石 家 庄 铁 道 大 学

**实 验 报 告**

**实习一 线性表应用**

题 目： 约瑟夫退圈

班 级： 信1901 –4

姓 名： 闫竞存

学 号：20194127

日 期：2020.10.28

**1. 实验题目**

约瑟夫（Joeph）问题的一种描述是：编号为 1,2,…,n 的 n 个人按顺时针方向围坐一圈，每人持有一个密码（正整数）。一开始任选一个正整数作为报数上限值 m，从第一个人开始按顺时针方向自 1 开始顺序报数，报到 m 时停止报数。报 m 的人出列，将他的密码作为新的 m 值，从他在顺时针方向上的下一个人开始重新从 1 报数，如此下去，直至所有人全部出列为止。试设计一个程序求出出列顺序。

**2. 需求分析**

任务：利用单向循环链表存储结构模拟约瑟夫退圈过程，按照出列的顺序印出各人的编号。

（1）程序输入的为不超过int范围的正整数，分别是

（2）输出的形式

（3）程序所能达到的功能

（4）测试数据

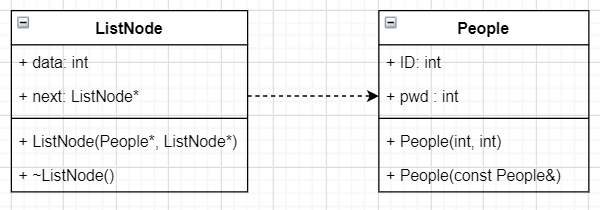
m 的初值为 20；

密码：3，1，7，2，4，8，4

（正确的结果应为 6，1，4，7，2，3，5）

**3. 概要设计**

本程序中一共包含了两个类：



**4. 部分代码详细设计**

主程序main() 算法：

int main() {

    int flag = 0; // 输入正确检查

    int n = 0, m = 0;

    cout << "----------------------\n       约瑟夫退圈\n----------------------" << endl;

    do{

        cout << "指定人数 n : ";

        cin >> n;

        if (n <= 0) {

            flag = 1;

            cout << "人数应该大于 0, 请重新输入" << endl;

        } else flag = 0;

    } while (flag);

    flag = 0;

    do {

        cout << "指定初始上报的上限值 m : ";

        cin >> m;

        if (m < 0) {

            flag = 1;

            cout << "初始上报的上限值应该大于等于0，请重新输入" << endl;

        } else flag = 0;

    } while (flag);

    flag = 0;

    cout << "输入这些人的密码 ：" << endl;

    int temp;

    ListNode\* t = NULL; //临时节点

    for (int i = 1; i <= n; i++) { //数值录入

        do {

            cin >> temp;

            if (temp < 0) {

                flag = 1;

                cout << "密码应该大于等于0，请重新输入" << endl;

            } else flag = 0;

        } while (flag);

        if (i == 1) {

            head = new ListNode(new People(i, temp), NULL);

            t = head;

        } else if (i == n) {

            t->next = new ListNode(new People(i, temp), head);

            t = t->next;

        } else {

            t->next = new ListNode(new People(i, temp), NULL);

            t = t->next;

        }

    }

    int\* outOrder = new int[n];

    int cntOutNumber = 0;

    int totalNumber = n;

    cout << "-------------------------" << endl;

    while (t && t->next != t) {

        for (int i = 1; i < m; i++) t = t->next;

        ListNode\* p = t->next;

        if (p) {

            m = p->data->pwd;

            outOrder[cntOutNumber++] = p->data->ID; // 记录出列顺序

            cout << p->data->ID << " 出列，当前队列: ";

            t->next = p->next;

            // 输出当前队列的值

            int temp = --totalNumber;

            ListNode\* pt = t;

            while (temp--) {

                cout << pt->data->ID << ' ';

                pt = pt->next;

            }

            cout << endl;

            delete p;

        } else break;

    }

    if (t) cout << t->data->ID << " 出列" << "，队列已空" << endl;

    outOrder[cntOutNumber++] = t->data->ID; //最后一个数字入队

    cout << "-------------------------\n出列顺序为：";

    for (int i = 0; i < n; i++) cout << outOrder[i] << " ";

    cout << "\n-------------------------" << endl;

    return 0;

}

**5. 遇到的问题以及解决方案**

**遇到的问题：**在最后一个元素出列的时候，由于我的出列算法为: “将要删除的元素的前一个结点指向其后面的结点”，但是我在设计链表的时候没有设计头指针，导致了最后一个元素无法在循环中删除。

**解决方案:** 在循环之外删除这个结点

**6. 用户使用说明**

界面输出【指定人数 n (n > 0):】 此时用户输入合法的n表示参与约瑟夫退圈的人数，并敲击回车

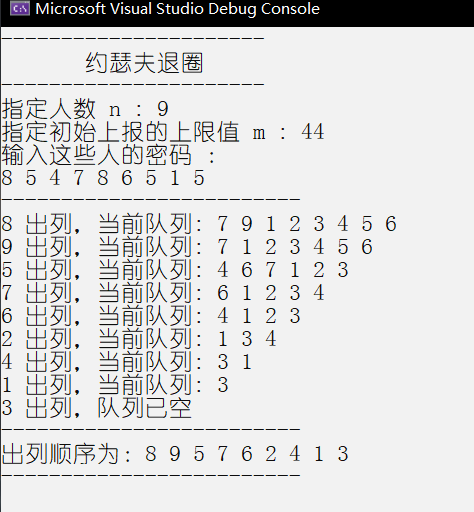
界面输出【指定初始上报的上限值 m (m >= 0):】此时用户输入合法的m值表示初始上报的上限值，并敲击回车

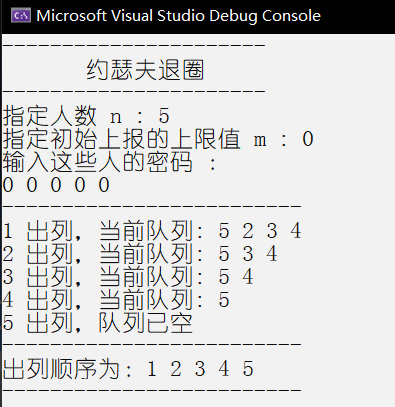
界面输出【输入这些人的密码 (密码 > =0)：】此时用户输入n个合法的正整数值，并以空格、tab或者回车分隔，输入完毕之后敲击回车。

此时如果用户操作无误，则界面会输出【出列顺序：】和约瑟夫退圈的顺序，即算法的运行结果。

**7. 测试结果**







**容错测试**

