2.6 错误码