**东莞城市学院**

**实 验 报 告**

**课程名称：** 数据结构

**实验名称：** 单链表的基本操作

**学 院：** 计算机与信息学院

**姓 名：** 张宇航

**学 号：** 202035020145

**专业班级：** 2020级华为创新班

**指导教师：** 彭 刚

2021 **年** 09 **月** 24 **日**

**实验报告成绩评定表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 评定项目 | 内 容 | 满 分 | 评 分 | 总 分 |
| 实验态度 | 实验认真，态度端正，遵守纪律，出勤情况。 | 20 |  |  |
| 实验过程 | 按实验要求完成各种功能或操作，代码书写规范，注释清晰，设计严谨，运行结果正确。 | 40 |  |
| 报告撰写 | 报告字迹整洁、内容丰富、条理清楚；图、表、文字表达准确规范，上交及时。 | 40 |  |
| 总成绩 | 采用五级分制：优、良、中、及格、不及格 |  | | |
| **评语：**  **指导老师： 年 月 日** | | | | |

实验报告正文

1. **实验目的**

1．掌握单链表的存储特点及其实现。

2．掌握单链表的插入、删除算法及其应用算法的程序实现。

1. **实验内容及结果**

实验内容

1. 编写一个完整的程序，实现单链表的生成、插入、删除、输出等基本操作。
2. 随机产生一组元素，建立一个带头结点的单向链表。
3. 遍历单链表并输出元素的值。
4. 将值为 e 的结点插入到表的第 i 个结点的位置上。
5. 读取单链表中第 i 个结点的数据域到 e。
6. 删除单链表的第 i 个结点。
7. 编写在非递减有序单链表中插入一个元素使链表结点仍有序的函数，并利用该函数建
8. 立一个非递减有序单链表。
9. 利用算法 6 建立两个非递减有序单链表，然后用算法 7 合并成一个非递减有序链表。

在主函数中分别调试上述算法。

1. 源程序

(可编辑)

/\* Created by 张宇航.

 \* NAME:LinkList

 \* DATE:2021/9/23

 \* MyWeb:zhang666zyh.github.io

 \* Talk is cheap,show me the code!!!

 \*/

*#include* <stdio.h>

*#include* <stdlib.h>

*#define* OK 1

*#define* ERROR 0

*typedef* *int* Status;

*typedef* *int* ElemType;

*typedef* *struct* LNode {

    ElemType elem;

*struct* LNode *\**next;

}LNode, *\**LinkList;

//初始化单链表（带头结点）

Status initList(LinkList *&*L) {

    L = *new* LNode;

    L->next = *NULL*;

*return* OK;

}

//在 L 中第 i 个元素之前插入数据元素 e

Status ListInsert\_L(LinkList *&*L,*int* i,ElemType e){

    LNode\* p,\*s;

    p = L;

*int* j = 0;

*while*(p && j < (i - 1)){

        p = p->next;

        ++j;

    }

*if*(!p || j > (i - 1)){

        printf("ERROR!!!\n");

    }

    s = *new* LNode;

    s->elem = e;

    s->next = p->next;

    p->next = s;

*return* OK;

}

// 遍历并输出全部元素

*void* display(LinkList L) {

    LinkList temp = L;//将 temp 指针重新指向头结点

*while* (temp->next) {

        temp = temp->next;

        printf("%d ", temp->elem);

    }

    printf("\n");

}

//将线性表 L 中第 i 个数据元素删除

Status ListDelete\_L(LinkList *&*L,*int* i,ElemType *&*e){

    LNode\* p = L;

    LNode\* q = *new* LNode;

*int* j = 0;

*while*(p && j < (i - 1)){

        p = p->next;

        ++j;

    }

*if*(!p || j > (i - 1)){

        printf("Exception!!!\n");

*return* ERROR;

    }

    q = p->next;

    p->next = q->next;

    delete q;

*return* OK;

}

// 取值，读取单链表中第 i 个元素的数据值

Status GetElem\_L(LinkList L,*int* i,ElemType *&*e){

    LinkList p = L->next;

*int* j = 1;

*while*(p && j < i){

        p = p->next;

        ++j;

    }

*if*(!p || j > i){

        printf("Exception!!!\n");

*return* ERROR;

    }

    e = p->elem;

*return* OK;

}

//在有序单链表中插入元素，链表仍然有序

Status insertLinkOrder(LinkList *&*L,ElemType e){

    LNode\* pre,\*p,\*s;

    pre = L,p = L->next;

    s = *new* LNode;

    s->elem = e;

    s->next = *NULL*;

*while*(p != *NULL* && p->elem < e){

        pre = p;

        p = p->next;

    }

    s->next = pre->next;

    pre->next = s;

*return* OK;

}

//两个非递减有序单链表 La 和 Lb 合并成一个非递增有序链表 Lc

*void* MergeOrderList (LinkList *&*La, LinkList *&*Lb, LinkList *&*Lc){

    LNode\* p = La->next;

    LNode\* q = Lb->next;

    LNode\* r;

    Lc = La;

    Lc->next = *NULL*;

    free(Lb);

    r = Lc;

*while*(p != *NULL* && q != *NULL*){

*if*(p->elem <= q->elem){

            r->next = p;

            p = p->next;

            r = r->next;

        }*else*{

            r->next = q;

            q = q->next;

            r = r->next;

        }

    }

    r->next = *NULL*;

*if*(p != *NULL*){

        r->next = p;

    }

*if*(q != *NULL*){

        r->next = q;

    }

}

*int* main() {

    LinkList L = *NULL*;

    printf("初始化链表为：\n");

    initList(L);

*for* (*int* i=1;i<=10;i++)

        ListInsert\_L(L,i,rand()%100+1);

    display(L);

    printf("在第 5 的位置插入元素 120：\n");

    ListInsert\_L(L, 5, 120);

    display(L);

    printf("第 6 个元素的值为：");

    ElemType val;

*if* (GetElem\_L(L, 6,val)== OK) {

        printf("%d\n", val);

    }

*else* {

        printf("没有该元素");

    }

    ElemType delElem;

    printf("删除第 3 个元素:\n");

*if* (ListDelete\_L(L, 3, delElem)==OK)

        display(L);

*else*

    printf("删除元素失败");

    printf("生成有序链表 La 为：\n");

    LinkList La = *NULL*;

    initList(La);

    srand(2);

*for* (*int* i=1;i<10;i++){

        insertLinkOrder(La,rand()%100+1);

    }

    display(La);

    printf("生成有序链表 Lb 为：\n");

    LinkList Lb = *NULL*;

    initList(Lb);

    srand(10);

*for* (*int* i=1;i<10;i++){

        insertLinkOrder(Lb,rand()%100+1);

    }

    display(Lb);

    printf("合并后的序链表 Lc 为：\n");

    LinkList Lc = *NULL*;

    initList(Lc);

    MergeOrderList(La,Lb,Lc);

    display(Lc);

    system("pause");

*return* 0;

}

1. 上机实验截图

PS:这里通过dos命令运行exe文件,因为如果使用编译器直接运行中文会乱码,所以只 能先将当前cmd.exe中的编码改成UTF-8再执行LinkList.exe文件,并不是拿过别 人编译过的可执行文件应付作业.



1. **总结及体会**

（本次实验遇到的主要问题及解决办法，实验体会及收获。）

本次作业是实现单链表的基本操作,没什么体会,巩固基础的题目,主要帮助我学习 C++的语法.