《数据结构》实验报告

题目：编制一个演示约瑟夫环的程序

姓名：冯兴

学号：15051416

班级：计科三班

指导教师：徐翀

1. 需求分析
2. 本演示程序中，链表结点的数据类型限定为整形数字，从理论中不能出现数字0或者负数，若出现，与数字1效果相同。输出的运算结果是一串数字
3. 演示程序以用户和计算机的对话方式执行，即在计算机终端上显示“提示信息”之后，由用户在键盘输入演示程序中规定的运算命令；相应的输入数据和运算结果显示在后。
4. 程序执行的命令：

(1).设置初始密码

(2)设置人数

(3)设置密码

(4)构造循环链表

4.测试数据

(1).m的初值为20；n=7,7个人的密码依次为:3,1,7,2,4,8,4

出列顺序:6,1,4,7,2,3,5

(2). m的初值为1；n=6,6个人的密码依次为: 9,8,6,4,2,5

出列顺序: 4,3,1,2,5,6

(3. m的初值为15;n=8,8个人的密码依次为: 4,5,9,6,2,8,7,1

出列顺序: 7,8,2,3,5,4,6,1

二、概要设计

为实现上述功能，需要一个循环链表，为此需要一个抽象数据类型

1. 循环链表的抽象数据类型定义为：

ADT CirculList{

数据类型：D={ai｜ai∈Int,i=1,2,…..,n,n>=0}

数据关系：R1={<ai-1,ai>｜ai-1,ai ∈D,i=2,….,n}

基本操作：

CreateList(n,&L)

操作结果：构造一个长度为n的循环链表L。

DeleteNode(p,& e,&x)

初始条件：p不为空

操作结果：删除p结点的下一个结点，并用e返回密码，x返回位序。

DestroyList(L)

初始条件：循环链表L已经存在。

操作结果：销毁表L。

}ADT CirculList

1. 本程序包含四个模块：
2. 主程序模块：

void main(){

初始化;

do{

接受命令;

处理命令;

}while(“命令”=”退出”);

}

1. 循环链表单元模块——实现循环表的抽象数据类型;
2. 结点结构单元模块——定义循环表的结点结构。
3. 各模块之间的调用关系如下：

主程序模块

循环链表模块

结点结构单元模块

* 1. 详细设计

1. 元素类型、结点类型和指针类型

struct Node

{

int data; //密码

int x; //位序

Node \*next;

};

1. 循环链表基本操作设置如下：

void CreateList(int n,CirculList &L) //创建n个结点的循环链表

{

int x;

L = new Node();

L->x = 1;

cout << "输入密码" << endl;

cin >> x;

L->data = x;

Node \*pl = L, \*p;

for (int i = 0; i < n - 1; i++)

{

p = new Node();

cin >> x;

p->x = i + 2;

p->data = x;

pl->next = p;

pl = p;

}

pl->next = L;

}

void DeleteNode(Node \*p, int &e,int &x) //删除结点

{

Node \*q = p->next;

e = q->data;

x = q->x;

p->next = q->next;

delete(q);

}

void DestoryList(CirculList &L,int n) //循环链表的销毁

{

Node \*p = L;

while (n--&&L)

{

p = L->next;

delete(L);

L = p;

}

}

1. 主函数以及其他函数

void run(CirculList &L, int &m, int n) //执行约瑟夫环步骤

{

Node \*p = L;

while (!(p->next == L)) //p指针从起点向前移一位，防止第一个报数就是1

{

p = p->next;

}

for (int i = 0; i < n; i++){

int j;

for (int i = 0; i < m - 1; i++) //移动到要删除的前一个节点

{

p = p->next;

}

DeleteNode(p, m, j);

cout << j << endl;

p = p->next; //删除之后移动到后一位

}

cout << "出列完成";

L = NULL; //L变为野指针，所以要赋值为空

}

void main()

{

int m,n;

cout << "请输入初始密码：";

cin >> m;

cout << "请输入人数：";

cin >> n;

CirculList cl;

CreateList(n,cl);

cout << "开始出列" << endl;

run(cl, m, n);

}

1. 函数的调用关系

main

CreateList run DestoryList

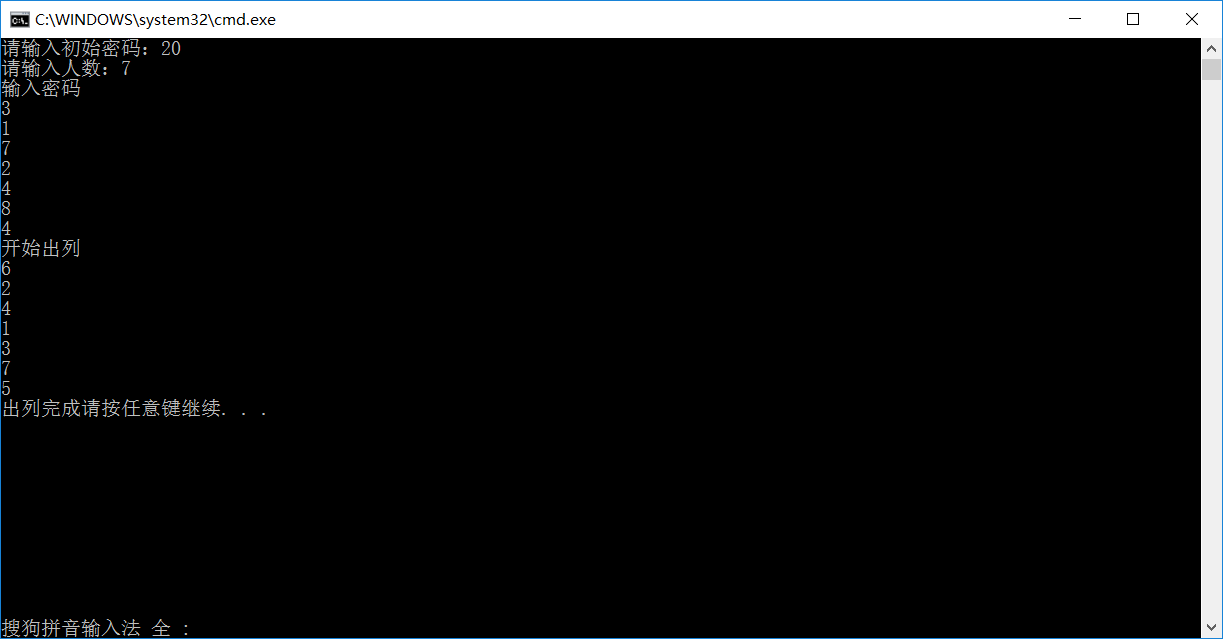
MakeNode DeleteNode

* 1. 调试分析

1. 本程序的核心算法其实为run,刚开始因为分析不足，早期版本没有考虑初始密码是1的情况，导致运行错误。所以开始运行时先把指针先往前移一位。
2. 由于采用的是循环链表，没有设置终点，当约瑟夫(run)执行完毕，会导致L变为野指针，导致后来执行销毁链表时发生错误。所以执行run函数后把L设置为空指针，并在DestoryList中增加空指针的判断。
3. 有时候忽略细节，总是少打一个分号，中英文弄错，变量名用错，导致调试浪费了一些时间。让我明白了细节决定成败。
4. 初始版本并没有设计循环链表这一抽象类型，导致结构混乱，层次不分明。
5. 算法的时空分析

由于采用的是简单的链表结构，操作基本，所以算法时间度比较合理。如果想要进一步减小复杂度，则要找到其中的规律，求出数学公式。但是本例要求模拟约瑟夫的过程，故不做考虑。其中基本操作DeleteNode时间复杂度为O(n),则run的时间复杂度为O(n\*m),基本操作DestoryList算法时间复杂度也为O(n).

1. 本实习作业采用数据抽象的程序设计方法，讲程序划分为三个结构：主控模块，循环链表，元素结点，使得修改更新时思路更清晰，实现调试顺利，各模块具有较好的重用性，我自己得到了一次良好的程序设计训练，让我对链表这种数据结构有了更深的了解。
   1. 用户手册
2. 本程序运行环境为Windows操作系统，执行文件为yuesefu.exe
3. 界面如下：



* 1. 测试数据

输入初始密码：键入“20”

设置人数：键入“7”

输入密码：键入”3,1,7,2,4,8,4”

输出：6,2,4,1,3,7,5

输入初始密码：键入“10”

设置人数：键入“6”

输入密码：键入”9,8,6,4,2,5”

输出：4,3,1,2,5,6

输入初始密码：键入“15”

设置人数：键入“8”

输入密码：键入”4,5,9,6,2,8,7,1”

输出：7,8,2,3,5,4,6,1

* 1. 附录

源程序文件名清单：

yuesefu.cpp //源代码文件…