**上海理工大学程序设计Java综合实验报告**

|  |  |
| --- | --- |
| **作品描述（自行设计系统的简介）** | 此次 Java 程序设计作品名为 “SeaBattle2”，是一个简易的海战游戏。游戏目的在于猜测敌方战船的位置并给予打击。  游戏项目中包含五个文件，其中 BattleShip.java 与 GameBoard.java 文件中各自包含了一个类，使用了面向对象编程的思路，分别创建了战船与游戏板两个类，并抽象出了它们的属性与方法，是项目中的核心类。  StartFrame.java, GameFrame.java 与 PVEModeFrame.java 这三个文件主要使用了 swing 技术，为这个游戏项目创建了简易的用户交互界面,并且确定了整个操作流程。其中，StartFrame.java 实现了游戏的主界面，GameFrame.java 和 PVEModeFrame.java 则分别实现了游戏的普通模式(Original Mode)和人机对战模式(PVE Mode) ，人机对战模式又分为简单和困难两个难度等级。在这三个文件中，每个文件也包含了若干个非公有类，以实现弹出对话框以告知用户一些信息的功能。 |
| **研究思路** | （1）选题意义 |
| 这个选题源自《Head First Java》第96页介绍的攻击网站游戏的概念。这个游戏的思路相对简单，实现起来也需要用到本学期的大部分内容。所以在没有读过书中这部分内容的情况下可以使用课上所学的面向对象编程思想以及使用 swing 进行图形界面开发的内容来实现游戏相关的功能。 |
| （2）课题研究思路 |
| BattleShip 类与 GameBoard 类使用面向对象的思想抽象出了战舰和游戏板的方法以及属性，并使用了 Scanner 类实现了在这两个类中获取用户输入的方法，也使用了 ArrayList 类动态数组的功能。  其余三个类都使用了 swing 来实现图形界面。StartFrame.java 与 GameFrame.java 中使用了文件输入输出的知识实现普通模式的最高分保存与读取的功能。在人机对战模式(PVEModeFrame.java)中，使用线程的知识模拟电脑在思考时的暂停。 |
| （3）本课题研究的创新之处和预期成果 |
| 本课题研究的创新之处在于人机对战模式的加入，以及该模式的两种难度的选项。  根据预期，在简单难度下，电脑的回合时，会选择整个游戏板的范围内没有打击过的位置落子；而在困难难度下，如果电脑上一次打击命中，电脑的下一个打击位置会从上次命中的位置的上下左右中选取一个没有打击过的位置落子。 |
| **自定义类** | BattleShip 类，抽象出了战船的属性和方法。  成员变量有整型的 size，整型的二维数组 loc 还有整型的 hitNum。其中，size 存储了战船的大小（3或2），loc 存储了战船在游戏板中的位置，hitNum 存储了战船被命中的次数（用来判断是否被击沉）。  这个类包含七个方法：   1. 构造方法接收一个整型参数并用它初始化成员变量 size，同时将 hitNum 初始化为 0。 2. getRandomLoc 方法接收游戏板的最大长和宽作为参数并生成随机的二维位置数组作为返回值，GameBoard 类中的 isLocAvailable 方法会用到这个返回值，判断这个位置是否可以使用。 3. 若这个位置可以使用，则再作为参数传回给 setLoc 方法，确定船的最终位置。 4. isHit 方法接收一个打击位置数组，判断是否命中这艘船，并将结果以布尔类型的形式返回。 5. isSink 方法通过比较成员变量 hitNum 与 size 的大小判断船是否被击沉，并将结果以布尔变量的形式返回。 6. getSize 和 getLoc 方法返回战船的大小和位置信息。 |
|  |
| GameBoard 类，抽象出了游戏板的属性和方法。  成员变量有整型的 BOARDROW，BOARDCOL，shipNum，hitNum 和 numSink，还有二维整型数组 gameBoard 和 BattleShip 类的数组 battleShips。其中，BOARDROW 和 BOARDCOL 存储了游戏板的长和宽，为了方便的改变游戏板的大小；shipNum 存储了游戏板中战船的数量；hitNum 用来统计玩家在这块游戏板上打击的次数，以计算出最终的得分；numSink 存储了游戏板上已经被击沉的战船的数量；gameBoard 数组存储了游戏板上每个格子所包含的信息；battleShips 数组包含了这个游戏板上所有的战船。  这个类包含十个方法：   1. 其中有两个构造函数，默认构造函数是给普通模式使用的，另一个构造函数接收一个整型参数作为游戏板中的战船数量。这两个构造函数都初始化了成员变量 gameBoard 和 battleShips，区别在于默认构造函数同时确定了每个战船的位置，而在当对战模式中船的位置需要玩家来确定时，就会使用到另外一个构造函数。 2. addOneShip 接收一个整型参数 shipNum （表示战船的编号）和一个 ArrayList （表示战船的位置），这个方法的功能通过这两个参数确定单个战船的位置，用于对战模式中玩家确定船的位置。 3. arrangeShips 方法可以安排好所有战船的位置，位置为随机生成。 4. isLocAvailable 接收一个整型参数 shipNo（战船的编号），方法内部会用到 BattleShip 类的 getRandomLoc 方法来生成一个位置，在判断这个位置是否符合要求之后将结果以布尔变量的形式返回。 5. fireCannon 方法接收一个整型的打击位置数组，方法中遍历所有的战船对象，判断这个打击位置是否命中战船。 6. isWin 方法比较成员变量 numSink 和 shipNum 以判断是否获胜，并将结果以布尔变量的形式返回。 7. getScore，getRowNum 和 getColNum 方法分别返回得分以及游戏板的长和宽。 |
| GameFrame 类是游戏普通模式的界面，继承自 JFrame，实现了 ActionListener 接口。  有两个成员变量，GameBoard 类的对象 gameBoard 和 JButton 对象的二维数组 buttons。其中 buttons 数组中的每一个 JButton 都代表 gameBoard 游戏板中的一个格子。  这个类包含四个方法：   1. 构造方法确定了这个 JFrame 的一些基本信息，还有大小标题等信息，同时调用了另外一个 addButtons 方法添加了所有的按钮。 2. addButtons 方法为界面添加了所有的按钮，并给所有按钮添加了事件监听器。 3. saveHighScore 方法使用了 Java 的文件输入输出，实现了保存历史最高的十个分数的功能。保存的文件是在 src 文件夹中的 HighScore.txt 文件。 4. actionPerformed 方法实现了玩家点击按钮之后的操作，包括命中或者为命中后的操作以及获胜时保存最高分和弹出对话框的操作。 |
| PVEModeFrame 类是游戏人机对战模式的界面，继承自 JFrame，实现了 Action Listener 接口。  有十一个成员变量，其中 GameBoard 类的成员 computerBoard 和 playerBoard 分别是电脑和玩家的游戏板；JButton 类型的二维数组 computerButtons 和 playerButtons 分别是电脑和玩家游戏板的按钮；JTextArea 类的对象 instrctionArea 和stateArea 分别用于显示指示面板和状态面板；布尔类型的变量 isHard 表示是否是困难模式。整型的 shipCounter，shipLenCounter以及 ArrayList 类的对象 tempLoc，blocksTaken 都是起到辅助作用的成员变量。  这个类包含十一个方法：   1. 构造方法接收一个布尔型的参数，表示判断难度是简单还是困难，并初始化了必要的成员变量，调用了之后的 addButtons 方法和 addTextArea 方法添加按钮和文本区，也对标题和大小等进行了设置，并且将电脑的游戏板上的按钮全部设置为不可点击。 2. addTextArea 和 addButtons 方法为界面添加了文本区和所有的按钮，并为按钮加上了事件监听器。 3. addShips 方法接收两个整型变量 row 和 col，表示玩家添加的船的横坐标和纵坐标，这个方法根据这个坐标以及成员变量 shipCounter 和 shipLenCounter 的辅助确定船的位置。同时具有简单的判断玩家所选位置是否合法的功能。 4. isLocOk 方法在添加船的时候使用，会接收一个整型位置数组，判断这个位置是否符合规则，并将结果以布尔类型的形式返回。 5. playerMove 方法表示玩家的回合，接收两个整型变量 row 和 col，表示玩家打击位置的横坐标和纵坐标，返回值为玩家是否命中。 6. computerMove 方法表示电脑的回合，接收的参数是控制难度的布尔类型参数。 7. easyComputer 表示简单的电脑的行为，即打击点的选取是在整个游戏板大小范围内随机的（但是不包括已经打击了的格子，用成员变量 blocksTaken 控制），返回值为通过这个方法所决定的打击位置。 8. hardComputer 表示困难的电脑的行为，即在上一次打击命中的情况下，优先选择上下左右的格子打击。由于需要上一次打击位置的信息，所以接收两个整型参数，表示上一次打击位置的横坐标和纵坐标，返回值为通过这个方法所决定的打击位置。 9. arrayListContains 方法起到一个查重的作用。由于 ArrayList 类的 contains 方法使用的是 equals 来判断，所以不能判断元素的值是相等，因此需要重写。接收需要判断的 ArrayList 和一个元素作为参数，返回值为布尔类型，表示这个元素是否和 ArrayList 中的一个元素具有相等的值 10. actionPerformed 方法实现了玩家点击按钮时的操作，包括在添加战船阶段和游戏正式开始之后的阶段。添加战船阶段结束之后和游戏开始阶段之前会将 playerButtons 全部设置为不可点击而 computerButtons 全部设置为可以点击。 |
| StartFrame 是整个游戏的主界面，继承自 JFrame，实现了 ActionListener 接口。  成员变量为三个 JButton 类的成员 start，highScore 和 quit。分别表示开始，最高分和退出三个按钮。  这个类包含两个方法：   1. 构造方法，初始化并设置了所有的三个按钮，每个按钮都设置了图片以增加美观性，图片背景的实现是使用了一个重写了 paintComponent 方法的 JPanel 类，类名为 ImagePanel。所有的图片文件都位于项目根目录下 “/src/Image” 文件夹中。 2. actionPerformed 方法实现了在点击每个按钮时会有的事件，点击 start 按钮会弹出模式选择对话框（ModeChooseDialog 类），点击 quit 按钮会关闭窗口并退出，点击 highScore 按钮会弹出最高分对话框（HighScoreDialog 类）。 |
| 还有七个类，其中六个是用来实现对话框的类。分别是普通模式获胜时弹出的胜利对话框 WinDialog，人机对战模式获胜时以及失败时弹出的对话框 PlayerWinDialog 和 PlayerLoseDialog，游戏的主界面下的最高分对话框 HighScoreDialog，模式选择对话框 ModeChooseDialog 以及难度选择对话框 DifficultyChooseDialog。这六个类都继承自 JDialog，并实现了 ActionListener 接口。  最后一个类名为 ImagePanel，继承自 JPanel，重写了 paintComponent 方法以实现在 JPanel 上添加图片的功能，因为 JPanel 没有直接设置图片的方法。 |
| 界面设计 | 这个项目主要有三个界面，StartFrame 所实现的主界面，GameFrame 所实现的普通模式界面以及 PVEModeFrame 所实现的人机对战模式界面。  游戏主界面主体是一张背景图，浮于背景图上方有三个图片按钮 Start， High Score 和 Quit。所有的图片文件都位于项目根目录下 “/src/Image” 文件夹中。  普通模式界面为若干个按钮所组成的网格，每个按钮在点击之后都会有相应的变化提示命中与否，若是命中，按钮会变成一张画有 “X” 符号的图片，若是没有命中，按钮会变成另一种颜色。  人机对战模式界面为两个类似普通模式的网格中夹着指示文本区以及状态文本区，会对当前的状态以及玩法做出一定的提示。左边网格表示电脑的游戏板（需要进行打击），右边网格表示玩家的游戏板（需要放置战船）。 |
| **界面控制的主要代码段** | 以下是主界面主要的控制代码。  public class StartFrame extends JFrame implements ActionListener{  JButton start, highScore, quit;    public StartFrame(){  this.setBounds(200, 200, 860, 538);  this.setTitle("SeaBattle2");  this.setBackground(new Color(255, 255, 255));  this.setDefaultCloseOperation(DISPOSE\_ON\_CLOSE);    ImagePanel backgroundPanel = new ImagePanel();    start = new JButton();  start.setIcon(new ImageIcon("src/Image/startSmallClean.png"));  start.setBounds(107, 20, 54, 25);  start.setBorderPainted(false);  highScore = new JButton();  highScore.setIcon(new ImageIcon("src/Image/highScoreSmallClean.png"));  highScore.setBounds(107, 50, 54, 25);  highScore.setBorderPainted(false);  quit = new JButton();  quit.setIcon(new ImageIcon("src/Image/quitSmallClean.png"));  quit.setBounds(107, 80, 54, 25);  quit.setBorderPainted(false);    start.addActionListener(this);  highScore.addActionListener(this);  quit.addActionListener(this);  highScore.setContentAreaFilled(false);  start.setContentAreaFilled(false);  quit.setContentAreaFilled(false);  start.setOpaque(false);  quit.setOpaque(false);  highScore.setOpaque(false);  backgroundPanel.add(start);  backgroundPanel.add(highScore);  backgroundPanel.add(quit);  this.add(backgroundPanel, BorderLayout.CENTER);    this.setVisible(true);  }    public void actionPerformed(ActionEvent e){  if (e.getSource() == start){  System.out.println("Start");  ModeChooseDialog modeChooseDialog = new ModeChooseDialog();  }  if (e.getSource() == highScore){  System.out.println("HighScore");  try {  HighScoreDialog highScoreDialog = new HighScoreDialog();  } catch (NumberFormatException e1) {  e1.printStackTrace();  } catch (IOException e1) {  e1.printStackTrace();  }  }  if (e.getSource() == quit){  System.out.println("Quit");  this.dispose();  }  }  public static void main(String[] args) {  StartFrame startFrame = new StartFrame();  }  } |
| **系统使用操作过程** | 运行 StartFrame.java，会出现项目的主界面，若要查看普通模式的历史最高分，点击 High Score 按钮，若要退出，点击 Quit 按钮，若要开始游戏，点击 Start 按钮。点击了 Start 按钮之后会弹出模式选择对话框，在 Original Mode（普通模式）和 PVE Mode （人机对战模式）中选择一个。若是选择了 PVE Mode，会弹出难度选择对话框，在 Easy Mode（简单） 和 Hard Mode（困难）中选择一个。  若是进行普通模式的游戏，默认会有四艘船，两艘船的大小为 2，剩下两艘的大小为 3。猜测敌方战船的位置并点击该位置的按钮，若是命中会出现一个“X”的符号，若是没有命中，按钮会变色。在将全部战船都击沉之后，会弹出胜利对话框，对话框中会显示次此游戏的分数，若是历史前十个最高分，会保存下来，可以在主界面的最高分对话框中看到。已经打击过的按钮会被设置为不可点击。  若是进行人机对战模式的游戏，在选择了难度之后，首先需要放置己方的战船，默认一共四艘船，前两艘的大小为 2，后两艘的大小为 3，指示文本区会提示你当前放置的是第几艘船，并会显示它的大小。要放置船，点击右侧玩家游戏板的相应位置的按钮即可（此时左侧电脑游戏板无法点击）。一艘船的放置位置必须连续，且不能斜着放，否则指示文本区会提示无法放置。已经放置过的位置的按钮会被设置为不可点击。  在己方战船放置完毕之后，右侧玩家游戏板无法点击，左侧电脑游戏板可以点击，游戏正式开始。猜测电脑战船的位置并点击相应位置的按钮每次命中都会奖励一次额外的打击机会。已经打击过的按钮会被设置为不可点击。 |
| **正确评价本次实验的结果** | 本次实验的结果基本都可以符合预期，最高分能正常保存以及读取，普通模式以及人机对战模式的困难和简单两个难度也都能正常的运行，没有出现致命错误。  人机对战模式在玩家放置战船的阶段在一些罕见的情况下会出现无法继续的错误，如下图所示：  在这种情况下，已经放置好了两艘船，大小都是 2（图中编号为战船的编号）。若是下一艘船放置的位置为红色箭头所指的位置，那这艘船的第二个格子必须选择在红色箭头所指格子的右侧格子，但是由于第三艘船的大小是 3，所以需要第三个格子。因为已经选中的格子的按钮已经被设置为不可点击，又因为一艘战船的放置不能有间隔，所以其他地方的蓝色方块都不可以点击，若是点击了，上方的文本区就会出现错误提示。  于是在这种情况下，游戏将无法继续。  暂时还没想到较为经济的解决方式。  还存在一个可以优化的地方，所有自己写的弹出窗口类，都可以用 javax.swing 包中 JOptionPane 类的 showMessageDialog 方法来代替，可以减少相当多的代码量。  最后，游戏界面的美化也许还不够吧。 |
| **教师评语** |  |

**本次实验者相关信息：**

**所在学院：中英国际学院 专业：电子信息科学与技术（第二专业 计算机科学与技术）**

**学号：1524010127 姓名：潘一滔**

**成绩： 任课教师：**

**完成实验时间：2017/6/5**