Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №5**

**Дисциплина: Программирование для мобильных платформ**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.М. Нагалевский

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Т.А. Приходько

**Тема работы:** Разработка графического приложения игры лото с использование многопоточности.

**Постановка задачи:**

Четыре игрока строго по очереди выставляют свои фишки (у каждого свой цвет фишки) на общую доску, о чем каждый выдает сообщение, например: «Красный, фишка З». По окончании (когда все клетки заполнены) сообщается, кто закончил первым.

**Описание решения:**

1. Класс LotoGame: Этот класс представляет главное окно игры. Он создает игровую доску, создает четырех игроков, инициализирует таймер и управляет ходами игроков.
2. Класс Player: Этот класс представляет игрока. У каждого игрока есть имя, цвет фишек, игровая доска и количество размещенных фишек. Игроки ходят поочередно, пытаясь разместить фишку на доске.
3. Класс Board: Этот класс представляет игровую доску. Он содержит ячейки (плитки) и отслеживает состояние игры. Для предотвращения коллизий при размещении фишек на доске, используется многопоточность и синхронизированные методы.
4. Класс LotoTile: Этот класс представляет фишку с номером и цветом.
5. В игре используется таймер (javax.swing.Timer), чтобы позволить игрокам ходить по очереди. Каждый тик таймера вызывает метод placeTile() у текущего игрока, который пытается разместить фишку на доске. После каждого хода обновляется отображение доски.
6. Игра продолжается, пока все ячейки на доске не будут заполнены. По окончании игры таймер останавливается, и объявляется победитель, тот игрок, который первым заполнил все ячейки.

На рисунке 1 представлена UML-диаграмма классов.

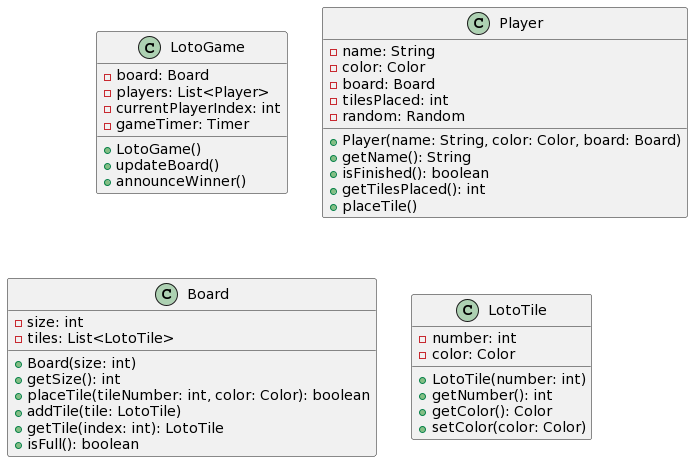


Рисунок 1 – UML-диаграмма

На рисунке 2 – 3 представлены итоги выполнения программы.

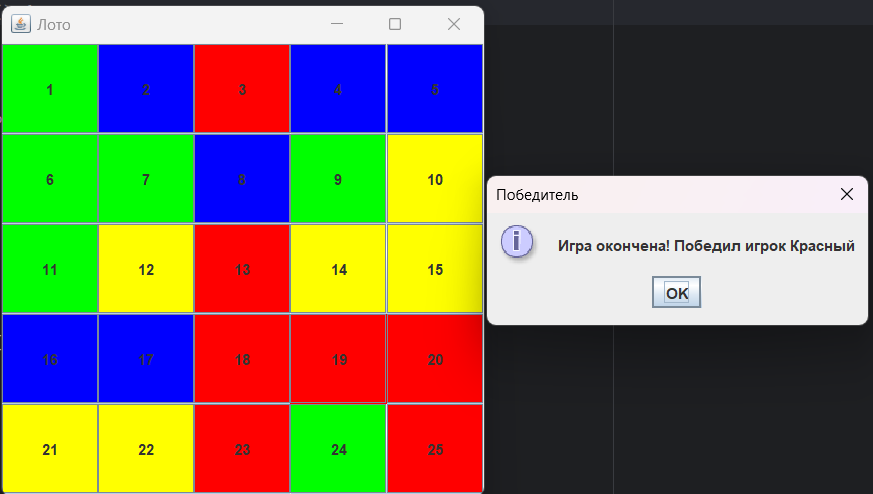


Рисунок 2 – Пример 1.



Рисунок 3 – Пример 2.

**Листинг программ**

Файл LotoGame.java

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Random;  
  
public class LotoGame extends JFrame {  
 private Board board;  
 private List<Player> players;  
 private int currentPlayerIndex;  
 private final int boardSize = 5;  
 private Timer gameTimer;  
  
 public LotoGame() {  
 board = new Board(boardSize);  
 players = new ArrayList<>();  
 players.add(new Player("Красный", Color.*RED*, board));  
 players.add(new Player("Синий", Color.*BLUE*, board));  
 players.add(new Player("Зеленый", Color.*GREEN*, board));  
 players.add(new Player("Желтый", Color.*YELLOW*, board));  
 currentPlayerIndex = 0;  
  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setSize(400, 400);  
 setLocationRelativeTo(null);  
 setLayout(new GridLayout(boardSize, boardSize));  
 setTitle("Лото");  
  
 for (int i = 0; i < boardSize \* boardSize; i++) {  
 JButton button = new JButton();  
 button.setPreferredSize(new Dimension(50, 50));  
 add(button);  
 board.addTile(new LotoTile(i + 1));  
 }  
  
 gameTimer = new Timer(1000, new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 Player currentPlayer = players.get(currentPlayerIndex);  
 if (!currentPlayer.isFinished()) {  
 currentPlayer.placeTile();  
 updateBoard();  
 }  
  
 currentPlayerIndex++;  
  
 if (currentPlayerIndex >= players.size()) {  
 currentPlayerIndex = 0;  
 }  
  
 if (board.isFull()) {  
 gameTimer.stop();  
 announceWinner();  
 }  
 }  
 });  
  
 gameTimer.start();  
 }  
  
 private void updateBoard() {  
 for (int i = 0; i < board.getSize(); i++) {  
 LotoTile tile = board.getTile(i);  
 JButton button = (JButton) getContentPane().getComponent(i);  
 if (tile.getColor() != null) {  
 button.setBackground(tile.getColor());  
 button.setText(String.*valueOf*(tile.getNumber()));  
 }  
 }  
 }  
  
 private void announceWinner() {  
 int maxTilesPlaced = -1;  
 String winner = "";  
  
 for (Player player : players) {  
 int tilesPlaced = player.getTilesPlaced();  
 if (tilesPlaced > maxTilesPlaced) {  
 maxTilesPlaced = tilesPlaced;  
 winner = player.getName();  
 }  
 }  
  
 JOptionPane.*showMessageDialog*(this, "Игра окончена! Победил игрок " + winner, "Победитель", JOptionPane.*INFORMATION\_MESSAGE*);  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(() -> {  
 LotoGame game = new LotoGame();  
 game.setVisible(true);  
 });  
 }  
}

Файл LotoTile.java

import java.awt.\*;  
  
class LotoTile {  
 private int number;  
 private Color color;  
  
 public LotoTile(int number) {  
 this.number = number;  
 this.color = null;  
 }  
  
 public int getNumber() {  
 return number;  
 }  
  
 public Color getColor() {  
 return color;  
 }  
  
 public void setColor(Color color) {  
 this.color = color;  
 }  
}

Файл Board.java

import java.awt.\*;  
import java.util.ArrayList;  
import java.util.List;  
  
class Board {  
 private int size;  
 private List<LotoTile> tiles;  
  
 public Board(int size) {  
 this.size = size;  
 tiles = new ArrayList<>();  
 }  
  
 public int getSize() {  
 return size \* size;  
 }  
  
 public synchronized boolean placeTile(int tileNumber, Color color) {  
 for (LotoTile tile : tiles) {  
 if (tile.getNumber() == tileNumber && tile.getColor() == null) {  
 tile.setColor(color);  
 return true;  
 }  
 }  
 return false;  
 }  
  
 public synchronized void addTile(LotoTile tile) {  
 tiles.add(tile);  
 }  
  
 public LotoTile getTile(int index) {  
 return tiles.get(index);  
 }  
  
 public boolean isFull() {  
 for (LotoTile tile : tiles) {  
 if (tile.getColor() == null) {  
 return false;  
 }  
 }  
 return true;  
 }  
}

Файл Player.java

import java.awt.\*;  
import java.util.Random;  
  
class Player {  
 private String name;  
 private Color color;  
 private Board board;  
 private int tilesPlaced;  
 private Random random;  
  
 public Player(String name, Color color, Board board) {  
 this.name = name;  
 this.color = color;  
 this.board = board;  
 this.tilesPlaced = 0;  
 this.random = new Random();  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public boolean isFinished() {  
 return tilesPlaced >= board.getSize();  
 }  
  
 public int getTilesPlaced() {  
 return tilesPlaced;  
 }  
  
 public void placeTile() {  
 int tileNumber;  
 boolean tilePlaced;  
 do {  
 tileNumber = random.nextInt(board.getSize()) + 1;  
 tilePlaced = board.placeTile(tileNumber, color);  
 } while (!tilePlaced);  
  
 tilesPlaced++;  
 System.*out*.println(name + ", фишка " + tileNumber);  
 }  
}