Homework：

**1、　用 GUI 改写策略模式的例子。**

答：对于本题，我使用了 JavaFX 进行图形化界面设计，我设计的游戏运行逻辑如下：

1) 用户启动应用程序，加载并显示游戏界面。
2) 用户选择策略（赢取策略或概率策略），然后点击“开始游戏”按钮开始游戏。
3) 游戏开始，初始化玩家和计算机的策略、得分，并更新界面状态。
4) 用户选择出拳（石头、剪刀、布），游戏进行一轮：
   - 根据用户选择生成 Hand 对象。
   - 调用计算机的 nextHand()方法生成计算机的出拳。
   - 比较出拳结果，更新游戏状态和得分。
   - 更新界面显示游戏结果和当前得分。
5) 用户可以选择重新开始游戏：
   - 点击“重新开始游戏”按钮，重新初始化游戏。
6) 在游戏进行中，用户不能切换策略，只能重新开始游戏以选择不同策略。

游戏界面如图 1 所示：


图 1 猜拳游戏初始界面

GameController 类负责处理用户交互和游戏逻辑。

- 初始化界面元素——initialize()方法：
  - 将两个策略选择按钮（winningStrategyRadioButton 和 probStrategyRadioButton）加入到同一个 ToggleGroup 中，以确保它们互斥。
  - startButton 按钮：调用 startGame()方法，初始化游戏。
  - restartButton 按钮：调用 restartGame()方法，重新开始游戏。
  - 石头、剪刀、布按钮：调用 playGame(int humanMove)方法，根据玩家的选择进行游戏。
  - 禁用 restartButton 按钮，确保在游戏开始前不能点击。
- 开始游戏——startGame()方法：
  - 根据用户选择的策略初始化玩家和计算机的策略。
  - 初始化玩家和计算机的得分。
  - 更新界面，显示游戏开始信息。

- 禁用 startButton 按钮，防止在游戏开始后重复点击。
- 启用 restartButton 按钮，允许重新开始游戏。
- 禁用策略选择按钮，防止在游戏进行中切换策略。

- 重新开始游戏——restartGame()方法：
  - 设置各种按钮能否响应点击以重新初始化游戏。
- 进行游戏——playGame(int humanMove)方法：
  - 根据玩家选择的出拳（石头、剪刀、布），生成对应的 Hand 对象。
  - 调用计算机玩家的 nextHand()方法，生成计算机的出拳。
  - 比较玩家和计算机的出拳结果，并更新游戏状态（胜负、平局）。
  - 根据游戏结果更新得分和界面显示。
- 更新得分——updateScore()方法：
  - 更新界面上的得分标签，显示当前玩家和计算机的得分。

接着可以运行程序以测试：

首先选取 WinningStrategy，它会根据前一轮是否获胜来决定下一次出拳的结果。如果上一次没有获胜，它会随机选择一个新的出拳；如果上一次获胜，它会继续使用相同的出拳。


图 2 选取 Winning Strategy 并开始游戏


图 3 第一次出拳


图 4 第一次猜拳结果


图 5 第二次仍选择石头


图 6 第三次出剪刀，由 Winning Strategy 可知电脑会出布

接着点击重新开始游戏并选取 ProbStrategy，它使用一个三维数组 history 来记录每种出拳结果的出现次数，并根据这些统计数据来计算下一次出拳的概率。


图 7 选择 ProbStrategy


图 8 第一次出拳


图 9 第二次出拳


图 10 第三次出拳

## 2、 附录

### 1） Game.java

```
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Game extends Application {
    public static void main(String[] args) {
        launch(args);
    }

    @Override
    public void start(Stage primaryStage) throws Exception {
        FXMLLoader loader = new
FXMLLoader(getClass().getResource("game.fxml"));
        primaryStage.setTitle("猜拳游戏");
        primaryStage.setScene(new Scene(loader.load()));
        primaryStage.show();
    }
}
```

### 2） GameController.java

```
package com.example.game;

import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;

public class GameController {
    @FXML
    private RadioButton winningStrategyRadioButton;
    @FXML
    private RadioButton probStrategyRadioButton;
    @FXML
    private ToggleGroup strategyToggleGroup;
    @FXML
    private Button rockButton;
    @FXML
    private Button paperButton;
    @FXML
    private Button scissorsButton;
```

```
    @FXML
    private Button startButton;
    @FXML
    private Button restartButton;
    @FXML
    private Label resultLabel;
    @FXML
    private Label scoreLabel;
    private Player humanPlayer;
    private Player computerPlayer;
    private int humanScore = 0;
    private int computerScore = 0;

    @FXML
    public void initialize() {
        strategyToggleGroup = new ToggleGroup();
        winningStrategyRadioButton.setToggleGroup(strategyToggleGroup);
        probStrategyRadioButton.setToggleGroup(strategyToggleGroup);
        startButton.setOnAction(event -> startGame());
        restartButton.setOnAction(event -> restartGame());
        rockButton.setOnAction(event -> playGame(Hand.HANDVALUE_GUU));
        paperButton.setOnAction(event -> playGame(Hand.HANDVALUE_PAA));
        scissorsButton.setOnAction(event ->
playGame(Hand.HANDVALUE_CHO));
        restartButton.setDisable(true);
    }

    private void startGame() {
        Strategy strategy;
        if (winningStrategyRadioButton.isSelected()) {
            strategy = new WinningStrategy((int) (Math.random() * 1000));
        } else {
            strategy = new ProbStrategy((int) (Math.random() * 1000));
        }
        humanPlayer = new Player("Human", strategy);
        computerPlayer = new Player("Computer", strategy);

        humanScore = 0;
        computerScore = 0;
        updateScore();
        resultLabel.setText("游戏开始！选择你的出拳。");

        startButton.setDisable(true);
        restartButton.setDisable(false);
```

```
        winningStrategyRadioButton.setDisable(true);
        probStrategyRadioButton.setDisable(true);
    }

    private void restartGame() {
        startButton.setDisable(false);
        restartButton.setDisable(true);
        winningStrategyRadioButton.setDisable(false);
        probStrategyRadioButton.setDisable(false);
    }

    private void playGame(int humanMove) {
        Hand humanHand = Hand.getHand(humanMove);
        Hand computerHand = computerPlayer.nextHand();

        String result;
        if (humanHand.isStrongerThan(computerHand)) {
            result = "你赢了! " + humanHand + " beats " + computerHand;
            humanPlayer.win();
            computerPlayer.lose();
            humanScore++;
        } else if (humanHand.isWeakerThan(computerHand)) {
            result = "你输了! " + computerHand + " beats " + humanHand;
            humanPlayer.lose();
            computerPlayer.win();
            computerScore++;
        } else {
            result = "平手! 都选择出 " + humanHand;
            humanPlayer.even();
            computerPlayer.even();
        }

        updateScore();
        resultLabel.setText(result);
    }

    private void updateScore() {
        scoreLabel.setText("得分 - 玩家: " + humanScore + " | 电脑: " +
computerScore);
    }
}
```

3) game.fxml

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.RadioButton?>
<?import javafx.scene.layout.GridPane?>
<GridPane xmlns:fx="http://javafx.com/fxml/1"
xmlns="http://javafx.com/javafx/8.0.171"
fx:controller="com.example.game.GameController" alignment="center"
hgap="10" vgap="10">
    <padding>
        <Insets top="20" right="20" bottom="20" left="20"/>
    </padding>
    <Label text="选择电脑使用的策略:" GridPane.rowIndex="0"
GridPane.columnIndex="0"/>
    <RadioButton fx:id="winningStrategyRadioButton" text="Winning
Strategy" GridPane.rowIndex="0" GridPane.columnIndex="1"/>
    <RadioButton fx:id="probStrategyRadioButton" text="Probability
Strategy" GridPane.rowIndex="0" GridPane.columnIndex="2"/>
    <Button fx:id="startButton" text="开始游戏" GridPane.rowIndex="1"
GridPane.columnIndex="0"/>
    <Button fx:id="restartButton" text="重新开始游戏"
GridPane.rowIndex="1" GridPane.columnIndex="1" GridPane.columnSpan="2"/>
    <Label text="你选择出:" GridPane.rowIndex="2"
GridPane.columnIndex="0"/>
    <Button fx:id="rockButton" text="石头" GridPane.rowIndex="2"
GridPane.columnIndex="1"/>
    <Button fx:id="paperButton" text="布" GridPane.rowIndex="2"
GridPane.columnIndex="2"/>
    <Button fx:id="scissorsButton" text="剪刀" GridPane.rowIndex="2"
GridPane.columnIndex="3"/>

    <Label fx:id="resultLabel" text="结果将在这里展示"
GridPane.rowIndex="3" GridPane.columnIndex="0" GridPane.columnSpan="4"/>
    <Label fx:id="scoreLabel" text="得分: 玩家 0 - 0 电脑"
GridPane.rowIndex="4" GridPane.columnIndex="0" GridPane.columnSpan="4"/>
</GridPane>
```

4) Hand.java

```
package com.example.game;

public class Hand {
```

```
    public static final int HANDVALUE_GUU = 0;
    public static final int HANDVALUE_CHO = 1;
    public static final int HANDVALUE_PAA = 2;
    public static final Hand[] hand = new Hand[]{new Hand(0), new Hand(1),
new Hand(2)};
    private static final String[] name = new String[]{"石头", "剪刀", "布
"};
    private int handvalue;

    private Hand(int handvalue) {
        this.handvalue = handvalue;
    }

    public static Hand getHand(int handvalue) {
        return hand[handvalue];
    }

    public boolean isStrongerThan(Hand h) {
        return this.fight(h) == 1;
    }

    public boolean isWeakerThan(Hand h) {
        return this.fight(h) == -1;
    }

    private int fight(Hand h) {
        if (this == h) {
            return 0;
        } else {
            return (this.handvalue + 1) % 3 == h.handvalue ? 1 : -1;
        }
    }

    public String toString() {
        return name[this.handvalue];
    }
}
```

5) Player.java

```
package com.example.game;

public class Player {
```

```
    private String name;
    private Strategy strategy;
    private int wincount;
    private int losecount;
    private int gamecount;

    public Player(String name, Strategy strategy) {
        this.name = name;
        this.strategy = strategy;
    }

    public Hand nextHand() {
        return this.strategy.nextHand();
    }

    public void win() {
        this.strategy.study(true);
        ++this.wincount;
        ++this.gamecount;
    }

    public void lose() {
        this.strategy.study(false);
        ++this.losecount;
        ++this.gamecount;
    }

    public void even() {
        ++this.gamecount;
    }

    public String toString() {
        return "[" + this.name + ":" + this.gamecount + " games, " +
this.wincount + " win, " + this.losecount + " lose" + "]";
    }
}
```

6) Strategy.java

```
package com.example.game;

public interface Strategy {
```

```
    Hand nextHand();
    void study(boolean var1);
}
```

7) ProbStrategy.java

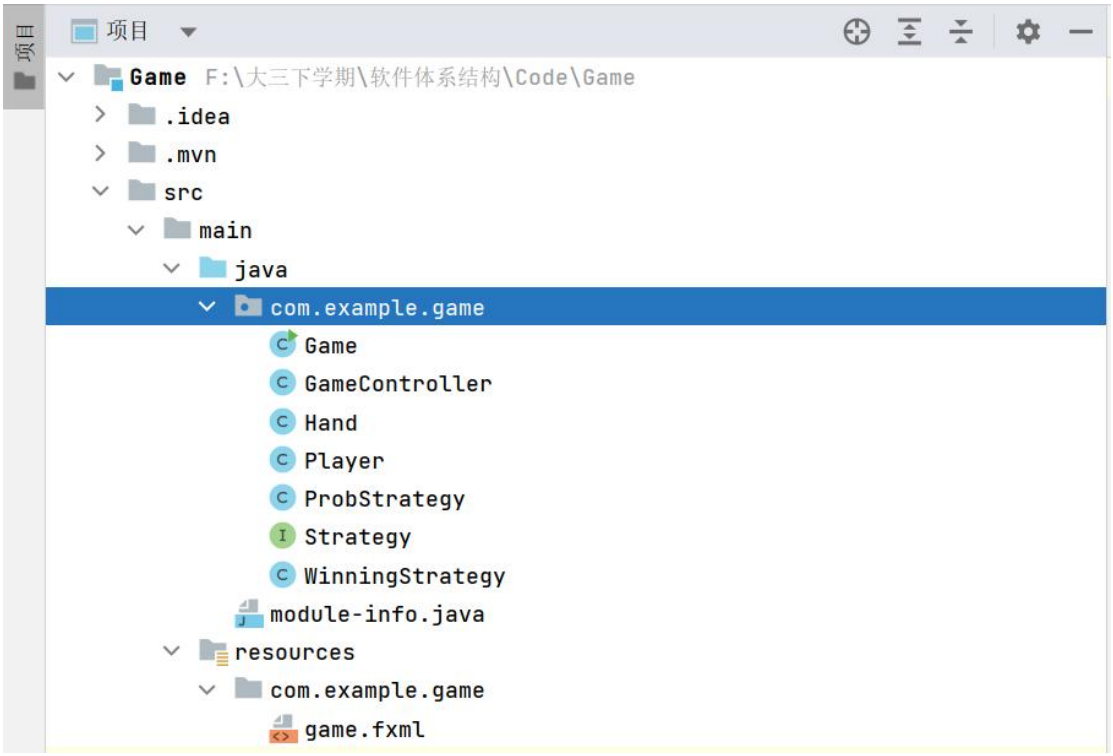
```
package com.example.game;

import java.util.Random;

public class ProbStrategy implements Strategy {
    private Random random;
    private int prevHandValue = 0;
    private int currentHandValue = 0;
    private int[][] history = new int[][]{{1, 1, 1}, {1, 1, 1}, {1, 1, 1}};

    public ProbStrategy(int seed) {
        this.random = new Random((long)seed);
    }

    public Hand nextHand() {
        int bet =
this.random.nextInt(this.getSum(this.currentHandValue));
        byte handvalue;
        if (bet < this.history[this.currentHandValue][0]) {
            handvalue = 0;
        } else if (bet < this.history[this.currentHandValue][0] +
this.history[this.currentHandValue][1]) {
            handvalue = 1;
        } else {
            handvalue = 2;
        }

        this.prevHandValue = this.currentHandValue;
        this.currentHandValue = handvalue;
        return Hand.getHand(handvalue);
    }

    private int getSum(int hv) {
        int sum = 0;

        for(int i = 0; i < 3; ++i) {
            sum += this.history[hv][i];
        }
```

```
        return sum;
    }

    public void study(boolean win) {
        int var10002;
        if (win) {
            var10002 =
this.history[this.prevHandValue][this.currentHandValue]++;
        } else {
            var10002 =
this.history[this.prevHandValue][(this.currentHandValue + 1) % 3]++;
            var10002 =
this.history[this.prevHandValue][(this.currentHandValue + 2) % 3]++;
        }

    }
}
```

8) WinningStrategy.java

```
package com.example.game;

import java.util.Random;

public class WinningStrategy implements Strategy {
    private Random random;
    private boolean won = false;
    private Hand prevHand;

    public WinningStrategy(int seed) {
        this.random = new Random((long)seed);
    }

    public Hand nextHand() {
        if (!this.won) {
            this.prevHand = Hand.getHand(this.random.nextInt(3));
        }

        return this.prevHand;
    }

    public void study(boolean win) {
        this.won = win;
    }
}
```

9） 程序结构


项目
项目
Game F:\大三下学期\软件体系结构\Code\Game
.idea
.mvn
src
main
java
com.example.game
Game
GameController
Hand
Player
ProbStrategy
Strategy
WinningStrategy
module-info.java
resources
com.example.game
game.fxml