

廈門大學



信息学院软件工程系

《JAVA 程序设计》实验报告

实验四

姓名：庾晓萍

学号：20420192201952

学院：信息学院

专业：软件工程

完成时间：2022/3/18

一、实验目的及要求

（一）实验目的

- 1、熟悉数组的使用
- 2、熟悉简单的 JavaFX 图形界面

（二）实验要求

- 1、下周前五前将工程文档和实验报告打包上传到 FTP

二、实验题目及实现过程

一、基本题目：

题目 1：

(Game of Craps) Write an application that runs 1,000,000 games of craps and answers the following questions:

- (a) How many games are won on the first roll, second roll, ..., twentieth roll and after the twentieth roll?
- (b) How many games are lost on the first roll, second roll, ..., twentieth roll and after the twentieth roll?
- (c) What are the chances of winning at craps? [Note: You should discover that craps is one of the fairest casino games. What do you suppose this means?]
- (d) What is the average length of a game of craps?
- (e) Do the chances of winning improve with the length of the game?

二、Craps 的定义

Craps is a game played with a pair of dice. In the game of craps, the shooter (the player with the dice) rolls a pair of dice and the number of spots showing on the two upward faces are added up. If the opening roll (called the ‘coming out roll’) is a 7 or 11, the shooter wins the game. If the opening roll results in a 2 (snake eyes), 3 or 12 (box cars), the shooter loses, otherwise known as ‘crapping out’. If the shooter rolls a 4, 5, 6, 8, 9 or 10 on the opening roll, then he or she must roll the same number before rolling a 7 to win the game. For example, if the shooter rolls a 6 on the come out roll, a 10 on the second roll and a 7 on the third roll, the shooter loses since he rolled a 7 before rolling another 6. If, however, he rolled a 6 on the third roll, he wins the game.

(一) 实验环境

操作系统: Windows 10;

IDE: Eclipse Java 2018-12

编程语言: Java;

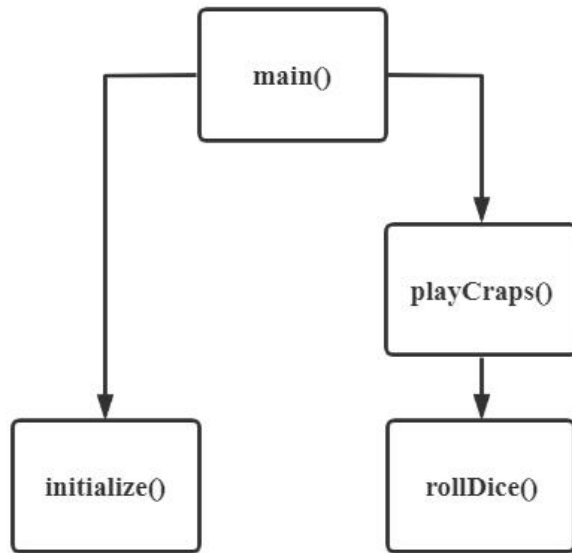
(二) 实现过程

(1) 设计类

设计了名为 `GameCraps` 的类, 其中包括 `main` 方法, 用来实现用户输入输出与方法调用。用户输入一个整数代表游戏进行的轮数。`main` 函数将调用 `initialize` 方法进行初始化, 即将储存各种数据的变量置零, 再在循环中调用 `playCraps` 方法来完成每一局的游戏, 而在 `playCraps` 方法中, 每次扔掷骰子将通过 `rollDice` 方法来完成。

(2) 调用关系

调用关系如下图，即 `main` 方法可以对 `initialize` 方法和 `playCraps` 方法进行调用
`playCraps` 方法可以对 `rollDice` 方法进行调用。



(3) 问题思考

-----Start-----

请输入游戏次数（输入0表示退出）：1000000

游戏进行共1000000轮：

你赢得游戏493789轮，获胜的概率为：49.38%

平均每局游戏所用步数：3.37

赢得游戏所用总步数：1448540

输掉游戏所用总步数：1922370

赢得/输掉一轮游戏所用步数分布如下：

走 1步： 赢得游戏： 222608轮	输掉游戏： 110637轮
走 2步： 赢得游戏： 77462轮	输掉游戏： 111622轮
走 3步： 赢得游戏： 55451轮	输掉游戏： 79269轮
走 4步： 赢得游戏： 39177轮	输掉游戏： 56920轮
走 5步： 赢得游戏： 28096轮	输掉游戏： 41104轮
走 6步： 赢得游戏： 19943轮	输掉游戏： 29685轮
走 7步： 赢得游戏： 14524轮	输掉游戏： 21562轮
走 8步： 赢得游戏： 10239轮	输掉游戏： 15549轮
走 9步： 赢得游戏： 7375轮	输掉游戏： 11060轮
走10步： 赢得游戏： 5376轮	输掉游戏： 7905轮
走11步： 赢得游戏： 3857轮	输掉游戏： 5758轮
走12步： 赢得游戏： 2636轮	输掉游戏： 4135轮
走13步： 赢得游戏： 2002轮	输掉游戏： 2990轮
走14步： 赢得游戏： 1427轮	输掉游戏： 2122轮
走15步： 赢得游戏： 1019轮	输掉游戏： 1659轮
走16步： 赢得游戏： 698轮	输掉游戏： 1175轮
走17步： 赢得游戏： 484轮	输掉游戏： 847轮
走18步： 赢得游戏： 387轮	输掉游戏： 597轮
走19步： 赢得游戏： 257轮	输掉游戏： 451轮
20以上： 赢得游戏： 771轮	输掉游戏： 1164轮

(a) How many games are won on the first roll, second roll, ..., twentieth roll and after the twentieth roll?

由游戏运行结果可知，进行 1000000 轮游戏，其中走一步赢得游戏的局数为 222608 轮、走两步赢得游戏的轮数为 77462 轮……（具体见上图）

(b) How many games are lost on the first roll, second roll, ..., twentieth roll and after the twentieth roll?

由游戏运行结果可知，进行 1000000 轮游戏，其中走一步输掉游戏的局数为 110637 轮、走两步输掉游戏的轮数为 111622 轮……（具体见上图）

(c) What are the chances of winning at craps? [Note: You should discover that craps is one of the fairest casino games. What do you suppose this means?]

由游戏运行结果可知，进行 1000000 轮游戏，计算得到游戏获胜的概率为 49.38%，接近 0.5，可以说 craps 是相对公平的游戏，即获胜概率和失败概率几乎相等。

(d) What is the average length of a game of craps?

由游戏运行结果可知，进行 1000000 轮游戏，平均每轮游戏的步数为 3.37，即平均三步多可以完成一局游戏。

(e) Do the chances of winning improve with the length of the game?

```
-----Start-----  
请输入游戏次数（输入0表示退出）： 100  
游戏进行共100轮：  
你赢得游戏46轮，获胜的概率为： 46.00%
```

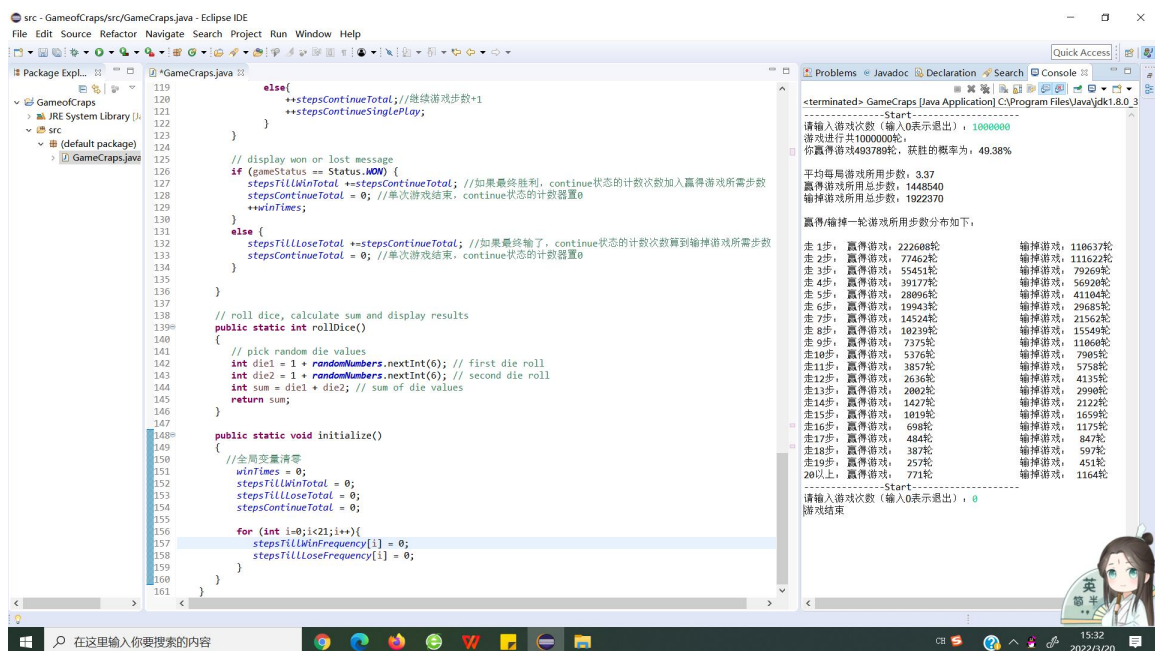
```
-----Start-----  
请输入游戏次数（输入0表示退出）： 10000  
游戏进行共10000轮：  
你赢得游戏4897轮，获胜的概率为： 48.97%
```

```
请输入游戏次数（输入0表示退出）： 1000000  
游戏进行共1000000轮：  
你赢得游戏493370轮，获胜的概率为： 49.34%
```

运行代码，发现随着游戏局数增多，获胜概率确实增加了。

（三） 过程截图

（1）全屏截图



The screenshot shows the Eclipse IDE with the `GameCraps.java` file open. The code implements a craps game simulation. The console output shows the results of running the program, including the number of games played, the total number of steps, the win/loss frequency, and a table of win/loss steps.

```
src - GameofCraps/src/GameCraps.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer: src - GameofCraps - GameCraps.java
119         else{
120             ++stepsContinueTotal; //继续游戏步数+1
121             ++stepsContinueSinglePlay;
122         }
123     }
124     // display won or lost message
125     if (gameStatus == Status.WON) {
126         stepsFillWinTotal += stepsContinueTotal; //如果最终胜利, continue状态的计数次数加入赢得游戏所需步数
127         stepsContinueTotal = 0; //单次游戏结束, continue状态的计数置0
128         ++winTimes;
129     }
130     else {
131         stepsFillLoseTotal += stepsContinueTotal; //如果最终输了, continue状态的计数次数算到输掉游戏所需步数
132         stepsContinueTotal = 0; //单次游戏结束, continue状态的计数置0
133     }
134 }
135
136 // roll dice, calculate sum and display results
137 public static int rollDice()
138 {
139     // pick random die values
140     int die1 = 1 + randomNumbers.nextInt(6); // first die roll
141     int die2 = 1 + randomNumbers.nextInt(6); // second die roll
142     int sum = die1 + die2; // sum of die values
143     return sum;
144 }
145
146 public static void initialize()
147 {
148     //全局变量清零
149     winTimes = 0;
150     stepsFillWinTotal = 0;
151     stepsFillLoseTotal = 0;
152     stepsContinueTotal = 0;
153     for (int i=0; i<21; i++){
154         stepsFillWinFrequency[i] = 0;
155         stepsFillLoseFrequency[i] = 0;
156     }
157 }
158
159
160 }
161
```

Console Output:

```
<terminated> GameCraps [Java Application] C:\Program Files\Java\jdk1.8.0_3
Start
请输入游戏次数 (输入0表示退出): 1000000
游戏进行共1000000局;
你赢得游戏463789局, 获胜的概率为: 49.38%

平均每局游戏所用步数: 3.37
赢得游戏所用总步数: 1448540
输掉游戏所用总步数: 1922370

赢得/输掉一轮游戏所用步数分布如下:

走 1步: 赢得游戏: 222608轮      输掉游戏: 110637轮
走 2步: 赢得游戏: 77462轮      输掉游戏: 111622轮
走 3步: 赢得游戏: 55451轮      输掉游戏: 79269轮
走 4步: 赢得游戏: 39177轮      输掉游戏: 56920轮
走 5步: 赢得游戏: 28096轮      输掉游戏: 41104轮
走 6步: 赢得游戏: 19943轮      输掉游戏: 29685轮
走 7步: 赢得游戏: 14524轮      输掉游戏: 21562轮
走 8步: 赢得游戏: 10239轮      输掉游戏: 15549轮
走 9步: 赢得游戏: 7375轮       输掉游戏: 11668轮
走10步: 赢得游戏: 5376轮       输掉游戏: 7905轮
走11步: 赢得游戏: 3857轮       输掉游戏: 5758轮
走12步: 赢得游戏: 2636轮       输掉游戏: 4135轮
走13步: 赢得游戏: 2002轮       输掉游戏: 2908轮
走14步: 赢得游戏: 1427轮       输掉游戏: 2122轮
走15步: 赢得游戏: 1019轮       输掉游戏: 1659轮
走16步: 赢得游戏: 698轮        输掉游戏: 1175轮
走17步: 赢得游戏: 484轮        输掉游戏: 847轮
走18步: 赢得游戏: 387轮        输掉游戏: 597轮
走19步: 赢得游戏: 257轮        输掉游戏: 451轮
走20以上: 赢得游戏: 771轮      输掉游戏: 1164轮

请输入游戏次数 (输入0表示退出): 0
游戏结束
```

(2) 运行结果

1. 用户输入 1000000 表示进行游戏 1000000 局, 输出游戏进行总轮数、玩家赢得游戏局数、玩家获胜概率、平均每局游戏所用步数、赢得游戏所用总步数、输掉游戏所用总步数, 最后以表格形式输出赢得/输掉一轮游戏所用步数的分布。

```

-----Start-----
请输入游戏次数（输入0表示退出）： 1000000
游戏进行共1000000轮：
你赢得游戏493789轮，获胜的概率为： 49.38%

平均每局游戏所用步数： 3.37
赢得游戏所用总步数： 1448540
输掉游戏所用总步数： 1922370

赢得/输掉一轮游戏所用步数分布如下：

走 1步： 赢得游戏： 222608轮          输掉游戏： 110637轮
走 2步： 赢得游戏： 77462轮           输掉游戏： 111622轮
走 3步： 赢得游戏： 55451轮           输掉游戏： 79269轮
走 4步： 赢得游戏： 39177轮           输掉游戏： 56920轮
走 5步： 赢得游戏： 28096轮           输掉游戏： 41104轮
走 6步： 赢得游戏： 19943轮           输掉游戏： 29685轮
走 7步： 赢得游戏： 14524轮           输掉游戏： 21562轮
走 8步： 赢得游戏： 10239轮           输掉游戏： 15549轮
走 9步： 赢得游戏： 7375轮            输掉游戏： 11060轮
走10步： 赢得游戏： 5376轮            输掉游戏： 7905轮
走11步： 赢得游戏： 3857轮            输掉游戏： 5758轮
走12步： 赢得游戏： 2636轮            输掉游戏： 4135轮
走13步： 赢得游戏： 2002轮            输掉游戏： 2990轮
走14步： 赢得游戏： 1427轮            输掉游戏： 2122轮
走15步： 赢得游戏： 1019轮            输掉游戏： 1659轮
走16步： 赢得游戏： 698轮             输掉游戏： 1175轮
走17步： 赢得游戏： 484轮             输掉游戏： 847轮
走18步： 赢得游戏： 387轮             输掉游戏： 597轮
走19步： 赢得游戏： 257轮             输掉游戏： 451轮
20以上： 赢得游戏： 771轮            输掉游戏： 1164轮

-----Start-----
请输入游戏次数（输入0表示退出）： 0
游戏结束

```

题目 2:

(Airline Reservations System) A small airline has just purchased a computer for its new automated reservations system. You've been asked to develop the new system. You're to write an application to assign seats on each flight of the airline's only plane (capacity: 10 seats).

Your application should display the following alternatives: Please type 1 for First Class and Please type 2 for Economy. If the user types 1, your application should assign a seat in the first- class section (seats 1–5). If the user types 2, your application should assign a seat in the economy section (seats 6–10). Your

application should then display a boarding pass indicating the person's seat number and whether it's in the first-class or economy section of the plane.

Use a one-dimensional array of primitive type boolean to represent the seating chart of the plane. Initialize all the elements of the array to false to indicate that all the seats are empty. As each seat is assigned, set the corresponding element of the array to true to indicate that the seat is no longer available.

Your application should never assign a seat that has already been assigned. When the economy section is full, your application should ask the person if it's acceptable to be placed in the first-class section (and vice versa). If yes, make the appropriate seat assignment. If no, display the message "Next flight leaves in 3 hours."

(一) 实验环境

操作系统: Windows 10;

IDE: Eclipse Java 2018-12

编程语言: Java;

(二) 实现过程

(1) 设计类

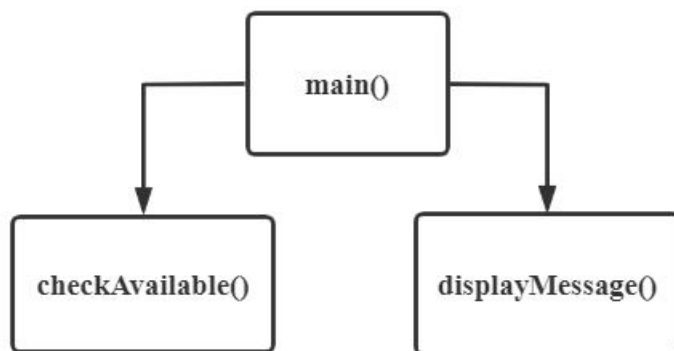
设计了用于投掷航空公司预订系统的 `AirlineReservationsSystem` 类, 包括 `main` 方法, 用来显示选择菜单, 读取用户输入, 并调用相关方法。键入 1 为头等舱, 键入 2 为经济舱。如果用户输入 1, 程序在座位 1-5 分配一个座位。如果用户输入 2, 程序在座位 6-10 分配一个座位。其中 `displayMessage` 用于输出登机卡信息, 显示乘客的座位号码, 以及在头等舱还是经济舱。

`checkAvailable` 用于检查可以使用的座位。使用原始类型布尔值的一维数组来表示平面的座位图。将数组中的所有元素初始化为 `false`, 以表示所有座位都

是空的。当分配每个座位时，将数组的相应元素设置为 `true`，以表明该座位不再可用。

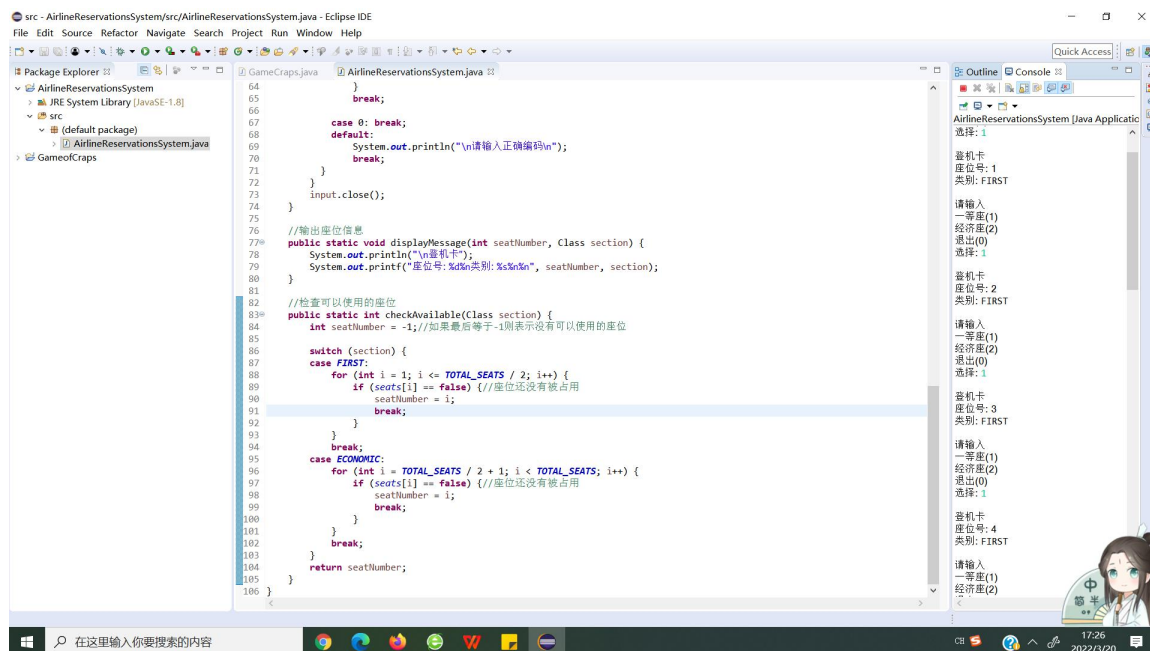
(2) 调用关系

调用关系如下图，即 `main` 方法可以对 `displayMessage` 方法和 `checkAvailable` 方法进行调用。



(三) 过程截图

(1) 全屏截图



(2) 运行结果

1. 用户选择座位类型，输出登机卡

```
请输入
一等座(1)
经济座(2)
退出(0)
选择: 1

登机卡
座位号: 1
类别: FIRST
```

2. 某类座位已满时，询问是否要更换其他种类的座位

```
请输入
一等座(1)
经济座(2)
退出(0)
选择: 1

一等座已满!
你需要更换成经济座吗?
1 - Yes
2 - Not
输入选择: 1

登机卡
座位号: 6
类别: ECONOMIC
```

3. 没有更多座位时，输出航班已满

```
请输入
一等座(1)
经济座(2)
退出(0)
选择: 2

没有更多座位了，航班已满
```

题目 3:

(Turtle Graphics) The Logo language made the concept of turtle graphics famous. Imagine a mechanical turtle that walks around the room under the control of a Java application. The turtle holds a pen in one of two positions, up or down. While the pen is down, the turtle traces out shapes as it moves, and while the pen is up, the turtle moves about freely without writing anything. In this problem, you'll simulate the operation of the turtle and create a computerized sketchpad.

Use a 20-by-20 array floor that's initialized to zeros. Read commands from an array that contains them. Keep track of the current position of the turtle at all times and whether the pen is currently up or down. Assume that the turtle always starts at position (0, 0) of the floor with its pen up. The set of turtle commands your application must process are shown in Fig. 7.29.

Command	Meaning
1	Pen up
2	Pen down
3	Turn right
4	Turn left
5,10	Move forward 10 spaces (replace 10 for a different number of spaces)
6	Display the 20-by-20 array
9	End of data (sentinel)

Fig. 7.29 | Turtle graphics commands.

Suppose that the turtle is somewhere near the center of the floor. The following “program” would draw and display a 12-by-12 square, leaving the pen in the up position:

```

2
5,12
3
5,12
3

```

5,12

3

5,12

1

6

9

As the turtle moves with the pen down, set the appropriate elements of array floor to 1s. When the 6 command (display the array) is given, wherever there's a 1 in the array, display an asterisk or any character you choose. Wherever there's a 0, display a blank.

Write an application to implement the turtle graphics capabilities discussed here. Write several turtle graphics programs to draw interesting shapes. Add other commands to increase the power of your turtle graphics language.

（一）实验环境

操作系统：Windows 10;

IDE: Eclipse Java 2018-12

编程语言：Java;

（二）实现过程

（1）设计类

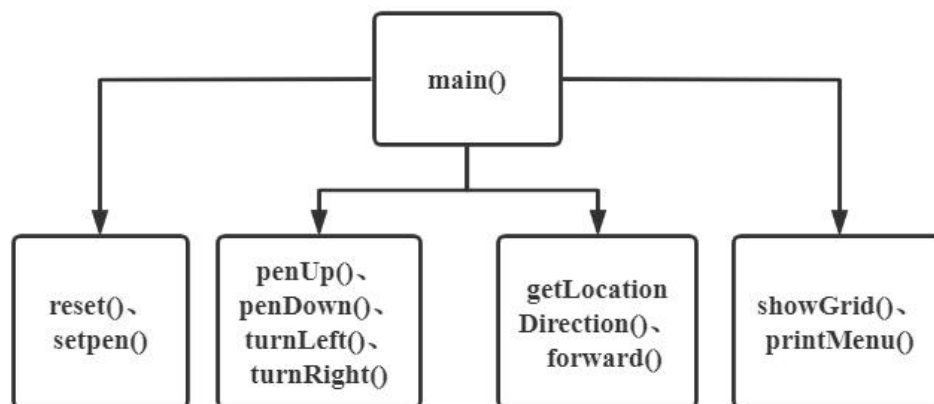
① 设计了用于绘制海龟图形的 TurtleGraphics 类。构造函数可以调用 reset 方法，将网格和坐标重置为默认值。setpen 方法可以更改画笔图案。penUp 和 penDown 可以设置画笔是否处于绘制状态。turnLeft、turnRight 可以改变画笔所朝的方向。getLocationDirection 可以打印画笔所在位置与画笔的方向。

forward 可以使画笔在 drawDir 方向移动 n 步。showGrid 可以打印最终绘制的图形。printMenu 可以打印命令菜单。

② 设计了用于完成一次绘图的 Drawer 类。Drawer 类中 new 了一个 TurtleGraphics 对象，在 main 方法调用该对象的各种方法实现绘图。

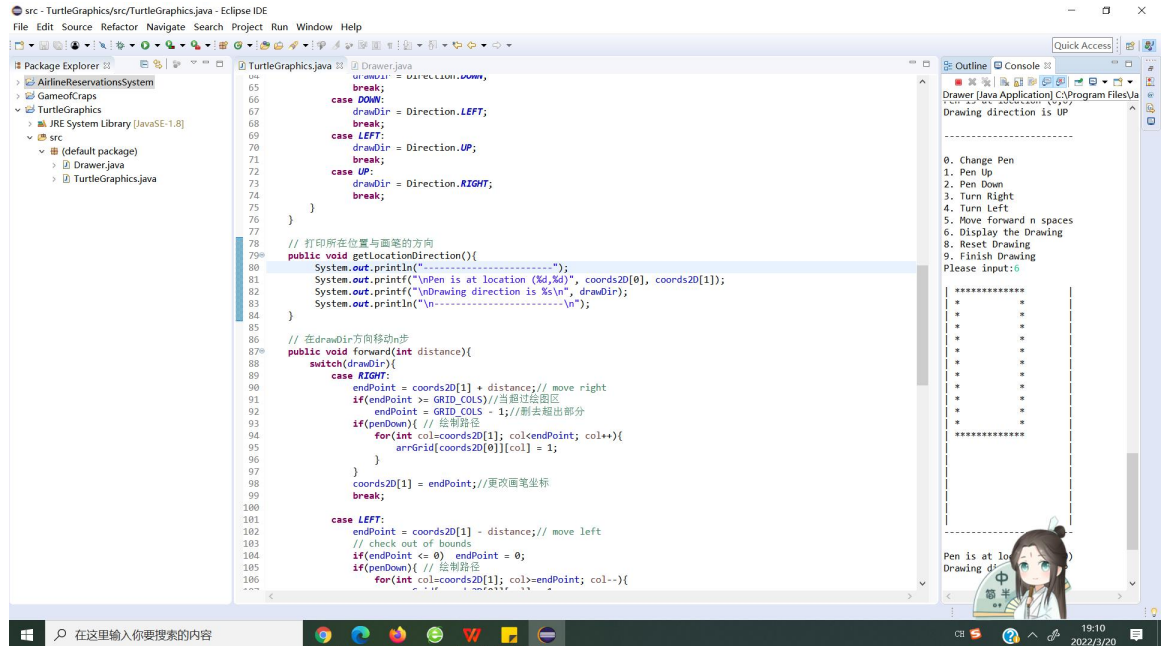
(2) 调用关系

调用关系如下图，即 Drawer 类的主方法可以对 TurtleGraphics 类中的 reset、setpen、penUp、penDown、turnLeft、turnRight、getLocationDirection、forward、showGrid、showGrid 方法进行调用。



(三) 过程截图

(1) 全屏截图



(2) 运行结果

- 提示画笔的位置，以及画笔的位置，用户选择操作 2，即落下画笔。

```
-----
Pen is at location (0,0)
Drawing direction is RIGHT
-----
```

```
0. Change Pen
1. Pen Up
2. Pen Down
3. Turn Right
4. Turn Left
5. Move forward n spaces
6. Display the Drawing
8. Reset Drawing
9. Finish Drawing
Please input:2
-----
```

- 输入三次命令 5 12 3，代表连续绘制 3 组*（12 个），绘制后右转。再输入 6 得到打印图形。

```
-----  
0. Change Pen  
1. Pen Up  
2. Pen Down  
3. Turn Right  
4. Turn Left  
5. Move forward n spaces  
6. Display the Drawing  
8. Reset Drawing  
9. Finish Drawing  
Please input:6
```

```
*****  
*           *  
*           *  
*           *  
*           *  
*           *  
*           *  
*           *  
*           *  
*           *  
*           *  
*           *  
*           *  
*****
```

```
-----  
Pen is at location (0,0)  
Drawing direction is UP  
-----
```

3. 添加了命令可以更改画图案


```
-----  
0. Change Pen  
1. Pen Up  
2. Pen Down  
3. Turn Right  
4. Turn Left  
5. Move forward n spaces  
6. Display the Drawing  
8. Reset Drawing  
9. Finish Drawing  
Please input:0  
  
Current pen: *  
Enter new pen character: 8  
|-----  
  
Pen is at location (0,0)  
Drawing direction is UP
```

题目 4:

(Enhancing Class Time2) Modify class Time2 to include a tick method that increments the time stored in a Time2 object by one second. Provide method incrementMinute to increment the minute by one and method incrementHour to increment the hour by one. Write a program that tests the tick method, the incrementMinute method and the incrementHour method to ensure that they work correctly. Be sure to test the following cases:

- (a) incrementing into the next minute,
- (b) incrementing into the next hour and
- (c) incrementing into the next day (i.e., 11:59:59 PM to 12:00:00 AM).

(一) 实验环境

操作系统: Windows 10;

IDE: Eclipse Java 2018-12

编程语言：Java;

（二）实现过程

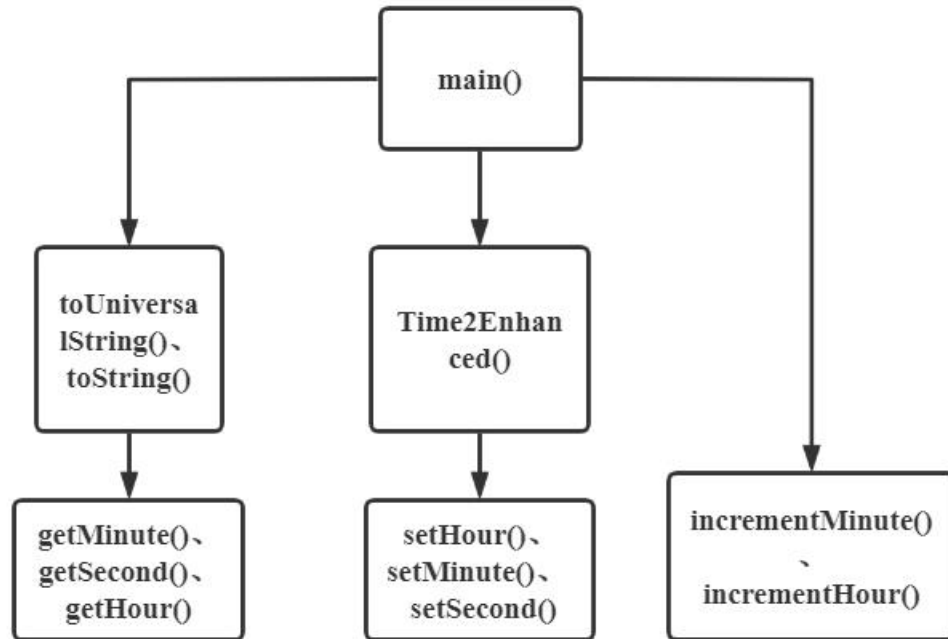
（1）设计类

① 设计了用于存储和修改时间的 `Time2Enhanced` 类。构造函数可以通过传入不同数量的参数来将每个实例变量进行不同的初始化操作。`setTime` 方法可以调用 `setHour`、`setMinute`、`setSecond` 方法来检验并设置小时、分钟、秒。`incrementMinute`、`incrementHour` 方法可以用来增加小时和分钟的时间。`toUniversalString` 和 `toString` 方法可以得到通用时间和标准时间。`getMinute`、`getSecond`、`getHour` 方法可以得到当前时间的分钟、秒和小时。

② 设计了用于完成时间滴答测试的 `Test` 类。`Drawer` 类中 `new Time2Enhanced` 对象，在 `main` 方法调用 `Time2Enhanced` 的构造函数初始化、调用 `toUniversalString` 方法转化成通用时间，调用 `toString` 方法转换为标准时间。调用 `incrementMinute` 方法实现分钟的增加，调用 `incrementHour` 方法实现小时的增加。

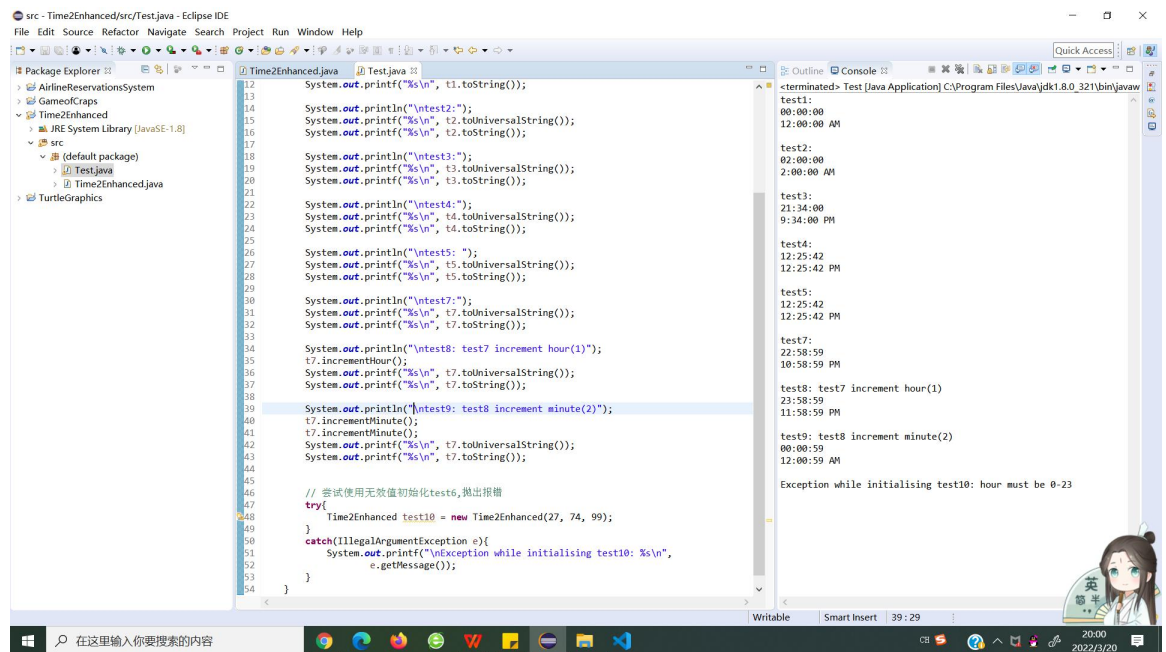
（2）调用关系

调用关系如下图，即 `Test` 类的 `main` 方法可以对 `Time2Enhanced` 类中的构造函数、`toUniversalString` 和 `toString` 方法、`incrementMinute`、`incrementHour` 方法进行调用。构造函数可以对 `Time2Enhanced` 类中 `setHour`、`setMinute`、`setSecond` 方法进行调用。`toUniversalString` 和 `toString` 方法可以调用 `Time2Enhanced` 类中的 `getMinute`、`getSecond`、`getHour` 方法。



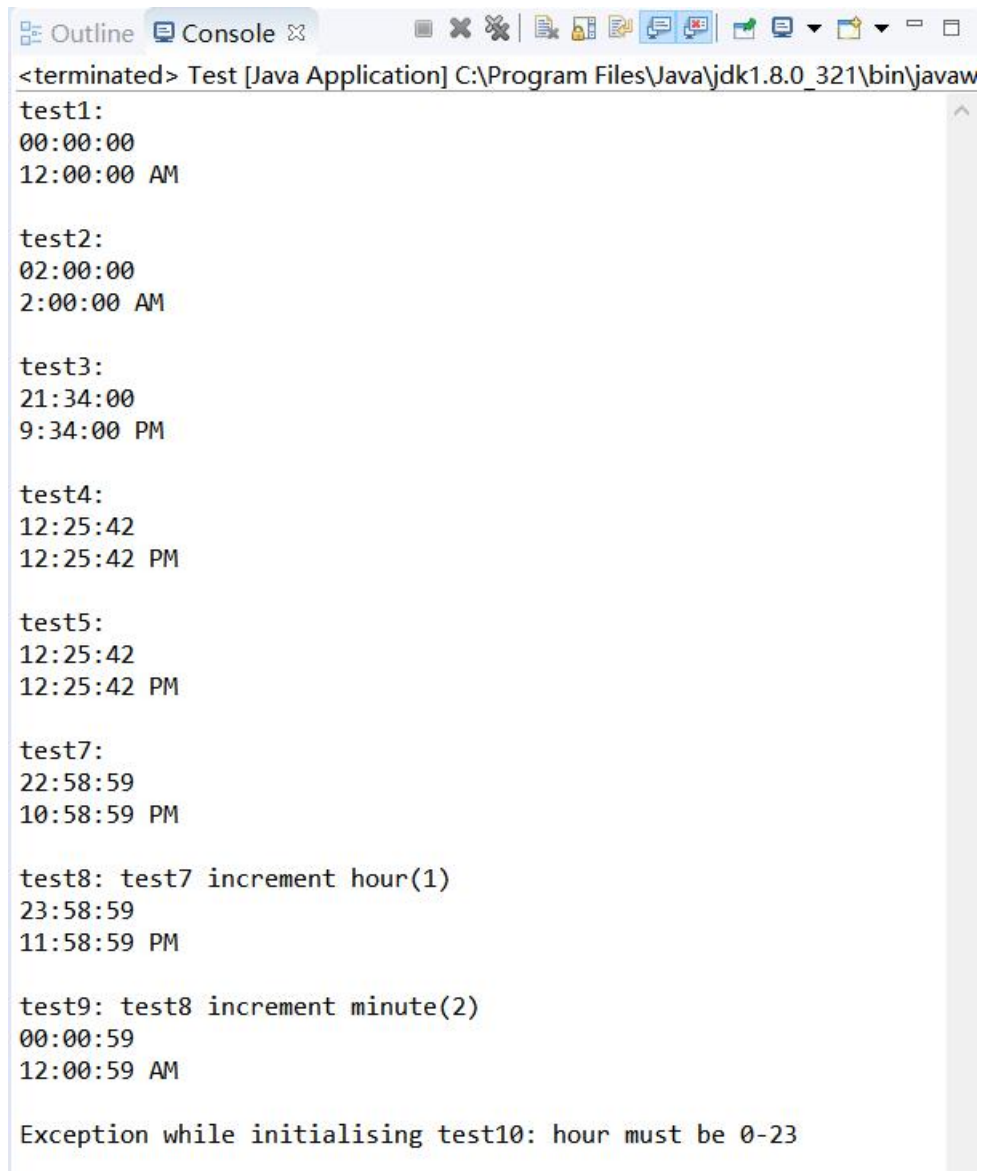
(三) 过程截图

(1) 全屏截图



(2) 运行结果

1. test1 是所有参数设为默认值的结果，test2 是小时是自定义，其他参数设置为默认值的结果，test3 是小时和分钟自定义，秒设置为默认值的结果。test4 是小时、分钟、秒都是自定义的结果。test5 是由 test4 指定得到。
2. test8 成功测试了增加时间到接下来的一个小时（即晚上 10:58:59 至晚上 11:58:59）
3. test9 成功测试了增加时间到第二天（即晚上 11:58:59 至上午 12:00:59）。
4. 尝试使用无效值初始化 test6，成功抛出报错 “hour must be 0-23”



```
<terminated> Test [Java Application] C:\Program Files\Java\jdk1.8.0_321\bin\javaw
test1:
00:00:00
12:00:00 AM

test2:
02:00:00
2:00:00 AM

test3:
21:34:00
9:34:00 PM

test4:
12:25:42
12:25:42 PM

test5:
12:25:42
12:25:42 PM

test7:
22:58:59
10:58:59 PM

test8: test7 increment hour(1)
23:58:59
11:58:59 PM

test9: test8 increment minute(2)
00:00:59
12:00:59 AM

Exception while initialising test10: hour must be 0-23
```

题目 5:

(Addition App) Create a JavaFX version of the addition program. Use two TextFields to receive the user's input and a Button to initiate the calculation. Display the results in a Label. Since TextField method getText returns a String, you must convert the String the user enters to an int for use in calculations. Recall that the static method parseInt of class Integer takes a String argument representing an integer and returns the value as an int.

（一）实验环境

操作系统：Windows 10;

IDE：Eclipse Java 2018-12

编程语言：Java;

（二）实现过程

（1）设计类

① AdditionApp 类中，继承了 Application 抽象类并重写了 start()方法，在该方法中，Stage 就是 JavaFX 工具中用来表示整个图形工具界面窗口的类，在该类中需要加入一个 Scene（场景）来进行填充，而所有的组件、元素都是构建在 Scene 中的。另外，在 JavaFX 8 中支持代码与布局和样式分离，所以在文件中通过 FXMLLoader 的 load()方法引入了一个外联的 AdditionApp.fxml 文件，在此 fxml 文件中就可以专心编写图形界面布局和组件相关功能。

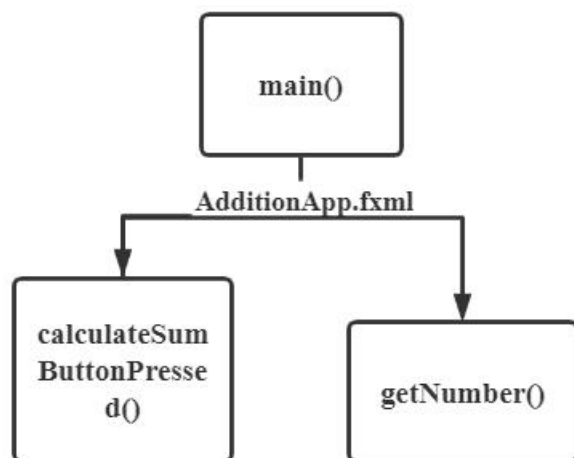
```
1 import java.io.IOException;
2
3
4
5
6
7
8
9 public class AdditionApp extends Application {
10     @Override
11     //重写start方法
12     public void start(Stage stage) throws IOException {
13         //使用FXMLLoader加载器，加载名为AdditionApp.fxml”的文件
14         Parent root = FXMLLoader.load(getClass().getResource("AdditionApp.fxml"));
15         Scene scene = new Scene(root); //创建一个场景
16         stage.setTitle("Add"); //为图形界面窗口设置标题
17         stage.setScene(scene); //为图形界面窗口设置场景
18         stage.show(); //将图形界面设置为可见
19     }
20     public static void main(String[] args) {
21         //通过Application抽象类的launch()方法启动程序
22         launch(args);
23     }
24 }
```

② AdditionApp.fxml 文件中设置了画布的宽度为 250，字体大小为 18size，控制文件为 AdditionAppControllor.java。同时设置了两个 TextField，一个的 id 是 number1TextField，显示文字为 Number 1。另一个的 id 是 number2TextField，显示文字为 Number 2。设置了一个 Button，显示文本是 Calculate。

③ 在 AdditionAppController 类中，通过 calculateSumButtonPressed 方法来计算输入的两个数的相加结果。通过 getNumber 方法来获取用户输入。使用 TextField 对象的 getText 方法来获取文本内容，调用 String 类中的 trim 方法来减少用户输入的多余空格，用 Integer.parseInt 将 String 型对象转换为一个整型对象。同时，当捕获到错误时，将提示用户应该输入一个整数。

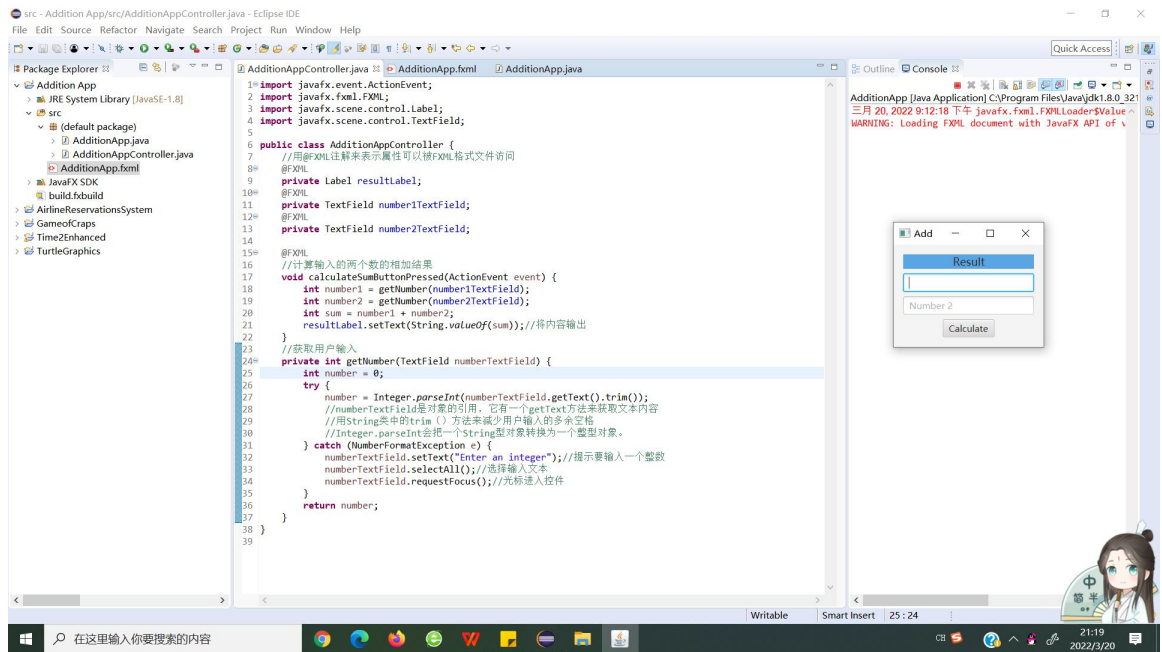
(2) 调用关系

调用关系如下图，即 main 方法可以通过加载 AdditionApp.fxml 来对 AdditionAppController 类中的方法进行调用。AdditionAppController 类中，通过 calculateSumButtonPressed 方法来计算输入的两个数的相加结果。通过 getNumber 方法来获取用户输入。



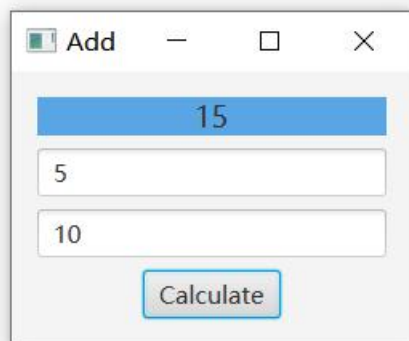
(三) 过程截图

(1) 全屏截图

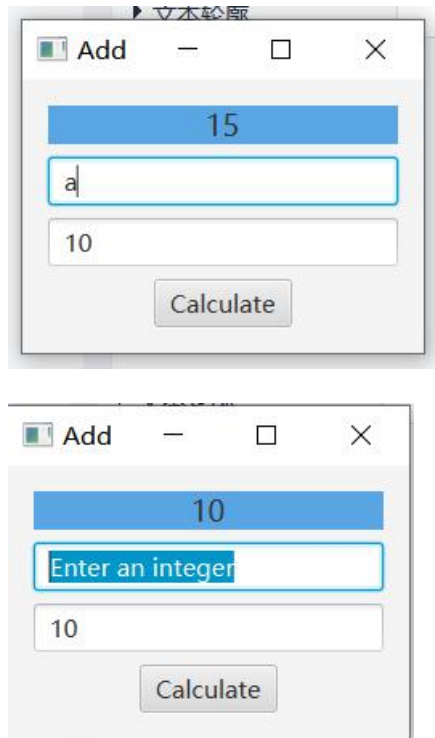


(2) 运行结果

1. 输入数字 5 和 10 计算出正确结果。



2. 当输入不是数字时，清除输入并输出错误。



扩展题目：

(7.30 Card Shuffling and Dealing) Modify Fig. 7.11 to deal a five-card poker hand. Then modify class `DeckOfCards` of Fig. 7.10 to include methods that determine whether a hand contains

I.a pair (一对 5 张中 2 张的数字一样, 另外 3 张不相同)

II.two pairs (两对 4 张中两两数字一样, 余下 1 张不同)

III.three of a kind(e.g., three jacks 三条 5 张中 3 张的数字一样)

IV.four of a kind(e.g.,four aces 四条)

V.a flush(i.e.,all five cards of the same suit,同花 五张花色一样)

VI.a straight(i.e.,five cards of consecutive face values, 顺子 五张数字相连)

VII.a full house(i.e., two cards of one face value and three cards of another face value 葫芦 3 张数字一样, 另外 2 张数字一样)



2. (7.31 Card Shuffling and Dealing) Use the methods developed in Exercise 7.30 to write an application that deals two five-card poker hands, evaluates each hand and determines which is better.
3. (7.32 Project: Card Shuffling and Dealing) Modify the application developed in Exercise 7.31 so that it can simulate the dealer. The dealer's five-card hand is dealt "face down," so the player cannot see it. The application should then evaluate the dealer's hand, and, based on the quality of the hand, the dealer should draw one, two or three more cards to replace the corresponding number of unneeded cards in the

original hand. The application should then reevaluate the dealer's hand. [Caution: This is a difficult problem!]

4. (Project: Card Shuffling and Dealing) Modify the application developed in Exercise 7.32 so that it can handle the dealer's hand automatically, but the player is allowed to decide which cards of the player's hand to replace. The application should then evaluate both hands and determine who wins. Now use this new application to play 20 games against the computer. Who wins more games, you or the computer? Have a friend play 20 games against the computer. Who wins more games? Based on the results of these games, refine your poker-playing application. (This, too, is a difficult problem.) Play 20 more games. Does your modified application play a better game?

5.(Project: Card Shuffling and Dealing) Modify the application of Figs. 7.9 –7.11 to use Face and Suit enum types to represent the faces and suits of the cards.

Declare each of these enum types as a public type in its own source-code file. Each Card should have a Face and a Suit instance variable. These should be initialized by the Card constructor. In class DeckOfCards, create an array of Faces that's initialized with the names of the constants in the Face enum type and an array of Suits that's initialized with the names of the constants in the Suit enum type. [Note: When you output an enum constant as a String, the name of the constant is displayed.]

(一) 实验环境

操作系统: Windows 10;

IDE: Eclipse Java 2018-12

编程语言: Java;

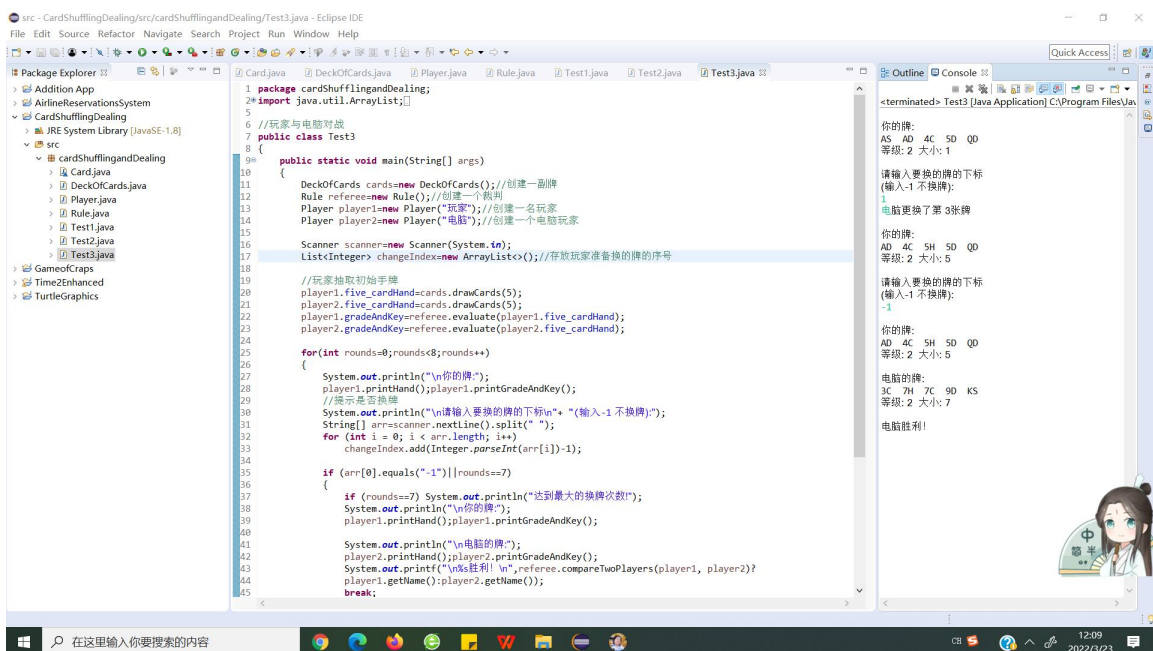
(二) 实现过程

(1) 设计类

- ① **Card** 类代表一张扑克牌,通过两参数的构造函数初始化卡片的面和花色。同时有 **getFace**、**getSuit** 方法返回卡牌的花色和大小。**toString** 方法返回 **Card** 的字符串表示形式
- ② **DeckOfCards** 类表示一副扑克牌,调用构造函数填充纸牌。**shuffe** 方法用于洗牌, **drawCards** 方法用于发牌。
- ③ **Rule** 类储存规则,相当于裁判。**compareTwoPlayers** 方法用于比较两个牌手不同的得分。将调用 **isPair**、**isTwoPairs**、**isTOAK**、**isFOAK**、**isFlush**、**isStraight**、**isFullHouse** 方法判断是否是一对、两对、三条、四条、同花、顺子、葫芦。
- ④ **Player** 类表示玩家,通过 **change** 方法进行换牌,通过 **autoChange** 方法进行自动换牌
- ⑤ **Test1** 类可以实现随机抽牌,判断大小等级。**Test2** 类可以建立两个玩家对战,并输出胜负结果。**Test3** 类可以实现玩家与电脑对战,并拥有玩家选择换牌的功能。

(三) 过程截图

(1) 全屏截图

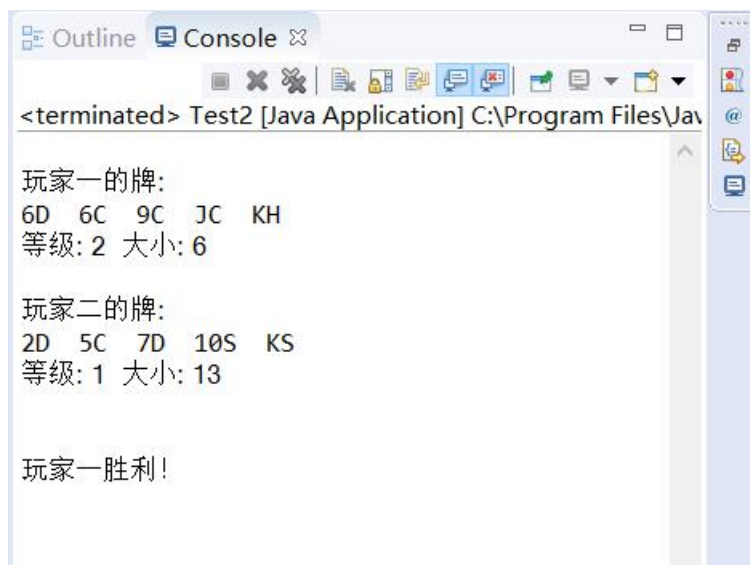


(2) 运行结果

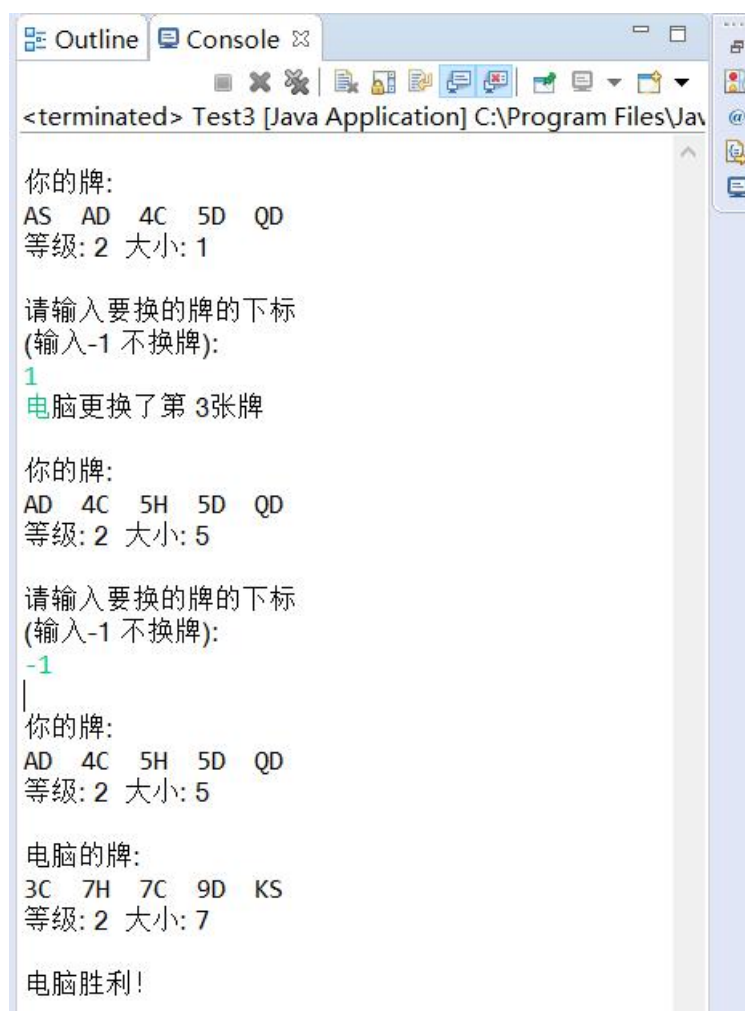
(1) Test1 中实现随机抽牌，判断并判断抽到的牌的大小等级。



(2) Test2 类可以建立两个玩家对战，并输出胜负结果。



(3) Test3 实现玩家与电脑对战，并且玩家可以选择换牌。



三、实验总结与心得记录

在本次实验过程中，我练习了控制结构，熟悉了 java 的语法，熟悉了 java 类的定义，实例化和调用。我也熟悉了简单的枚举和简单的 JavaFX 图形界面，体会到了 JAVA 语言的优点。