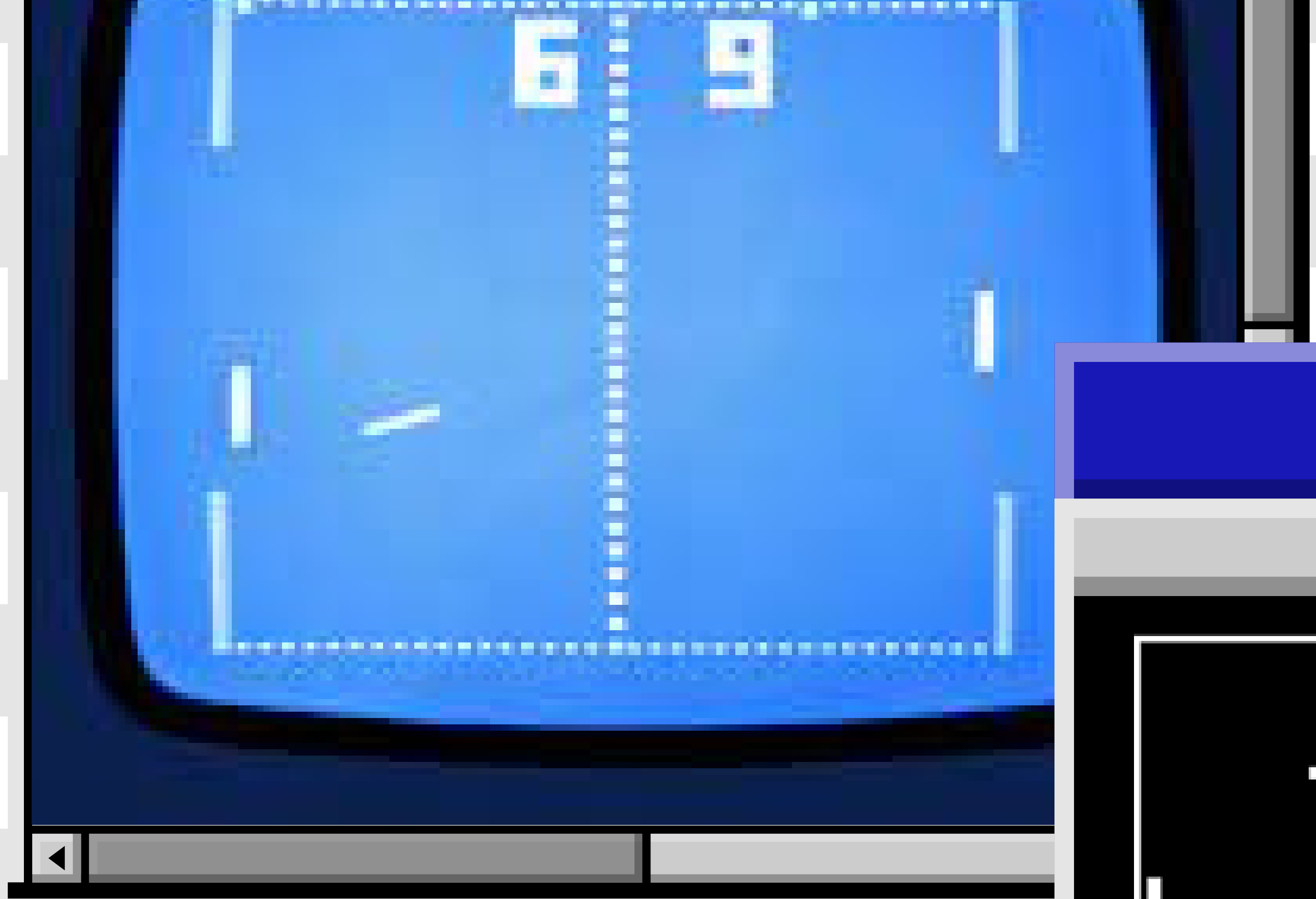
PONG GAME

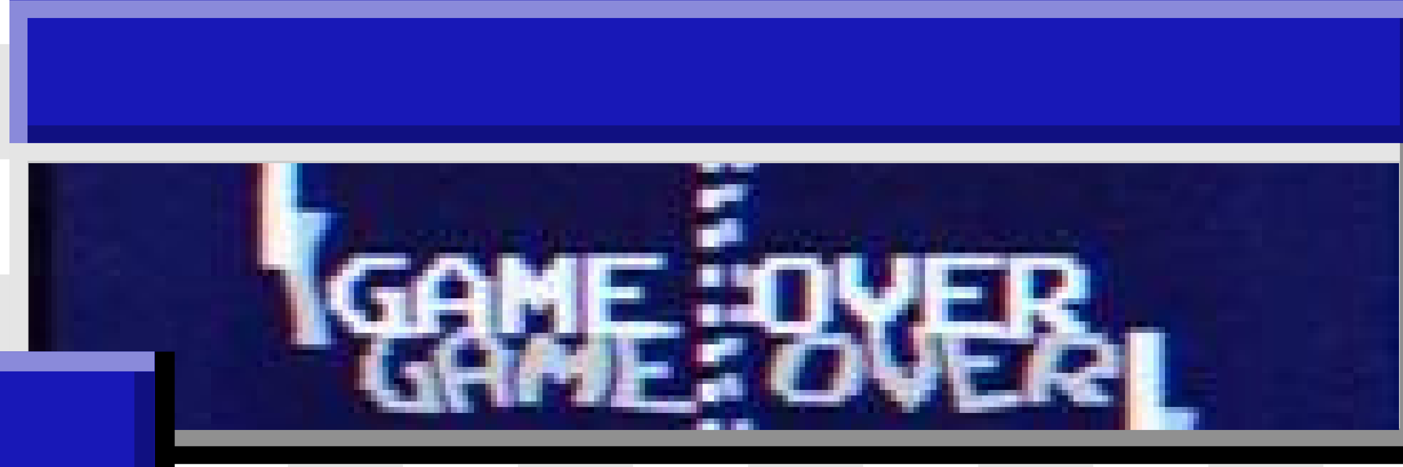
START

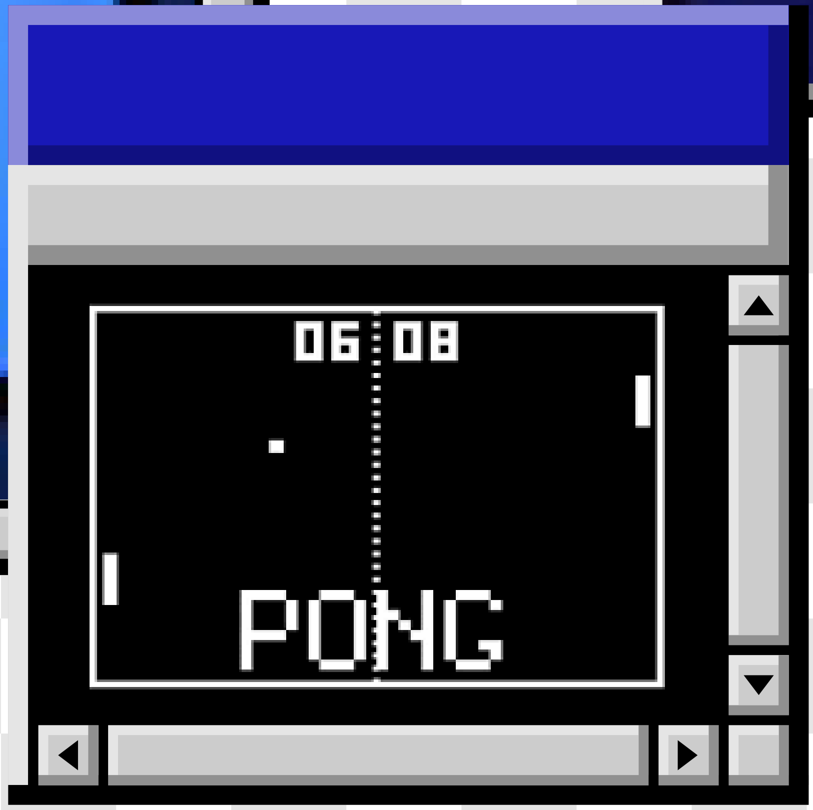
Intermediate Programming

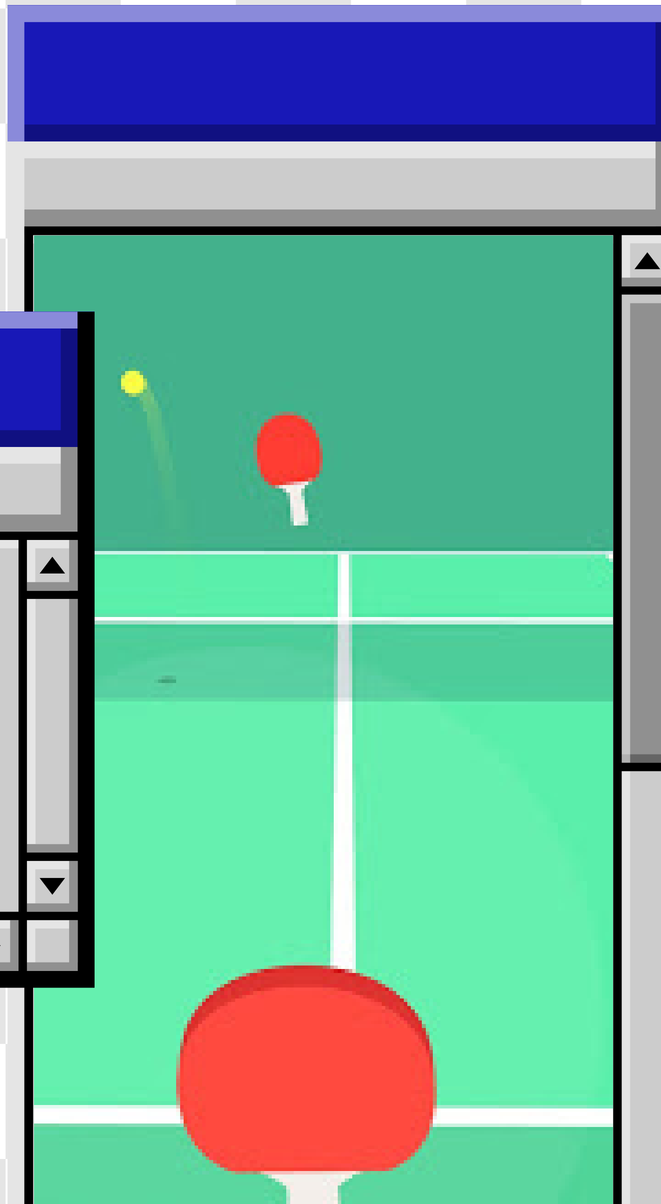
National University-Manila

11:11PM









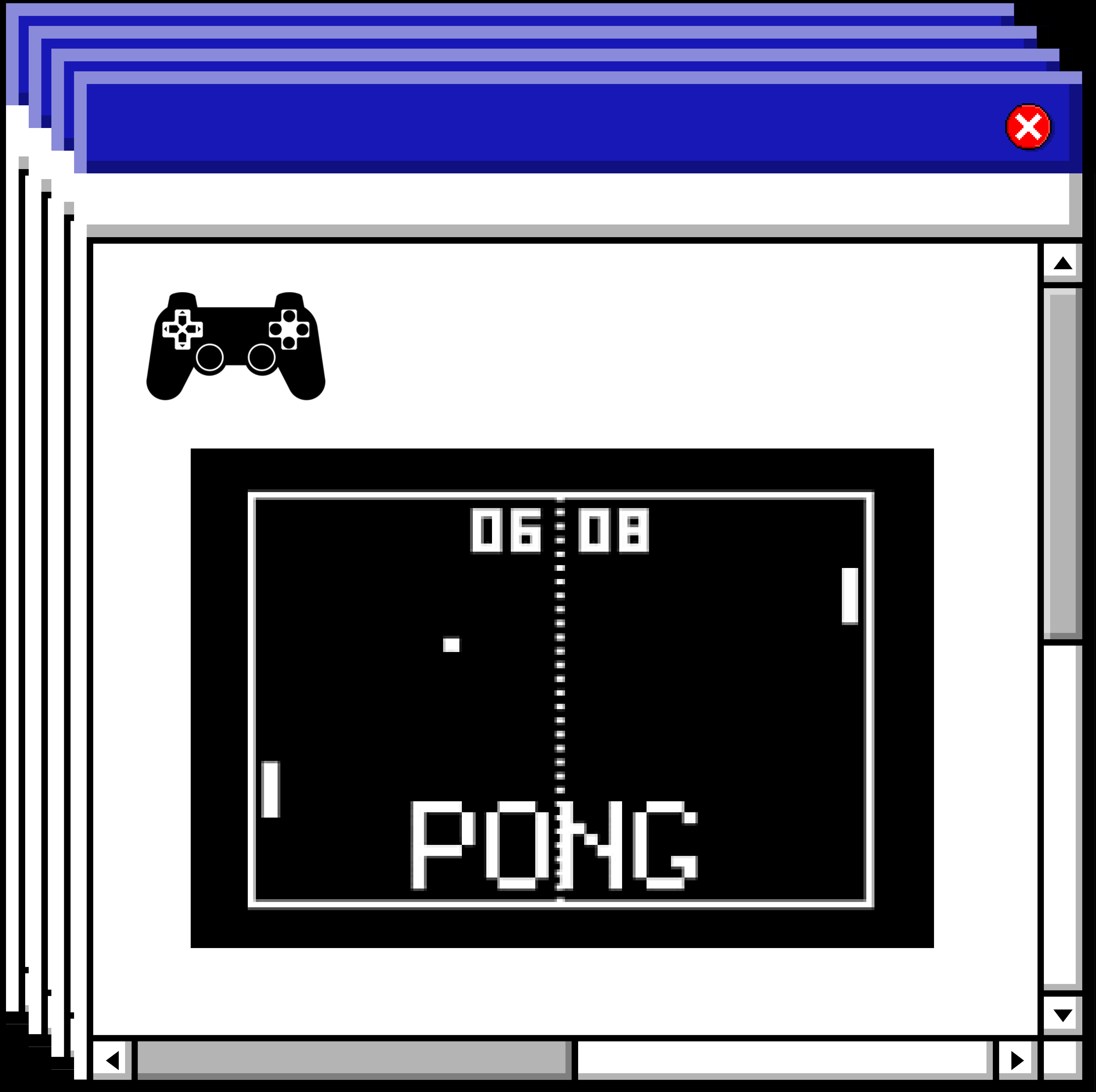
Back to Agenda Page

PONG GAME

Pong

Pong is a classic arcade video game that was first released in 1972. It's a two-player game in which players control virtual paddles to hit a ball back and forth across a simple 2D playing field. The objective is to score points by forcing the other player to miss the ball, with the first player to reach a certain number of points declared the winner.

The game is often credited with popularizing video gaming as a mainstream entertainment medium, and it has since become a cultural icon. Today, it's available in many forms, from simple mobile apps to more advanced console and PC versions, and it's still enjoyed by players of all ages around the world.



National University-Manila

Back to Agenda Page

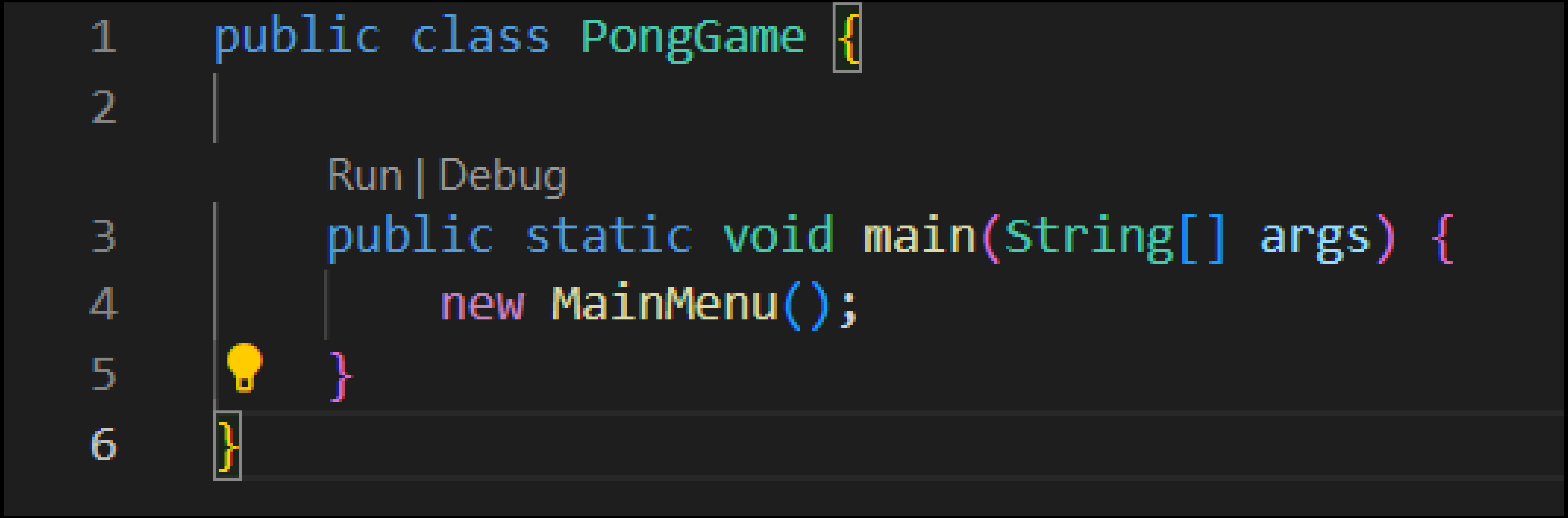
THE GRAPHICAL USER INTERFACE

START

National University-Manila

11:11PM

PongGame.java

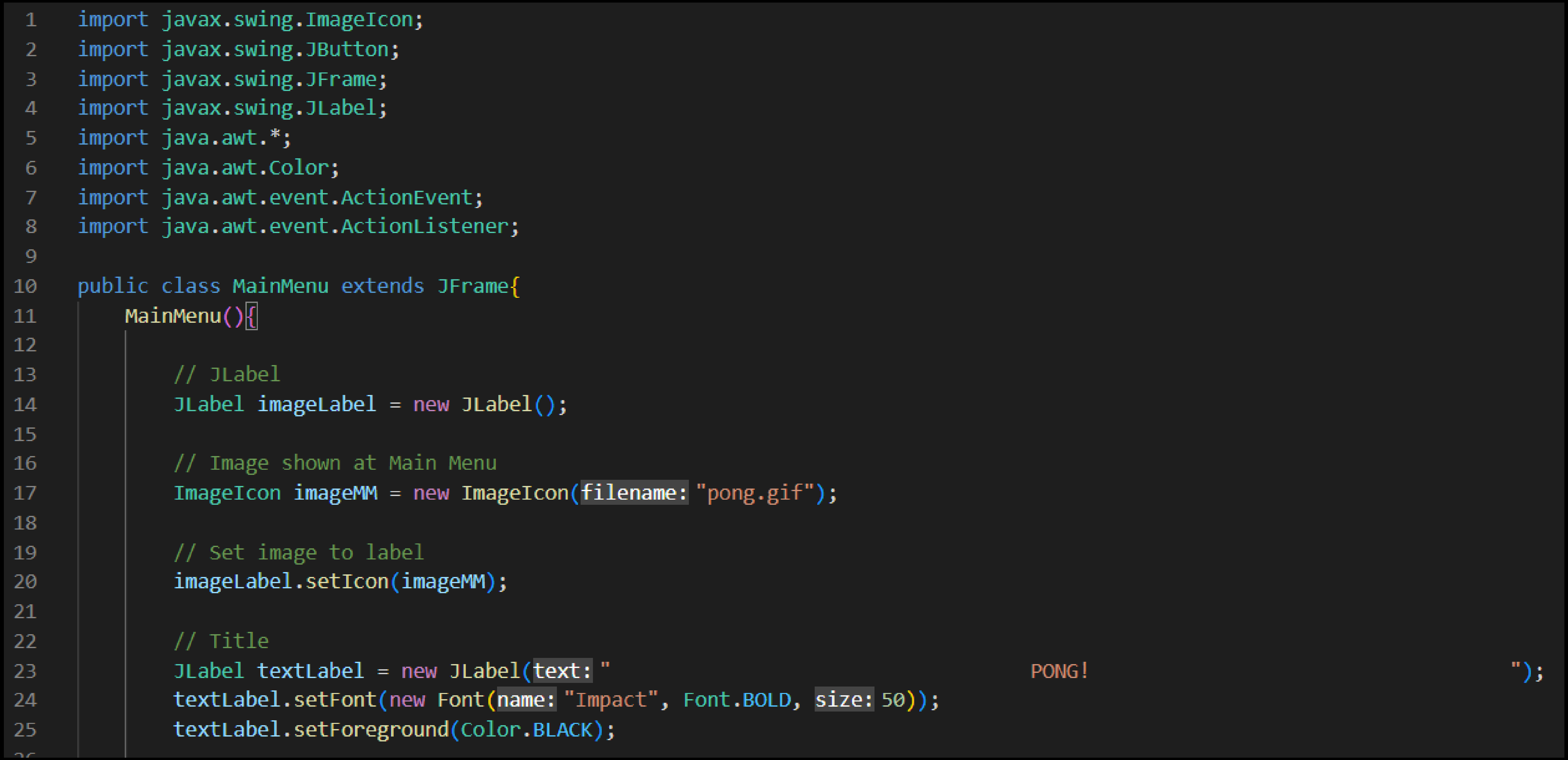


```
public class PongGame {

    Run | Debug
    public static void main(String[] args) {
        new MainMenu();
    }
}
```

Back to Agenda Page

Pong Game class contains the code that sets up the basic structure for a Pong game and launches the main menu for the player to start playing.

MainMenu.java

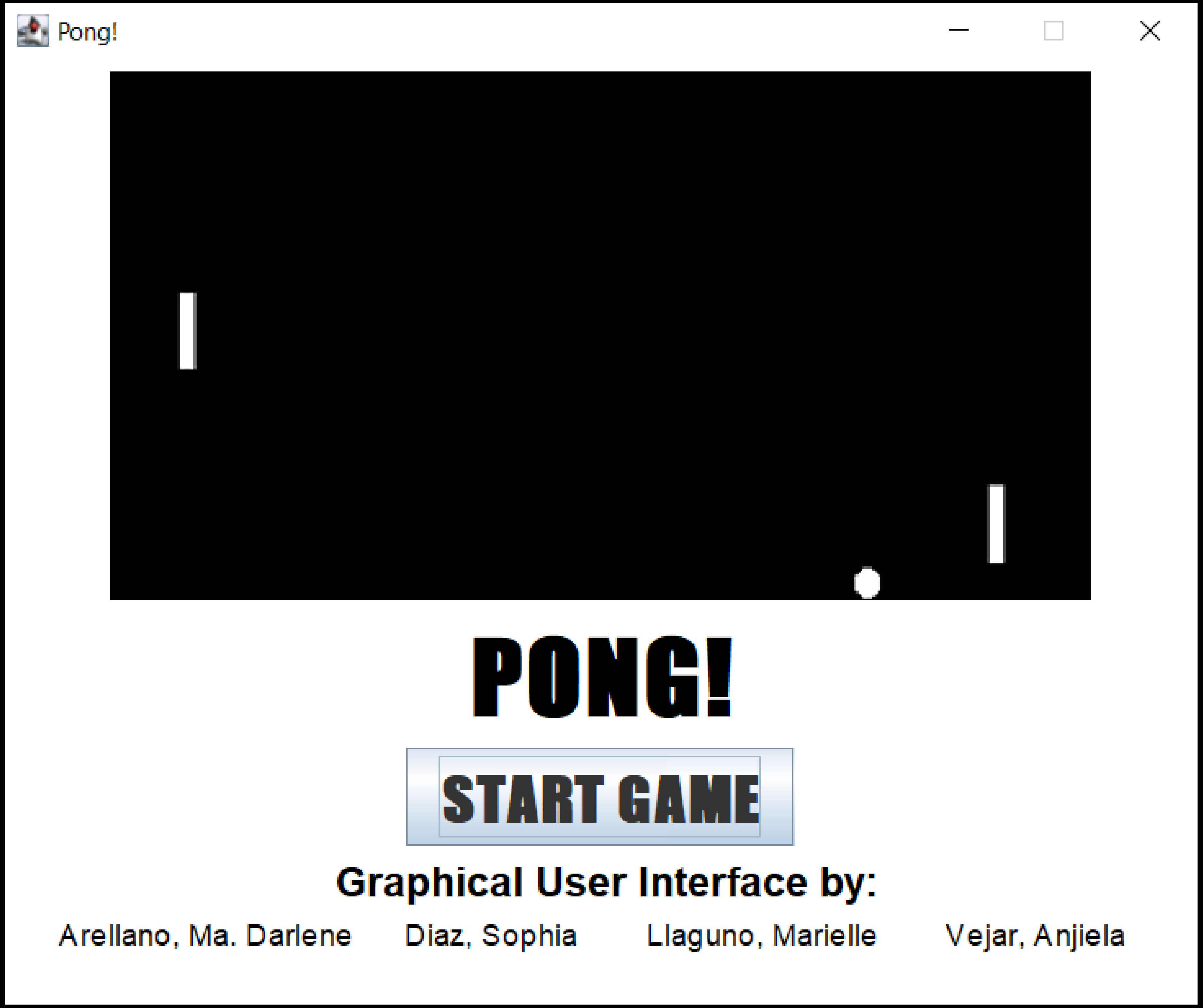


```java
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import java.awt.*;
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MainMenu extends JFrame{
    MainMenu(){

        // JLabel
        JLabel imageLabel = new JLabel();

        // Image shown at Main Menu
        ImageIcon imageMM = new ImageIcon(filename: "pong.gif");

        // Set image to label
        imageLabel.setIcon(imageMM);

        // Title
        JLabel textLabel = new JLabel(text: "                                   PONG!                                   ");
        textLabel.setFont(new Font(name: "Impact", Font.BOLD, size: 50));
        textLabel.setForeground(Color.BLACK);
```

The code appears to be creating a graphical user interface (GUI) for a Pong game. The GUI includes a JLabel that displays an image of the game, a title JLabel that displays the game's name, a JButton labeled "START GAME," and several JLabels that display the names of the GUI creators. Additionally, an event handler is created for the start button that opens the customizations frame and closes the main menu frame.

MainMenu.java

MENU



PONG!

START GAME

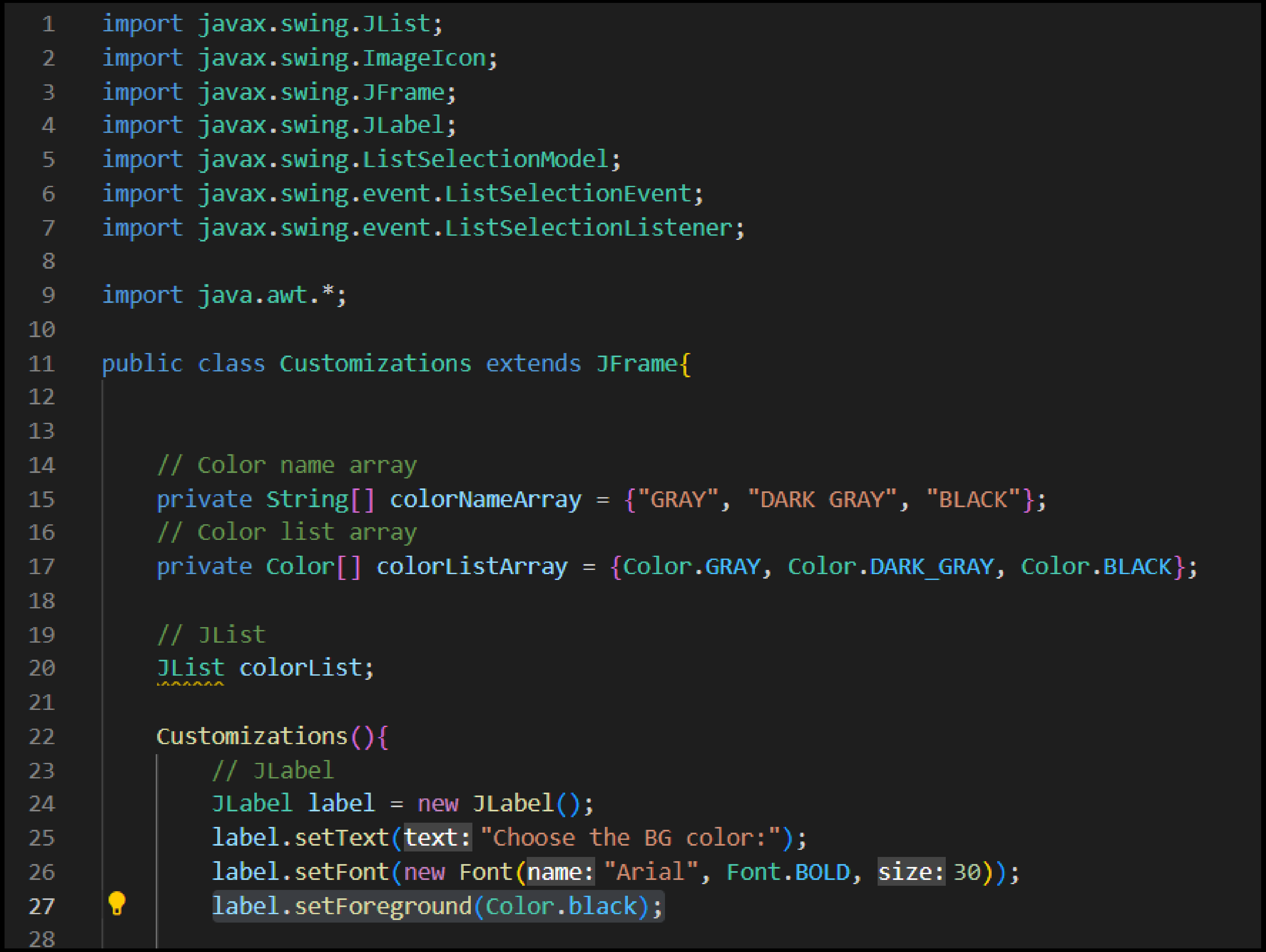
Graphical User Interface by:

Arellano, Ma. Darlene Diaz, Sophia Llaguno, Marielle Vejar, Anjiela

Back to Agenda Page

Customizations.java

MENU



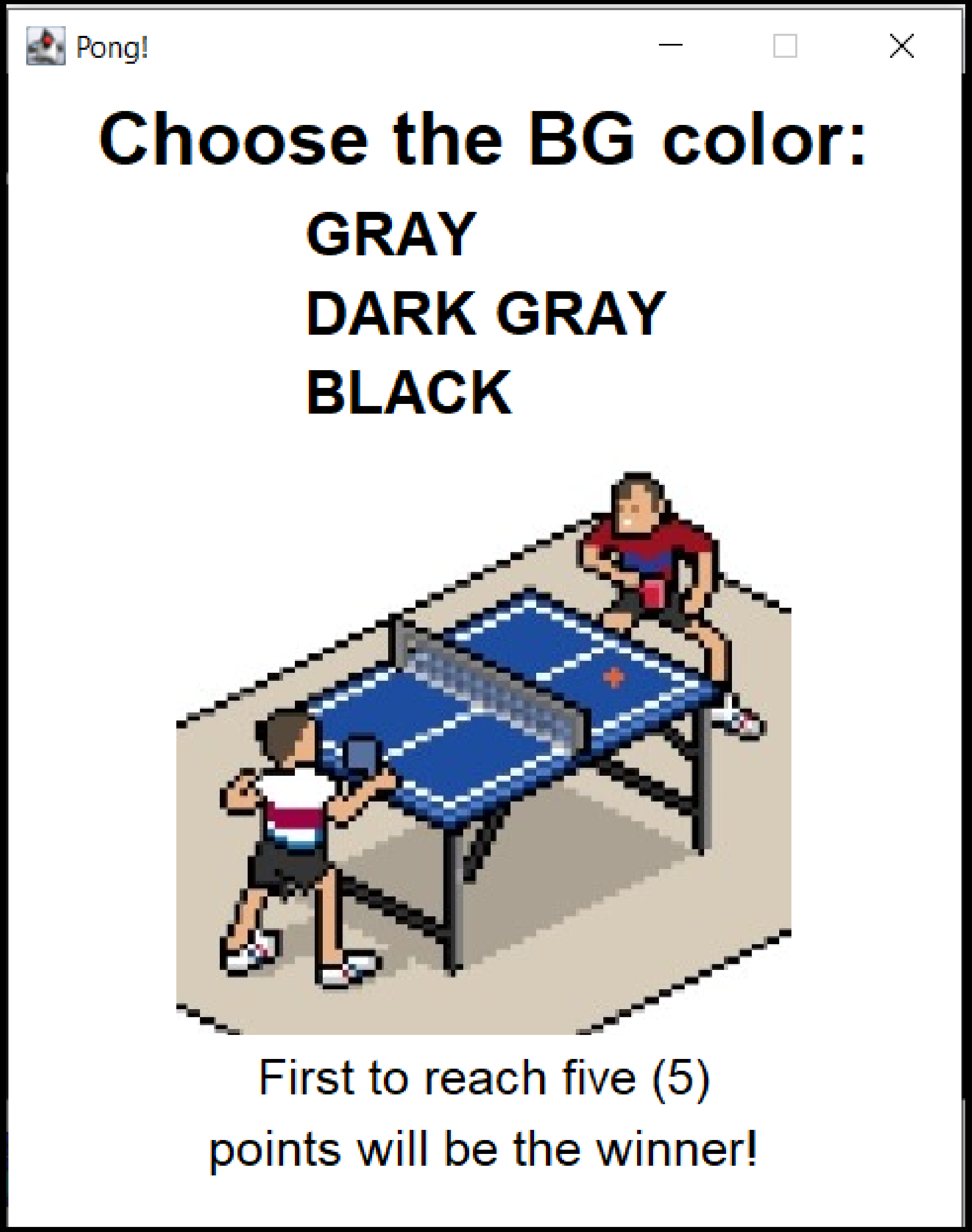
```java
import javax.swing.JList;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ListSelectionModel;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;

import java.awt.*;

public class Customizations extends JFrame{


    // Color name array
    private String[] colorNameArray = {"GRAY", "DARK GRAY", "BLACK"};
    // Color list array
    private Color[] colorListArray = {Color.GRAY, Color.DARK_GRAY, Color.BLACK};

    // JList
    JList colorList;

    Customizations(){
        // JLabel
        JLabel label = new JLabel();
        label.setText(text: "Choose the BG color:");
        label.setFont(new Font(name: "Arial", Font.BOLD, size: 30));
        label.setForeground(Color.black);

```

Back to Agenda Page

Customizations class extends 'JFrame'. It contains a constructor method that creates and sets up a GUI (graphical user interface) for a game of Pong. The GUI contains 'JLabel' that display the text "Choose the BG color:" and a 'JList' that display color options. It also creates a 'JLabel' that display directions for the game and an image of two people playing Pong.

Customizations.java

MENU


Pong!

Choose the BG color:

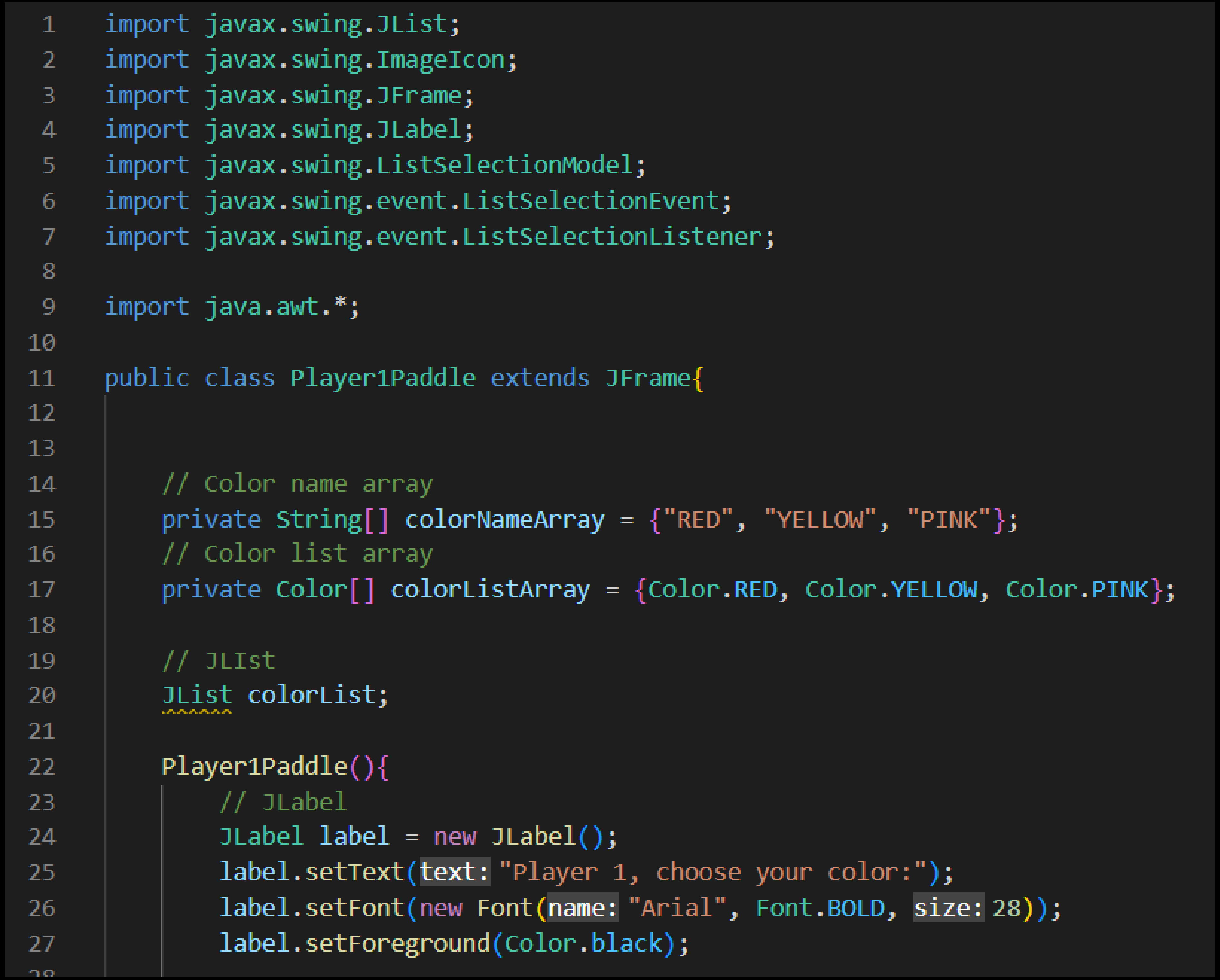
GRAY
DARK GRAY
BLACK

First to reach five (5)
points will be the winner!

Back to Agenda Page

Player1Paddle.java

MENU



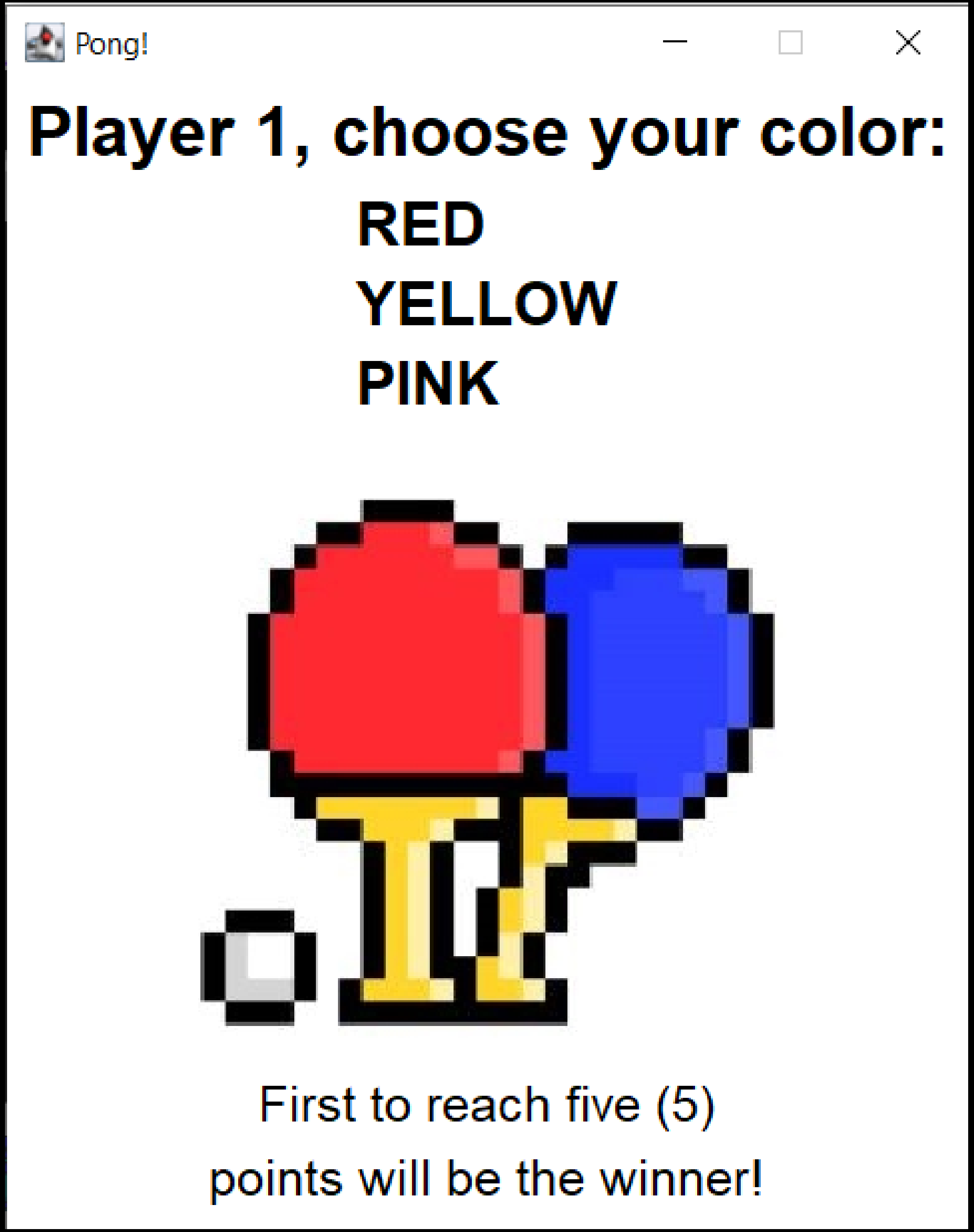
```java
import javax.swing.JList;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ListSelectionModel;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;

import java.awt.*;

public class Player1Paddle extends JFrame{


    // Color name array
    private String[] colorNameArray = {"RED", "YELLOW", "PINK"};
    // Color list array
    private Color[] colorListArray = {Color.RED, Color.YELLOW, Color.PINK};

    // JLIst
    JList colorList;

    Player1Paddle(){
        // JLabel
        JLabel label = new JLabel();
        label.setText(text: "Player 1, choose your color:");
        label.setFont(new Font(name: "Arial", Font.BOLD, size: 28));
        label.setForeground(Color.black);
```

Back to Agenda Page

It creates a GUI window with a label prompting the player to choose a color from a list of three options displayed as a JList. The selected color is used to set the color of the player's paddle in a game of Pong. The GUI window also includes an image, directions, and a fixed size. The class includes an EventHandler class that listens for changes in the selected color and triggers the creation of a new Player1Paddle window while closing the current window.

Player1Paddle.java

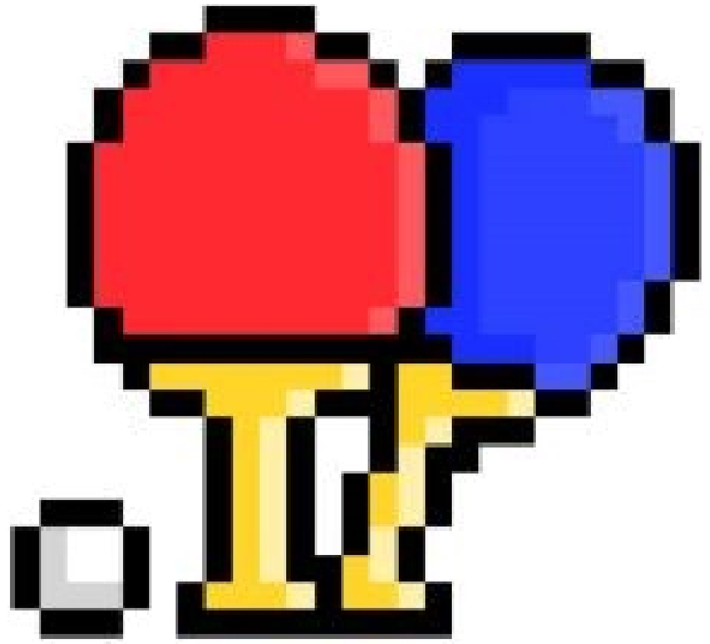
MENU



Pong!

Player 1, choose your color:

RED
YELLOW
PINK

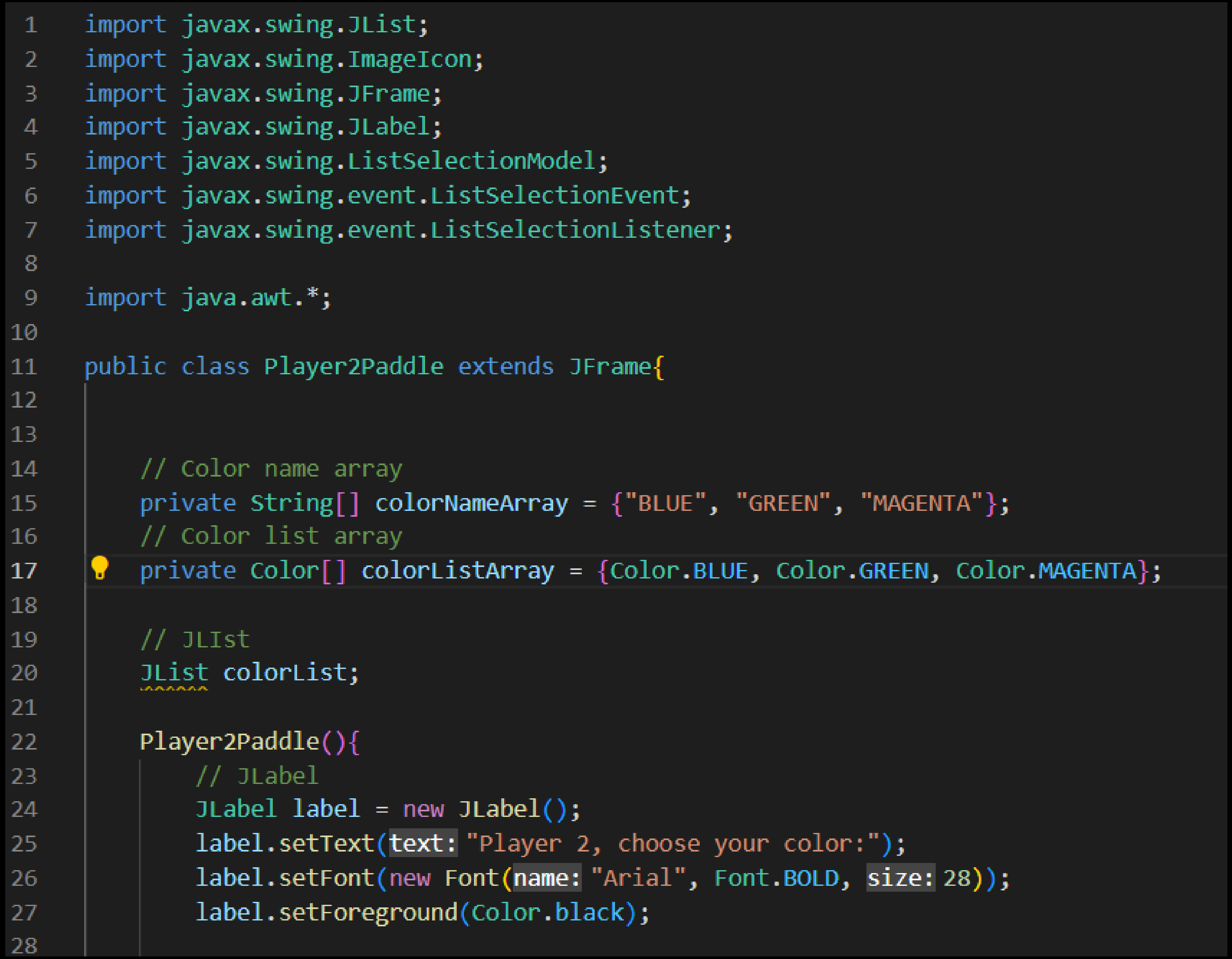


First to reach five (5)
points will be the winner!

Back to Agenda Page

Player2Paddle.java

MENU



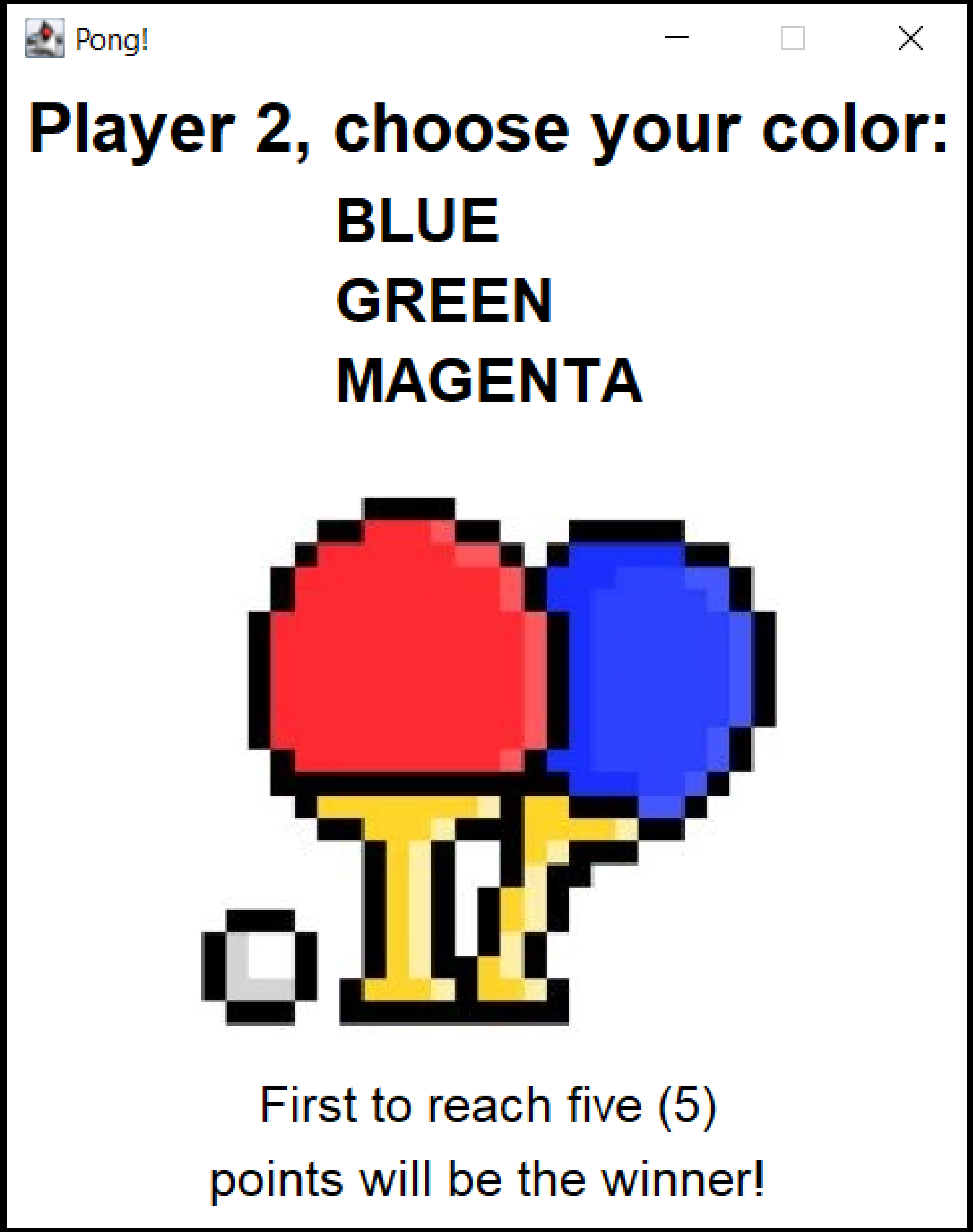
```java
import javax.swing.JList;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ListSelectionModel;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;

import java.awt.*;

public class Player2Paddle extends JFrame{


    // Color name array
    private String[] colorNameArray = {"BLUE", "GREEN", "MAGENTA"};
    // Color list array
    private Color[] colorListArray = {Color.BLUE, Color.GREEN, Color.MAGENTA};

    // JLIst
    JList colorList;

    Player2Paddle(){
        // JLabel
        JLabel label = new JLabel();
        label.setText(text: "Player 2, choose your color:");
        label.setFont(new Font(name: "Arial", Font.BOLD, size: 28));
        label.setForeground(Color.black);

```

Back to Agenda Page

It creates a GUI window with a label prompting the player to choose a color from a list of three options displayed as a JList. The selected color is used to set the color of the player's paddle in a game of Pong. The GUI window also includes an image, directions, and a fixed size. The class includes an EventHandler class that listens for changes in the selected color and triggers the creation of a new Player2Paddle window while closing the current window.

Player2Paddle.java

MENU


Pong!

Player 2, choose your color:

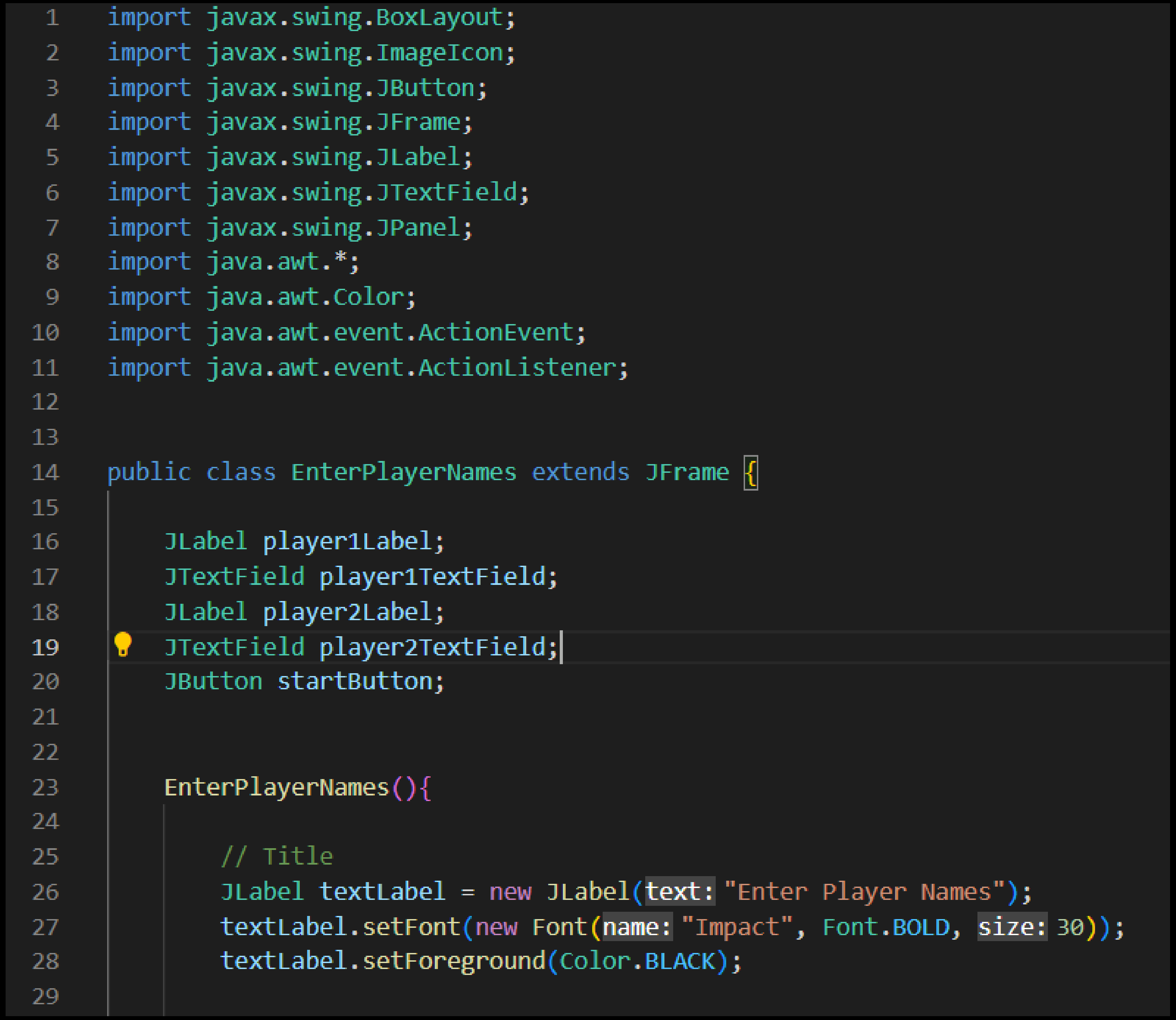
BLUE
GREEN
MAGENTA

First to reach five (5)
points will be the winner!

Back to Agenda Page

EnterPlayerNames.java

MENU



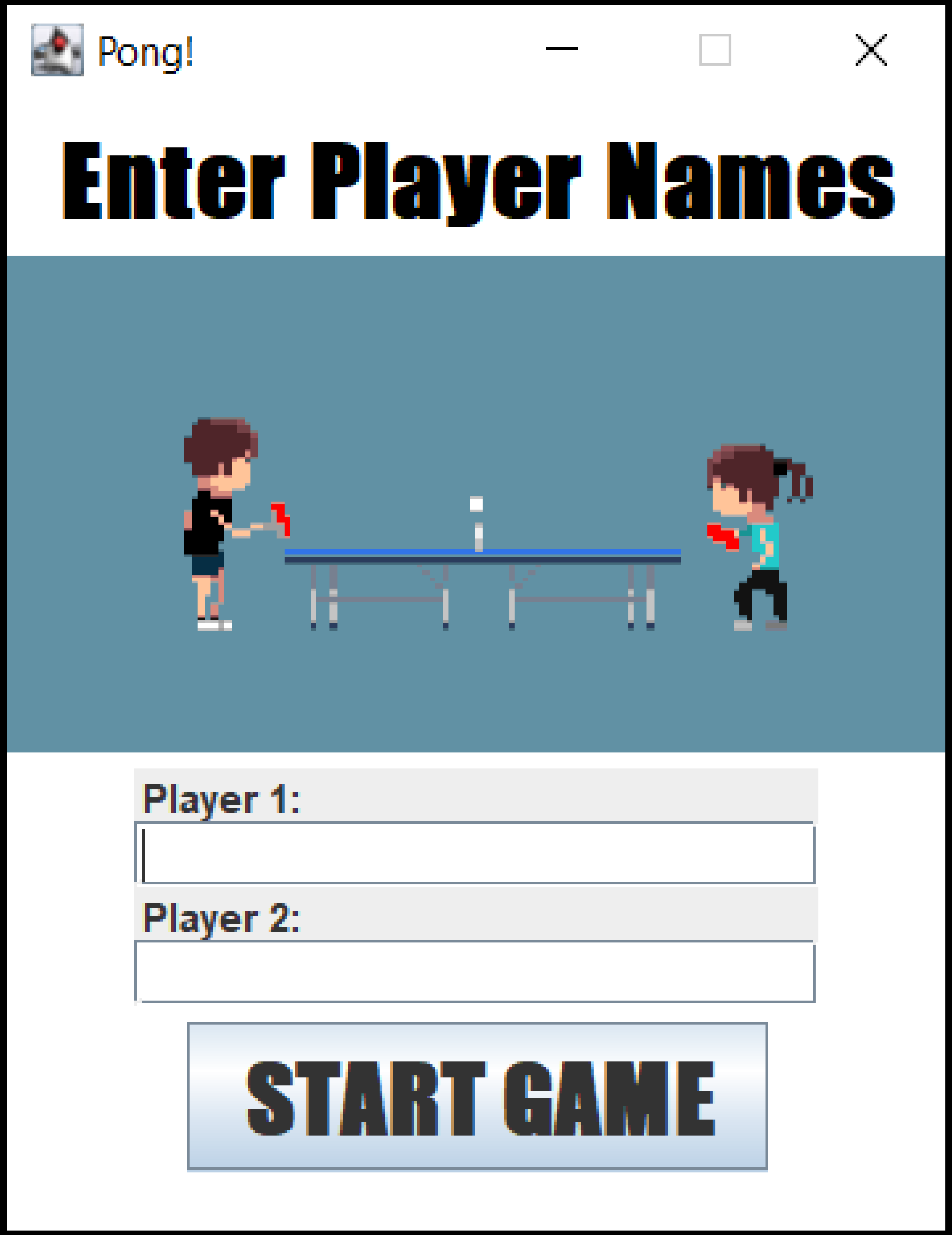
```java
import javax.swing.BoxLayout;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JPanel;
import java.awt.*;
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;


public class EnterPlayerNames extends JFrame {

    JLabel player1Label;
    JTextField player1TextField;
    JLabel player2Label;
    JTextField player2TextField;
    JButton startButton;


    EnterPlayerNames(){

        // Title
        JLabel textLabel = new JLabel(text: "Enter Player Names");
        textLabel.setFont(new Font(name: "Impact", Font.BOLD, size: 30));
        textLabel.setForeground(Color.BLACK);

```

Back to Agenda Page

This specific code creates a JFrame for entering player names. It includes two JLabels and two JTextFields for players to enter their names, an image and a "START GAME" JButton. It also includes an event handler that retrieves player names from the JTextFields and launches the game frame, passing in the player names. The JFrame has a white background, is not resizable, and terminates the program on close.

EnterPlayerNames.java

MENU



Pong!

Enter Player Names

Player 1:

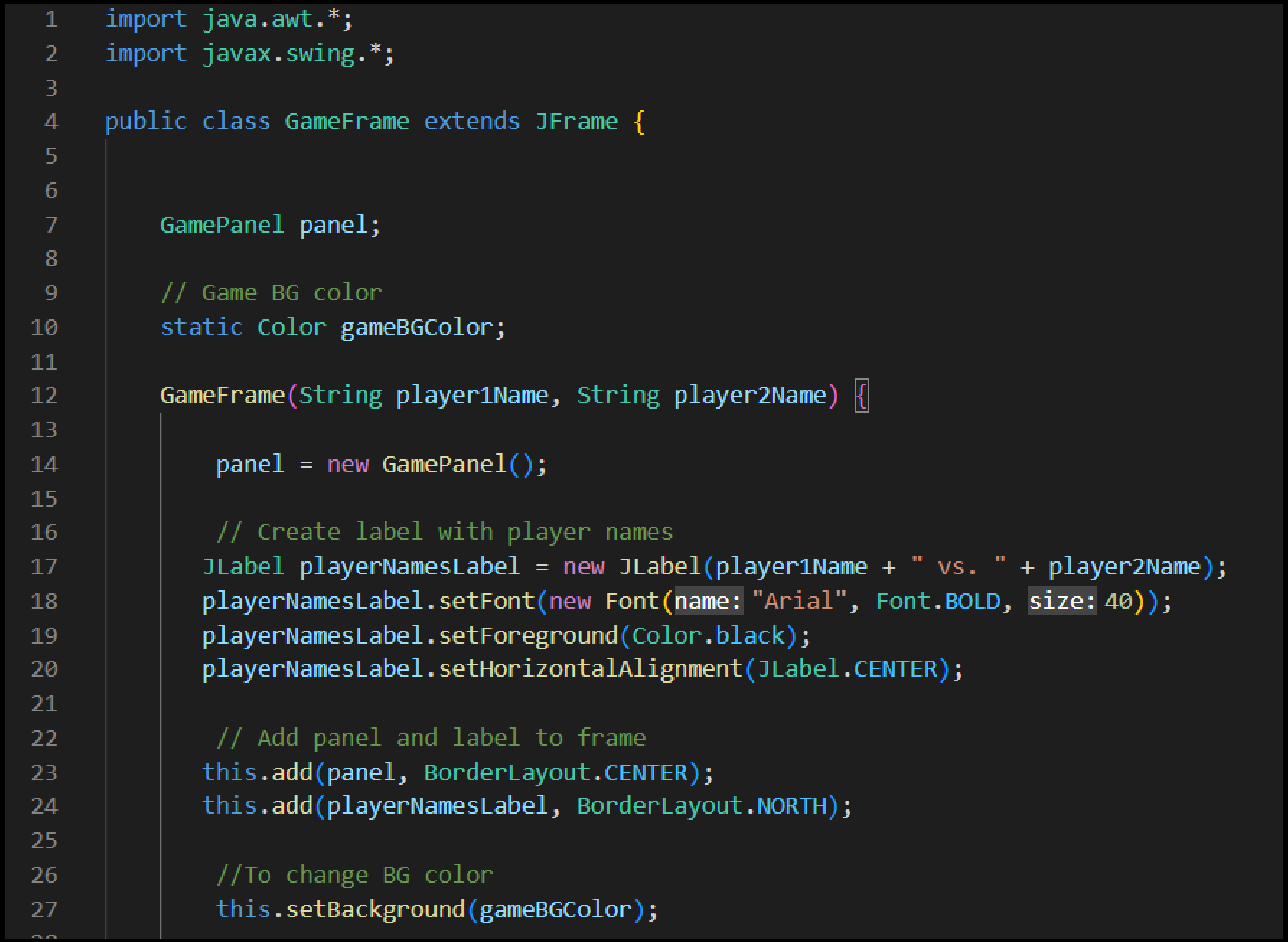
Player 2:

START GAME

Back to Agenda Page

GameFrame.java

MENU



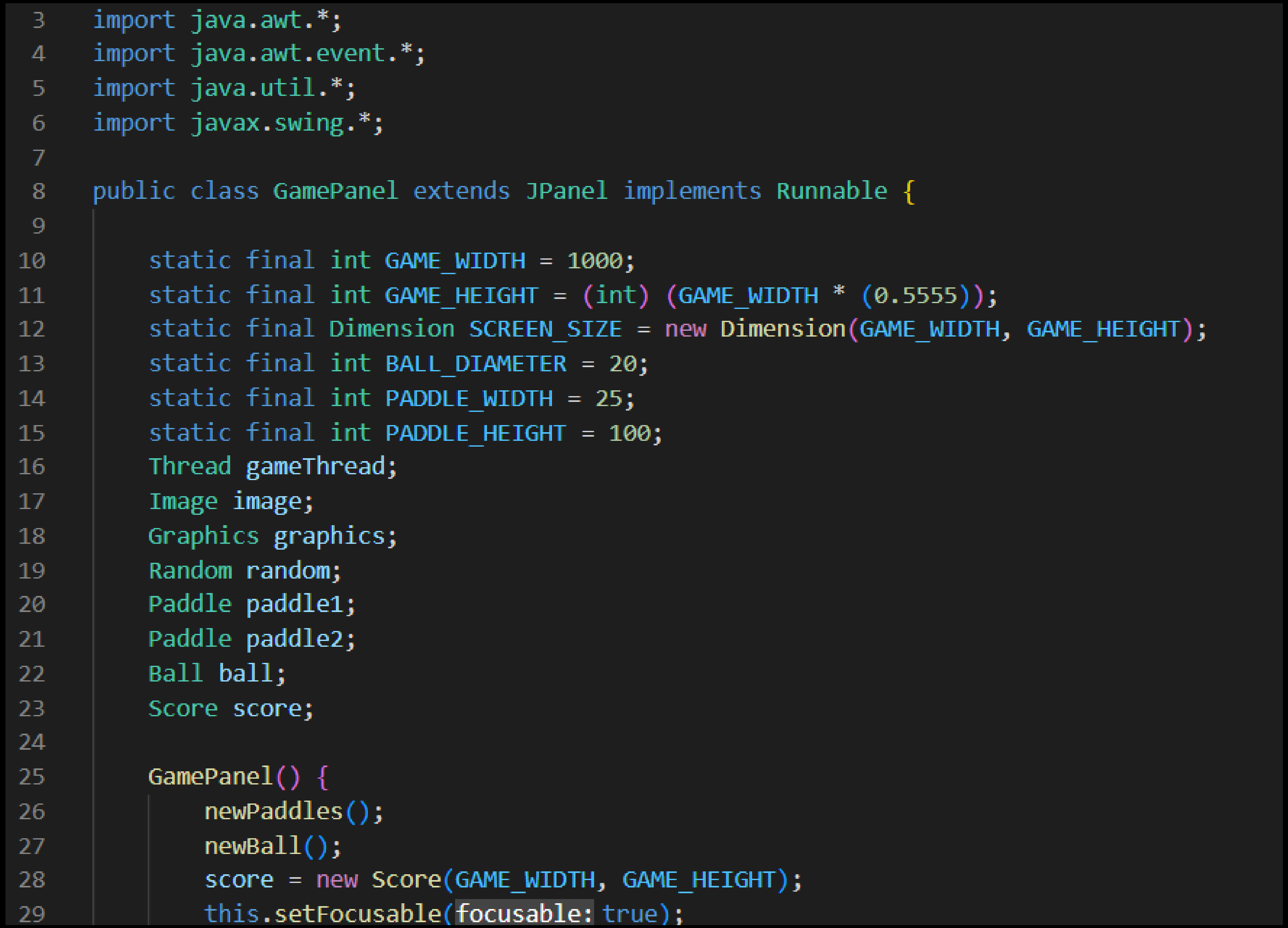
```java
import java.awt.*;
import javax.swing.*;

public class GameFrame extends JFrame {


    GamePanel panel;

    // Game BG color
    static Color gameBGColor;

    GameFrame(String player1Name, String player2Name) {

        panel = new GamePanel();

        // Create label with player names
       JLabel playerNamesLabel = new JLabel(player1Name + " vs. " + player2Name);
       playerNamesLabel.setFont(new Font(name: "Arial", Font.BOLD, size: 40));
       playerNamesLabel.setForeground(Color.black);
       playerNamesLabel.setHorizontalAlignment(JLabel.CENTER);

        // Add panel and label to frame
       this.add(panel, BorderLayout.CENTER);
       this.add(playerNamesLabel, BorderLayout.NORTH);

        //To change BG color
        this.setBackground(gameBGColor);
```

Back to Agenda Page

The GameFrame class extends JFrame and includes a GamePanel object and a static Color object. The constructor initializes the GamePanel object, creates a JLabel object with player names, and adds them to the frame. The background color is set using gameBGColor.

GamePanel.java

MENU

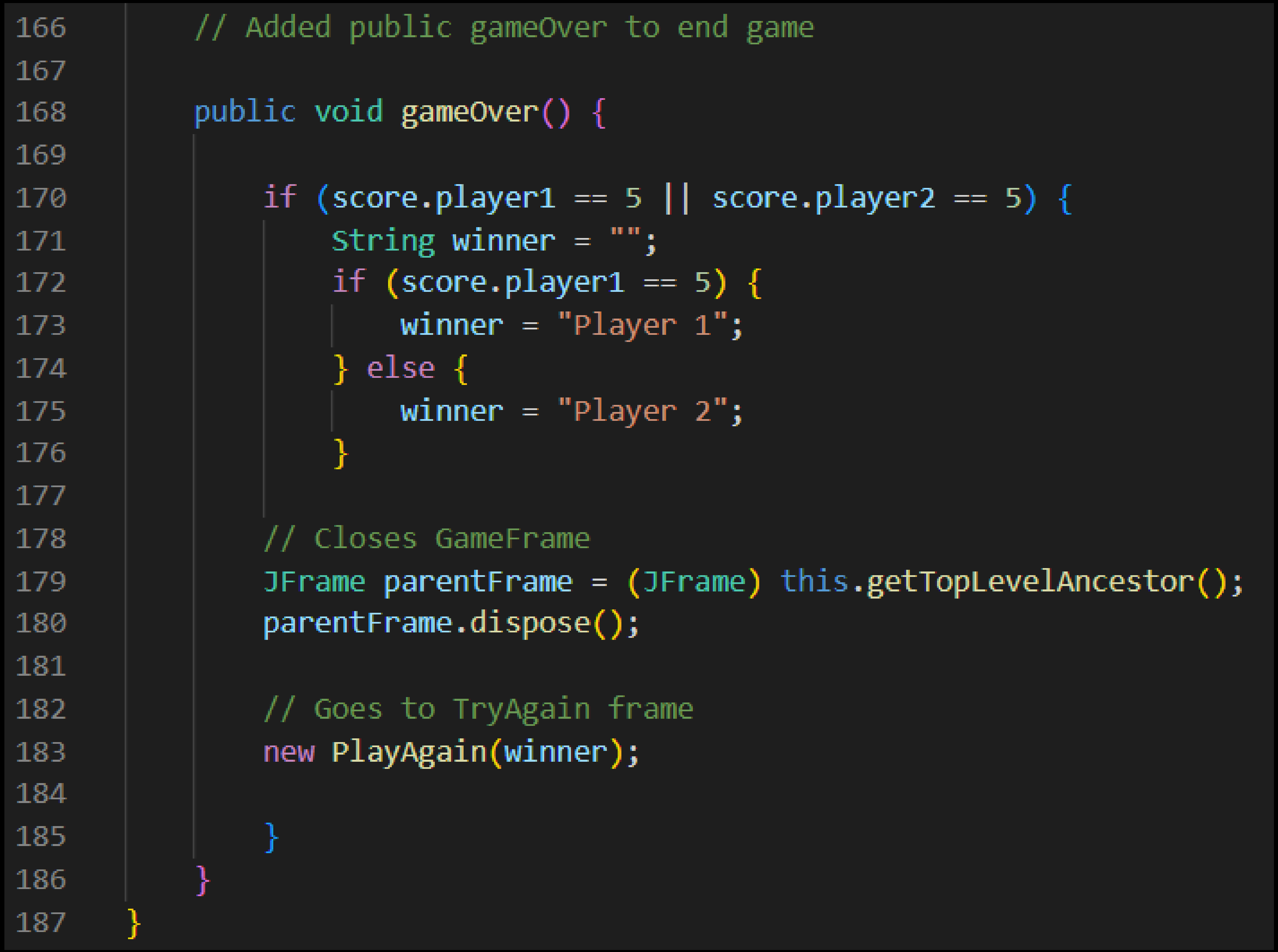


```java
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;

public class GamePanel extends JPanel implements Runnable {

    static final int GAME_WIDTH = 1000;
    static final int GAME_HEIGHT = (int) (GAME_WIDTH * (0.5555));
    static final Dimension SCREEN_SIZE = new Dimension(GAME_WIDTH, GAME_HEIGHT);
    static final int BALL_DIAMETER = 20;
    static final int PADDLE_WIDTH = 25;
    static final int PADDLE_HEIGHT = 100;
    Thread gameThread;
    Image image;
    Graphics graphics;
    Random random;
    Paddle paddle1;
    Paddle paddle2;
    Ball ball;
    Score score;

    GamePanel() {
        newPaddles();
        newBall();
        score = new Score(GAME_WIDTH, GAME_HEIGHT);
        this.setFocusable(focusable: true);
```

Back to Agenda Page

The GamePanel is a custom JPanel (Java Swing component) that is used to construct a game display window. To generate a new thread for the game loop, it extends the JPanel class and implements the Runnable interface. Moreover, it offers several techniques for manipulating the objects, drawing game elements like the paddles, ball, and score, and checking for object collisions. The game window is displayed by adding the GamePanel to a JFrame.

GamePanel.java - gameOver()

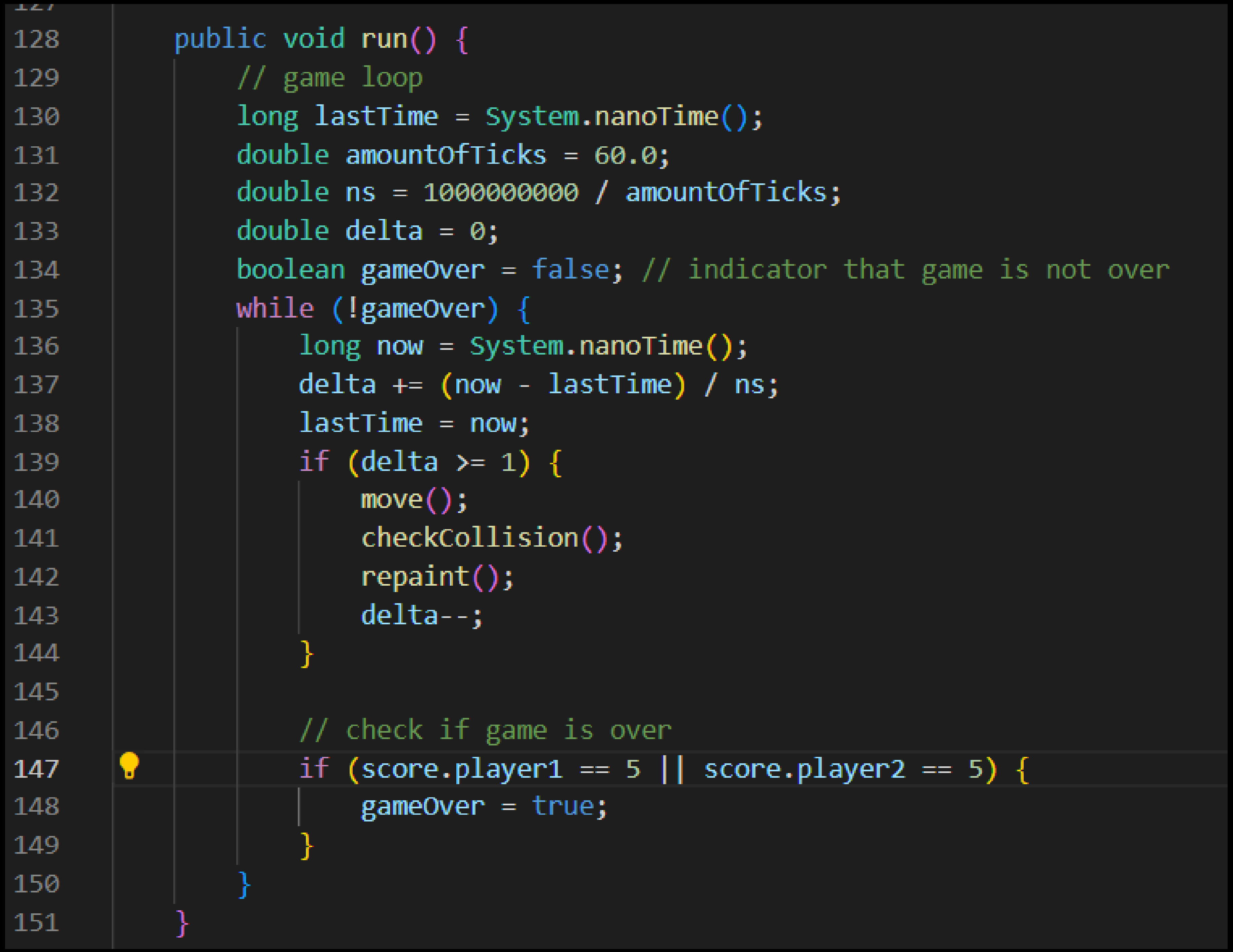


```java
    // Added public gameOver to end game

    public void gameOver() {

        if (score.player1 == 5 || score.player2 == 5) {
            String winner = "";
            if (score.player1 == 5) {
                winner = "Player 1";
            } else {
                winner = "Player 2";
            }

        // Closes GameFrame
        JFrame parentFrame = (JFrame) this.getTopLevelAncestor();
        parentFrame.dispose();

        // Goes to TryAgain frame
        new PlayAgain(winner);

        }
    }
}
```

Back to Agenda Page

public void gameOver was added to the GamePanel class . The method "gameOver( )" is responsible for checking if either player 1 or player 2 has reached a score of 5 points, and if so, it initiates the end of the game. The code assigns the name of the player with the greatest score to the variable "winner" if either player has a score of 5. The current game window (GameFrame) is then terminated by deleting its parent JFrame object.

GamePanel.java - run()



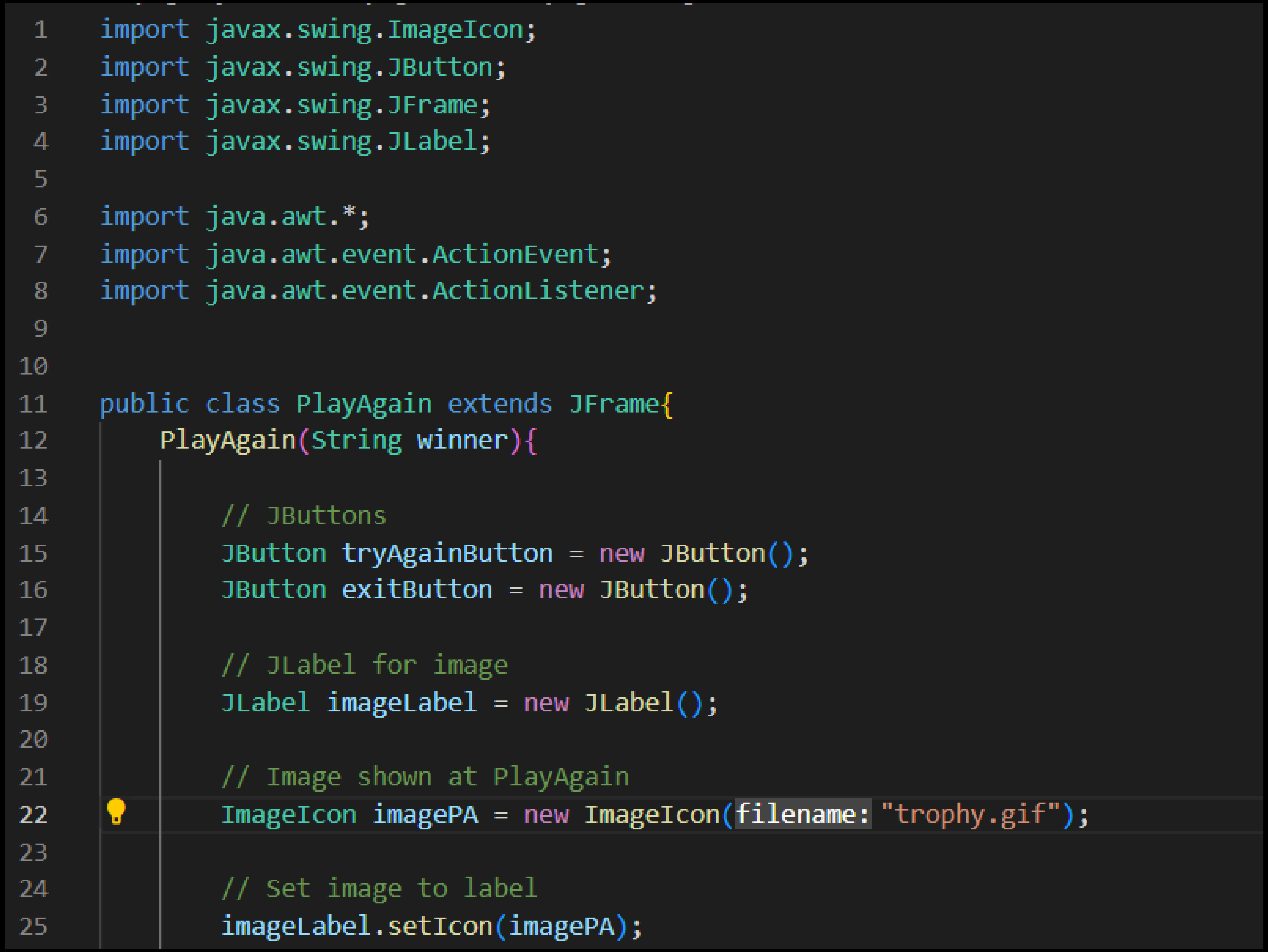
```java
    public void run() {
        // game loop
        long lastTime = System.nanoTime();
        double amountOfTicks = 60.0;
        double ns = 1000000000 / amountOfTicks;
        double delta = 0;
        boolean gameOver = false; // indicator that game is not over
        while (!gameOver) {
            long now = System.nanoTime();
            delta += (now - lastTime) / ns;
            lastTime = now;
            if (delta >= 1) {
                move();
                checkCollision();
                repaint();
                delta--;
            }

            // check if game is over
            if (score.player1 == 5 || score.player2 == 5) {
                gameOver = true;
            }
        }
    }
```

Back to Agenda Page

public void run contains a game loop that runs while the game is not over. The loop terminates when the game is over, which is indicated by either player1 or player2 reaching a score of 5.

PlayAgain.java

MENU



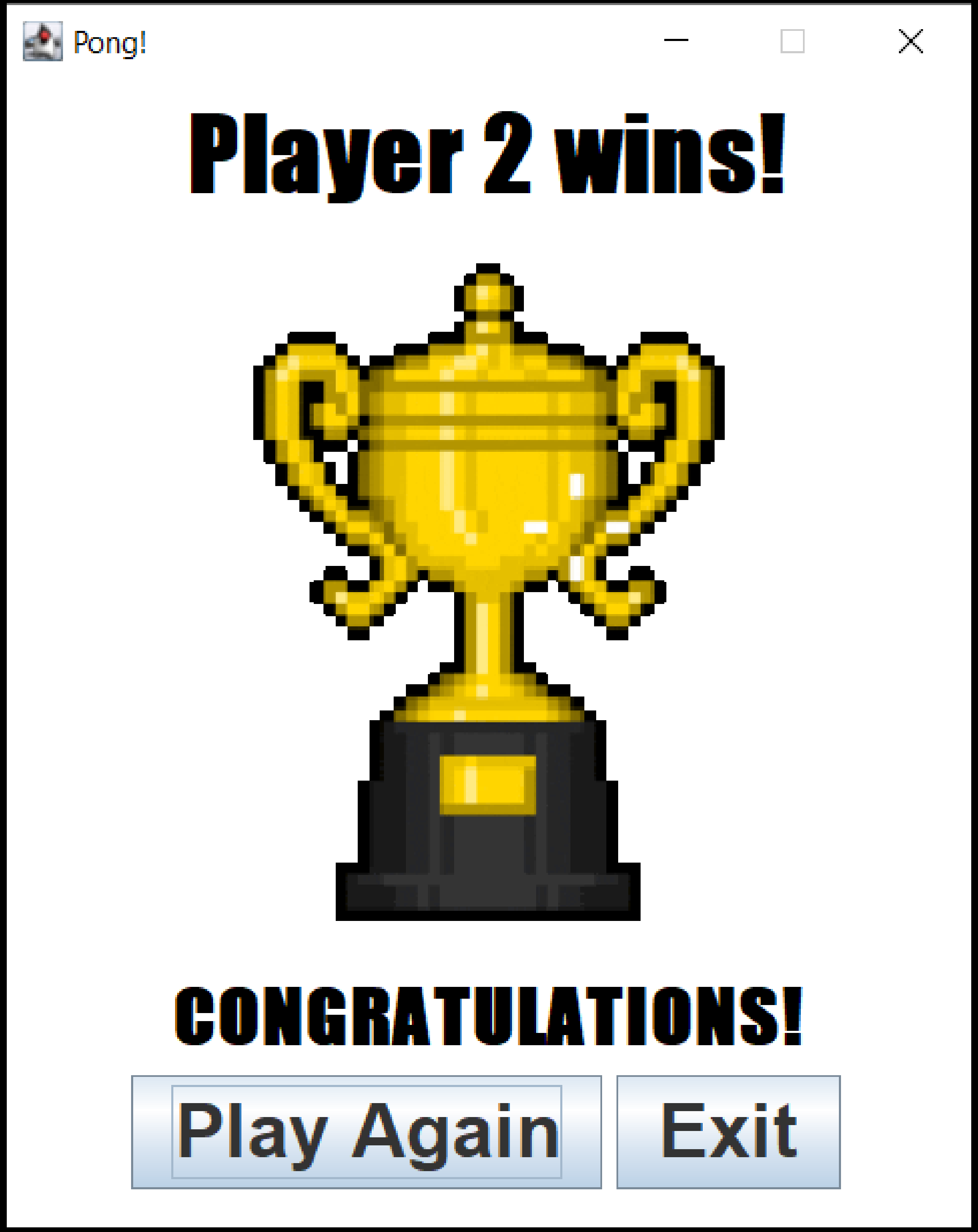
```java
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;


public class PlayAgain extends JFrame{
    PlayAgain(String winner){

        // JButtons
        JButton tryAgainButton = new JButton();
        JButton exitButton = new JButton();

        // JLabel for image
        JLabel imageLabel = new JLabel();

        // Image shown at PlayAgain
        ImageIcon imagePA = new ImageIcon(filename: "trophy.gif");

        // Set image to label
        imageLabel.setIcon(imagePA);
```

Back to Agenda Page

The PlayAgain class generates a new window asking the user players if they want to restart the game or close it. The "winner" variable is passed as a parameter to the PlayAgain class' constructor so that it can display the winning player's name on the screen.

PlayAgain.java

MENU


Pong!

Player 2 wins!

CONGRATULATIONS!

Back to Agenda Page

GAME DEMO

Back to Agenda Page

Thank you!

Thank you all for attending today's presentation.
I hope you found it informative and engaging..

PongGame. java

Use these design resources in your Canva Presentation. Happy designing!

Don't forget to delete this page before presenting.