# 实验五

## 1. 问题定义及需求分析

### 1.1 问题描述

##### Josephus排列问题定义如下：假设n个竞赛者排成一个环形。给定一个正整数m≤n，从第1人开始，沿环计数，第m人出列。这个过程一直进行到所有人都出列为止。最后出列者为优胜者。全部出列次序定义了1，2，…n的一个排列。称为（n，m）Josephus排列。例如，（7，3）Josephus排列为3,6,2,7,5,1,4。

### 1.2 实验要求

##### 设计求解Josephus排列问题程序。

###### (1)采用顺序表、单链表或双向循环链表等数据结构。

###### (2)采用双向循环链表实现Josephus排列问题，且奇数次顺时针轮转，偶数次逆时针轮转。

###### (3)推荐采用静态链表实现Josephus排列问题。

###### (4)实现STL的双向循环链表类。

###### (5)双向循环链表类的简单应用。

### 1.3 输入

#### 1.3.1 输入的形式

##### 两个整数，n和m，定义同题中

#### 1.3.2 输入值的范围

### 1.4 输出形式

##### 一个由n个数字组成的序列，按照出列先后排序。

### 1.5 程序的功能

##### 输出Josephus排列，且若m为奇数则顺时针旋转，若m为偶数则逆时针旋转。

### 1.6 测试数据（样例）

#### 1.6.1 输入

##### 待补充

#### 1.6.2 输出

##### 待补充

## 2.概要设计

### 2.1 抽象数据类型定义（抄书上的ADT，只抄用到的函数就行）

### 2.2 主程序流程

##### (0)初始化变量

##### (1)在屏幕上展示问题情境

##### (2)处理输入数据

##### (3)把所有人都加入循环链表

##### (4)判断m的奇偶

##### (5)根据上一步的判断来执行循环，模拟报数并出圈

### 2.3 模块调用关系（图片）

## 3.详细设计

### 3.1 定义数据类型及存储结构（代码）

### 3.2 每个函数及操作的代码 （代码）

## 4.调试分析

### 4.1 遇到的问题及分析

#### 4.1.1 问题

##### (1)在每一次报数后，如何知道上一轮谁出圈了？如何表达出圈人的位置？

##### (2)在删除元素的时候，总是发生数组越界导致程序崩溃。

#### 4.1.2 分析

##### (1)在CList类中加一个指针（STL迭代器），来记录当前操作的元素和它的位置。

##### (2)删除当前元素后，它的迭代器里面没有元素，因此在删除的时候还要把迭代器向前或向后移动。这里把删除分成了两个函数，erase\_prev()和erase\_next()，分别对应删除后把迭代器往前移动和往后移动。

### 4.2 算法时空分析

#### 4.2.1 时间

##### (1)添加元素

###### 由链表的性质，是O(1)

##### (2)删除元素

###### 由链表的性质，是O(1)

##### (3)总体算法时间复杂度

###### 因为一共有n人要出圈，而每个人出圈前要先经过m个人，因此总体的时间复杂度为O(n\*m)。

#### 4.2.2 空间

##### 要存n个人，所以是O(n)。

#### 4.2.3 改进

##### 在网上查阅资料得知，如果只想知道最后一个出圈的人，则可通过O(n)的时间复杂度解决。

###### 我们将明确解出k=2时的问题。对于k!=2的情况，我们在下面给出一个一般的解法。

###### 设f(n)为一开始有n个人时，生还者的位置(注意：最终的生还者只有一个)。走了一圈以后，所有偶数号码的人被杀。再走第二圈，则新的第二、第四、……个人被杀，等等；就像没有第一圈一样。如果一开始有偶数个人，则第二圈时位置为x的人一开始在第2x-1个位置。因此位置为f(2n)的人开始时的位置为2f(n)-1。这便给出了以下的递推公式：

###### 如果一开始有奇数个人，则走了一圈以后，最终是号码为1的人被杀。于是同样地，再走第二圈时，新的第二、第四、……个人被杀，等等。在这种情况下，位置为x的人原先位置为2x+1。这便给出了以下的递推公式：

###### 如果我们把n和f(n)的值列成表，我们可以看出一个规律：

###### 从中可以看出，f(n)是一个递增的奇数数列，每当n是2的幂时，便重新从f(n)=1开始。因此，如果我们选择m和l，使得 $ n=2^{m}+l $ 且 $ 0l<2^{m} $，那么 $ f(n)=2l+1。 $ 注意：2^m是不超过n的最大幂，l是留下的量。显然，表格中的值满足这个方程。我们用数学归纳法给出一个证明。

###### 定理：如果 $ n=2^{m}+l $ 且 $ 0l<2^{m} $ ，则 $ f(n)=2l+1 $ 。

###### 证明：对n应用数学归纳法。n=1的情况显然成立。我们分别考虑n是偶数和n是奇数的情况。

###### 如果n是偶数，则我们选择 $ l\_{1} $ 和 $ m\_{1} $ ，使得 $ n/2=2^{{m\_{1}}}+l\_{1} $ ，且 $ 0l\_{1}<2^{{m\_{1}}} $ 。注意 $ l\_{1}=l/2 $ 。我们有 $ f(n)=2f(n/2)-1=2((2l\_{1})+1)-1=2l+1 $ ，其中第二个等式从归纳假设推出。

###### 如果n是奇数，则我们选择 $ l\_{1} $ 和 $ m\_{1} $ ，使得 $ (n-1)/2=2^{{m\_{1}}}+l\_{1} $ ，且 $ 0l\_{1}<2^{{m\_{1}}} $ 。注意 $ l\_{1}=(l-1)/2 $ 。我们有 $ f(n)=2f((n-1)/2)+1=2((2l\_{1})+1)+1=2l+1 $ ，其中第二个等式从归纳假设推出。证毕。

###### 答案的最漂亮的形式，与n的二进制表示有关：把n的第一位移动到最后，便得到f(n)。如果n的二进制表示为 $ n=b\_{0}b\_{1}b\_{2}b\_{3}b\_{m} $ ，则 $ f(n)=b\_{1}b\_{2}b\_{3}b\_{m}b\_{0} $ 。这可以通过把n表示为 $ 2^{m}+l $ 来证明。

###### 一般情况下，考虑生还者的号码从n-1到n的变化, 我们可以得到以下的递推公式(编号从0开始)：

###### 这种方法的运行时间是O(n)。

### 4.3 经验和体会

## 5.使用说明

## 6.测试结果

## 7.附录

### 7.1 个人负责的部分

### 7.2 整个程序