**约瑟夫生死环项目文档**

**项目简介：**

约瑟夫生者死者游戏的大意是：30个旅客同乘一条船，因为严重超载，加上风高浪大危险万分；因此船长告诉乘客，只有将全船一半的旅客投入海中，其余人才能幸免于难。无奈，大家只得统一这种方法，并议定30个人围成一圈，由第一个人开始，依次报数，数到第9人，便将他投入大海中，然后从他的下一个人数起，数到第9人，再将他投入大海，如此循环，直到剩下15个乘客为止。问哪些位置是将被扔下大海的位置。

**项目功能说明：**

本游戏的数学建模如下：假如N个旅客排成一个环形，依次顺序编号1, 2, …, N。从某个指定的第S号开始。沿环计数，每数到第M个人就让器出列，且从下一个人开始重新计数，继续进行下去。这个过程一直进行到剩下K个旅客为止。（要求采用单循环链表）

本游戏要求用户输入的内容包括：

1. 旅客的个数，也就是N的值；
2. 离开旅客的间隔书，也就是M的值；
3. 所有旅客的序号作为一组数据要求存放在某种数据结构中。

本游戏要求输出的内容是包括：

1. 离开旅客的序号；
2. 剩余旅客的序号。

**数据结构：**

单循环链表

**具体实现：**

1. 基本构成：

结构体Player: (int) pos 玩家所处初始位置

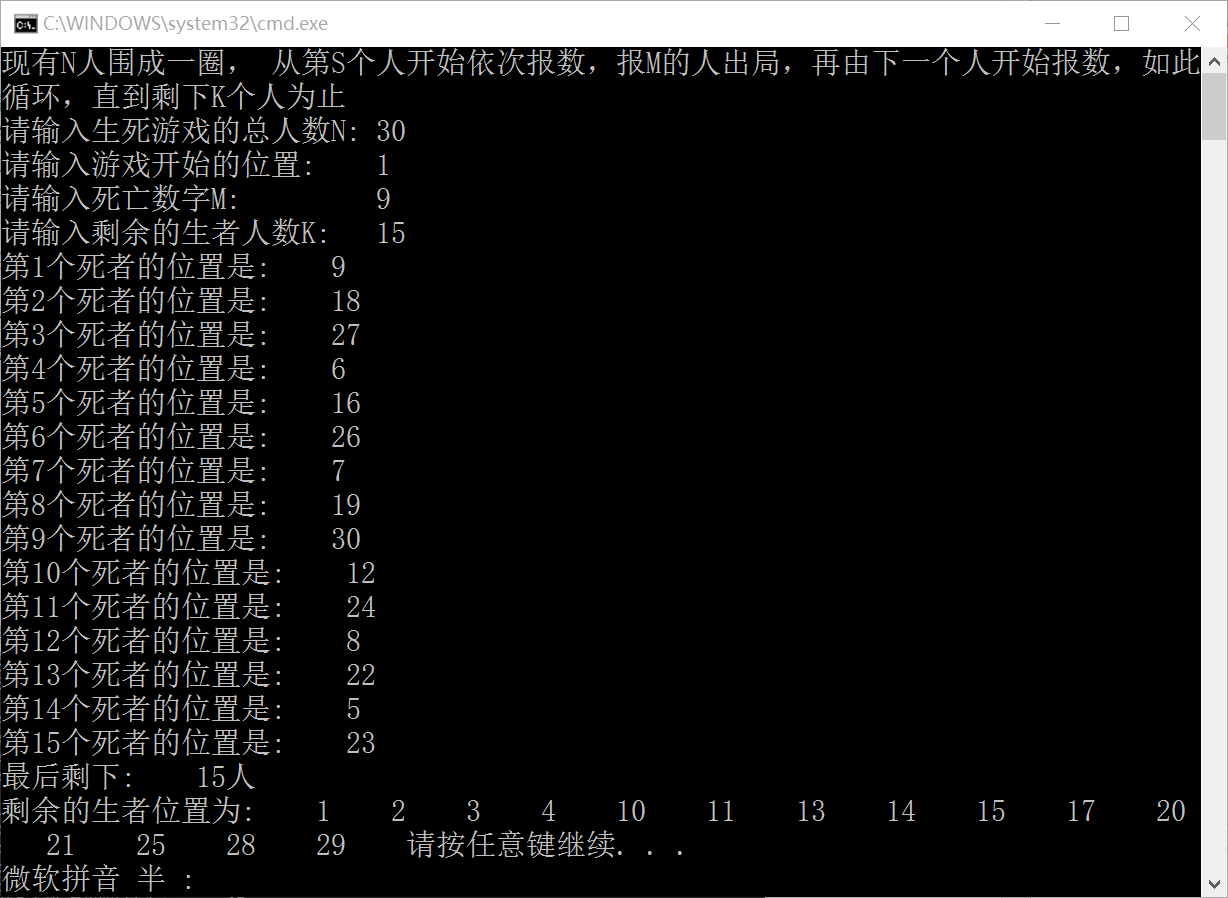
(struct \*)尾指针

核心函数：Player \* solve(int n, int s, int m, int k)

1. 方法实现具体说明

将每个Player结构体作为一个节点，构成一个单循环链表，定义一个临时指针移动寻找应该被删去的那个结构体。当找到之后，将它之前的结构体直接链接到它的下一个，这样一来，那个被删去的结构体就脱离了这个体系，相当于死亡。最后，当人数达到要求，便返回一个还构成圈的结构体指针，相当于返回了现在还存活的人数。简单地进行排序之后就可以输出了。

**程序运行说明：**

****

**收获：**

在约瑟夫生死环的编写过程中，我对单循环链表的认识更加深刻了。通过将首尾链表链接在一起，神奇地实现了环形数据结构，巧妙地解决了约瑟夫生死环问题，希望自己在数据结构的学习过程中可以接触越来越多神奇的数据结构。