# 一、问题描述

编号为1，2，3，……，n的n个人按顺时针方向围坐一圈，每人持有一个密码。一开始任选一个数作为报数上限值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。

# 二、程序模块

# 三、思路分析

1. 基本思路：顺序存储和链表存储。我们首先按照原始的方法（即按顺序依次遍历）写出了程序。

* 顺序存储方法每轮只需按下标找一次元素，但平均需要移动n/2个元素。
* 链表存储方法加入了正逆遍历的判定，每轮平均需要遍历n/4个元素，并执行一次删除操作。另外为了避免频繁使用malloc函数造成内存碎片化，链表存储实际上采用了静态链表存储。

经过分析，这两种方法的时间复杂度均为；空间复杂度近似为o(n)。

1. 降低算法的时间复杂度。由于数组的大面积元素移动和链表的多个元素遍历是我们不愿意看到的。我们将整体的大环进行适当的拆分，这样就可以在拆分的小部分中进行较少数量的元素移动或元素遍历。至此，我们得出的分组遍历的方法。

* 顺序存储方法的数组有两层：第一层有A个元素，代表A个组；每个一层元素对应一个二层的数组，数组中有n/A个元素。遍历时，首先根据各组元素个数确定目标组序号，平均需要A/2次；再根据下标确定元素位置，平均进行n/2A次元素移动。经过数学分析，确定最佳分组个数A=。
* 链表存储方法的存储结构与上面类似，只是依照静态链表结构加入了前驱后继。遍历时，首先遍历当前元素所在小组，平均需要n/2A次；然后根据密码值确定遍历方向，再遍历大组得到目标组序号，平均需要A/4次；然后在小组中进行遍历得到目标元素，平均需要n/2A次。经过数学分析，确定最佳分组个数A=2。

经过分析，这两种方法对时间成本进行了优化，时间复杂度为；略微牺牲了一些空间，但空间复杂度仍为o(n)。

1. 在分组操作实现之后，思考如何将复杂度降到，对于此时间复杂度很自然想到二叉树的结构。我们的二叉树构造过程如下，底层为个数为，前n个节点值为1，剩下的节点值为0,的数组，向上传递，父节点为两个子节点的值之和，直至顶层。每层之间的父子节点的关系是由其在数组中的序号维持的，二叉树可视化结构如下：

删除节点的方法也是向上传递，根据上下层的序号的关系寻找父节点来更新数值，对于第三个元素删除后，树的可视化如图：

节点查找方法分为向左查找与向右查找，对于向右查找的方式：向上遍历，当前的位置为左兄弟节点，判断右兄弟节点是否大于当前需要移动的值，若是，进入右兄弟节点，进行向下遍历操作，若否，进入父节点，更新密码值，继续进行判断；当前位置是右节点，直接进入父节点，重复当前操作。向下遍历，判断左子节点是否包含有比当前密码值更多的节点，若否，进入右子节点，重复当前操作，若是，进入左子节点，重复当前操作。

查找的最坏与平均情况时间复杂度均为，删除的最坏与平均情况时间复杂度均为，故总时间复杂度为，空间复杂度为。

对于向左向右的时间分析：考虑到构建树的结构时，我们将不满的位置值初始化为0，故右侧的一部分的值是无意义的，故，向右搜索时势必要进行不必要的比较判断，从而增加时间的开销。故向左搜索的时间往往优于向右搜索。

1. 除了受到二叉树的启发，我们也尝试从别的角度出发进行分析。分析GRL算法（分组数组）实则是将约瑟夫环的一维数组转化为一个2D平面，利用面上的横纵坐标实现下一个出列元素的迅速定位。那我们可以将维数提升，比如说将将其抽象为一个三维的立体空间，利用x,y,z三方向坐标定位，四维可以理解维有多个三维的立方体，更多维可以以此类推。于是进行抽象化，提升维数，并尝试确定维数为多少的时候，它的定位速度是最快的。设维数为m，那么每一个维度的个数为。为了简化问题，我们假设每一次寻找都是最坏情况（即在每一个维度上寻找都需要到最后一个才能找到），所以每一次寻找的次数就是

对(1)式求导得到

当(2)式值为0时，就求得了f(m)的最小值，此时 ，。所以每一个维度的大小也就求出来了，就等于自然对数e。因而我们将维度大小限定在2和3，经过计算，当维度为3时，每一次查找的次数是最少的。

再开始进行算法的结构设计，将一维数组抽象为维，那么每一个元素的查找都是从最高维找起，每一维的不同分量值都是该分区的总元素个数。其实更加形象化一点，可以发现这个多维的抽象模型更像是一个三叉树，我们拿举例。

首先建立三叉树，如下图：

再进行删除元素的操作。首先将密码与当前位置进行相关计算得出下一个元素的绝对位置，如算出的下一个将要删除的元素是第3个，其过程如下图：

首先将最高维记录的元素个数减一，降低维数，在第2维进行搜素，发现在第一个分区就有目标元素，便将其个数减一并降低维数到第一维，再进行基本的删除操作。至此，一次的删除操作便已结束，删除n个便是将这一过程重复n遍，所以其复杂度也是。

# 四、实现方法

共采用了七种方法进行约瑟夫环的演示，依次为：

①常规数组——ARR

②循环链表——LDL

③分组数组——GRY

④分组静态链表——2D

⑤二叉树（向右）——BIR

⑥二叉树（向左）——BIL

⑦三叉树——MD3

最优算法：三叉树算法