武汉纺织大学

数学与计算机学院

数据结构实验报告

2019~2020学年第一学期

|  |  |
| --- | --- |
| 专 业 班 级 | 物联网二班 |
| 学生学号 | 1802220113 |
| 学生姓名 | 梁少伟 |
| 任 课 教 师 | 叶鹏 |
| 实验指导教师 | 叶鹏 |
| 实验地点 | 实训楼 |
| 填写时间 | 2019/9/25 |
|  | |

**填写说明**

1. 报告需要学生填写的内容包括：实验要求、完整的实验结果记录、实验结果分析、实验总结。实验评价由教师填写。
2. 各部分填写内容的要求
   * **实验要求**主要填写源代码。
   * **实验结果记录**中主要反映程序运行后的效果。
   * **实验结果分析**中主要就程序运行结果是否符合预期进行分析，对于达不到预期的分析其产生的原因。
   * **实验总结**中主要阐述在实验中遇到了哪些问题，如何解决的。
3. 实验报告及源代码的提交方式
   * 报告和源代码一并发送到trustie平台。
   * 附件中应有两个文件：一是word版实验报告，**实验报告的命名规则**是：“学号\_姓名\_班级\_实验X”。另一个是压缩文件。它是将整个工程压缩成一个文件，这个**压缩文件的命名规则**是：“学号\_姓名\_班级\_实验X\_源代码”。

**实验1 顺序表的基本操作**

**一、实验目的**

1.掌握使用VC++上机调试线性表的基本方法；

2.掌握线性表的基本操作：插入、删除、查找等运算在顺序存储结构上的实现。

二、**实验内容**

顺序表的基本操作的实现。在此基础上实现教材中的算法2.7。对于每个顺序表的基本操作和算法2.7，都必须输入不少于两种数据进行测试。

头文件（head.h）

#include <stdio.h>

#include<stdlib.h>

#define Size 20

typedef struct list

{

int\* head;

int length;

int size;

}list;

list initlist();

list create(list t1);

list addlist(list t, int elem, int add);

list dellist(list t, int add);

int selectlist(list t, int elem);

list amendlist(list t, int elem, int newElem);

list emerge(list t1, list t2, list t3);

void ListDelete(list t);

void displaylist(list t);

void menu();

list sort(list t);

函数实现（function.cpp）

#include"head.h"

list initTable()

{

list

t;

t.head = (int\*)malloc(Size \* sizeof(int));

if (!t.head)

{

printf("FAILURE");

exit(0);

}

t.length = 0;

t.size = Size;

return t;

}

list create(list t1)

{

for (int i = 1; i <= 4; i++)

{

int n;

printf("输入元素");

scanf\_s("%d", &n);

t1.head[i - 1] = n;

t1.length++;

}

return t1;

}

list addTable(list t, int elem, int add)

{

if (add > t.length + 1 || add < 1)

{

printf("PROBLEM");

return t;

}

if (t.length >= t.size)

{

t.head = (int\*)realloc(t.head, (t.size + 1) \* sizeof(int));

if (!t.head)

{

printf("存储分配失败");

}

t.size += 1;

}

for (int i = t.length - 1; i >= add - 1; i--)

{

t.head[i + 1] = t.head[i];

}

t.head[add - 1] = elem;

t.length++;

return t;

}

list delTable(list t, int add)

{

if (add > t.length || add < 1)

{

printf("被删除元素的位置有误");

exit(0);

}

for (int i = add; i < t.length; i++)

{

t.head[i - 1] = t.head[i];

}

t.length--;

return t;

}

int selectTable(list t, int elem)

{

for (int i = 0; i < t.length; i++)

{

if (t.head[i] == elem)

{

return i + 1;

}

}

return -1;

}

list amendTable(list t, int elem, int newElem)

{

int add = selectTable(t, elem);

t.head[add - 1] = newElem;

return t;

}

list sort(list t)

{

int temp;

for (int i = 0; i < t.length; i++)

for (int j = i; j < t.length; j++)

{

if (t.head[i] > t.head[j])

{

temp = t.head[j];

t.head[j] = t.head[i];

t.head[i] = temp;

}

}

return t;

}

list emerge(list t1, list t2, list t3)

{

int i = 0, j = 0;

while (i < t1.length && j < t2.length)

{

if (t1.head[i] < t2.head[j])

{

t3.head[t3.length] = t1.head[i];

i++;

}

else

{

t3.head[t3.length] = t2.head[j];

j++;

}

t3.length++;

}

while (i < t1.length)

{

t3.head[t3.length] = t1.head[i];

i++;

t3.length++;

}

while (j < t2.length)

{

t3.head[t3.length] = t2.head[j];

j++;

t3.length++;

}

return t3;

}

void ListDelete(list t)

{

free(t.head); t.head = NULL; printf("已销毁\n");

}

void displayTable(list t)

{

for (int i = 0; i < t.length; i++)

{

printf("%d", t.head[i]);

}

printf("\n");

}

void menu()

{

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

printf(" 1.建立顺序表La 2.建立顺序表Lb 3.合并La和Lb生成Lc \n");

printf(" 4.查找Lc表中的元素 5.删除Lc表中的元素 6.在Lc表中插入元素 \n");

printf(" 7.删毁Lc表 8.退出 \n");

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

}

主函数（main.cpp）

#include "head.h"

int main() {

list la = initTable();

list lb = initTable();

list lc = initTable();

menu();

int choice = 1;

while (choice) {

printf("选择项目序号：");

scanf\_s("%d", &choice);

switch (choice) {

case 1: {

la = create(la);

la = sort(la);

displayTable(la);

break;

}

case 2: {

lb = create(lb);

lb = sort(lb);

displayTable(lb);

break;

}

case 3: {

lc = emerge(la, lb, lc);

displayTable(lc);

break;

}

case 4: {

int index;

printf("输入要查找的元素（下标）:");

scanf\_s("%d", &index);

int add = selectTable(lc, index);

printf("%d\n", add);

break;

}

case 5: {

int index;

printf("输入要删除的元素（下标）：");

scanf\_s("%d", &index);

lc = delTable(lc, index);

displayTable(lc);

break;

}

case 6: {

int index, num;

printf("输入插入的序号和数据：");

scanf\_s("%d %d", &index, &num);

lc = addTable(lc, index, num);

displayTable(lc);

break;

}

case 7: {

ListDelete(lc);

break;

}

default:

exit(0);

}

}

return 0;

**三、实验要求**  
1.认真阅读和理解本实验的算法。

2.上机运行本程序。（源程序）

**四、完整的实验结果记录**

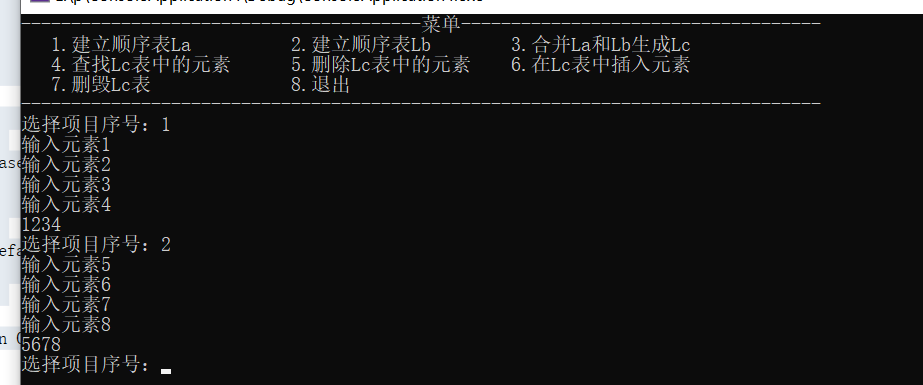
第一组测试数据：

1创建la，lb：

单链表la元素为（1，2，3，4）

单链表lb元素为（5，6, 7, 8）

输出结果:正确（如图1.1）

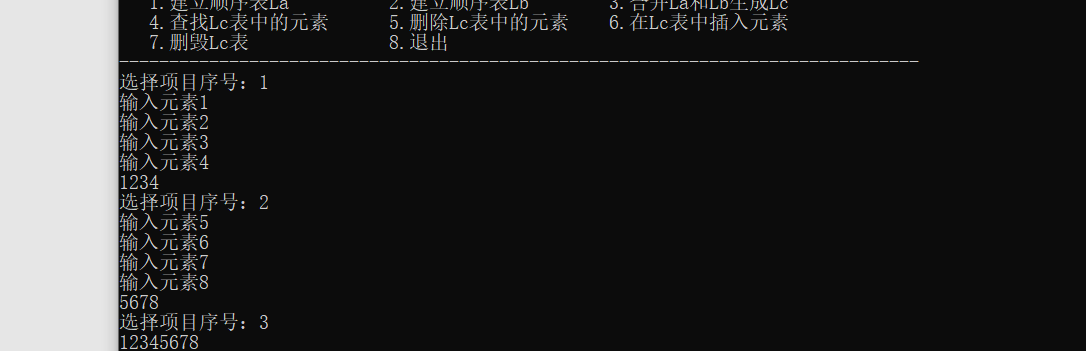


**图1.1 la，lb链表创建的实验结果**

2合并la，lb为lc

结果应该显示为：12345678

输出结果：正确（如图1.2）

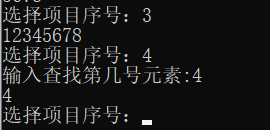


**图1.2 lc链表创建的实验结果**

3查找lc中的第四个元素

结果应该显示为：4

输出结果：正确（如图1.3）

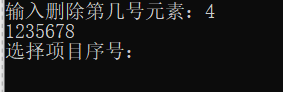


**图1.3 lc链表查找第四号元素创建的实验结果**

4删除lc中第四个元素

结果应该显示为：1235678

输出结果：正确（如图1.4）

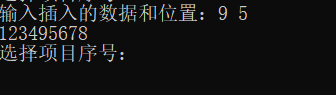


**图1.4 lc链表删除第四号元素创建的实验结果**

5在lc中插入元素

把9插入第五元素的位置应该显示：123495678

输出结果：正确（如图1.5）

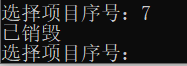


**图1.5 在lc链表中第五元素位置插入9的实验结果**

6毁灭lc

结果应该显示为：删除成功

输出结果：正确（如图1.6）



**图1.6 毁灭lc的实验结果**

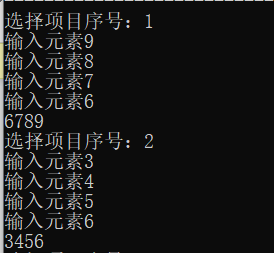
第二组测试数据：

1创建la，lb：

单链表la元素为（9,8,7,6）

单链表lb元素为（3,4,5,6）

输出结果:正确（如图2.1）



**图2.1 la，lb链表创建的实验结果**

2合并la，lb为lc

结果应该显示为：34566789

输出结果：正确（如图2.2）

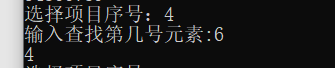


**图2.2 lc链表创建的实验结果**

3查找lc中的第6个元素

结果应该显示为：6

输出结果：错误，（如图2.3）

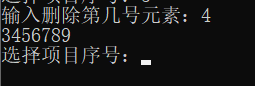


**图2.3 lc链表查找第6号元素创建的实验结果**

4删除lc中第四个元素

结果应该显示为：345789

输出结果：正确（如图2.4）

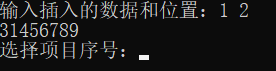


**图2.4 lc链表删除第四号元素创建的实验结果**

5在lc中插入元素

把9插入第五元素的位置应该显示：31456789

输出结果：正确（如图2.5）

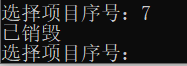


**图2.5 在lc链表中第五元素位置插入9的实验结果**

6毁灭lc

结果应该显示为：删除成功

输出结果：正确（如图2.6）



**图2.6 毁灭lc的实验结果**

**五、实验结果分析**

本实验中基本都符合要求，但是第二次试验中查找元素出错了

**六、实验总结**

这是第一次把函数分开写，感觉有点难，还有出错的时候也不是很容易的找出来，百度搜索也查不到，向舍友进行咨询，最终换了一种思路，最终才解决掉。

**七、实验评价**（教师）