实验报告

题目：解决约瑟夫环问题。

班级：电信1808 姓名：陈佐臣 学号：U201813471 完成日期：2019.3.25

1需求分析：

编号为1，2，……n的n个人按顺时针方向围成一圈，每个人持有一个密码（正整数）。一开始任选一个正整数作为上限值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，从他的顺时针方向下一个人开始重新从1开始报数，如此下去，直到所有人出列为止。

2：概要设计:

首先typedef一个结构体 ，然后建立一个单链表，再将其建成一个循环链表，以此来模拟约瑟夫环。再对链表进行处理。

3：详细设计：

1：typedef一个包含密码，序号以及结构指针的结构体。

2：建立循环单链表。

3：数据输入，输入序号以及对应的密码，初始上限（开始时m的值）

4：找到每次出列的人并记下他的序号，密码，然后把他free掉，直到只剩下一个人为止。

4：调试分析：

问题：对于结点的删除delete不能很好的使用，改为free。

算法的时空分析：时间复杂度为o（n.m）.

经验及体会：学会了循环链表的使用。

5:用户使用说明：

根据提示输入人数i，然后依次输入对应的人的密码，就可得到出列顺序。

7：附录：

#include<stdio.h>

#include<stdlib.h>

typedef struct LNode{

int data;//密码

int order;//序号

struct LNode\*next;

} LNode;

void main()

{

struct LNode\*head;//head为头指针

struct LNode\*p1,\*p2;

int i,j,m,n;

printf("输入人数");//输入人数

scanf("%d",&n);

//为了简化代码,建表的时候使用一个空的Node作为表头.

p1=(struct LNode\*)malloc(sizeof(LNode));

head=p1;

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

{

p1->next=(struct LNode\*)malloc(sizeof(LNode));

printf("输入第i个人的密码:",i);//输入密码

scanf("%d",&p1->next->data);

p1->next->order=i;

p1=p1->next;

}

p1->next=head->next;//将链表首尾相连,删除空表头.

p2=head->next;//此时p2指向链表的头

free (head);

printf("输入上限:");

scanf("%d",&m);

printf("出列顺序:");

while (p2->next!=p2)//人数大于1

{

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

{

p1=p2;

p2=p2->next;

}

printf("%d",p2->order);//第m个人出列

m=p2->data;

p1->next=p2->next;//删除第m个结点

free (p2);

p2=p1->next;

}

printf ("%d",p2->order);//输出队列最后一个人

free (p2);//清空队列的最后一个

}