**实习报告**

题目：约瑟夫环问题

班级：1503041 姓名：孙天艺 学号：15030410038 完成日期：2016.9.30

1. **需求分析**
2. 本演示程序中，人数n应为任意的，首先应输入一个赋值给初始报数上限m，程序应能自动保存出列人的序号和将出列的人所持的密码赋给m，而后再次作为报数上限，如此循环，直至所有人都出列为止。
3. 演示程序以用户和计算机的对话方式执行，即在计算机终端显示“提示信息”之后，由用户在键盘上输入演示程序中规定的运算命令；相应的输入数据（滤去输入中非法字符）和运算结果显示在其后。
4. 程序执行的命令包括：

（1）构造循环单链表；（2）输入数据；（3）执行报数，储存出列人的序号，删除出列人的信息以及把出列人的密码赋给m并输出出列人的序号；（4）结束。

1. 测试数据：

n=7，7个人的密码依次为：3，1，7，2，4，8，4，首先m值为6，则正确的出列顺序为6，1，4，7，2，3，5。

1. **概要设计**

整个程序通过循环单链表实现。

用循环逐个计算要删除的节点，并且读取该节点的信息密码password，然后以此password作为下一次循环用到的密码，继续计数。

1. **详细设计**

#include<stdio.h>

#include<stdlib.h>

#define MAX\_NODE\_NUM 30

#define TRUE 1

#define FALSE 0

typedef struct NodeType

{

int number;

int password;

struct NodeType \*next;

}NodeType;

/\* 创建单向循环链表 \*/

static void CreaList(NodeType \*\*, const int);

/\* 运行"约瑟夫环"问题 \*/

static void StatGame(NodeType \*\*, int);

/\* 得到一个结点 \*/

static NodeType \*GetNode(const int, const int);

/\* 测试链表是否为空, 空为TRUE，非空为FALSE \*/

static unsigned EmptyList(const NodeType \*);

int main(void)

{

int n, m;

NodeType \*pHead=NULL;

while(1)

{

printf("输入总的人数 n(<=%d):",MAX\_NODE\_NUM);

scanf("%d",&n);

printf("初始循环的密码为:");

scanf("%d",&m);

if(n>MAX\_NODE\_NUM)

{

printf("数字太大，请重新输入!\n");

continue;

}

else break;

}

CreaList(&pHead,n);

printf("\n每个结点退出的序列号和密码\n");

StatGame(&pHead,m);

return 0;

}

static void CreaList(NodeType \*\*ppHead, const int n)

{

int i,iCipher;

NodeType \*pNew, \*pCur;

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

{

printf("第%d个人的密码为:",i);

scanf("%d", &iCipher);

pNew=GetNode(i,iCipher);

if(\*ppHead==NULL)

{

\*ppHead=pCur=pNew;

pCur->next=\*ppHead;

}

else

{

pNew->next=pCur->next;

pCur->next=pNew;

pCur=pNew;

}

}

}

static void StatGame(NodeType \*\*ppHead, int iCipher)

{

int iCounter, iFlag=1,i=1;

NodeType \*pPrv, \*pCur, \*pDel;

pPrv=pCur=\*ppHead;

while(pPrv->next!=\*ppHead) pPrv=pPrv->next;

while(iFlag)

{

for(iCounter=1;iCounter<iCipher;iCounter++)

{

pPrv=pCur;

pCur=pCur->next;

}

if(pPrv==pCur) iFlag=0;

pDel=pCur;

pPrv->next=pCur->next;

pCur=pCur->next;

iCipher=pDel->password;

printf("第%d个退出的是序列号为%d的人,其密码为:%d\n",i, pDel->number,pDel->password);

free(pDel);

++i;

}

\*ppHead=NULL;

}

static NodeType \*GetNode(const int iId,const int iCipher)

{

NodeType \*pNew;

pNew=(NodeType \*)malloc(sizeof(NodeType));

if(!pNew)

{

printf("错误，内存不足!\n");

exit(-1);

}

pNew->number=iId;

pNew->password=iCipher;

pNew->next=NULL;

return pNew;

}

static unsigned EmptyList(const NodeType \*pHead)

{

if(!pHead)

{

printf("列表为空!\n");

return TRUE;

}

return FALSE;

}

1. **调试分析**
2. 由于对循环链表算法的理解不够透彻，在建造循环链表的前期，指针指向混乱，导致程序无法通过编译。
3. 刚开始时曾忽略一些变量参数的标识“&”，使调试程序费时不少，今后应重视确定参数的变量和赋值属性的区分和标识。
4. 在循环链表的使用中，由于指针占主导地位，理解节点的前后关系有点复杂，偶尔会出现小错误。以后吸取教训，在复杂循环时，最好通过图形的辅助，以使编程过程思路清晰。
5. 本实习作业采用数据抽象的程序设计方法，将程序分成三个层次结构：节点和指针、循环链表和主程序模块，使得设计时思路清晰，实现时调试顺利，各模块具有较好的可重用性，确实得到了一次良好的程序设计训练。
6. **测试结果**

