****

**《数据结构》**

**实验报告**

实验名称：约瑟夫环输出元素

姓名：龙金河

学号：1825122033

专业：软件工程

任课教师：罗伟

计算机科学与技术学院软件工程专业二班2018级

2019年9月26日

1.需求分析：

1.问题描述：约瑟夫问题的一种描述是：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的值，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。

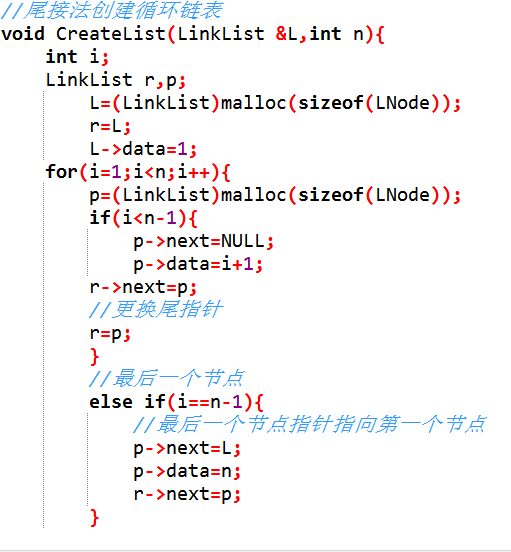
２.基本要求：利用单向循环链表存储结构模拟此过程，按照出列的顺序印出各人的编号。

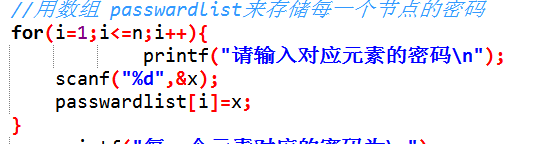
３.测试数据：m的初值为20；n=7，7个人的密码依次为：3,1,7,2,4,8,4,首先m值为6（正确的出列顺序应为6,1,4,7,2,3,5）。

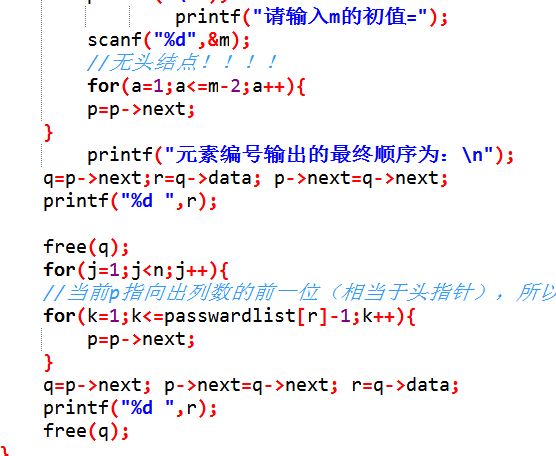
４.实现提示：程序运行后，首先要求用户指定初始报数上限值，然后读取各人的密码。可设n≤30。此题所用的循环链表中不需要“头结点”，请注意空表和非空表的界限。

1. 项目设计：

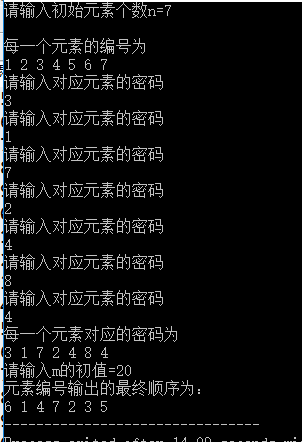
实验主要是通过构造循环链表来实现的，从链表开头的节点依次读取报数，报到密码的那个节点被删除，并输出其所存储的编号，依次类推，直到所有的节点都被删除，所有编号全部输出。本程序主要是两个方法，一个是用尾插法构造循环链表，另一个实现约瑟夫环输出。我是利用节点的data来存储编号，第一个节点对应的data就是编号1，以此类推，每个节点存储一个编号。最后一个节点的指针指向第一个节点。具体代码如下图

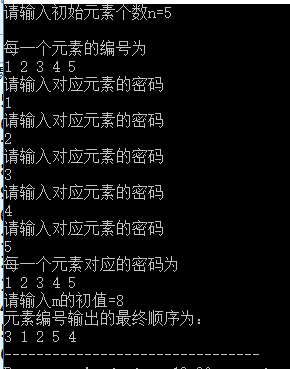


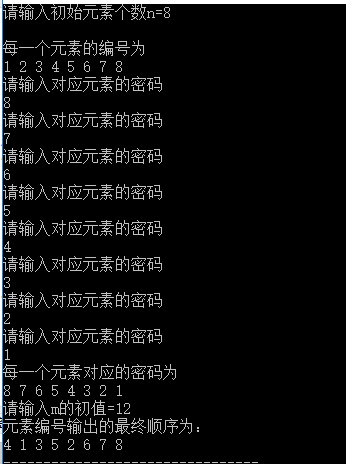
然后我是利用了一个数组来存储每一个编号所对应的密码，如下图

最后是报数删除节点过程，先得到第一个m的值，开始报数找到第一个要删除的节点并删除，且记录并输出其编号，利用其编号所对应的密码再次报数删除下一个节点，做循环依次类推，具体代码如下图

1. 测试结果:







经过验算，三组数据均正确。

4.总结分析

这个实验练习了链表的使用，做这个实验还是花了较长的时间的，第一个难点就是循环链表的构造，第二个难点当然是循环删除节点，中间也是出现过许多的错误，经过不断的测试与改进最终实现了约瑟夫环。

5.源代码

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

//定义符号常量

#define LIST\_INIT\_SIZE 50

#define LISTINCREMENT 10

#define OK 1

#define ERROR 0

#define OVERFLOW -2

//定义元素类型

typedef int ElemType;

typedef struct LNode{

ElemType data;

struct LNode \*next;

}LNode,\*LinkList;

//尾接法创建循环链表

void CreateList(LinkList &L,int n){

int i;

LinkList r,p;

L=(LinkList)malloc(sizeof(LNode));

r=L;

L->data=1;

for(i=1;i<n;i++){

p=(LinkList)malloc(sizeof(LNode));

if(i<n-1){

p->next=NULL;

p->data=i+1;

r->next=p;

//更换尾指针

r=p;

}

//最后一个节点

else if(i==n-1){

//最后一个节点指针指向第一个节点

p->next=L;

p->data=n;

r->next=p;

}

}

}

//输出节点的编号

void PrintList(LinkList &L,int n)

{

LinkList p;p=L;

int i;

for(i=1;i<=n;i++){

printf("%d ",p->data);

p=p->next;

}

}

void Run(LinkList &L,int n){

LinkList p,q;p=L;

int i,j,m,k,r,x,a,passwardlist[n];

//用数组 passwardlist来存储每一个节点的密码

for(i=1;i<=n;i++){

printf("请输入对应元素的密码\n");

scanf("%d",&x);

passwardlist[i]=x;

}

printf("每一个元素对应的密码为\n");

for(i=1;i<=n;i++){

printf("%d ",passwardlist[i]);

}

printf("\n");

printf("请输入m的初值=");

scanf("%d",&m);

//无头结点！！！！

for(a=1;a<=m-2;a++){

p=p->next;

}

printf("元素编号输出的最终顺序为：\n");

q=p->next;r=q->data; p->next=q->next;

printf("%d ",r);

free(q);

for(j=1;j<n;j++){

//当前p指向出列数的前一位（相当于头指针），所以循环次数和前面不同

for(k=1;k<=passwardlist[r]-1;k++){

p=p->next;

}

q=p->next; p->next=q->next; r=q->data;

printf("%d ",r);

free(q);

}

}

int main(){

int n;

printf("请输入初始元素个数n=");

scanf("%d",&n);

printf("\n");

LNode \*pl = (LNode \*)malloc(sizeof(LNode));

CreateList(pl,n);

printf("每一个元素的编号为\n");

PrintList(pl,n);

printf("\n");

Run(pl,n);

}