```
public class BrickBreaker {
        public class Main{
            package com.mycompany.brickbreaker;

import javax.swing.JFrame;

public class BrickBreaker {
    public static void main(String[] args) {
        JFrame obj = new JFrame();
        GamePlay gameplay = new GamePlay();
        obj.setBounds(10, 10, 700, 600);
        obj.setTitle("BrickBreaker");
        obj.setResizable(false);
        obj.setVisible(true);
        obj.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        obj.add(gameplay);
    }
}}

public class MapGenerator {
    public class MapGenerator {
        public int map[][];
        public int bricksWidth;
        public int bricksHeight;

        public MapGenerator(int row, int col) {
            map = new int[row][col];
            for (int[] map1 : map) {
                for (int j = 0; j < map[0].length; j++) {
                    map1[j] = 1;
                }
            }
            bricksWidth = 540 / col;
            bricksHeight = 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.red);
                        g.fillRect(j * bricksWidth + 80, i * bricksHeight + 50, bricksWidth,
bricksHeight);

                        g.setStroke(new BasicStroke(3));
                        g.setColor(Color.black);
                        g.drawRect(j * bricksWidth + 80, i * bricksHeight + 50, bricksWidth,
bricksHeight);

                    }
                }

            }
        }

        public void setBricksValue(int value, int row, int col) {
            map[row][col] = value;

        }

    }
}

public class GamePlay {
    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 = 8;
    private int playerX = 310;
    private int ballposX = 120;
    private int ballposY = 350;
    private int ballXdir = -1;
    private int ballYdir = -2;
    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) {
        g.setColor(Color.black);
        g.fillRect(1, 1, 692, 592);

        map.draw((Graphics2D) g);

        g.setColor(Color.yellow);
        g.fillRect(0, 0, 3, 592);
        g.fillRect(0, 0, 692, 3);
        g.fillRect(691, 0, 3, 592);

        g.setColor(Color.white);
        g.setFont(new Font("serif", Font.BOLD, 25));
        g.drawString("" + score, 590, 30);

        g.setColor(Color.yellow);
        g.fillRect(playerX, 550, 100, 8);

        // ball
        g.setColor(Color.GREEN);
        g.fillOval(ballposX, ballposY, 20, 20);

        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 Score: " + score, 190, 300);

            g.setFont(new Font("serif", Font.BOLD, 30));
            g.drawString("   Press Enter to Restart", 190, 340);
        }
        if (totalbricks == 0) {
            play = false;
            ballYdir = -2;
            ballXdir = -1;
            g.setColor(Color.red);
            g.setFont(new Font("serif", Font.BOLD, 30));
            g.drawString("    Game Over: " + score, 190, 300);

            g.setFont(new Font("serif", Font.BOLD, 30));
            g.drawString("   Press Enter to Restart", 190, 340);

        }

        g.dispose();

    }
```

```
        @Override
        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;
                }

                A: 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.bricksWidth + 80;
                            int brickY = i * map.bricksHeight + 50;
                            int bricksWidth = map.bricksWidth;
                            int bricksHeight = map.bricksHeight;

                            Rectangle rect = new Rectangle(brickX, brickY, bricksWidth,
bricksHeight);
                            Rectangle ballrect = new Rectangle(ballposX, ballposY, 20, 20);
                            Rectangle brickrect = rect;

                            if (ballrect.intersects(brickrect)) {
                                map.setBricksValue(0, i, j);
                                totalbricks--;
                                score += 5;
                                if (ballposX + 19 <= brickrect.x || ballposX + 1 >= brickrect.x +
bricksWidth) {
                                    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();
        }

        @Override
        public void keyTyped(KeyEvent e) {

        }

        @Override
        public void keyReleased(KeyEvent e) {

        }

        @Override
        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) {
                    ballposX = 120;
                    ballposY = 350;
                    ballXdir = -1;
                    ballYdir = -2;
                    score = 0;
                    playerX = 310;
                    totalbricks = 21;
                    map = new MapGenerator(3, 7);

                    repaint();
                }
            }

        }

        public void moveRight() {
            play = true;
            playerX += 20;
        }

        public void moveLeft() {
            play = true;
            playerX -= 20;
        }

    }
}

}
```