



# **BRICK BREAKER GAME**



## **A PROJECT REPORT**

*Submitted by*

**AFIYA J (2303811710422004)**

*in partial fulfillment of requirements for the award of the course*

**CGB1201 - JAVA PROGRAMMING**

*In*

**COMPUTER SCIENCE AND ENGINEERING**

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY**

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

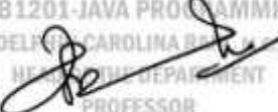
**SAMAYAPURAM-621112**

**NOVEMBER- 2024**

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY**  
**(AUTONOMOUS)**  
**SAMAYAPURAM-621112**

**BONAFIDECERTIFICATE**

Certified that this project report on “**BRICK BREAKER GAME**” is the bonafide work of **AFIYA J (2303811710422004)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



CGB1201-JAVA PROGRAMMING  
Dr. A. DELPHIN CAROLINA RANI, M.E., Ph.D.,  
HEAD OF THE DEPARTMENT  
PROFESSOR

**SIGNATURE**

Dr. A. Delphin Carolina Rani, M.E., Ph.D.,

**HEAD OF THE DEPARTMENT**

**PROFESSOR**

Department of CSE

K. Ramakrishnan College of Technology

(Autonomous)

Samayapuram-621112.



CGB1201-JAVA PROGRAMMING  
Mr. M. SARAVANAN, M.E.,  
SUPERVISOR  
ASSISTANT PROFESSOR

**SIGNATURE**

Mr. M. Saravanan, M.E.,

**SUPERVISOR**

**ASSISTANT PROFESSOR**

Department of CSE

K. Ramakrishnan College of Technology

(Autonomous)

Samayapuram-621112.

Submitted for the viva-voce examination held on 02.12.2024



CGB1201-JAVA PROGRAMMING  
Mr. M. SARAVANAN, M.E.,  
SUPERVISOR  
ASSISTANT PROFESSOR

**INTERNAL EXAMINER**



CGB1201-JAVA PROGRAMMING  
Dr. P. SENTHILSELVI, M.E., Ph.D.,  
INTERNAL EXAMINER  
PROFESSOR  
8138-SCE, TRICHY.

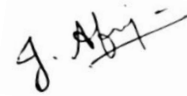
**EXTERNAL EXAMINER**

## DECLARATION

I declare that the project report on **“BRICK BREAKER GAME”** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201- JAVA PROGRAMMING**.

.

**Signature**

A handwritten signature in black ink, appearing to read 'g. Afiya', with a horizontal line extending from the end.

**AFIYA J**

Place: Samayapuram

Date: 02.12.2024

## ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project guide **Mr. M. SARAVANAN, M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

## **VISION OF THE INSTITUTION**

To serve the society by offering top-notch technical education on par with global standards

## **MISSION OF THE INSTITUTION**

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

## **VISION OF DEPARTMENT**

To be a center of eminence in creating competent software professionals with research and innovative skills.

## **MISSION OF DEPARTMENT**

**M1: Industry Specific:** To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

**M2: Research:** To prepare students for research-oriented activities.

**M3: Society:** To empower students with the required skills to solve complex technological problems of society.

## **PROGRAM EDUCATIONAL OBJECTIVES**

### **1. PEO1: Domain Knowledge**

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

### **2. PEO2: Employability Skills and Research**

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

### **3. PEO3: Ethics and Values**

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

### **PROGRAM SPECIFIC OUTCOMES (PSOs)**

#### **PSO 1: Domain Knowledge**

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

#### **PSO 2: Quality Software**

To apply software engineering principles and practices for developing quality software for scientific and business applications.

#### **PSO 3: Innovation Ideas**

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

### **PROGRAM OUTCOMES (POs)**

Engineering students will be able to:

**1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

**2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

**3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations

**4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
- 8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## **ABSTRACT**

The Brick Breaker Game is a Java-based interactive arcade game designed to showcase core programming concepts, graphical rendering, and event-driven mechanics. The game involves a paddle controlled by the player, a ball that moves dynamically, and a grid of bricks that are destroyed upon collision with the ball. The primary objective is to clear all bricks while preventing the ball from falling below the paddle. The game ends when all bricks are destroyed or the player loses all lives.

This project leverages Java's Swing framework for graphical user interface (GUI) rendering and animation. Key features include collision detection using `Rectangle2D`, event handling for paddle movement, and real-time updates to score and game state.

The game offers an engaging experience while providing insights into GUI development, game mechanics, and efficient coding practices. It serves as an excellent learning tool for understanding Java's capabilities in creating interactive applications.



### ABSTRACT WITH POs AND PSOs MAPPING

#### CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

| ABSTRACT  | POs MAPPED   | PSOs MAPPED  |
|---|--|--|
| The Brick Breaker Game is a classic arcade game where the player controls a paddle to bounce a ball and break bricks arranged on the screen. The game challenges the player with progressively harder levels, power-ups, and obstacles. The ball's speed increases as the game advances, requiring the player to use strategy and quick reflexes. | <b>PO1 -3</b><br><b>PO2 -3</b><br><b>PO3 -3</b><br><b>PO4 -3</b><br><b>PO5 -3</b><br><b>PO6 -3</b><br><b>PO7 -3</b><br><b>PO8 -3</b><br><b>PO9 -3</b><br><b>PO10 -3</b><br><b>PO11-3</b><br><b>PO12 -3</b> | <b>PSO1 -3</b><br><b>PSO2 -3</b><br><b>PSO3 -3</b> |

Note: 1- Low, 2-Medium, 3- High

## TABLE OF CONTENTS

| CHAPTER NO. | TITLE                                | PAGE NO. |
|-------------|--------------------------------------|----------|
|             | <b>ABSTRACT</b>                      | VIII     |
| <b>1</b>    | <b>INTRODUCTION</b>                  | 1        |
|             | 1.1 Objective                        | 1        |
|             | 1.2 Overview                         | 1        |
|             | 1.3 Java Programming concepts        | 2        |
| <b>2</b>    | <b>PROJECT METHODOLOGY</b>           | 3        |
|             | 2.1 Proposed Work                    | 3        |
|             | 2.2 Block Diagram                    | 3        |
| <b>3</b>    | <b>MODULE DESCRIPTION</b>            | 4        |
|             | 3.1 Game Initialization Module       | 4        |
|             | 3.2 Ball Module                      | 4        |
|             | 3.3 Paddle Module                    | 4        |
|             | 3.4 Brick Module                     | 4        |
|             | 3.5 Game Logic and Control Module    | 5        |
| <b>4</b>    | <b>CONCLUSION &amp; FUTURE SCOPE</b> | 6        |
|             | 4.1 Conclusion                       | 6        |
|             | 4.2 Future Scope                     | 6        |
|             | <b>REFERENCES</b>                    | 18       |
|             | <b>APPENDIX A (SOURCE CODE)</b>      | 7        |
|             | <b>APPENDIX B (SCREENSHOTS)</b>      | 15       |

# CHAPTER 1

## INTRODUCTION

### 1.1 Objective

The objective of the brick breaker game is for the player to control a paddle and bounce a ball to break all the bricks on the screen. The player must avoid letting the ball fall off the bottom of the screen, which would result in a loss of a life. Each brick requires one or more hits to break, and the player earns points for every brick destroyed. As the player progresses through the levels, the difficulty increases, with more complex brick layouts, faster ball speeds, or additional obstacles. The ultimate goal is to clear all the bricks while keeping the ball in play and achieving the highest score possible.

### 1.2 Overview

Brick Breaker is a classic arcade-style game where the player controls a paddle at the bottom of the screen to bounce a ball that breaks bricks arranged above. The objective is to clear all the bricks by hitting them with the ball. The ball bounces off walls and the paddle, and the player must prevent it from falling off the screen. As the player progresses through levels, the bricks become more complex, and new obstacles or power-ups are introduced.

Power-ups can change the paddle's size, speed, or ability to shoot lasers, while some bricks may require multiple hits to break. The game typically features multiple lives, and losing all of them results in a game over.

Originally launched as Breakout in 1976, it has since inspired numerous versions and remains popular on many platforms, from arcade machines to mobile app.

### 1.3 Java Programming Concepts

- **Classes and Objects** are the building blocks of any Java program. In a Brick Breaker game, you will create classes that define the various entities in the game (e.g., Ball, Paddle, Brick, and Game).
- **Inheritance** : Used to create different types of bricks that share common properties (like position and size), but each brick can have its own unique behavior (e.g., indestructible or power-up).
- **Control flow(loops and conditionals):** Loops are used to continuously check for game events such as moving the ball, checking for collisions, and updating the game state. Conditionals (if, else) are used to make decisions, such as when the ball hits the paddle, bricks, or walls.

- **Event handling:** In Java, event handling like ActionListener is used for handle time and Keylistener is used for handle user input
- **Swing and AWT:** Component like JFrame ,JLabel, JButton and Jpanel are used to build the user interface.
- **Array list:** ArrayLists and other collections are useful for managing dynamic collections of objects, such as the collection of bricks in the game.

## CHAPTER 2

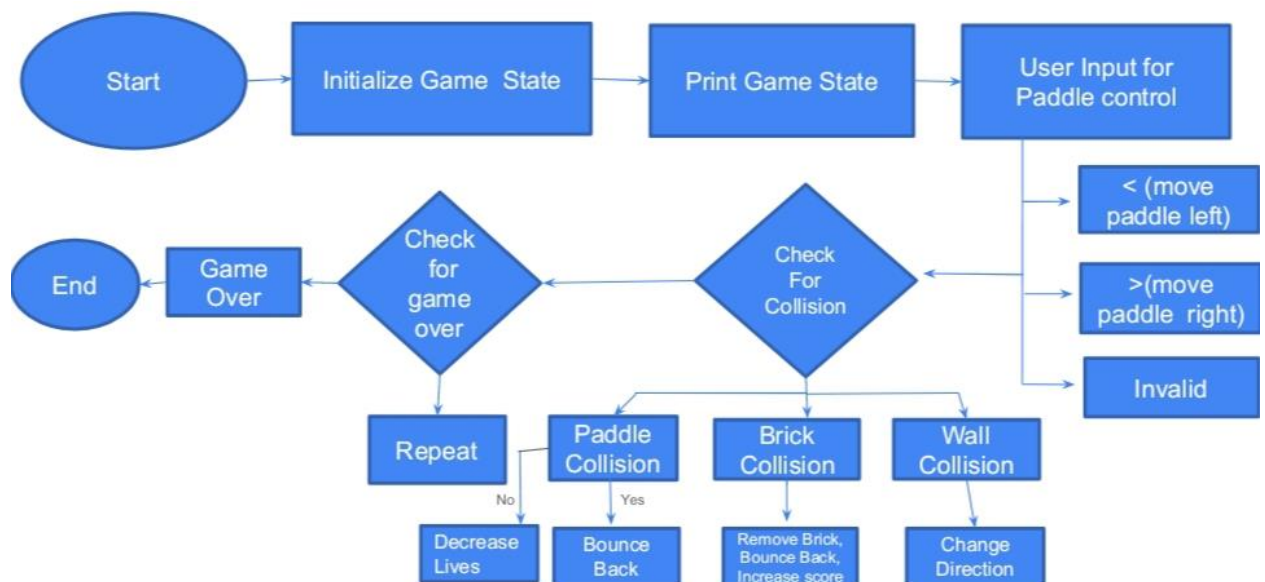
### PROJECT METHODOLOGY

#### 2.1 Proposed Work

The proposed work for the Brick Breaker Game game includes designing core game mechanics such as paddle movement, ball interaction with bricks, and defining the goal of clearing all bricks. Multiple levels will be created with increasing difficulty, and a scoring system will track the player's progress. Key game components include the ball, paddle, and different types of bricks (normal, indestructible, power-up).

The user interface will feature a start screen, HUD (score, lives, level), and game-over screen. Input handling will allow paddle control via keyboard or mouse, and game events like ball collisions will be managed throughout the game play.

#### 2.2 Block Diagram



## **CHAPTER 3**

### **MODULE DESCRIPTION**

#### **1.1 Game Initialization Module**

Sets up the game window, initializes objects (ball, paddle, bricks), and defines initial conditions like score and lives. It also sets up input handling for user controls.

Game Initialization Module is a Main Menu Screen. It allows players to select options like Start Game, View Instructions, Adjust Settings, or Exit, providing a structured and organized entry point to the game. This enhances user experience and adds a professional touch.

#### **1.2 Ball Module**

Ball Module makes the ball move and interact with the paddle and bricks. It ensures the ball bounces off walls and the paddle, and it detects brick collisions to update the game state.

The Ball Module could be ball speed variation, where the ball gradually speeds up over time or after breaking certain bricks. This increases the challenge and keeps the gameplay dynamic.

#### **1.3 Paddle Module**

Paddle Module gives the player control over the paddle and interacts with the ball to keep it in play. It also resizes the paddle during power-ups and ensures it remains on-screen. An extra point for the Paddle Module could be the addition of power-ups.

For example, when certain bricks are destroyed, they could drop power-ups that temporarily increase the paddle size, allow the paddle to shoot lasers, or make it move faster. This adds variety and excitement to the gameplay by giving the player new abilities.

#### **1.4 Brick Module**

Brick Module handles the creation and destruction of bricks, which are the obstacles the player must break to win. It also manages power-ups and special brick effects.

## **1.5 Game Logic and Control Module**

The Game Logic and Control Module is responsible for managing the overall flow of the game, including game progression, scoring, level transitions, and win/loss conditions. It tracks the player's score, which increases as bricks are broken, and manages the player's lives, decreasing each time the ball is missed. When a level is completed, it transitions to the next, often increasing the difficulty by speeding up the ball or changing the brick layout. The module ensures the game properly ends when the player runs out of lives or completes all levels, displaying appropriate messages for win or loss conditions. This module is crucial for maintaining the structure of the game, ensuring smooth gameplay and progression.

## **CHAPTER 4**

### **CONCLUSION & FUTURE SCOPE**

#### **4.1 CONCLUSION**

In conclusion, Brick Breaker is a simple yet highly addictive arcade game that combines easy-to-learn mechanics with progressively challenging gameplay. By using a paddle to bounce a ball and break bricks, players enjoy satisfying, skill-based action that keeps them engaged through increasing difficulty and power-ups. Its timeless appeal lies in its balance of simplicity, challenge, and quick play sessions, making it a favorite among casual gamers across various platforms. Brick Breaker fosters hand-eye coordination and reflexes, making it not only fun but also a light mental exercise. The game's ability to provide endless replay value, thanks to random brick patterns and power-up combinations, ensures that it remains enjoyable every time it's played.

#### **4.2 FUTURE SCOPE**

The future scope of brick breaker games holds significant potential for innovation as technology and gaming trends evolve. We can expect advancements in gameplay mechanics, with more sophisticated physics engines that allow for realistic ball behavior, like spin, velocity, and trajectory.

The rise of cloud gaming and cross-platform play could bring brick breaker games to a broader audience, making them more accessible and interconnected across devices. The future of brick breaker games will likely involve a blend of cutting-edge technology, deeper interactivity, and more engaging gameplay, ensuring the genre remains fresh and appealing to players of all ages.



## APPENDIX A

### (SOURCE CODE)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.awt.geom.Rectangle2D;

public class AdvancedBrickBreaker extends JPanel implements ActionListener {
    private Timer timer;
    private int ballX = 200, ballY = 200, ballDX = 2, ballDY = 3, ballSize = 20;
    private int paddleX = 300, paddleY = 550, paddleWidth = 120, paddleHeight = 10;
    private boolean[][] bricks;
    private int brickRows = 5, brickCols = 7, brickWidth = 100, brickHeight = 30, brickPadding = 10;
    private int lives = 3, score = 0, totalBricks;
    private boolean gameRunning = true;
    public AdvancedBrickBreaker() {
        bricks = new boolean[brickRows][brickCols];
        totalBricks = brickRows * brickCols;
        for (int i = 0; i < brickRows; i++) {
            for (int j = 0; j < brickCols; j++) {
                bricks[i][j] = true;
            }
        }
        timer = new Timer(10, this);
        timer.start();
        setFocusable(true);
        addKeyListener(new KeyAdapter() {
            public void keyPressed(KeyEvent e) {
                if (e.getKeyCode() == KeyEvent.VK_LEFT) {
                    paddleX -= 20;
                    if (paddleX < 0) paddleX = 0;
                }
            }
        });
    }
}
```

```

        } else if (e.getKeyCode() == KeyEvent.VK_RIGHT) {
            paddleX += 20;
            if(paddleX>getWidth()-paddleWidth)paddleX = getWidth() - paddleWidth;
        }
    }
});
}

protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    // Background
    g.setColor(Color.BLACK);
    g.fillRect(0, 0, getWidth(), getHeight());
    // Ball
    g.setColor(Color.RED);
    g.fillOval(ballX, ballY, ballSize, ballSize);
    // Paddle
    g.setColor(Color.BLUE);
    g.fillRect(paddleX, paddleY, paddleWidth, paddleHeight);
    // Bricks
    g.setColor(Color.GREEN);
    for (int i = 0; i < brickRows; i++) {
        for (int j = 0; j < brickCols; j++) {
            if (bricks[i][j]) {
                int brickX = j * (brickWidth + brickPadding) + 50;
                int brickY = i * (brickHeight + brickPadding) + 50;
                g.fillRect(brickX, brickY, brickWidth, brickHeight);
            }
        }
    }
    // Score and Lives
    g.setColor(Color.WHITE);
    g.setFont(new Font("Arial", Font.BOLD, 16));
    g.drawString("Score: " + score, 10, 20);
    g.drawString("Lives: " + lives, getWidth() - 80, 20);
    // Game Over Message

```

```

if (!gameRunning) {
    g.setFont(new Font("Arial", Font.BOLD, 36));
    g.setColor(Color.RED);
    g.drawString("GAME OVER", getWidth() / 2 - 100, getHeight() / 2);
    g.drawString("Score: " + score, getWidth() / 2 - 70, getHeight() / 2 + 40);
}
}

public void actionPerformed(ActionEvent e) {
    if (!gameRunning) {
        timer.stop();
        return;
    }
    // Ball Movement
    ballX += ballDX;
    ballY += ballDY;
    // Ball-Wall Collision
    if (ballX < 0 || ballX > getWidth() - ballSize) {
        ballDX = -ballDX;
    }
    if (ballY < 0) {
        ballDY = -ballDY;
    }
    // Ball-Paddle Collision
    if (new Rectangle2D.Double(ballX, ballY, ballSize, ballSize)
        .intersects(paddleX, paddleY, paddleWidth, paddleHeight)) {
        ballDY = -ballDY;
    }
    // Ball-Brick Collision
    for (int i = 0; i < brickRows; i++) {
        for (int j = 0; j < brickCols; j++) {
            if (bricks[i][j]) {
                int brickX = j * (brickWidth + brickPadding) + 50;
                int brickY = i * (brickHeight + brickPadding) + 50;
                Rectangle2D brickRect = new Rectangle2D.Double(brickX, brickY,
brickWidth, brickHeight);

```

```

        if (brickRect.intersects(ballX, ballY, ballSize, ballSize)) {
            bricks[i][j] = false;
            ballDY = -ballDY;
            score += 10;
            totalBricks--;
            if (totalBricks == 0) {
                gameRunning = false;
            }
            break;
        }
    }
}

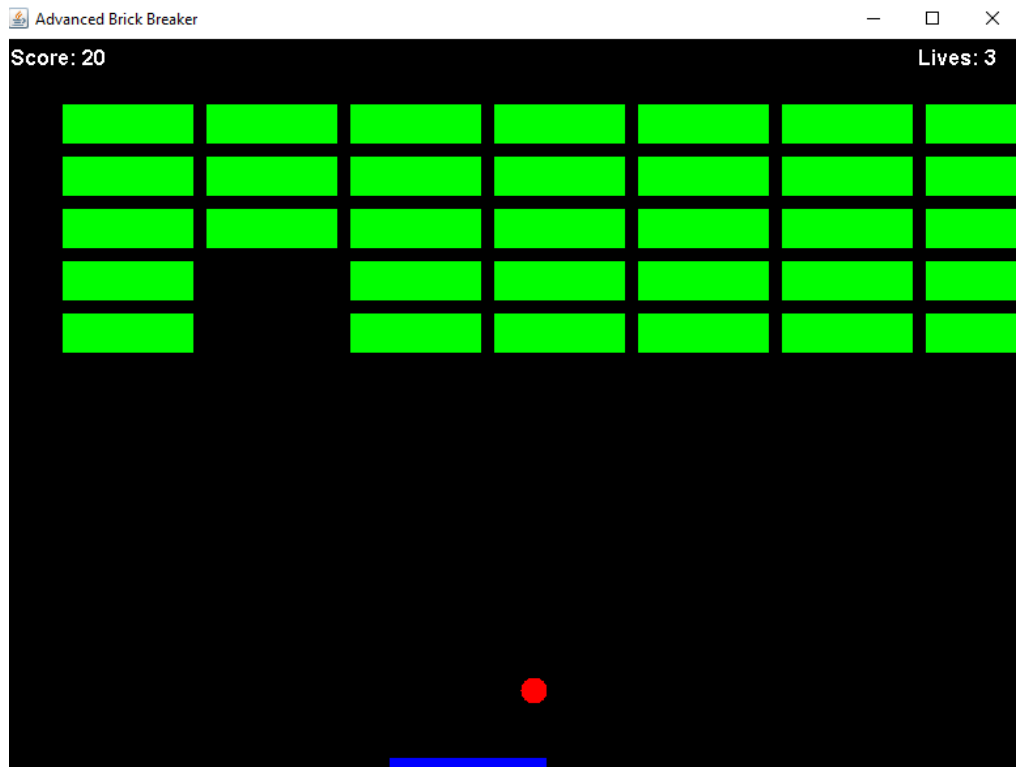
// Ball Misses Paddle
if (ballY > getHeight()) {
    lives--;
    if (lives == 0) {
        gameRunning = false;
    } else {
        resetBallAndPaddle(); } }repaint();}

private void resetBallAndPaddle() {
    ballX = getWidth() / 2;
    ballY = getHeight() / 2;
    ballDX = 2 + (int) (Math.random() * 3);
    ballDY = 3 + (int) (Math.random() * 3);
    paddleX = (getWidth() - paddleWidth) / 2;
}

public static void main(String[] args) {
    JFrame frame = new JFrame("Advanced Brick Breaker");
    AdvancedBrickBreaker game = new AdvancedBrickBreaker();
    frame.add(game);
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
}

```

## APPENDIX B (SCREENSHOTS)



## **REFERENCES**

### **BOOKS**

- Head First Java by Kathy Sierra and Bert Bates – Great for learning Java basics and object-oriented programming.
- Java: A Beginner's Guide by Herbert Schildt – Covers Java fundamentals, including GUI programming with Swing

### **WEBSITES**

- Oracle Java Documentation – Official documentation for Java programming.
- GeeksforGeeks – Articles and tutorials on Java Swing and game development.
- w3schools Java Tutorial – A beginner-friendly platform for Java basics.

### **YOUTUBE CHANNELS**

- CodeNCode: Provides tutorials on Java and game development.
- Bro Code: Includes Java Swing tutorials with real-world examples.
- ProgrammingKnowledge: Offers comprehensive Java programming tutorials.
- Apna College: Explains Java concepts in a simple and engaging way.
- Tech with Tim: Includes game programming content for various languages, including Java.