项目文档

谈瑞

项目二：约瑟夫生者死者游戏

专业班级：软件工程 4 班 学号：1452775 课目：数据结构课程设计
电话：18936361545 电子邮件：tanrui106@163.com 个人网站：http://guitoubing.top

目录

项目简介

项目概要

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。

项目功能及要求

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。

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。

项目结构

本项目采用单循环链表实现，基本上就是考察链表的插入、删除等基本操作，应该说写起来很顺畅，按照题设思路一步一步往下做就能写成功。

首先通过用户输入约瑟夫环的几个限制条件总人数、开始位置、死亡数字、剩余人数，同时判断输入是否合理，不合理则要求重新输入。而后通过这些数据构建链表完成游戏过程最后输出死亡的人以及剩余人的位置。

项目类的实现

Josepth 类

采用正交循环双向链表构建迷宫

	类成员	作用
public 成员	Josepth ();	Josepth 类的默认构造函数，此处用户输入约瑟夫环的限制条件
	~Josepth ();	Josepth 类的默认析构函数，在程序结束后释放链表中所有在游戏中存活下的结点
	void setNumbers(int, const int);	设置限制条件，同时判断输入是否合理，若不合理则重新输入
	Passenger* setJosepthNode(int);	递归设置约瑟夫环结点，最后构建约瑟夫环
	Passenger* setBeginNode(int);	根据开始位置设置开始结点
	Passenger* deleteNode(Passenger*);	删除结点，即杀人
	void startKilling(Passenger*);	游戏开始入口
private 成员	Passenger *present, *head, *tail;	头尾结点、当前结点
	int amount, firstIndex, deathNumber, amountLeft;	约瑟夫环的限制条件

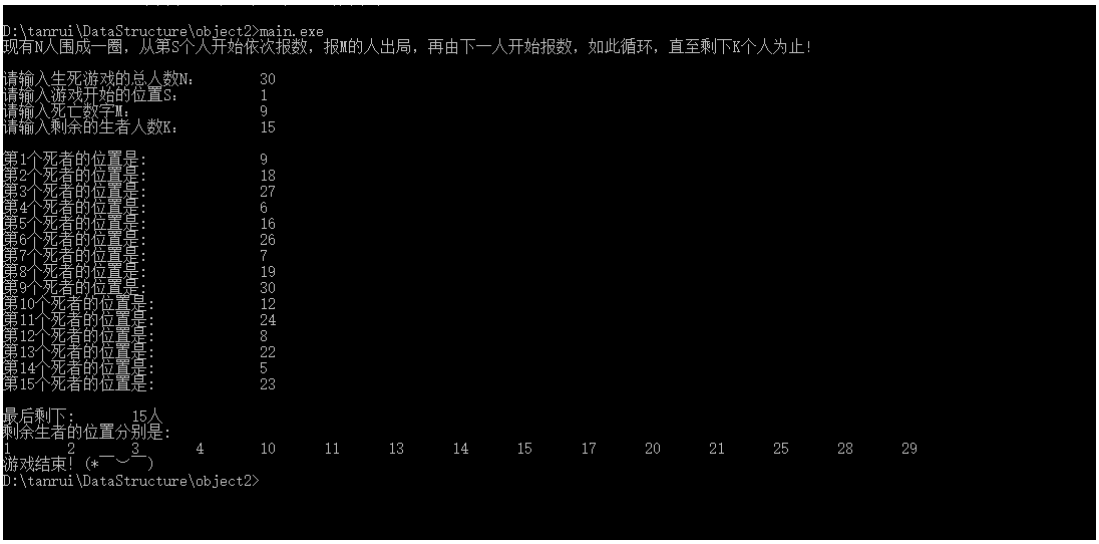
Passenger 类

Passenger 类存储了迷宫的结点

	类成员	作用
public 成员	Passenger();	Passenger 类的默认构造函数
	Passenger(int, Passenger*);	Passenger 类的重载构造函数，通过参数 number 及 next 指针构造实例
	~Passenger();	Passenger 类的默认析构函数
private 成员	int number;	旅客的位置编号
	Passenger* next;	指向下一个旅客的指针

运行测试

正常运行



检测可能出现的输入错误

对于给定的总人 N 数来说，S、M 和 K 均被限定下来了，看代码段：

```
case 0: {
        while (_input < 1 || _input > MAXAMOUNT) {
            cout << "输入的总人数有误，请重新输入：";
            cin >> _input;
        }
        amount = _input;
        break;
    }
```

总人数 amount 的取值范围为 1-MAXAMOUNT（这里宏定义为 20000）闭区间

```
case 1: {
        while (_input < 1 || _input > amount) {
            cout << "输入的开始位置有误，请重新输入：";
            cin >> _input;
        }
        firstIndex = _input;
        break;
    }
```

开始位置 firstIndex 的取值范围为 1-amount 闭区间

```
case 2: {
        while (_input <= 1) {
            cout << "输入的死亡数字有误，请重新输入：";
            cin >> _input;
        }
        if (_input > amount) _input = _input % amount;
        deathNumber = _input;
        break;
    }
```

死亡数字 deathNumber 的取值范围为>1 的整数，若 deathNumber 大于 amount，则将其置为用其对 amount 取模的结果即可。

```
case 3: {
        while (_input >= amount || _input <= 0) {
            cout << "输入的剩余人数有误，请重新输入：";
            cin >> _input;
        }
        amountLeft = _input;
        break;
    }
```

剩余的人数 amountLeft 的取值范围为 1-amount 的开区间

如图所示，错误的输入均可被检测出，并要求重新输入。