**\_\_\_\_2019\_\_\_\_\_\_级\_\_\_\_\_信1901-1\_\_\_\_\_班\_\_\_\_\_2020\_\_年\_\_\_\_11\_\_\_月\_\_13\_\_\_日**

**姓名\_\_\_\_\_李朴凡\_\_\_\_\_\_\_ 学号\_\_\_\_\_\_20194311\_\_\_\_\_\_ 电话\_\_\_\_\_18333135009\_\_\_\_\_\_\_\_**

**1．实验题目**

**约瑟夫环**

[问题描述]

约瑟夫（Joeph）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。

[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。

[测试数据]

m的初值为20；密码：3，1，7，2，4，8，4（正确的结果应为6，1，4，7，2，3，5）。

[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。设n≤30。

[选作内容]

向上述程序中添加在顺序结构上实现的部分。

**2．需求分析**

本演示程序用c++编写，完成单向循环链表的生成，向单向循环链表中插入元素，从单向循环链表中删除元素

输入的形式和输入值的范围：需要输入初始报数上限值m，人数n,n个密码。在所有输入中，元素的值都是整数，n<=30.

② 输出的形式：输出最后出列的人的顺序。

③ 程序所能达到的功能：完成约瑟夫问题

④ 测试数据：

输入m=20,n=7,密码依次为3，1，7，2，4，8，4

**3．概要设计**

1）为了实现上述程序功能，需要定义单链表的抽象数据类型：

typedef struct set{//结点

int password;

int num;

struct set \* next;

}Node;

基本操作：初始化循环链表

phead = NULL;

操作结果：循环链表为空，头指针指向空

基本操作：向循环链表中插入密码

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

{

if(phead == NULL)

{//空

phead = &head;

phead->num = j;//存编号

p0 = phead;

phead->next = phead;//循环

}

else{

p1 = (Node\*)malloc(sizeof(Node));//动态创建一个结构体

p1->next = p0;

phead->next = p1;

phead = p1;

phead->num = j;//编号

}

cin>>phead->password;

j++;//更新编号

}

基本操作：找到需要出列的人

while(p0){

if(p0->num == 0){//不存在了

p0 = p0->next;

continue; }

if(k == m && p0->num!= 0){//有人出列

a[i++] = p0->num;//编号

m = p0->password;//更新密码

p0->num = 0;//此人不存在了

k = 0;

}

k++;

if(i == (j - 1)) break;//满了

p0 = p0->next;

}

操作结果：将出列的人密码赋为0

**4．详细设计**

**#include<bits/stdc++.h>**

**using namespace std;**

**typedef struct set{//结点**

**int password;**

**int num;**

**struct set \* next;**

**}Node;**

**Node head,\*phead,\*p0,\*p1;**

**int main ()**

**{ int n,m;**

**int i,j = 1,k=1;**

**int flag = 1;**

**int a[40];**

**phead = NULL;//初始化循环链表**

**cin>>n>>m;**

**for (i = 0;i<n;i++)**

**{**

**if(phead == NULL)**

**{//空**

**phead = &head;**

**phead->num = j;//存编号**

**p0 = phead;**

**phead->next = phead;//循环**

**}**

**else{**

**p1 = (Node\*)malloc(sizeof(Node));//动态创建一个结构体**

**p1->next = p0;**

**phead->next = p1;**

**phead = p1;**

**phead->num = j;//编号**

**}**

**cin>>phead->password;**

**j++;//更新编号**

**}**

**i=0;**

**while(p0){**

**if(p0->num == 0){//不存在了**

**p0 = p0->next;**

**continue; }**

**if(k == m && p0->num!= 0){//有人出列**

**a[i++] = p0->num;//编号**

**m = p0->password;//更新密码**

**p0->num = 0;//此人不存在了**

**k = 0;**

**}**

**k++;**

**if(i == (j - 1)) break;//满了**

**p0 = p0->next;**

**}**

**for(j = 0;j<i;j++) cout<<a[j]<<' ';**

**}**

**5．调试分析**

运行稳定，在建立链表的时候要细心，规范循环结构。对头节点的控制进行了多次的调试。对于数据的储存和删除也进行了多次的修改。

**6．使用说明**

先确定使用的人数范围，然后赋予每个人独自的密码，然后再规定起始读数范围。最后由程序作出最终的排序结果。

**7．测试结果**

