

**数 据 结 构 实 验**

**(2015级信息安全专业适用)**

**计算机科学与技术学院**

**数组结构课程组**

**2016年1月**

目 录

[1基于顺序存储结构的线性表实现 1](#_Toc440831634)

[1.1 实验目的 1](#_Toc440831635)

[1.2 线性表基本运算定义 1](#_Toc440831636)

[1.3 实验任务 2](#_Toc440831637)

[2基于二叉链表的二叉树实现 3](#_Toc440831638)

[2.1 实验目的 3](#_Toc440831639)

[2.2 二叉树基本运算定义 3](#_Toc440831640)

[2.3 实验任务 5](#_Toc440831641)

[参考文献 6](#_Toc440831642)

[附录A 顺序表实现框架程序清单 7](#_Toc440831643)

[附录B 数据元素的文件读写 10](#_Toc440831644)

[附录C 线性表模板实例 11](#_Toc440831645)

# 1基于顺序存储结构的线性表实现

## 1.1 实验目的

通过实验达到⑴加深对线性表的概念、基本运算的理解；⑵熟练掌握线性表的逻辑结构与物理结构的关系；⑶物理结构采用顺序表,熟练掌握线性表的基本运算的实现。

## 1.2 线性表基本运算定义

依据最小完备性和常用性相结合的原则，以函数形式定义了线性表的初始化表、销毁表、清空表、判定空表、求表长和获得元素等12种基本运算，具体运算功能定义如下。

⑴初始化表：函数名称是InitaList(L)；初始条件是线性表L不存在已存在；操作结果是构造一个空的线性表。

⑵销毁表：函数名称是DestroyList(L)；初始条件是线性表L已存在；操作结果是销毁线性表L。

⑶清空表：函数名称是ClearList(L)；初始条件是线性表L已存在；操作结果是将L重置为空表。

⑷判定空表：函数名称是ListEmpty(L)；初始条件是线性表L已存在；操作结果是若L为空表则返回TRUE,否则返回FALSE。

⑸求表长：函数名称是ListLength(L)；初始条件是线性表已存在；操作结果是返回L中数据元素的个数。

⑹获得元素：函数名称是GetElem(L,i,e)；初始条件是线性表已存在，1≤i≤ListLength(L)；操作结果是用e返回L中第i个数据元素的值。

⑺查找元素：函数名称是LocateElem(L,e,compare())；初始条件是线性表已存在；操作结果是返回L中第1个与e满足关系compare（）关系的数据元素的位序，若这样的数据元素不存在，则返回值为0。

⑻获得前驱：函数名称是PriorElem(L,cur\_e,pre\_e)；初始条件是线性表L已存在；操作结果是若cur\_e是L的数据元素，且不是第一个，则用pre\_e返回它的前驱，否则操作失败，pre\_e无定义。

⑼获得后继：函数名称是NextElem(L,cur\_e,next\_e)；初始条件是线性表L已存在；操作结果是若cur\_e是L的数据元素，且不是最后一个，则用next\_e返回它的后继，否则操作失败，next\_e无定义。

⑽插入元素：函数名称是ListInsert(L,i,e)；初始条件是线性表L已存在且非空，1≤i≤ListLength(L)+1；操作结果是在L的第i个位置之前插入新的数据元素e。

⑾删除元素：函数名称是ListDelete(L,i,e)；初始条件是线性表L已存在且非空，1≤i≤ListLength(L)；操作结果：删除L的第i个数据元素，用e返回其值。

⑿遍历表：函数名称是ListTraverse(L,visit())，初始条件是线性表L已存在；操作结果是依次对L的每个数据元素调用函数visit()。

## 1.3 实验任务

采用顺序表作为线性表的物理结构，实现§1.2的基本运算。其中ElemType为数据元素的类型名，具体含义可自行定义，其它有关类型和常量的定义和引用详见文献[1]的p10。

要求构造一个具有菜单的功能演示系统。其中，在主程序中完成函数调用所需实参值的准备和函数执行结果的显示，并给出适当的操作提示显示。附录A提供了简易菜单的框架。

演示系统可选择实现线性表的文件形式保存。其中，①需要设计文件数据记录格式，以高效保存线性表数据逻辑结构(D,{R})的完整信息；②需要设计线性表文件保存和加载操作合理模式。附录B提供了文件存取的方法。

演示系统可选择实现多个线性表管理。

撰写本次实验报告，作为课程实验报告第一章的内容，其内容至少包括问题描述、系统设计、系统实现和实验小结。实验报告需要按照规范格式要求规范排版，详见“[2016-数据结构实验报告格式示例(2014级信息安全专业).docx](2016-数据结构实验报告格式示例（2014级信息安全专业）.docx)”。

演示系统的源程序应按照代码规范增加注释和排版，目标程序务必是可以独立于IDE运行的EXE文件。

按照公告的时间及时提交电子档实验资料，所有资料存储于每位同学自己的相应文件夹下，其文件夹名称格式为“专业班级-学号姓名-n”。如：IS1402-U201414999李某某-n。其中，n表示第n次实验报告。

资料至少包括实验报告、实验源程序和实验目标程序。根据需要还可以增加测试用例文件等等。

# 2基于二叉链表的二叉树实现

## 2.1 实验目的

通过实验达到⑴加深对二叉树的概念、基本运算的理解；⑵熟练掌握二叉树的逻辑结构与物理结构的关系；⑶以二叉链表作为物理结构，熟练掌握二叉树基本运算的实现。

## 2.2 二叉树基本运算定义

依据最小完备性和常用性相结合的原则，以函数形式定义了二叉树的初始化二叉树、销毁二叉树、创建二叉树、清空二叉树、判定空二叉树和求二叉树深度等20种基本运算，具体运算功能定义如下。

⑴初始化二叉树：函数名称是InitBiTree(T)；初始条件是二叉树T不存在；操作结果是构造空二叉树T。

⑵销毁二叉：树函数名称是DestroyBiTree(T)；初始条件是二叉树T已存在；操作结果是销毁二叉树T。

⑶创建二叉树：函数名称是CreateBiTree(T,definition)；初始条件是definition 给出二叉树T的定义；操作结果是按definition构造二叉树T。

⑷清空二叉树：函数名称是ClearBiTree (T)；初始条件是二叉树T存在； 操作结果是将二叉树T清空。

⑸判定空二叉树：函数名称是BiTreeEmpty(T)；初始条件是二叉树T存在；操作结果是若T为空二叉树则返回TRUE，否则返回FALSE。

⑹求二叉树深度：函数名称是BiTreeDepth(T)；初始条件是二叉树T存在；操作结果是返回T的深度。

⑺获得根结点：函数名称是Root(T)；初始条件是二叉树T已存在；操作结果是返回T的根。

⑻获得结点：函数名称是Value(T,e)；初始条件是二叉树T已存在，e是T中的某个结点；操作结果是返回e的值。

⑼结点赋值：函数名称是Assign(T,&e,value)；初始条件是二叉树T已存在，e是T中的某个结点；操作结果是结点e赋值为value。

⑽获得双亲结点：函数名称是Parent(T,e) ；初始条件是二叉树T已存在，e是T中的某个结点；操作结果是若e是T的非根结点，则返回它的双亲结点指针，否则返回NULL。

⑾获得左孩子结点：函数名称是LeftChild(T,e)；初始条件是二叉树T存在，e是T中某个节点；操作结果是返回e的左孩子结点指针。若e无左孩子，则返回NULL。

⑿获得右孩子结点：函数名称是RightChild(T,e)；初始条件是二叉树T已存在，e是T中某个结点；操作结果是返回e的右孩子结点指针。若e无右孩子，则返回NULL。

⒀获得左兄弟结点：函数名称是LeftSibling(T,e)；初始条件是二叉树T存在，e是T中某个结点；操作结果是返回e的左兄弟结点指针。若e是T的左孩子或者无左兄弟，则返回NULL。

⒁获得右兄弟结点：函数名称是RightSibling(T,e)；初始条件是二叉树T已存在，e是T中某个结点；操作结果是返回e的右兄弟结点指针。若e是T的右孩子或者无有兄弟，则返回NULL。

⒂插入子树：函数名称是InsertChild(T,p,LR,c)；初始条件是二叉树T存在，p指向T中的某个结点，LR为0或1，,非空二叉树c与T不相交且右子树为空；操作结果是根据LR为0或者1，插入c为T中p所指结点的左或右子树，p 所指结点的原有左子树或右子树则为c的右子树。

⒃删除子树：函数名称是DeleteChild(T.p.LR)；初始条件是二叉树T存在，p指向T中的某个结点，LR为0或1。 操作结果是根据LR为0或者1，删除c为T中p所指结点的左或右子树。

⒄前序遍历：函数名称是PreOrderTraverse(T,Visit())；初始条件是二叉树T存在，Visit是对结点操作的应用函数；操作结果：先序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

⒅中序遍历：函数名称是InOrderTraverse(T,Visit))；初始条件是二叉树T存在，Visit是对结点操作的应用函数；操作结果是中序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

⒆后序遍历：函数名称是PostOrderTraverse(T,Visit))；初始条件是二叉树T存在，Visit是对结点操作的应用函数；操作结果是后序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

⒇按层遍历：函数名称是LevelOrderTraverse(T,Visit))；初始条件是二叉树T存在，Visit是对结点操作的应用函数；操作结果是层序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

## 2.3 实验任务

采用二叉链表表作为二叉树的物理结构，实现§2.2的基本运算。其中ElemType为数据元素的类型名，具体含义可自行定义，其它有关类型和常量的定义和引用详见文献[1]的p10。

要求构造一个具有菜单的功能演示系统。其中，在主程序中完成函数调用所需实参值的准备和函数执行结果的显示，并给出适当的操作提示显示。

演示系统可选择实现二叉树的文件形式保存。其中，①需要设计文件数据记录格式，以高效保存二叉树数据逻辑结构(D,{R})的完整信息；②需要设计线性表文件保存和加载操作合理模式。附录B提供了文件存取的方法。

演示系统可选择实现多个二叉树管理。可采用线性表的方式管理多个二叉树，线性表中的每个数据元素为一个二叉树的基本属性，至少应包含有二叉树的名称。基于顺序表实现的二叉树管理，其物理结构的参考设计如图2-1所示。



图2-1 多二叉树管理的物理结构示意图

演示系统的源程序应按照代码规范增加注释和排版，目标程序务必是可以独立于IDE运行的EXE文件。

撰写本次实验报告，作为课程实验报告第二章的内容，其内容至少包括问题描述、系统设计、系统实现和实验小结。实验报告需要按照规范格式要求规范排版，详见“[2016-数据结构实验报告格式示例(2014级信息安全专业).docx](2016-数据结构实验报告格式示例（2014级信息安全专业）.docx)”。

按照公告的时间及时提交电子档实验资料，所有资料存储于每位同学自己的相应文件夹下，其文件夹名称格式为“专业班级-学号姓名-n”。如：IS1402-U201414999李某某-n。其中，n表示第n次实验报告。

资料至少包括实验报告、实验源程序和实验目标程序。根据需要还可以增加测试用例文件等等。

# 参考文献

[1] 严蔚敏等.数据结构（C语言版）.清华大学出版社

[2] [Larry Nyhoff](http://www.calvin.edu/~nyhl/index.html). [ADTs, Data Structures, and Problem Solving with C++.](http://vig.prenhall.com/catalog/academic/product/0,1144,0131409093,00.html)Second Edition, [Calvin College](http://cs.calvin.edu/),2005

[3] 殷立峰. Qt C++跨平台图形界面程序设计基础. 清华大学出版社,2014:192～197

[4] 严蔚敏等.数据结构题集（C语言版）.清华大学出版社

# 附录A 顺序表实现框架程序清单

/\* Linear Table On Sequence Structure \*/

#include <stdio.h>

#include <malloc.h>

#include <stdlib.h>

/\*---------page 10 on textbook ---------\*/

#define TRUE 1

#define FALSE 0

#define OK 1

#define ERROR 0

#define INFEASTABLE -1

#define OVERFLOW -2

typedef int status;

typedef int ElemType; //数据元素类型定义

/\*-------page 22 on textbook -------\*/

#define LIST\_INIT\_SIZE 100

#define LISTINCREMENT 10

typedef struct{ //顺序表（顺序结构）的定义

ElemType \* elem;

int length;

int listsize;

}SqList;

/\*-----page 19 on textbook ---------\*/

status IntiaList(SqList & L);

//status DestroyList(SqList & L);

//status ClearList(SqList &L);

//status ListEmpty(SqList L);

//int ListLength(SqList L);

//status GetElem(SqList L,int i,ElemType & e);

//status LocateElem(SqList L,ElemType e); //简化过

//status PriorElem(SqList L,ElemType cur,ElemType & pre\_e);

//status NextElem(SqList L,ElemType cur,ElemType & next\_e);

//status ListInsert(SqList & L,int i,ElemType e);

status ListDelete(SqList & L,int i,ElemType & e);

status ListTrabverse(SqList L); //简化过

/\*--------------------------------------------\*/

void main(void){

SqList L; int op=1;

while(op){

system("cls"); printf("\n\n");

printf(" Menu for Linear Table On Sequence Structure \n");

printf("-------------------------------------------------\n");

printf(" 1. IntiaList 7. LocateElem\n");

printf(" 2. DestroyList 8. PriorElem\n");

printf(" 3. ClearList 9. NextElem \n");

printf(" 4. ListEmpty 10. ListInsert\n");

printf(" 5. ListLength 11. ListDelete\n");

printf(" 6. GetElem 12. ListTrabverse\n");

printf(" 0. Exit\n");

printf("-------------------------------------------------\n");

printf(" 请选择你的操作[0~12]:");

scanf("%d",&op);

switch(op){

case 1:

//printf("\n----IntiaList功能待实现！\n");

if(IntiaList(L)==OK) printf("线性表创建成功！\n");

else printf("线性表创建失败！\n");

getchar();getchar();

break;

case 2:

printf("\n----DestroyList功能待实现！\n");

getchar();getchar();

break;

case 3:

printf("\n----ClearList功能待实现！\n");

getchar();getchar();

break;

case 4:

printf("\n----ListEmpty功能待实现！\n");

getchar();getchar();

break;

case 5:

printf("\n----ListLength功能待实现！\n");

getchar();getchar();

break;

case 6:

printf("\n----GetElem功能待实现！\n");

getchar();getchar();

break;

case 7:

printf("\n----LocateElem功能待实现！\n");

getchar();getchar();

break;

case 8:

printf("\n----PriorElem功能待实现！\n");

getchar();getchar();

break;

case 9:

printf("\n----NextElem功能待实现！\n");

getchar();getchar();

break;

case 10:

printf("\n----ListInsert功能待实现！\n");

getchar();getchar();

break;

case 11:

printf("\n----ListDelete功能待实现！\n");

getchar();getchar();

break;

case 12:

//printf("\n----ListTrabverse功能待实现！\n");

if(!ListTrabverse(L)) printf("线性表是空表！\n");

getchar();getchar();

break;

case 0:

break;

}//end of switch

}//end of while

printf("欢迎下次再使用本系统！\n");

}//end of main()

/\*--------page 23 on textbook --------------------\*/

status IntiaList(SqList & L){

L.elem = (ElemType \*)malloc( LIST\_INIT\_SIZE \* sizeof (ElemType));

if(!L.elem) exit(OVERFLOW);

L.length=0;

L.listsize=LIST\_INIT\_SIZE;

return OK;

}

status ListTrabverse(SqList L){

int i;

printf("\n-----------all elements -----------------------\n");

for(i=0;i<L.length;i++) printf("%d ",L.elem[i]);

printf("\n------------------ end ------------------------\n");

return L.length;

}

# 附录B 数据元素的文件读写

#include <stdio.h>

#include <stdlib.h>

typedef struct {

char c; int d; float f;

} ElemType;

typedef struct {

ElemType elem[10]; int length;

} SqList;

SqList L={{{'a',1,1.1},{'b',2,2.2},{'c',3,3.3},{'d',4,4.4}}, 4};

int main(int argc, char \*argv[]) {

FILE \*fp; char filename[30]; int i;

printf("input file name: ");

scanf("%s",filename);

//写文件的方法

if ((fp=fopen(filename,"w"))==NULL)

{

printf("File open erroe\n ");

return 1;

}

fwrite(L.elem,sizeof(ElemType),L.length,fp);

//这里是1次性写入，对于其它物理结构，可通过遍历，逐个访问数据元素

//并写入到文件中

fclose(fp);

//读文件的方法

L.length=0;

if ((fp=fopen(filename,"r"))==NULL)

{

printf("File open erroe\n ");

return 1;

}

while(fread(&L.elem[L.length],sizeof(ElemType),1,fp))

L.length++;

//这里从文件中逐个读取数据元素恢复顺序表，对于不同的物理结构，可通过读//取的数据元素恢复内存中的物理结构。

fclose(fp);

return 0;

}

# 附录C 线性表模板实例

#include "iostream"

using namespace std;

typedef float ElemType;

template <typename T>

struct list{

T elem[10];

int length;

};

template <typename T1>

int ListInsert(struct list<T1> &L,int i, T1 x)

{

int j;

if(i<1 || i>L.length+1) return ERROR;

if(i>100) return OVERFLOW;

for(j=L.length-1;j>=i-1;j--)

L.elem[j+1]=L.elem[j];

L.elem[i-1]=x;

L.length++;

return OK;

}

template <typename T2>

void ListDisplay(struct list<T2> &L)

{

int i;

for(i=0;i<L.length;i++)

cout<<" "<<L.elem[i];

}

int main()

{

struct list<ElemType> L1;

L1.length=0;

ListInsert(L1,1,(ElemType)10.2);

ListInsert(L1,1,(ElemType)20);

ListDisplay(L1);

return 1;

}

# 附录D 数据结构实验评价指标

|  |  |  |
| --- | --- | --- |
| **评价指标** | **满分** | **评价标准** |
| 程序功能（40%） | 100 | 成绩=(实完成功能数/应完成功能数)\*100。其中，正常用例记0.8，异常用例记0.2。 |
| 程序规范（10%） | 100 | 格式规范：80，注释：80+，不规范：80-。 |
| 报告内容（40%） | 100 | 问题描述：20，数据结构、算法设计和理论分析：60，测试计划：20。 |
| 报告规范（10%） | 100 | 基本规范：80，规范：80+，不规范：80-。 |
| 逾期扣分 | 10 | 逾期提交：2/天。超过5天者本次实验记0。 |
| **综合成绩＝实验成绩×92%＋实验考勤×8%**  实验成绩＝(∑程序功能×40%＋程序规范×10%＋报告内容×40%＋报告规范×10%－逾期扣分)÷2 | | |

注：实验考勤原则上仅记录签到情况，不考虑任何请假情形。