# 实验六 Java多线程

**一、实验目的**

1. 掌握Java多线程的创建及其启动，多线程的两种常用创建方式及其区别；

2. 加深理解线程的生命周期及五种基本状态，分别是新建状态（New）、就绪状态（Runnable）、运行状态（Running）、阻塞状态（Blocked）、死亡状态（Dead）；

3. 掌握引起Java线程阻塞的主要方法，如：jion()方法、sleep()方法、yeild()方法；

4. 掌握线程安全及其解决机制，如：同步方法、同步代码块等。

**二、实验内容**

1. 通过继承Thread类的方法创建两个线程，在Thread构造方法中指定线程的名称分别为“Thread\_one”和”Thread\_two”；其中一个线程向控制台输出奇数数列（1、3、5、7、……），时间间隔0.5秒；另一个线程向控制台输出偶数数列（2、4、6、8、……），时间间隔0.3秒。

2. 通过实现Runnable接口的方法创建一个新线程，要求main线程打印10次“main”，时间间隔0.2秒；新线程打印5次“new”，时间间隔0.4秒。

3. 下列程序代码是模拟火车售票情况，开启两个窗口销售10张火车票，程序一次运行结果如图1所示。请思考：

程序运行是否有错，若有错，错在哪里？应该如何改正？

**public** **class** SaleTicketThread **extends** Thread{

**private** String name;

**private** **int** tickets=10;

**public** SaleTicketThread() {

}

**public** SaleTicketThread(String name) {

**this**.name=name;

}

**public** **int** getTickets() {

**return** tickets;

}

**public** **void** setTickets(**int** tickets) {

**this**.tickets = tickets;

}

@Override

**public** **void** run() {

**while**(tickets>0){

System.***out***.println(name+" 正在销售第 "+tickets-- +" 票");

try{ Thread.sleep(200); }catch(InterruptedException e){}

}

}

**public** **static** **void** main(String[] args) {

SaleTicketThread sale1=**new** SaleTicketThread("窗口1");

SaleTicketThread sale2=**new** SaleTicketThread("窗口2");

sale1.start();

sale2.start();

}

}

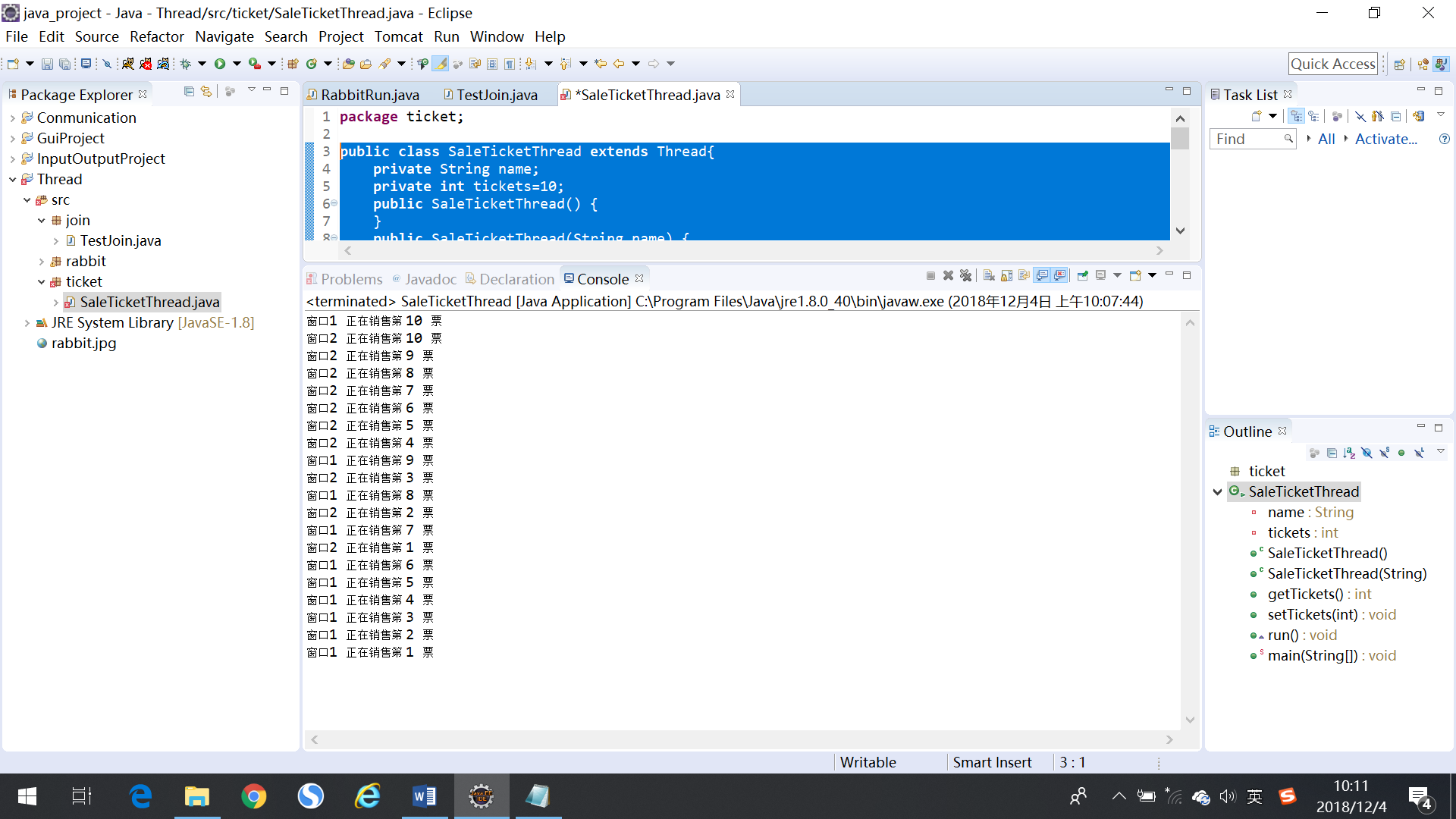


图1 第3题某次运行情况

**错的！同一个进程的多的线程，访问共同的数据冲突，sale1和sale2线程的同时访问tickets数据时冲突。为程序添加synchronized,使其sale1线程的访问tickets数据的时候sale2线程不访问tickets数据，以及为tickets加上static变量，使其实时更改。**

4.下列程序代码是使用实现Runnable接口模拟火车售票情况，售票员休息一秒再卖出一张火车票。现开启两个窗口销售10张火车票，程序一次运行结果如图2所示。请思考：

程序运行是否有错，若有错，错在哪里？应该如何改正？

**public** **class** SaleTikcetRunnable **implements** Runnable {

**private** **int** tickets=10;

@Override

**public** **void** run() {

**while**(tickets>0){

**try** {

Thread.*sleep*(1000);

System.***out***.println(Thread.*currentThread*().getName()+" 正在

销售第 "+tickets-- +" 票");

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

**public** **static** **void** main(String[] args) {

SaleTikcetRunnable ticket=**new** SaleTikcetRunnable();

Thread sale1=**new** Thread(ticket, "窗口1");

Thread sale2=**new** Thread(ticket, "窗口2");

sale1.start();

sale2.start();

}

}

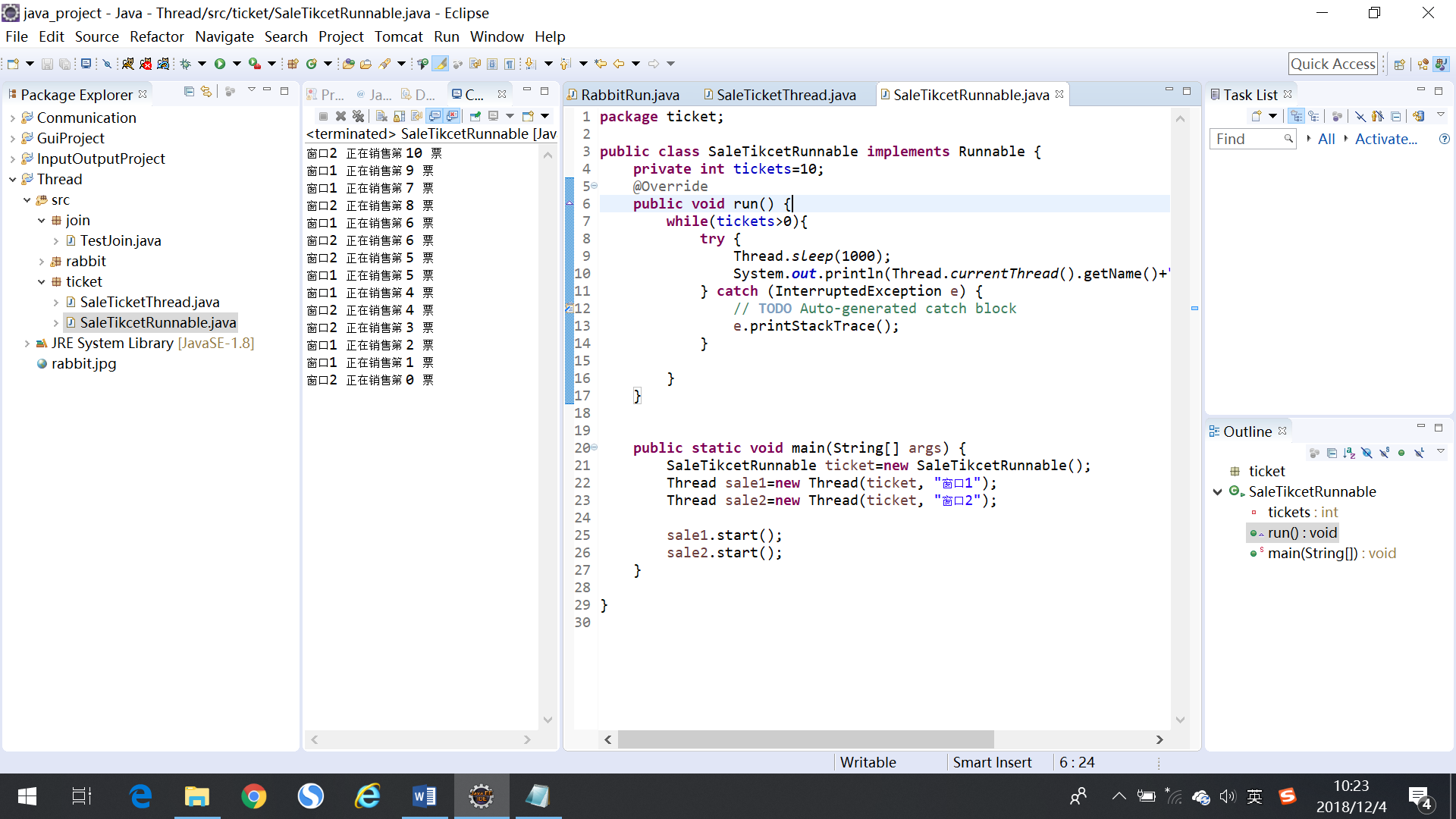


图2 第4题某次运行情况

**错的！**

5. 有一个抽奖池，该抽奖池中存放了奖励的金额，该抽奖池用一个数组int[] arr = {10,5,20,50,100,200,500,800,2,80,300};

创建两个抽奖箱(线程)设置线程名称分别为“抽奖箱1”，“抽奖箱2”，随机从arr数组中获取奖项元素并打印在控制台上，运行结果如图3所示。

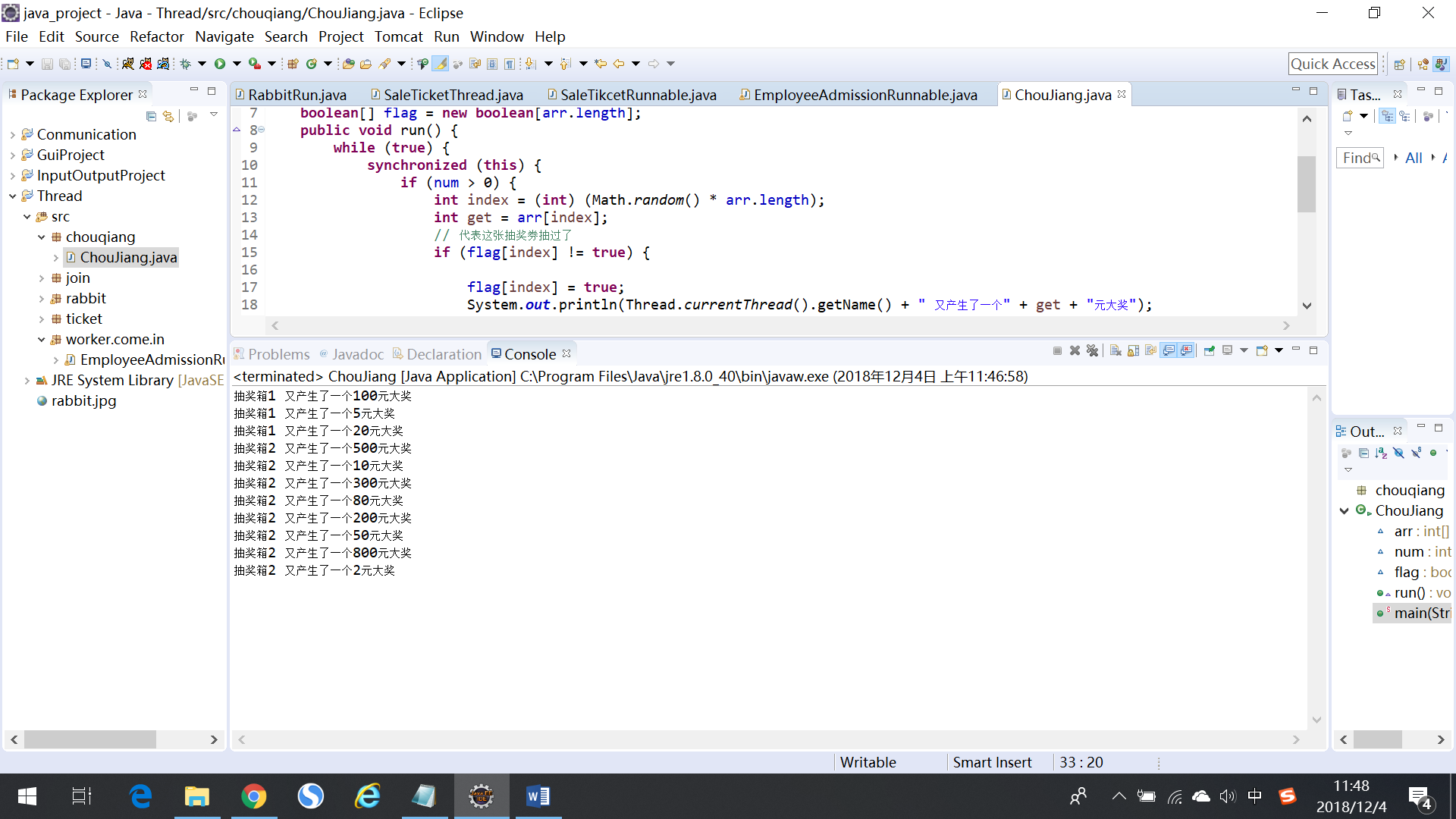


图3 模拟抽奖的运行结果

下列程序代码不完整，请补充完成。

**public** **class** ChouJiang **implements** Runnable {

//所有的奖券

**int**[] arr = { 10, 5, 20, 50, 100, 200, 500, 800, 2, 80, 300 };

**int** num = arr.length;

//记录对应的奖券是否抽过，为true表示已抽过、为false表示未抽过

**boolean**[] flag = **new** **boolean**[arr.length];

**public** **void** {

**while** (**true**) {

**synchronized** (**this**) {

**if** (num > 0) {

//产生指定范围（与奖券下标相一致）内的随机数

**int** index = (**int**) ( );

**int** get = arr[index];

// 代表这张抽奖券抽过了

**if** (flag[index] != **true**) {

flag[index] = **true**;

System.***out***.println( + " 又产生了一个" + get + "元大奖");

num--;

}

}**else**{

**return** ; //自动结束线程

}

}

}

}

**public** **static** **void** main(String[] args) {

ChouJiang c = **new** ChouJiang();

Thread t1 = **new** Thread(c, );

Thread t2 = **new** Thread(c, "抽奖箱2");

t1.start();

;

}

}

以下为选做题：

6. 编写如图4所示的界面，当程序运行时：

（1）每隔两秒钟在显示字母区域随机显示一个字母（如图所示，显示的字母是“g”）；

（2）用户在文本框中使用输入相应的字母，如果输入正确则得1分，否则得0分；

（3）并将用户的得分累计显示在的得分栏。

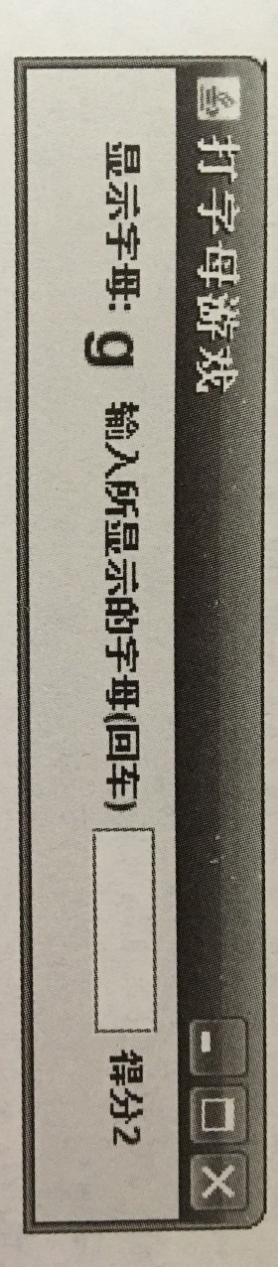


图4 字母游戏运行效果图

7. 模拟实现兔子散步，如图5所示，启动程序：

（1）单击开始按钮，兔子从左端往右方散步，当走到窗体的边界时，自动结束散步；

（2）单击停止按钮，兔子停止散步。

（3）思考：兔子在水平方向来来回回散步，如何实现？

（4）思考：兔子漫无目的地随机散步，如何实现？

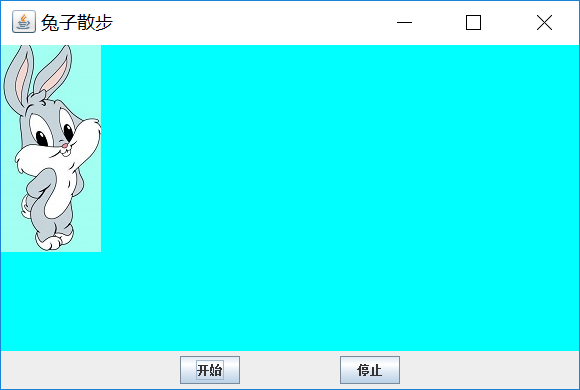


图5 兔子散步

8.请编写程序实现龟兔赛跑。

9.请编程实现打地鼠游戏。