一.问题分析

1. 处理的对象:假设有n个人，依次报数，每次将报到k的倍数或者个位数为k的人出列，依次重复处理
2. 实现的功能：找出重复报数中剩下的最后一个人
3. 处理后的结果： 找出重复报数中剩下的最后一个人将其编号进行输出。
4. 样例求解：

样例1中输入n为5，k为3，第一次报数，报到2,4的淘汰，第二次报数报到6,8的淘汰，最后剩余编号是3的胜出

样例2中输入n为7，k为3，第一次报数，报到3,6的淘汰，第二次报数报到9,12的淘汰，第三次报到13,15的淘汰，最后剩余编号是4的胜出。

二.数据结构和算法设计

1. 抽象数据类型设计

使用STL中的vector

1. 物理数据对象设计

使用STL中的vector

1. 算法思想的设计

游戏要求每次报到将K的倍数或者个位含有K的数人淘汰掉，然后重复报数直到只剩下最后一个人，输出最后一人编号。我们可以利用vector数组的长度可变，来将没有淘汰的人的编号存入动态数组vector中，然后调用函数resize()函数重新设置数组大小，再次循环报数，将没有淘汰的人的编号存入数组，重复以上几步，直到数组中只剩下一个元素，即退出循环，输出编号。

1. 关键功能的算法步骤

步骤：

首先将编号存入数组，然后在判断是否淘汰，如果不淘汰就将该编号覆盖原有的值，重置数组大小，重复以上步骤直到只剩下一个元素的时候就输出即胜出者的编号。

利用vector淘汰编号的伪代码：

for（j=1;j<n;j++）{

将编号存入动态数组；}

num=0

While（size！=1）{

len=数组此时的大小

初始化没有淘汰的人的编号的length=0；

for(p=0;p<len;p++){

num++;//模拟报数操作

if（不是k的倍数个位也不是k）

v[length++]=v[p];//覆盖掉原有的值

重新设置动态数组大小

}

三.算法性能分析

1. 时间复杂度：主程序中if else语句花费总的时间等于两个判断中较大那个，else中循环时间复杂度为○(n),在淘汰人数的循环，终止条件是动态数组大小为1时，退出循环，时间复杂度为○(nlogn+4),总的化简后为时间复杂度为○(nlogn)
2. 空间复杂度：定义了一个长度为n的动态数组，假设每个元素存储需要c个字节，在淘汰人数循环中每次数组的大小都在改变，空间复杂度就为O(cnlogn),化简后为O(nlogn)。