**Experiment 12:**

**Q 1: Write a program that bounces a blue ball inside a JPanel. The ball should begin moving with a mousePressed event. When the ball hits the edge of the JPanel, it should bounce off the edge and continue in the opposite direction. The ball should be updated using a Runnable.**

**Answer:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class BouncingBall {

    public static void main(String[] args) {

        JFrame frame = new JFrame("Bouncing Blue Ball");

        BallPanel panel = new BallPanel();

        frame.add(panel);

        frame.setSize(500, 400);

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        frame.setVisible(true);

    }

}

class BallPanel extends JPanel implements Runnable, MouseListener {

    private int x = 100, y = 100;

    private int dx = 3, dy = 3;

    private final int radius = 20;

    private boolean running = false;

    public BallPanel() {

        addMouseListener(this);

        setBackground(Color.WHITE);

    }

    @Override

    public void paintComponent(Graphics g) {

        super.paintComponent(g);

        g.setColor(Color.BLUE);

        g.fillOval(x, y, radius \* 2, radius \* 2);

    }

    @Override

    public void run() {

        while (running) {

            x += dx;

            y += dy;

            if (x <= 0 || x + radius \* 2 >= getWidth()) {

                dx = -dx;

            }

            if (y <= 0 || y + radius \* 2 >= getHeight()) {

                dy = -dy;

            }

            repaint();

            try {

                Thread.sleep(10);

            } catch (InterruptedException e) {

                e.printStackTrace();

            }

        }

    }

    public void mousePressed(MouseEvent e) {

        if (!running) {

            running = true;

            Thread t = new Thread(this);

            t.start();

        }

    }

    public void mouseClicked(MouseEvent e) {}

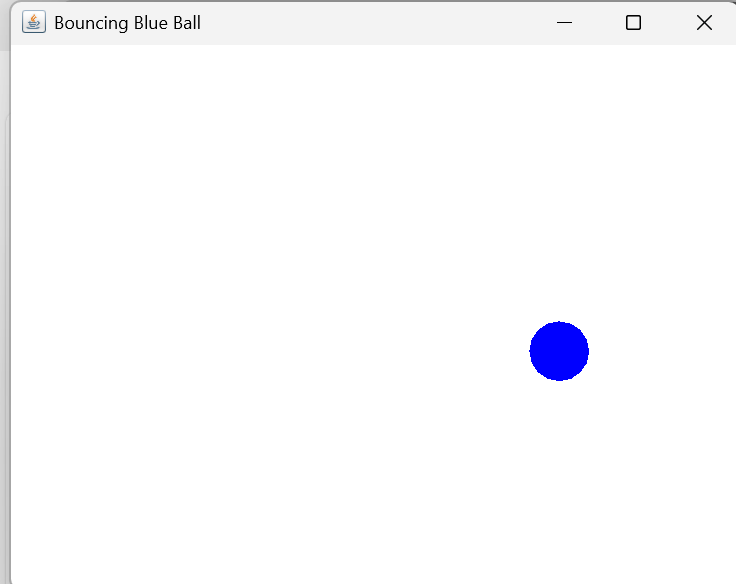
    public void mouseReleased(MouseEvent e) {}

    public void mouseEntered(MouseEvent e) {}

    public void mouseExited(MouseEvent e) {}

}

Output:



**Q 2: Create an application of Stopwatch using the concept of Multithreading.**

**Answer:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Stopwatch extends JFrame implements ActionListener {

    private JLabel timeLabel;

    private JButton startButton, stopButton, resetButton;

    private int hours = 0, minutes = 0, seconds = 0;

    private boolean running = false;

    private Thread stopwatchThread;

    public Stopwatch() {

        setTitle("Stopwatch");

        setSize(300, 150);

        setLayout(new FlowLayout());

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        timeLabel = new JLabel("00:00:00");

        timeLabel.setFont(new Font("Arial", Font.BOLD, 30));

        add(timeLabel);

        startButton = new JButton("Start");

        stopButton = new JButton("Stop");

        resetButton = new JButton("Reset");

        add(startButton);

        add(stopButton);

        add(resetButton);

        startButton.addActionListener(this);

        stopButton.addActionListener(this);

        resetButton.addActionListener(this);

        setVisible(true);

    }

    public void actionPerformed(ActionEvent e) {

        if (e.getSource() == startButton) {

            if (!running) {

                running = true;

                stopwatchThread = new Thread(() -> {

                    while (running) {

                        try {

                            Thread.sleep(1000);

                        } catch (InterruptedException ex) {

                            ex.printStackTrace();

                        }

                        seconds++;

                        if (seconds == 60) {

                            seconds = 0;

                            minutes++;

                        }

                        if (minutes == 60) {

                            minutes = 0;

                            hours++;

                        }

                        SwingUtilities.invokeLater(() -> {

                            timeLabel.setText(formatTime(hours, minutes, seconds));

                        });

                    }

                });

                stopwatchThread.start();

            }

        } else if (e.getSource() == stopButton) {

            running = false;

        } else if (e.getSource() == resetButton) {

            running = false;

            hours = 0;

            minutes = 0;

            seconds = 0;

            timeLabel.setText("00:00:00");

        }

    }

    private String formatTime(int h, int m, int s) {

        return String.format("%02d:%02d:%02d", h, m, s);

    }

    public static void main(String[] args) {

        new Stopwatch();

    }

}

**Output:**

