

**学生实验实习报告册**

|  |  |
| --- | --- |
| 学年学期： | 2022-2023 学年 🗹春**🞎**秋学期 |
| 课程名称： | 程序设计实训 |
| 学生学院： | 国际学院 |
| 专业班级： | 34082202 |
| 学生学号： | 2022214986 |
| 学生姓名： | 周昕阳 |
| 联系电话： | 18297704195 |

**重庆邮电大学教务处制**

目 录

[一.实验任务概述 1](#_Toc135851428)

[二.实验准备及方法 1](#_Toc135851429)

[三.实验设备及软件 1](#_Toc135851430)

[四.实验步骤、过程原始记录(数据、图表、计算等) 1](#_Toc135851431)

[(一).实验任务完成情况 2](#_Toc135851432)

[(二).任务成果展示 2](#_Toc135851433)

[1.流程图/时序图/类图 2](#_Toc135851434)

[2.设计思路必要说明 2](#_Toc135851435)

[3.实验结果 3](#_Toc135851436)

[五.实验体会和收获 3](#_Toc135851437)

[六.程序源代码 4](#_Toc135851438)

|  |  |  |  |
| --- | --- | --- | --- |
| **课程名称** | 程序设计实训 | **课程编号** | **A2131200** |
| **实验地点** | **综合实验大楼B409/410** | **实验时间** | **6月5日至6月9日** |
| **校外指导教师** | **无** | **校内指导教师** | **赵春泽** |
| **实验名称** | **程序设计实训** | | |
| **评阅人签字** |  | **成绩** |  |

# 一.实验任务概述

//报告者完成了什么写什么。

//分别描述实验1、2、3、4的任务

1.实验一：一群小孩围一圈，任意假定一个数，从第k个小孩起，顺时针方向数，每到第m个小孩时，该小孩便离开。小孩不断离开，圈子不断缩小。最后，剩下的一个小孩便是胜利者。由此条件，解决Josephus问题：最后胜利的是第几个小孩。实验一要求使用数组和链表分别完成代码实现，解决Josephus问题。

2.实验二：与实验一一样，实验二将解决Josephus问题。Josephus游戏从小孩问题中抽象而来，实验二将运用面向对象的程序设计方法，分析Josephus问题中的事物与关系，抽象出对象与联系建立模型，设计出类与关联，完成代码实现。

3.实验三：运用模块化程序设计方法，面向对象程序设计方法，完成打飞机游戏的代码实现。游戏程序中，用各种符号代表飞机子弹与敌机，显示飞机生命值与得分，，飞机撞上敌机生命值减1，减到0游戏结束。玩家可以通过w，a，s，d按键控制玩家飞机上下左右移动，按空格键发射子弹，子弹击中飞机计分。敌机位置随机出现。代码要求有多文件结构，采用面向过程与面向对象设计方法，分别对飞机展开程序框架设计。

4.实验四：XXXXX

# 二.实验准备及方法

// Tips：描述实验前自己为本次实验做了哪些准备，主要包括C++ \*\*理论知识的复习和巩固，实验所需C++代码的编辑和调试等。

//分别描述实验1、2、3、4的实验准备及方法

1.实验一：对C++理论知识进行了大体的复习，重新回顾了曾经的代码练习与知识点，掌握代码调试的方法。重点复习了C++中数组的知识，指针与地址、引用的知识，结构体的知识与链表的知识等等。再次熟悉了面向过程设计思想与方法。

2.实验二：复习java的相关理论知识，回顾调试代码的技巧。回顾并运用面向对象的程序设计思想与方法。学会类的抽象的方法以及找到类的关系并学会对其进行代码实现。

3.实验三：逻辑思维的训练，在面向过程设计的实现中充分理解代码的实现与编写逻辑。继续深入复习java中类的抽象与方法的编写，继续深入理解面向过程与面向对象的设计思想。能够精准巧妙地进行类的抽象，方法的实现，以及类之间关系的连接。

# 三.实验设备及软件

// Tips：描述此次实验中所使用的实验设备，以及相关的软件。如：XXX型号PC机，内存XXX兆，win10操作系统；软件有Visual studio 2013集成开发环境，Visual C++ 10.0工具套件。

Legion Y9000P IAH7H,内存32.0 GB，win11操作系统；软件有Visual Studio Code,MinGW-W64 GCC-8.1.0 C++编译器，JDK 8工具包。

# 四.实验步骤、过程原始记录(数据、图表、计算等)

// Tips：依据实验要求的内容，主要描述自己在实验过程中的实验步骤、主要实验测试数据以及期待结果。

## (一).实验任务完成情况

完成了基础实验和综合实验\*\*功能；\*\*程序绘制了流程图/类图/时序图，见图\*\*；得到\*\*\*结果，见图\*\*；主要实现代码见程序段\*\*\*

((完成了那些写那些))

//分别描述实验1、2、3、4的实验任务完成

（1）实验一：完成了基础实验和综合实验Josephus游戏功能面向过程实现；Josephus程序绘制了流程图，见[图1](#图1)；得到胜利小孩结果，见图，图；主要实现代码见程序段[表1 Josephus.cpp](#表1),[表2 Josephus2.cpp](#表2)。

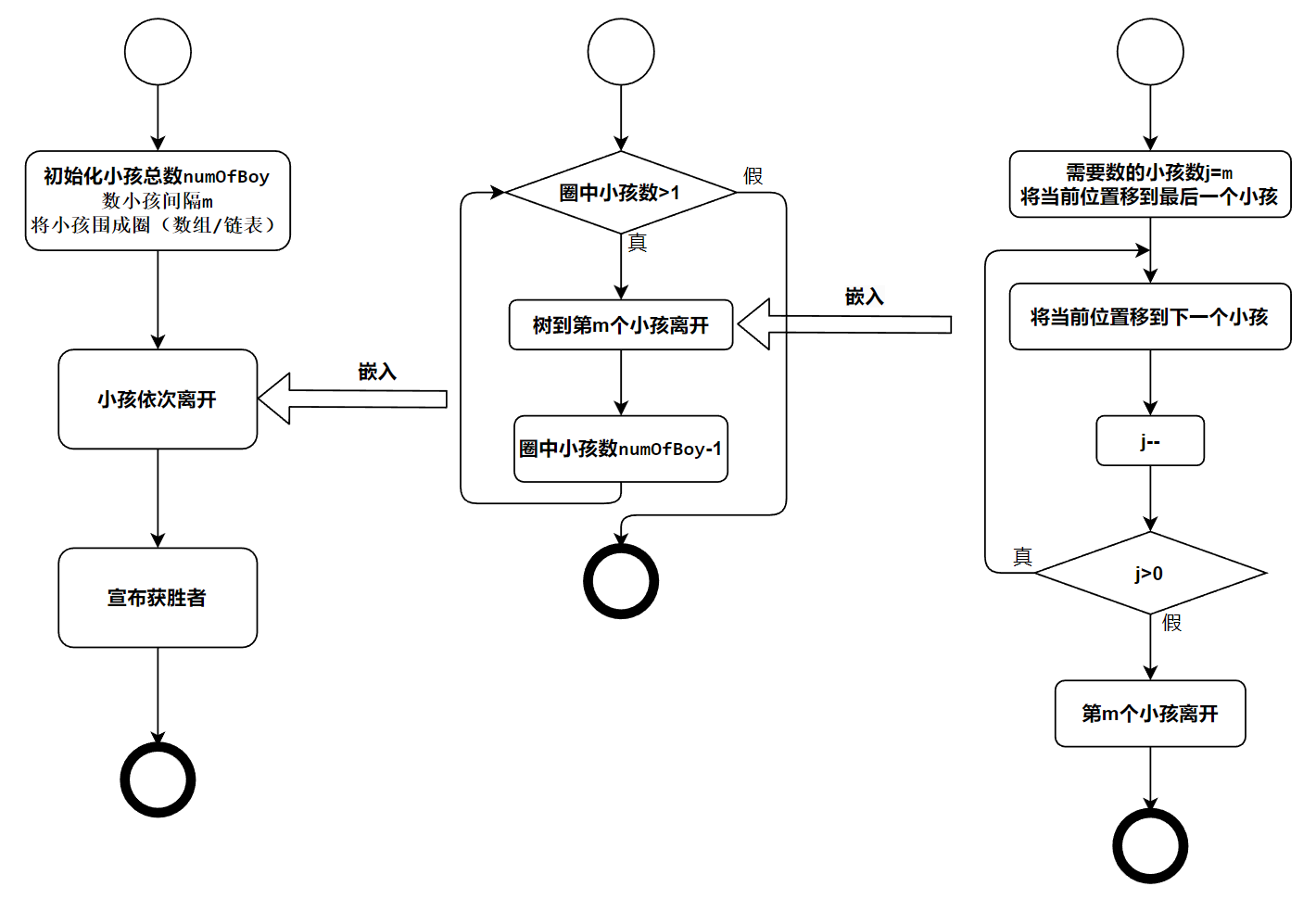
（2）实验二：完成了基础实验和综合实验Josephus游戏功能面向对象实现；Josephus程序绘制了类图，见；得到胜利小孩结果，见图，图；主要实现代码见程序段。

（3）实验三：XXXXX

## (二).任务成果展示

### 1.流程图/时序图/类图

//分别描述实验1、2、3、4的实验流程图/时序图/类图

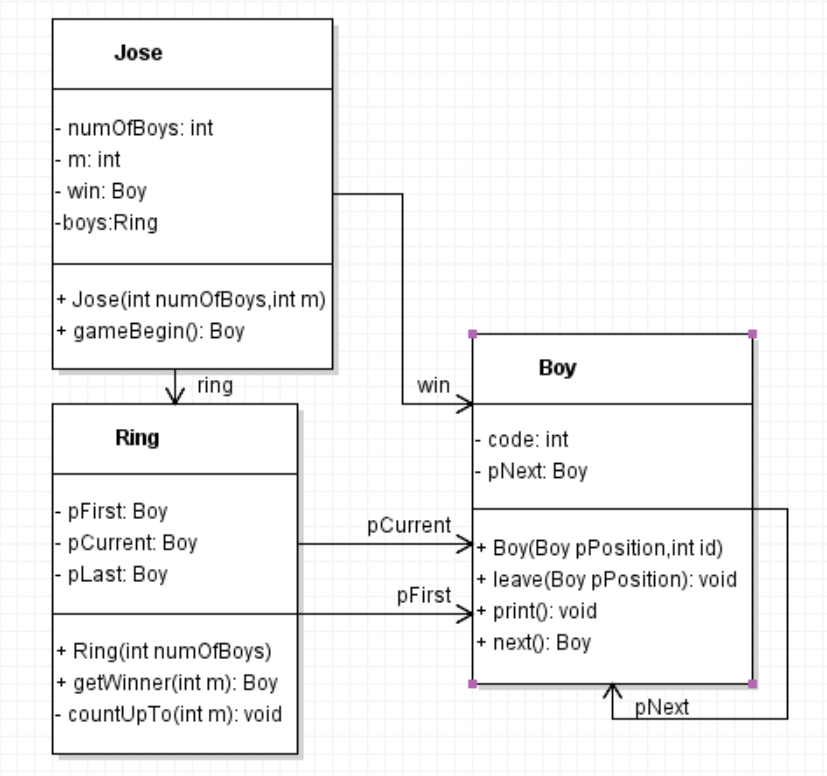
1. 实验一：面向过程实现Josephus问题

Josephus问题面向过程流程图如下。

图1 Josephus问题面向过程流程图

1. 实验二：面向对象实现Josephus问题

Josephus问题面向对象类图如下。

图2 Josephus问题面向对象类图

（3）实验三：XXXXX

### 2.设计思路必要说明

//分别描述实验1、2、3、4的实验设计思路

1. 实验一：面向过程实现Josephus问题

初始化全局变量小孩个数为numOfBoys，数小孩间隔为m。将小孩围成圈，可以通过数组或者链表进行实现，然后用一个大循环，判断圈中是否只剩一个小孩，如果只有一个小孩宣布小孩胜利，如果小孩不止1个，则进入小孩离开的循环区块。小孩离开中，先记录要数的小孩个数为j=m;创建小孩指示器指向最后一个小孩（为了第一次数数数到第一个小孩），进入循环体用j计数，循环3遍，每一遍将指示器移动到下一个小孩。循坏结束后，让小孩离开圆圈。记录小孩数-1，再进行大循环的判断，最后小孩只剩一个。

数组实现中，要实现数到最后一个小孩，下一个是第一个小孩，形成闭环，需要小孩指示器i，运行 **i = (i + 1) % numOfBoy;**取余语句来使指示器回到i=0。用数组存小孩编号和是否在圈内。用boys[i]=i+1来表示 第i+1个小孩，用boys[i]=0来表示第i+1个小孩离开。

链表实现中，要实现小孩围成圈，可以将每个小孩抽象为一个结构体，结构体中存两个属性，小孩编号和下一个小孩的指针。每个小孩依次指向下一个小孩，最后一个小孩指向第一个小孩，实现围成圈。小孩离开，只需要对链表中元素进行重新连接与删除。比如要删除编号为i的小孩，只需要将上一个小孩指向下一个小孩，然后再删除这个小孩的数据就能实现。

1. 实验二：面向对象实现Josephus问题

运用面向对象的程序设计思想，对问题进行抽象，抽象出Boy、Jose、Ring三个类。

（3）实验三：XXXXX

### 3.实验结果

// Tips：描述此次实验的结果，注意同实验要求进行比较，检查是否完全完成了实验要求的所有内容。可以将实验结果通过贴图的方式进行表述

如：

说明：按**图1**所示流程，该程序先初始化累加合sum的值为0，循环计算第n项的值保存在v中，并加到累加和sum中，直到第n项的绝对值() <0.00000001跳出循环，最后求出4与sum的乘积及得到的值。



**实现效果如图2所示**。

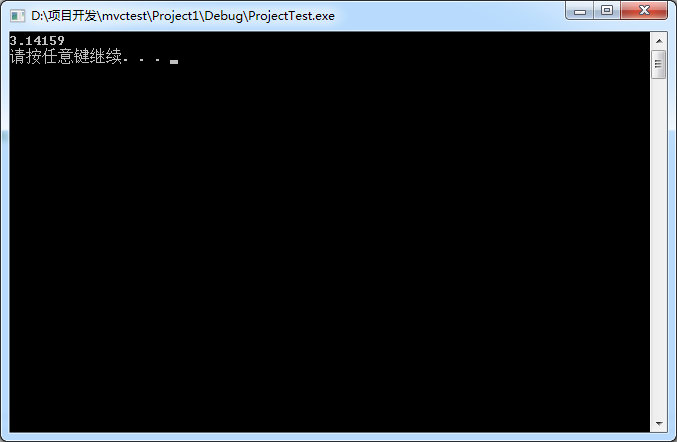


图2 求效果图



//分别描述实验1、2、3、4的实验结果

1. 实验一：面向过程实现Josephus问题

使用数组实现：

按图1所示流程，该程序初始化小孩数为7，数小孩间隔为3，输出小孩编号后，循坏进行小孩离开操作，小孩离开中循坏数小孩，每数3次，离开一个小孩，3，6，2，7，5，1号小孩依次离开，最后4号小孩获胜。

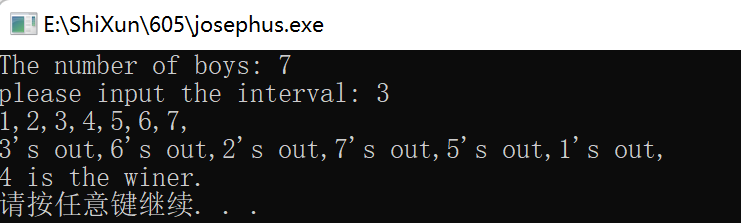
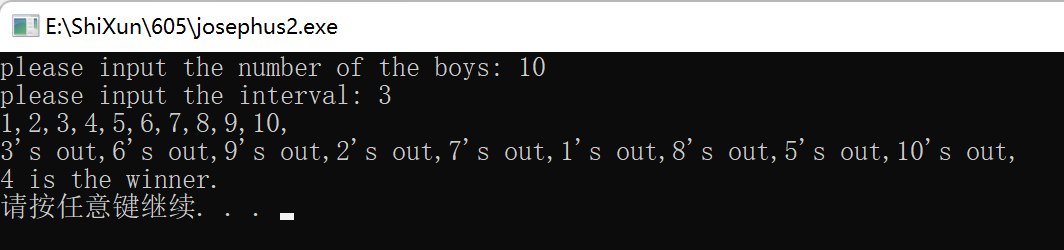
使用数组实现运行结果如下。

图 Josephus问题数组实现

使用链表实现：

 按图1所示流程，该程序初始化小孩数为10，数小孩间隔为3，输出小孩编号后，循坏进行小孩离开操作，小孩离开中循坏数小孩，每数3次，离开一个小孩，3，6，2，7，1，8，5，10号小孩依次离开，最后4号小孩获胜。

使用链表实现运行结果如下。

图 Josephus问题链表实现

（2）实验二：XXXXX

（3）实验三：XXXXX

（4）实验四：XXXXX

# 五.实验体会和收获

// Tips：描述此次实验的体会、感受和收获，如对于C++知识的学习、程序调试、测试数据构造等方面的感受等。

# 六.程序源代码

//Tips：将本次实验的所有程序源代码粘贴在此处。（应有必要注释）

//分别描述实验1、2、3、4的实验源代码

1. 实验一：Josephus问题

用数组实现源代码：

表1 Josephus.cpp

|  |
| --- |
| **#include <iostream>**  **using namespace std;**  **int main()**  **{**  **const int numOfBoy = 7; //小孩总数**  **int boys[numOfBoy];     //小孩数组**  **// n个小孩围成圈**  **int i = 0; //小孩位置指示器**  **for (i = 0; i < numOfBoy; i++)**  **{**  **boys[i] = i + 1;**  **}**  **cout << "The number of boys: " << numOfBoy << endl;**  **//输入小孩间隔**  **int m; // Josephus问题中的间隔m**  **cout << "please input the interval: ";**  **cin >> m;**  **//输出开始时的小孩编号**  **for (i = 0; i < numOfBoy; i++)**  **{**  **cout << boys[i] << ",";**  **}**  **cout << endl;**  **//小孩依次离开**  **i = numOfBoy - 1;                  //将i定位到最后一个小孩，下一次移动位能够得到i=0**  **for (int n = numOfBoy; n > 1; n--) //离开 numOfBoy - 1 个小孩**  **{**  **//数到第m个小孩离开**  **int j = m;**  **do**  **{**  **//将位置移到下一个小孩**  **i = (i + 1) % numOfBoy;**  **while (boys[i] == 0)**  **{**  **i = (i + 1) % numOfBoy;**  **}**  **j--; //数一个小孩**  **} while (j > 0);**  **//第m个小孩离开**  **cout << boys[i] << "'s out,"; //输出离开小孩编号**  **boys[i] = 0;                  //表示i小孩离开**  **}**  **cout << endl;**  **//宣布获胜者**  **for (i = 0; i < numOfBoy; i++)**  **{**  **if (boys[i] != 0)**  **{**  **cout << boys[i] << " is the winer." << endl;**  **break;**  **}**  **}**  **system("pause");**  **}** |

用链表实现：

表2 Josephus2.cpp

|  |
| --- |
| **#include <iostream>**  **using namespace std;**  **struct Boy //小孩节点**  **{**  **int code;**  **Boy \*pNext;**  **};**  **Boy \*pFirt = 0;    //第一个小孩指针**  **Boy \*pCurrent = 0; //当前小孩指针**  **Boy \*pivot = 0;    //上一个小孩指针**  **int main()**  **{**  **//输入小孩总数**  **int numOfBoy;**  **cout << "please input the number of the boys: ";**  **cin >> numOfBoy;**  **//输入小孩间隔**  **int m; // Josephus问题中的间隔m**  **cout << "please input the interval: ";**  **cin >> m;**  **// n个小孩围成圈**  **//添加第一个小孩**  **pFirt = new Boy;**  **pFirt->code = 1;**  **pCurrent = pFirt;**  **//添加其他小孩**  **for (int i = 1; i < numOfBoy; i++)**  **{**  **pivot = pCurrent;        //前一个小孩改为"当前的小孩"**  **pCurrent = new Boy;      //在当前的小孩添加一个小孩**  **pCurrent->code = i + 1;  //为加入的小孩编号**  **pivot->pNext = pCurrent; //前一个小孩指向当前的小孩**  **}**  **pCurrent->pNext = pFirt; //最后一个小孩指向第一个，此时pCurrent还在最后一个小孩**  **//输出开始时的小孩编号**  **for (int i = 0; i < numOfBoy; i++)**  **{**  **pCurrent = pCurrent->pNext;**  **cout << pCurrent->code << ",";**  **}**  **cout << endl;**  **//小孩依次离开,，此时pCurrent还在最后一个小孩**  **while (pCurrent->pNext != pCurrent)**  **{**  **int j;**  **j = m; //第m个小孩离开**  **do**  **{**  **//将位置移到下一个小孩**  **pivot = pCurrent;**  **pCurrent = pCurrent->pNext;**  **j--;         //数一个小孩**  **} while (j > 0); //循环m次表示数m个小孩**  **cout << pCurrent->code << "'s out,"; //输出离开小孩编号**  **pivot->pNext = pCurrent->pNext;      //表示该位置小孩离开**  **delete pCurrent;                     //删除这个小孩**  **pCurrent = pivot;                    // pCurrent退回到上一小孩**  **}**  **cout << endl;**  **//宣布获胜者**  **cout << pCurrent->code << " is the winner.";**  **cout << endl;**  **system("pause");**  **}** |

1. 实验二：Josephus游戏

使用面向对象思想实现：

表3 Josephus.java

|  |
| --- |
| **package Josephus;**  **import java.util.Scanner;**  **/\*\***  **\* Josephus game**  **\***  **\* @author xinyang Zhou**  **\* @version 1.0**  **\*/**  **public class Josephus {**  **/\*\***  **\* 运行Josephus问题**  **\***  **\* @param args A reference to a string array containing command-line arguments**  **\*/**  **public static void main(String[] args) {**  **Scanner input = new Scanner(System.in);**  **int numOfBoys;// 小孩总数**  **int interval;// 小孩间隔**  **Boy winnerBoy;// 获胜小孩**  **System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**  **System.out.println("这里是Josephus运算器!");**  **System.out.print("请输入小孩个数：");**  **numOfBoys = input.nextInt();**  **System.out.print("请输入间隔：");**  **interval = input.nextInt();**  **winnerBoy = new Jose(numOfBoys, interval).gameBegin();**  **System.out.print("获胜小孩");**  **winnerBoy.print();**  **input.close();**  **}**  **}** |

表4 Boy.java

|  |
| --- |
| **package Josephus;**  **/\*\***  **\* The Boy class of the Josephus.**  **\***  **\* @author Xinyang Zhou**  **\* @version 1.0**  **\*/**  **public class Boy {**  **// 小孩编号**  **private int code;**  **// 下一个小孩**  **private Boy pNext;**  **/\*\***  **\* 构造小孩**  **\***  **\* @param pPosition 用于连接为上一个小孩**  **\* @param id        小孩编号**  **\*/**  **public Boy(Boy pPosition, int id) {**  **code = id;**  **if (pPosition == null) {**  **this.pNext = this;// 只有一个小孩时，自己指向自己**  **} else {**  **this.pNext = pPosition.pNext;// 插入到小孩pPosition的后面**  **pPosition.pNext = this;**  **}**  **}**  **/\*\***  **\* 小孩离开**  **\***  **\* @param pPosition 离开小孩的上一个**  **\*/**  **public void leave(Boy pPosition) {**  **pPosition.pNext = this.pNext;// 上一个小孩的下一个接为离开小孩的下一个**  **System.out.println(code + "离开");**  **}**  **/\*\***  **\* 打印小孩ID**  **\*/**  **public void print() {**  **System.out.print("ID: " + code);**  **}**  **/\*\***  **\* 下一个小孩**  **\***  **\* @return 下一个小孩**  **\*/**  **public Boy next() {**  **return pNext;**  **}**  **}** |

表5 Jose.java

|  |
| --- |
| **package Josephus;**  **/\*\***  **\* The Jose class of the Josephus.**  **\***  **\* @author Xinyang Zhou**  **\* @version 1.0**  **\*/**  **public class Jose {**  **// 小孩总数**  **private int numOfBoys;**  **// 小孩间隔**  **private int m;**  **// 胜利的小孩**  **private Boy win;**  **// 小孩圆圈**  **private Ring boys;**  **/\*\***  **\* josephus问题的数据**  **\***  **\* @param numOfBoys 小孩总数**  **\* @param interval  间隔数**  **\*/**  **public Jose(int numOfBoys, int interval) {**  **this.numOfBoys = numOfBoys;**  **this.m = interval;**  **this.boys = new Ring(this.numOfBoys);// 小孩围一圈**  **}**  **/\*\***  **\* 游戏开始**  **\***  **\* @return 获胜小孩**  **\*/**  **public Boy gameBegin() {**  **win = boys.getWinner(m);**  **return win;**  **}**  **}** |

表6 Ring.java

|  |
| --- |
| **package Josephus;**  **/\*\***  **\* The Ring class of the Josephus.完成围小孩，数小孩，得到获胜小孩。**  **\***  **\* @author XinYang Zhou**  **\* @version 1.0**  **\*/**  **public class Ring {**  **// 第一个小孩**  **private Boy pFirst;**  **// 现在的小孩**  **private Boy pCurrent;**  **// 上一个小孩**  **private Boy pLast;**  **/\*\***  **\* 小孩围城圈**  **\***  **\* @param numOfBoys 小孩总数**  **\*/**  **public Ring(int numOfBoys) {**  **pFirst = new Boy(null, 1);// 第一个小孩**  **// 依次围上其他小孩**  **Boy pNewBoy = pFirst;**  **for (int i = 2; i <= numOfBoys; i++) {**  **pNewBoy = new Boy(pNewBoy, i);**  **}**  **// 将pCurrent移至最后一个小孩使得第一次数数pCurrent移到第一个**  **pCurrent = pNewBoy;**  **}**  **/\*\***  **\* 得到获胜小孩**  **\***  **\* @param m 间隔数**  **\* @return 获胜小孩**  **\*/**  **public Boy getWinner(int m) {**  **// 数小孩,知道只剩1个小孩**  **while (pCurrent != pCurrent.next()) {**  **// 每次数小孩的间隔为m。**  **countUpTo(m);**  **}**  **// 返回获胜者**  **return pCurrent;**  **}**  **/\*\***  **\* 往下数m个小孩，数到的小孩离开。**  **\***  **\* @param m 间隔数**  **\*/**  **private void countUpTo(int m) {**  **// 往下数m个小孩**  **for (int i = 0; i < m; i++) {**  **pLast = pCurrent;**  **pCurrent = pCurrent.next();**  **}**  **// 数到的小孩离开，同时pCurrent回到上一个小孩**  **pCurrent.leave(pLast);**  **pCurrent = pLast;**  **}**  **}** |

1. 实验三：打飞机游戏

面向过程实现：

表7 Fighter.java

|  |
| --- |
| **package FlyMe2TheMoon;**  **import java.util.Random;**  **import java.util.Scanner;**  **/\*\***  **\* The Fighter class of the FlyMe2TheMoon.**  **\***  **\* @author Xinyang Zhou**  **\* @version 1.0**  **\*/**  **public class Fighter {**  **private int height, width;// 游戏界面尺寸**  **private int fighter\_x, fighter\_y;// 飞机位置**  **private int bullet\_x, bullet\_y;// 子弹位置**  **private int enemy\_x, enemy\_y;// 敌机位置**  **private int score;// 分数**  **private int HP;// 生命值**  **/\*\***  **\* 构造打飞机游戏类**  **\*/**  **public Fighter() {**  **Random random = new Random();**  **score = 0;// 0分**  **HP = 5;// 5点生命值**  **// 游戏界面尺寸**  **height = 24;**  **width = 30;**  **// 飞机位置**  **fighter\_x = height - 5;**  **fighter\_y = width / 2;**  **// 敌机位置**  **enemy\_x = 0;**  **enemy\_y = (random.nextInt(width - 4) + 3);**  **}**  **/\*\***  **\* 画游戏图像**  **\*/**  **public void drawPicture() {**  **for (int x = 0; x < height + 2; x++) {**  **for (int y = 0; y < width + 2; y++) {**  **if (y == 0 || y == width + 1) {// 输出边框**  **System.out.print("|");**  **} else if (x == fighter\_x && (y >= fighter\_y - 2 && y <= fighter\_y + 2)) {// 画出飞机**  **System.out.print("#");**  **} else if (x == fighter\_x - 1 && y == fighter\_y) {// 画出飞机**  **System.out.print("A");**  **} else if (x == fighter\_x + 1 && (y == fighter\_y - 1 || y == fighter\_y + 1)) {// 画出飞机**  **System.out.print("I");**  **} else if (x == bullet\_x && y == bullet\_y) {// 画出子弹**  **System.out.print("\*");**  **} else if (x == enemy\_x && y == enemy\_y) {// 画出敌机**  **System.out.print("V");**  **} else {**  **System.out.print(" ");// 输出空格**  **}**  **}**  **System.out.println();**  **}**  **System.out.println("|                              |分数：" + score + " HP: " + HP);**  **}**  **/\*\***  **\* 控制飞机移动**  **\*/**  **public void operateFighter() {**  **Scanner Sc = new Scanner(System.in);**  **int speed = 0;// 控制敌机速度**  **do {// 控制飞机移动**  **String input = Sc.nextLine();**  **switch (input) {**  **case "w": {**  **if (fighter\_x > 2) {**  **fighter\_x--;**  **}**  **break;**  **}**  **case "a": {**  **if (fighter\_y > 3) {**  **fighter\_y--;**  **}**  **break;**  **}**  **case "s": {**  **if (fighter\_x < height - 1) {**  **fighter\_x++;**  **}**  **break;**  **}**  **case "d": {**  **if (fighter\_y < width - 2) {**  **fighter\_y++;**  **}**  **break;**  **}**  **case " ": {**  **bullet\_x = fighter\_x - 2;**  **bullet\_y = fighter\_y;**  **break;**  **}**  **}**  **ifSurvive();// 飞机是否存活**  **// 子弹移动**  **floatBullet();**  **ifEnemySurvive();// 敌机是否存活**  **// 敌机移动**  **if (speed < 2)**  **speed++;**  **if (speed == 2) {**  **enemyFighter();**  **speed = 0;**  **}**  **ifSurvive();// 飞机是否存活**  **ifEnemySurvive();// 敌机是否存活**  **// 更新图像**  **drawPicture();**  **} while (HP > 0);**  **Sc.close();**  **}**  **/\*\***  **\* 子弹移动**  **\*/**  **public void floatBullet() {**  **Random random = new Random();**  **bullet\_x--;**  **if (bullet\_x == enemy\_x && bullet\_y == enemy\_y) {**  **score++;// 加分**  **enemy\_x = 0;// 产生新敌机**  **enemy\_y = (random.nextInt(width - 4) + 3);**  **bullet\_x = -2;// 取消子弹**  **}**  **}**  **/\*\***  **\* 敌机移动**  **\*/**  **public void enemyFighter() {**  **Random random = new Random();**  **if (enemy\_x > height) {**  **// 产生新敌机**  **enemy\_x = 0;**  **enemy\_y = (random.nextInt(width - 4) + 3);**  **}**  **enemy\_x++;// 敌机下移**  **}**  **/\*\***  **\* 判断飞机是否存活**  **\*/**  **public void ifSurvive() {**  **Random random = new Random();**  **// 飞机被击中**  **if ((enemy\_x == fighter\_x - 1 && enemy\_y == fighter\_y)**  **|| (enemy\_x == fighter\_x && enemy\_y > fighter\_y - 3 && enemy\_y < fighter\_y + 3)) {**  **// 产生新敌机**  **enemy\_x = 0;**  **enemy\_y = (random.nextInt(width - 4) + 3);**  **// 刷新飞机位置**  **fighter\_x = height - 5;**  **fighter\_y = width / 2;**  **HP--;// 减少生命值**  **if (HP == 0) {**  **System.out.println("\*\*\*飞机被摧毁，游戏结束！\*\*\*");**  **System.exit(0);**  **}**  **}**  **}**  **/\*\***  **\* 判断敌机是否存活**  **\*/**  **public void ifEnemySurvive() {**  **Random random = new Random();**  **if (bullet\_x == enemy\_x && bullet\_y == enemy\_y) {// 击中敌机**  **score++;// 加分**  **bullet\_x = -2;// 取消子弹**  **// 重置敌机**  **enemy\_x = 0;**  **enemy\_y = (random.nextInt(width - 4) + 3);**  **}**  **}**  **}** |

表8 mainFighter.java

|  |
| --- |
| **package FlyMe2TheMoon;**  **/\*\***  **\* main of FlyMe2TheMoon**  **\***  **\* @author xinyang Zhou**  **\* @version 1.0**  **\*/**  **public class mainFighter {**  **/\*\***  **\* main函数（主模块）**  **\***  **\* @param args A reference to a string array containing command-line arguments**  **\*/**  **public static void main(String[] args) {**  **System.out.println("这里是打飞机小游戏~~");**  **// 实例打飞机类**  **Fighter FlyMe2TheMoon = new Fighter();**  **// 画出初始的图像**  **FlyMe2TheMoon.drawPicture();**  **// 打飞机游戏开始运作**  **FlyMe2TheMoon.operateFighter();**  **}**  **}** |

面向过程实现（运用时间控制敌机、子弹，运用窗口实现键盘监听）

表15

|  |
| --- |
|  |

表16

|  |
| --- |
|  |

面向对象实现：

表11 Fighter.java

|  |
| --- |
| **package FlyMe2TheMoon2;**  **/\*\***  **\* The Fighter class of the FlyMe2TheMoon.**  **\***  **\* @author Xinyang Zhou**  **\* @version 3.0**  **\*/**  **public class Fighter extends Picture {**  **private int fighter\_x, fighter\_y;// 飞机位置**  **private int HP;// 飞机生命值**  **private int score;// 分数**  **/\*\***  **\* 构造Fighter**  **\*/**  **public Fighter() {**  **createFighter();**  **HP = 5;// 初始生命值为5**  **score = 0;// 初始分数为0**  **}**  **/\*\***  **\* 设置飞机x坐标**  **\***  **\* @param fighter\_x 飞机x坐标**  **\*/**  **public void fighter\_xSetter(int fighter\_x) {**  **this.fighter\_x = fighter\_x;**  **}**  **/\*\***  **\* 设置飞机y坐标**  **\***  **\* @param fighter\_y 飞机y坐标**  **\*/**  **public void fighter\_ySetter(int fighter\_y) {**  **this.fighter\_y = fighter\_y;**  **}**  **/\*\***  **\* 设置飞机HP**  **\***  **\* @param HP 飞机HP**  **\*/**  **public void HPSetter(int HP) {**  **this.HP = HP;**  **}**  **/\*\***  **\* 设置分数**  **\***  **\* @param score 分数**  **\*/**  **public void scoreSetter(int score) {**  **this.score = score;**  **}**  **/\*\***  **\* @return 飞机HP**  **\*/**  **public int HPGetter() {**  **return HP;**  **}**  **/\*\***  **\* @return 得分**  **\*/**  **public int scoreGetter() {**  **return score;**  **}**  **/\*\***  **\* @return 飞机的x坐标**  **\*/**  **public int fighter\_xGetter() {**  **return fighter\_x;**  **}**  **/\*\***  **\* @return 飞机的y坐标**  **\*/**  **public int fighter\_yGetter() {**  **return fighter\_y;**  **}**  **/\*\***  **\* 创造飞机（刷新飞机位置）**  **\*/**  **public void createFighter() {**  **// 飞机位置**  **fighter\_x = super.heightGetter() - 5;**  **fighter\_y = super.widthGetter() / 2;**  **}**  **}** |

表12 Bullet.java

|  |
| --- |
| **package FlyMe2TheMoon2;**  **/\*\***  **\* The Bullet class of the FlyMe2TheMoon.**  **\***  **\* @author Xinyang Zhou**  **\* @version 3.0**  **\*/**  **public class Bullet {**  **private int bullet\_x, bullet\_y;// 子弹位置**  **/\*\***  **\* 子弹构造器**  **\*/**  **public Bullet(int fighter\_x, int fighter\_y) {**  **// 创造一个子弹**  **bullet\_x = fighter\_x - 2;**  **bullet\_y = fighter\_y;**  **}**  **/\*\***  **\* 设置子弹x坐标**  **\***  **\* @param bullet\_x 子弹x坐标**  **\*/**  **public void bullet\_xSetter(int bullet\_x) {**  **this.bullet\_x = bullet\_x;**  **}**  **/\*\***  **\* 设置子弹y坐标**  **\***  **\* @param bullet\_y 子弹y坐标**  **\*/**  **public void bullet\_ySetter(int bullet\_y) {**  **this.bullet\_y = bullet\_y;**  **}**  **/\*\***  **\* @return 子弹x坐标**  **\*/**  **public int bullet\_xGetter() {**  **return bullet\_x;**  **}**  **/\*\***  **\* @return 子弹x坐标**  **\*/**  **public int bullet\_yGetter() {**  **return bullet\_y;**  **}**  **/\*\***  **\* 子弹移动**  **\*/**  **public void floatBullet() {**  **bullet\_x--;// 子弹移动**  **}**  **}** |

表13 EnemyFighter.java

|  |
| --- |
| **package FlyMe2TheMoon2;**  **import java.util.Random;**  **/\*\***  **\* The enemyFighter class of the FlyMe2TheMoon.**  **\***  **\* @author Xinyang Zhou**  **\* @version 3.0**  **\*/**  **public class EnemyFighter extends Picture {**  **private int enemy\_x, enemy\_y;// 敌机位置**  **/\*\***  **\* 敌机构造器**  **\*/**  **public EnemyFighter() {**  **createEnemyFighter();**  **}**  **/\*\***  **\* @return 敌机x坐标**  **\*/**  **public int enemy\_xGetter() {**  **return enemy\_x;**  **}**  **/\*\***  **\* @return 敌机y坐标**  **\*/**  **public int enemy\_yGetter() {**  **return enemy\_y;**  **}**  **public void enemyFighterMove() {**  **enemy\_x++; // 敌机移动**  **}**  **/\*\***  **\* 创造敌机（刷新敌机位置）**  **\*/**  **public void createEnemyFighter() {**  **Random random = new Random();**  **// 产生新敌机**  **enemy\_x = 0;**  **enemy\_y = (random.nextInt(super.widthGetter() - 4) + 3);// 宽度限制敌机出现位置**  **}**  **}** |

表14 mainFlyMe2TheMoon.java

|  |
| --- |
| **package FlyMe2TheMoon2;**  **/\*\***  **\* main of FlyMe2TheMoon**  **\***  **\* @author xinyang Zhou**  **\* @version 3.0**  **\*/**  **public class mainFlyMe2TheMoon {**  **/\*\***  **\* main函数（主模块）**  **\***  **\* @param args A reference to a string array containing command-line arguments**  **\*/**  **public static void main(String[] args) {**  **System.out.println("这里是打飞机小游戏~~");**  **// 实例操作类**  **Operate FlyMe2TheMoon = new Operate();**  **// 打飞机游戏开始运作**  **FlyMe2TheMoon.operateFighter();**  **}**  **}** |

表15 Operate.java

|  |
| --- |
| **package FlyMe2TheMoon2;**  **import java.util.ArrayList;**  **import java.util.Scanner;**  **/\*\***  **\* The Operate class of the FlyMe2TheMoon.（控制程序）**  **\***  **\* @author Xinyang Zhou**  **\* @version 3.0**  **\*/**  **public class Operate {**  **private Fighter fighter;// 飞机**  **private ArrayList<Bullet> bullets = new ArrayList<Bullet>();// 子弹**  **private ArrayList<EnemyFighter> enemyFighters = new ArrayList<EnemyFighter>();// 敌机**  **public Operate() {**  **fighter = new Fighter();**  **bullets.add(new Bullet(-1, -1));**  **enemyFighters.add(new EnemyFighter());**  **// 打印初始图像**  **fighter.drawer(fighter.fighter\_xGetter(), fighter.fighter\_yGetter(),**  **bullets, enemyFighters, fighter.scoreGetter(), fighter.HPGetter());**  **}**  **/\*\***  **\* 控制飞机移动**  **\*/**  **public void operateFighter() {**  **Scanner scanner = new Scanner(System.in);**  **int speed = 0;// 控制敌机速度与敌机刷新**  **do {// 控制飞机移动与**  **String input = scanner.nextLine();**  **switch (input) {**  **case "w": {// 上**  **if (fighter.fighter\_xGetter() > 2) {**  **fighter.fighter\_xSetter(fighter.fighter\_xGetter() - 1);**  **}**  **break;**  **}**  **case "a": {// 左**  **if (fighter.fighter\_yGetter() > 3) {**  **fighter.fighter\_ySetter(fighter.fighter\_yGetter() - 1);**  **}**  **break;**  **}**  **case "s": {// 下**  **if (fighter.fighter\_xGetter() < fighter.heightGetter() - 1) {**  **fighter.fighter\_xSetter(fighter.fighter\_xGetter() + 1);**  **}**  **break;**  **}**  **case "d": {// 右**  **if (fighter.fighter\_yGetter() < fighter.widthGetter() - 2) {**  **fighter.fighter\_ySetter(fighter.fighter\_yGetter() + 1);**  **}**  **break;**  **}**  **case " ": {// 发射子弹**  **bullets.add(new Bullet(fighter.fighter\_xGetter(), fighter.fighter\_yGetter()));**  **ifEnemySurvive(bullets, enemyFighters);// 敌机是否存活**  **break;**  **}**  **case "b": {// 发射super子弹**  **for (int x = 0; x < 4; x++) {**  **for (int y = 0; y < 3; y++) {**  **bullets.add(new Bullet(fighter.fighter\_xGetter() - x, fighter.fighter\_yGetter() + y - 1));**  **}**  **}**  **ifEnemySurvive(bullets, enemyFighters);// 敌机是否存活**  **break;**  **}**  **}**  **ifSurvive(enemyFighters);// 飞机是否存活**  **// 子弹与飞机移动依赖与用户操作**  **pcOperate(speed);// 子弹与敌机运动**  **if (speed < 5) {**  **speed++;**  **} else if (speed == 5) {**  **enemyFighters.add(new EnemyFighter());// 添加飞机**  **speed = 0;// 更新速度值**  **}**  **} while (fighter.HPGetter() > 0);**  **scanner.close();**  **}**  **public void pcOperate(int speed) {**  **for (int i = 0; i < bullets.size(); i++) {**  **bullets.get(i).floatBullet(); // 子弹移动**  **if (bullets.get(i).bullet\_xGetter() == -1) {**  **bullets.remove(bullets.get(i));**  **i--;**  **}**  **}**  **ifEnemySurvive(bullets, enemyFighters);// 敌机是否存活**  **for (int i = 0; i < enemyFighters.size(); i++) {**  **if (speed % 2 == 0) {// 达到敌机移动速度**  **enemyFighters.get(i).enemyFighterMove(); // 敌机移动**  **ifEnemySurvive(bullets, enemyFighters);// 敌机是否存活**  **ifSurvive(enemyFighters);// 飞机是否存活**  **if (enemyFighters.get(i).enemy\_xGetter() > fighter.heightGetter()) {**  **enemyFighters.remove(enemyFighters.get(i));**  **i--;**  **}**  **}**  **}**  **// 更新图像**  **fighter.drawer(fighter.fighter\_xGetter(), fighter.fighter\_yGetter(),**  **bullets, enemyFighters, fighter.scoreGetter(), fighter.HPGetter());**  **}**  **/\*\***  **\* 判断飞机是否存活**  **\*/**  **public void ifSurvive(ArrayList<EnemyFighter> enemyFighters) {**  **for (int i = 0; i < enemyFighters.size(); i++) {**  **// 飞机撞上敌机**  **if ((enemyFighters.get(i).enemy\_xGetter() == fighter.fighter\_xGetter() - 1**  **&& enemyFighters.get(i).enemy\_yGetter() == fighter.fighter\_yGetter())**  **|| (enemyFighters.get(i).enemy\_xGetter() == fighter.fighter\_xGetter()**  **&& enemyFighters.get(i).enemy\_yGetter() > fighter.fighter\_yGetter() - 3**  **&& enemyFighters.get(i).enemy\_yGetter() < fighter.fighter\_yGetter() + 3)) {**  **enemyFighters.remove(enemyFighters.get(i)); // 删除敌机**  **enemyFighters.add(new EnemyFighter()); // 产生新敌机**  **fighter.createFighter(); // 刷新飞机位置**  **fighter.HPSetter(fighter.HPGetter() - 1);// 减少生命值**  **if (fighter.HPGetter() == 0) {**  **System.out.println("\*\*\*飞机被摧毁，游戏结束！\*\*\*");**  **System.exit(0);**  **}**  **}**  **}**  **}**  **/\*\***  **\* 判断敌机是否存活**  **\*/**  **public void ifEnemySurvive(ArrayList<Bullet> bullets, ArrayList<EnemyFighter> enemyFighters) {**  **for (Bullet bullet : bullets) {**  **for (EnemyFighter enemyFighter : enemyFighters) {**  **if (bullet.bullet\_xGetter() == enemyFighter.enemy\_xGetter()**  **&& bullet.bullet\_yGetter() == enemyFighter.enemy\_yGetter()) {// 击中敌机**  **fighter.scoreSetter(fighter.scoreGetter() + 1);// 加分**  **bullet.bullet\_xSetter(-2);// 取消子弹**  **enemyFighter.createEnemyFighter();// 重置敌机**  **}**  **}**  **}**  **}**  **}** |

表16 Picture.java

|  |
| --- |
| **package FlyMe2TheMoon2;**  **import java.util.ArrayList;**  **/\*\***  **\* The Picture class of the FlyMe2TheMoon.**  **\***  **\* @author Xinyang Zhou**  **\* @version 3.0**  **\*/**  **public class Picture {**  **private int height, width;// 游戏界面尺寸**  **/\*\***  **\* 构造游戏图像框架数据**  **\*/**  **public Picture() {**  **// 游戏界面尺寸**  **height = 24;**  **width = 30;**  **}**  **/\*\***  **\* @return 游戏高**  **\*/**  **public int heightGetter() {**  **return height;**  **}**  **/\*\***  **\* @return 游戏宽**  **\*/**  **public int widthGetter() {**  **return width;**  **}**  **/\*\***  **\* 画图像**  **\***  **\* @param fighter\_x     飞机x位置**  **\* @param fighter\_y     飞机y位置**  **\* @param bullets       子弹**  **\* @param enemyFighters 敌机**  **\* @param score         分数**  **\* @param HP            生命值**  **\*/**  **public void drawer(int fighter\_x, int fighter\_y, ArrayList<Bullet> bullets,**  **ArrayList<EnemyFighter> enemyFighters, int score, int HP) {**  **for (int x = 0; x < height + 2; x++) {**  **for (int y = 0; y < width + 2; y++) {**  **boolean printed = false;**  **// 输出边框**  **if (y == 0 || y == width + 1) {**  **System.out.print("|");**  **printed = true;**  **}**  **// 画出飞机**  **if (!printed && x == fighter\_x && (y >= fighter\_y - 2 && y <= fighter\_y + 2)) {**  **System.out.print("#");**  **printed = true;**  **} else if (!printed && x == fighter\_x - 1 && y == fighter\_y) {**  **System.out.print("A");**  **printed = true;**  **} else if (!printed && x == fighter\_x + 1 && (y == fighter\_y - 1 || y == fighter\_y + 1)) {**  **System.out.print("I");**  **printed = true;**  **}**  **// 画出子弹**  **for (Bullet bullet : bullets) {**  **if (!printed && x == bullet.bullet\_xGetter() && y == bullet.bullet\_yGetter()) {**  **System.out.print("\*");**  **printed = true;**  **break;**  **}**  **}**  **// 画出敌机**  **for (EnemyFighter enemyFighter : enemyFighters) {**  **if (!printed && x == enemyFighter.enemy\_xGetter() && y == enemyFighter.enemy\_yGetter()) {**  **System.out.print("V");**  **printed = true;**  **break;**  **}**  **}**  **// 输出空格**  **if (!printed) {**  **System.out.print(" ");**  **}**  **}**  **System.out.println();**  **}**  **System.out.println("|                              |分数：" + score + " HP: " + HP);**  **}**  **}** |