## 实习报告

题目:利用单向循环链表存储结构模拟约瑟夫环过程，按照出列顺序引出各人的编号。

班级：电信1809班 姓名：张浩 学号：U201813496 完成日期：2019.3.25

1. 需求分析：
2. 本演示程序，将环中的每个元素类比为循环链表的每一个结点。演示程序以用户

和计算机的对话方式执行，在计算机终端上显示提示信息之后，由用户在键盘上输入演示程序中规定的运算命令，相应地输入总共的结点数n，与初始的报数上限m，

运算结果显示在后。

1. 程序执行的命令包括：

1）、创建单向循环单链表，并返回循环链表的头指针。

2）、模拟约瑟夫环操作，并提取待移除结点的数据域，重新设定头指针，循环操作。

二、概要设计：

为实现上述程序功能，应以单向循环链表表示约瑟夫环。

1. 详细设计：

#include <stdio.h>

#include <stdlib.h>

typedef struct Node //定义结点{

int code;

int num;

struct Node\* next;

}Node,\*PNode;

PNode CreateNode(int n) //创立单向循环链表，并返回链表的头结点

{

PNode head=NULL,p=NULL,q;

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

{

PNode p=(Node\*)malloc(sizeof(Node));

printf("Input the %d code\t",i);

scanf("%d",&p->code);

p->num=i;

if(i==1) //创立头结点

{

q=p;

head=q;

}

if(i>1&&i<n) //创立中间的结点

{

q->next=p;

q=p;

}

if(i==n) //尾结点连接头结点，形成循环

{

q->next=p;

p->next=head;

}

}

return head;

}

int Joseph(PNode head,int m,int n)

{

PNode p=head;

int i=1,k=m,j=n; //i为计数器，k为报数上限，j为环中的总人数

while(j>0)

{

if(p->next==p)

{

printf("%d\t",p->num);

return 0;

}

if(i==k)

{

PNode s=p;

p=head;

while(p->next!=s) p=p->next; //寻找待删除结点的前一个结点

p->next=s->next;

head=s->next; //转移头指针，防止循环链表丢失

k=s->code;

i=0;

printf("%d\t",s->num);

free(s); //释放该结点的内存，相当如人数减一

j--;

}

i++;

p=p->next;

}

}

void main()

{

int m,n;

printf("Input the number of persons\n");

scanf("%d",&n);

printf("Input the m\n");

scanf("%d",&m);

PNode head=CreateNode(n);

Joseph(head,m,n);

}

1. 调试分析：
2. 由于对单向循环链表认识不足，在最初构建循环链表时导致尾结点连接头结点失败。
3. 算法的时空分析：

1）、在建立循环链表时，时间复杂度为O(n)；

2）、在移除元素时，时间复杂度为O(j);在寻找待释放结点时，时间复杂度为O(n);

五、测试结果：

1）、输入m=20，n=7，密码分别为3 1 7 2 4 8 4

输出结果：6 1 4 7 2 3 5

2）、输入m=1，n=5，密码分别为1 2 3 4 5

输出结果：1 2 4 3 5

3）、输入m=1,n=5,密码分别为：2 1 2 1 1

输出结果：1 3 5 2 4