Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

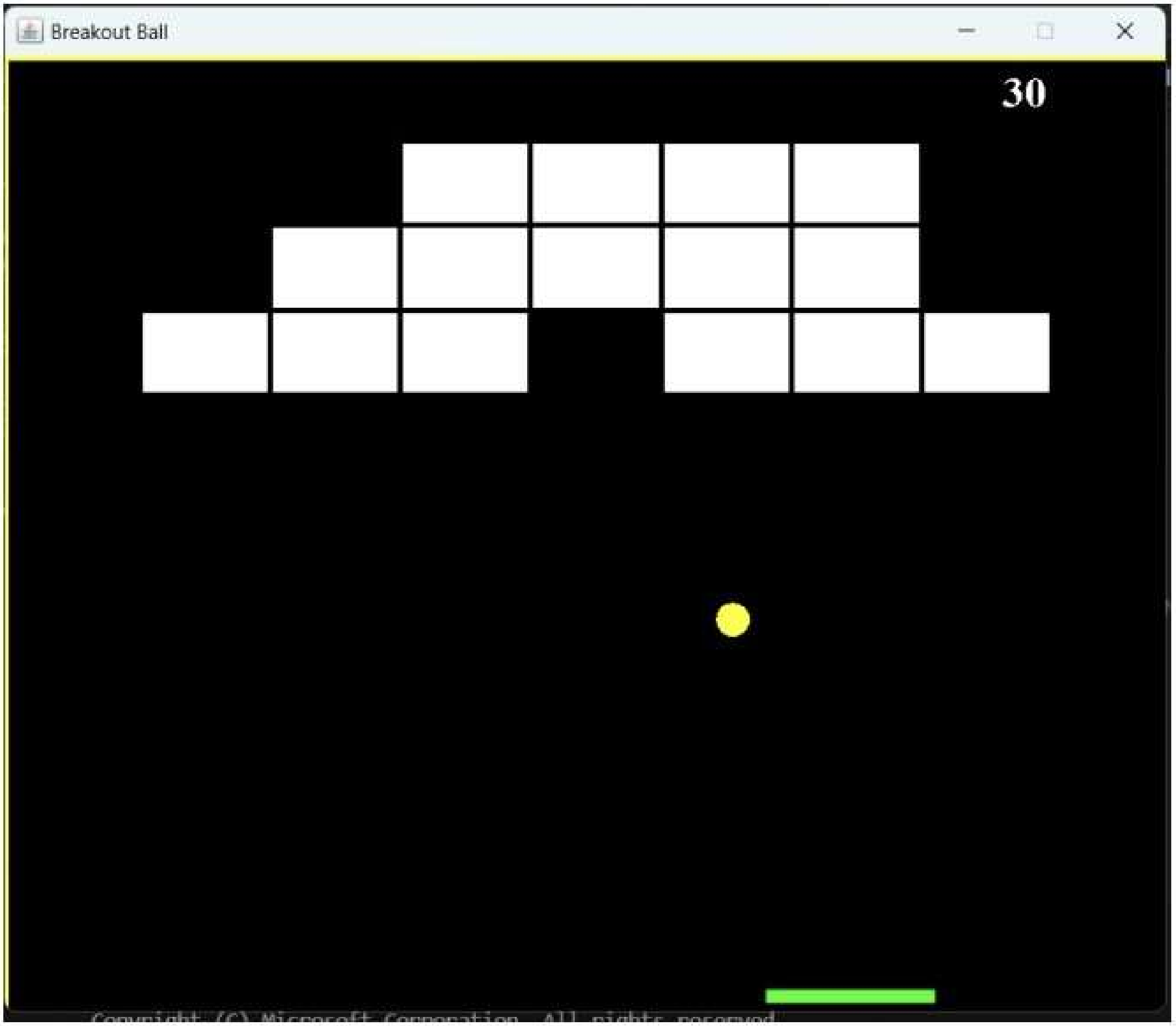
Experiment No. 12
Course Project based on the content of the syllabus.
Date of Performance:
Date of Submission:

Code:

```
package brickBracker;
 import javax.swing.JFrame;
 public class Main {
public static void main(String[] args) {
 JFrame obj = new JFrame();
 Gameplay gamePlay = new Gameplay();
 obj.setBounds(10, 10, 700, 600);
 obj.setTitle("Breakout Ball");
 obj.setResizable(false);
 obj.setVisible(true);
 obj.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 obj.add(gamePlay);
}
 } package brickBracker;
 import java.awt.Color;
import java.awt.Font;
 import java.awt.Graphics;
 import java.awt.Graphics2D;
import java.awt.Rectangle;
 import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
 import java.awt.event.KeyEvent;
 import java.awt.event.KeyListener;
 import javax.swing.JPanel;
 import javax.swing.Timer;
public class Gameplay extends JPanel implements KeyListener, ActionListener {
 private boolean play = false;
 private int score = 0;
private int totalBricks = 21;
 private Timer timer;
 private int delay = 15;
 private int playerX = 310;
 private int ballposX = 120;
private int ballposY = 350;
private int ballXdir = -2;
 private int ballYdir = -1;
 private MapGenerator map;
 public Gameplay() {
 map = new MapGenerator(3, 7);
 addKeyListener(this);
setFocusable(true);
 setFocusTraversalKeysEnabled(false);
 timer = new Timer(delay, this);
 timer.start();
 } public void paint(Graphics g) {
 super.paint(g);
 Background g.setColor(Color.black);
 g.fillRect(1, 1, 692, 592);
 // Drawing map
 map.draw((Graphics2D) g);
// Borders
g.setColor(Color.yellow);
g.fillRect(0, 0, 3, 592);
 g.fillRect(0, 0, 692, 3);
g.fillRect(691, 0, 3, 592);
// Scores
 g.setColor(Color.white);
 g.setFont(new Font("serif", Font.BOLD, 25));
 g.drawString("" + score, 590, 30);
// Paddle
 g.setColor(Color.green);
g.fillRect(playerX, 550, 100, 8);
 // The ball
 g.setColor(Color.yellow);
g.fillOval(ballposX, ballposY, 20, 20);
if (totalBricks <= 0) {
play = false;
 ballXdir = 0;
 ballYdir = 0;
g.setColor(Color.RED);
 g.setFont(new Font("serif", Font.BOLD, 30));
 g.drawString("YOU WON", 260, 300);
 }
 if (ballposY > 570) {
 play = false;
ballXdir = 0;
 ballYdir = 0;
 g.setColor(Color.RED);
g.setFont(new Font("serif", Font.BOLD, 30));
 g.drawString("Game over, Scores: " + score, 190, 300);
 g.setFont(new Font("serif", Font.BOLD, 20));
 g.drawString("Press Enter to Restart", 230, 350);
 }
 }
 public void actionPerformed(ActionEvent e) {
 timer.start();
if (play) {
 if (new Rectangle(ballposX, ballposY, 20, 20).intersects(new Rectangle(playerX, 550, 100, 8))) {
ballYdir = -ballYdir;
 }
 for (int i = 0; i < map.map.length; i++) {
for (int j = 0; j < map.map[0].length; j++) {
if (map.map[i][j] > 0) {
int brickX = j * map.brickWidth + 80;
int brickY = i * map.brickHeight + 50;
int brickWidth = map.brickWidth;
int brickHeight = map.brickHeight;
 Rectangle rect = new Rectangle(brickX, brickY, brickWidth, brickHeight);
Rectangle ballRect = new Rectangle(ballposX, ballposY, 20, 20);
 Rectangle brickRect = rect;
if (ballRect.intersects(brickRect)) {
map.setBrickValue(0, i, j);
 totalBricks--;
score += 5;
if (ballposX + 19 <= brickRect.x || ballposX + 1 >= brickRect.x + brickRect.width) {
 ballXdir = -ballXdir;
} else {
 ballYdir = -ballYdir;
 } break A;
}
 }
 }
 }
ballposX += ballXdir;
 ballposY += ballYdir;
 if (ballposX < 0) {
 ballXdir = -ballXdir;
 } if (ballposY < 0) {
 ballYdir = -ballYdir;
 } if (ballposX > 670) {
ballXdir = -ballXdir;
 }
 }
 repaint();
 }
 public void keyTyped(KeyEvent e) {
} public void keyReleased(KeyEvent e) {
} public void keyPressed(KeyEvent e) {
 if (e.getKeyCode() == KeyEvent.VK_RIGHT) {
if (playerX >= 600) {
playerX = 600;
 } else {
 moveRight();
 }
}
if (e.getKeyCode() == KeyEvent.VK_LEFT) {
if (playerX < 10) {
 playerX = 10;
 } else {
 moveLeft();
 }
 }
 if (e.getKeyCode() == KeyEvent.VK_ENTER) {
if (!play) {
 play = true;
 ballposX = 120;
 ballposY = 350;
 ballXdir = -1;
 ballYdir = -2;
playerX = 310;
 score = 0;
totalBricks = 21;
 map = new MapGenerator(3, 7);
 repaint();
 }
 }
 }
 public void moveRight() {
play = true;
 playerX += 20;
} public void moveLeft() {
 play = true;
playerX -= 20;
 }
}
```

```
package brickBracker;
 import java.awt.BasicStroke;
 import java.awt.Color;
 import java.awt.Graphics2D;
public class MapGenerator {
 public int map[][];
 public int brickWidth;
 public int brickHeight;
public MapGenerator(int row, int col) {
map = new int[row][col];
 for (int i = 0; i < map.length; i++) {
 for (int j = 0; j < map[0].length; j++) {
 map[i][j] = 1;
 }
 }
 brickWidth = 540 / col;
brickHeight = 150 / row;
 }
 public void draw(Graphics2D g) {
for (int i = 0; i < map.length; i++) {
 for (int j = 0; j < map[0].length; j++) {
if (map[i][j] > 0) {
 g.setColor(Color.white);
 g.fillRect(j * brickWidth + 80, i * brickHeight + 50, brickWidth, brickHeight);
 g.setStroke(new BasicStroke(3));
g.setColor(Color.black);
 g.drawRect(j * brickWidth + 80, i * brickHeight + 50, brickWidth, brickHeight);
}
 }
 }
 }
 public void setBrickValue(int value, int row, int col) {
 map[row][col] = value;
}
}
```

OUTPUT:



Conclusion:
Results: The Brick Breaker game has been successfully implemented with the following results and observations: 1. Gameplay Experience: The game offers an engaging and enjoyable gaming experience. Players can control the paddle's movement, bounce the ball off the paddle, and break bricks to score points. 2. Graphics and User Interface: The game features a visually appealing graphical user interface. The paddle, ball, and bricks are rendered smoothly. The game provides clear feedback to the player, including a scoring system and game over messages. 3. Game Logic: The game logic, including collision detection and handling, works as expected. The ball reacts appropriately to collisions with the paddle, walls, and bricks. The win and lose conditions are correctly implemented. 4. User Interaction: The game responds effectively to user input. Players can control the paddle using the left and right arrow keys. The "Enter" key allows them to restart the game after a win or loss. 5. Scoring System: The scoring system rewards players with points for breaking bricks. The score is updated on the screen in real-time, providing positive reinforcement to the player. In conclusion, the Brick Breaker game is a well-structured and functional 2D arcade game developed in Java. It provides a classic gaming

experience that challenges players' reflexes and hand-eye coordination. The following key points summarize the project: 1. Game Development: The project demonstrates the development of a 2D game using Java and the Swing library. It showcases fundamental game development concepts, including graphics rendering, user input handling, and collision detection. 2. User Engagement: The game successfully engages players with its intuitive controls and rewarding gameplay. Breaking bricks and aiming for a high score creates a sense of achievement. 18. 3. Educational Value: The project can serve as an educational resource for individuals learning game development or programming. It illustrates fundamental concepts of game design and object-oriented programming. 4. Future Enhancements: While the current game is fully functional, there is room for further improvements and additions. Future enhancements could include additional levels, power-ups, and enhanced graphics. 5. Open Source: The project is open-source and can be further developed or customized by the community. It serves as a foundation for creating various 2D ball-and-paddle games. In summary, the Brick Breaker game project has been successfully implemented, providing an entertaining gaming experience. It reflects the principles of game development and offers opportunities for expansion and customization. The project's open-source nature encourages collaboration and innovation in the realm of 2D game development