# 实验报告

题目：线性表及其应用——约瑟夫环

班级：网络工程 姓名：王硕 学号：1625131037

一、需求分析

1.本程序中，输入的数据为整数，输入数据的个数由主程序控制（本程序中为7）

2.演示程序以用户和计算机的对话方式进行，即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；相应的输入数据和运算结果显示在后。

3.程序执行的命令包括：1)创建循环链表，2)删除链表中的结点,3）按照实验1.1.2的要求执行算法并输出要求输出的数据

4.测试数据

m=20，n=7,输入的数据依次为3,1,7,2,4,8,4

二、概要设计

为实现上述程序功能，需要循环链表抽象数据类型。

**ADT** sqlist{

数据对象：D={ai| ai∈I，i=1,2,3,4…n，n≥0}

数据关系：{< ai-1, ai>| ai-1, ai ∈I}

基本操作：

\*creat(int n)

创建一个长度为n的不含头结点的循环链表，并返回其地址

\*del\_behind(linklist \*l)

删除链表\*l后一个结点，并返回新链表的地址

三、详细设计

1.链表的结构体

struct linklist

{

int code; //每个人的密码

int num; //每个人的位置

linklist \*next;

};

2.实现功能的函数

linklist \*creat(int n) //创建一个无头结点的循环链表

{

linklist \*l,\*s,\*r;

l=new linklist;

cin>>l->code;

l->num=1;

l->next=NULL;

r=l;

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

{

s=new linklist;

cin>>s->code;

s->num=i+1;

s->next=NULL;

r->next=s;

r=s;

}

r->next=l;

return l;

}

linklist \*del\_behind(linklist \*l) //删除该链表结点后的一个结点

{

linklist \*r;

r=l->next->next;

delete l->next;

l->next=r;

l=l->next;

return l;

}

Int main()

{

…

while(l->next!=l) //主程序中实现具体算法的部分

{

for(int i=0;i<n-1;i++) //使得l指向链表中最后一个结点，使下一个for循环的l->next指向当前链表的第一个结点

l=l->next;

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

l=l->next;

m=(l->next)->code;

cout<<l->next->num<<" ";

l=del\_behind(l);

n--;

}

…

}

四、调试分析

本次实验创建的是是一个无头结点的循环链表。在调试过程中遇到了很多诸如首次使用链表无法指向第一个结点的问题。后来证实无头结点的链表操作难度确实比有头结点链表的大。但本次实验仍保留了最初的设计，且功能达到设计要求。

## 附程序源代码：

#include<iostream>

using namespace std;

struct linklist

{

int code;

int num;

linklist \*next;

};

linklist \*creat(int n) //创建一个无头结点的循环链表

{

linklist \*l,\*s,\*r;

l=new linklist;

cin>>l->code;

l->num=1;

l->next=NULL;

r=l;

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

{

s=new linklist;

cin>>s->code;

s->num=i+1;

s->next=NULL;

r->next=s;

r=s;

}

r->next=l;

return l;

}

linklist \*del\_behind(linklist \*l) //删除该链表结点后的一个结点

{

linklist \*r;

r=l->next->next;

delete l->next;

l->next=r;

l=l->next;

return l;

}

int main()

{

linklist \*l,\*t;

int m=20,n=7;

l=new linklist;

l=creat(n);

t=l;

while(l->next!=l)

{

for(int i=0;i<n-1;i++) //使得l指向链表中最后一个结点，使下一个for循环的l->next指向当前链表的第一个结点

l=l->next;

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

l=l->next;

m=(l->next)->code;

cout<<l->next->num<<" ";

//if(l->next!=l)

//{

/\*linklist \*r;

r=(l->next)->next;

delete l->next;

l->next=r;

l=l->next;\*/

//}

l=del\_behind(l);

n--;

}

cout<<l->num;

cout<<endl;

return 0;

}