Topic: Multi-Threading with GUI

Name of Student: Kartik Banshi Katkar

Batch: B2 **Branch: SY-IT** **Roll No: 29**

Q. Code a system using Multi-Threading in Java to solve the producer consumer problem where there are two producer threads and one consumer threads. Store the produce in arraylist which is being produced.

Code:

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.LinkedList;
import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.SwingUtilities;
import javax.swing.table.DefaultTableModel;

public class ProducerConsumerGUIExample extends JFrame {
    private static final long serialVersionUID = 1L;
    private JPanel mainPanel;
    private JTable bufferTable;
    private DefaultTableModel tableModel;
    private JLabel statusLabel;
    private JButton startButton;
    private JButton stopButton;
    private Producer producer1;
    private Producer producer2;
```

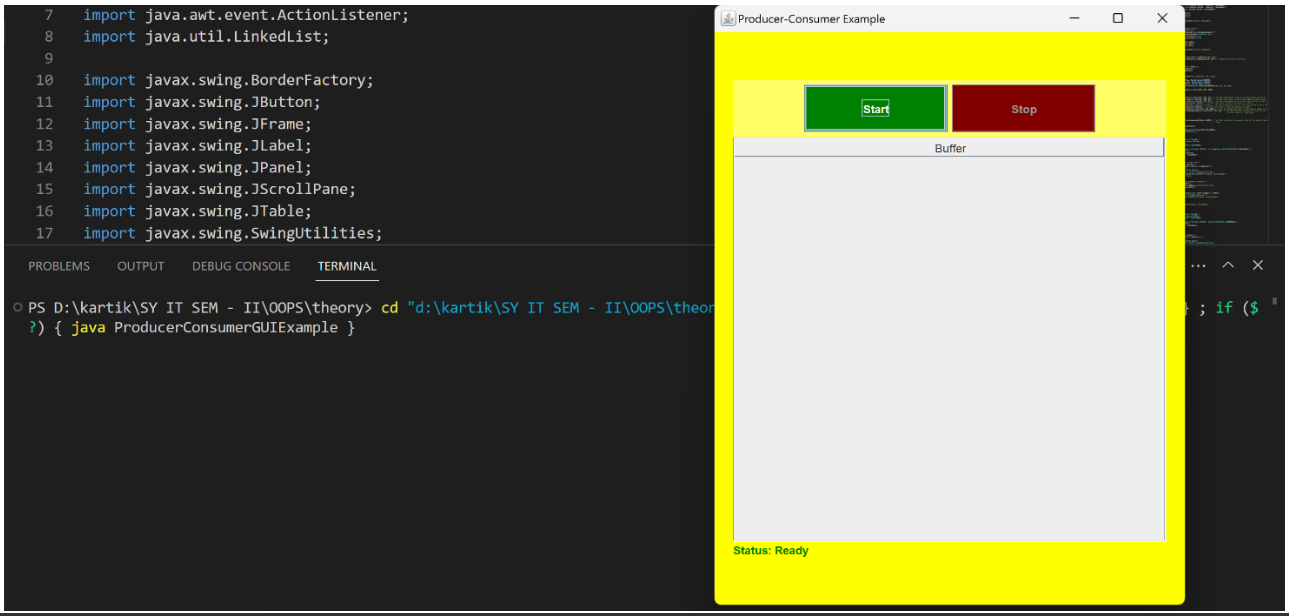
```java
private Consumer consumer;
private LinkedList buffer;
private int capacity = 5;
public ProducerConsumerGUIExample() {
  super("Producer-Consumer Example");

  buffer = new LinkedList<>();
  tableModel = new DefaultTableModel(new Object[][] {}, new String[] { "Buffer" }) {
    private static final long serialVersionUID = 1L;

    @Override
    public boolean isCellEditable(int row, int column) {
      return false;
    }
  };
  mainPanel = new JPanel(new BorderLayout());

  bufferTable = new JTable(tableModel);
  bufferTable.setRowSelectionAllowed(false);
  bufferTable.setColumnSelectionAllowed(false);
  bufferTable.getTableHeader().setReorderingAllowed(false);
  bufferTable.getColumnModel().getColumn(0).setResizable(false);
  JScrollPane scrollPane = new JScrollPane(bufferTable);

  statusLabel = new JLabel("Status: Ready");

  startButton = new JButton("Start");
  startButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      startButton.setEnabled(false);
      stopButton.setEnabled(true);

      producer1 = new Producer(buffer, capacity, tableModel);
      producer2 = new Producer(buffer, capacity, tableModel);
      consumer = new Consumer(buffer, tableModel);
```

```
        producer1.start();
        producer2.start();
        consumer.start();

        statusLabel.setText("Status: Running");
    }
});

stopButton = new JButton("Stop");
stopButton.setEnabled(false);
stopButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        startButton.setEnabled(true);
        stopButton.setEnabled(false);

        producer1.interrupt();
        producer2.interrupt();
        consumer.interrupt();

        statusLabel.setText("Status: Stopped");
    }
});
startButton.setPreferredSize(new Dimension(150, 50));
stopButton.setPreferredSize(new Dimension(150, 50));// changing the size of buttons
JPanel buttonPanel = new JPanel();
buttonPanel.add(startButton);
buttonPanel.add(stopButton);
buttonPanel.setBackground(new Color(255, 255, 102));
mainPanel.add(scrollPane, BorderLayout.CENTER);
mainPanel.add(statusLabel, BorderLayout.SOUTH);
mainPanel.add(buttonPanel, BorderLayout.NORTH);
mainPanel.setBorder(BorderFactory.createEmptyBorder(50, 20, 50, 20));


mainPanel.setBackground(new Color(255, 255, 102));
mainPanel.setBackground(new Color(255, 255, 0)); // set the background color of the main panel to Yellow
statusLabel.setForeground(new Color(0, 128, 0)); // set the text color of the status label to dark green
```

```
    startButton.setBackground(new Color(0, 128, 0)); // set the background color of the start button to dark green
    startButton.setForeground(Color.WHITE); // set the text color of the start button to white
    stopButton.setBackground(new Color(128, 0, 0)); // set the background color of the stop button to dark red
    stopButton.setForeground(Color.WHITE); // set the text color of the stop button to white
    bufferTable.setBackground(new Color(173, 216, 230)); // set the background color of the buffer table to light blue
    bufferTable.setSelectionBackground(new Color(225, 225, 0)); // set the selection background color of the
                                                                // buffer table to light gray
    bufferTable.setSelectionForeground(Color.BLACK); // set the selection foreground color of the buffer table to
                                                     // black
    this.setContentPane(mainPanel);
    this.pack();
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setLocationRelativeTo(null);
}
// The Producer class
static class Producer extends Thread {
    private LinkedList buffer;
    private int capacity;
    private DefaultTableModel tableModel;

    public Producer(LinkedList buffer, int capacity, DefaultTableModel tableModel) {
        this.buffer = buffer;
        this.capacity = capacity;
        this.tableModel = tableModel;
    }
    public void run() {
        for (int i = 1; i <= 10; i++) {
            synchronized (buffer) {
                while (buffer.size() == capacity) {
                    try {
                        buffer.wait();
                    } catch (InterruptedException e) {
                        System.out.println("Producer interrupted");
                        return;
                    }
                }
            System.out.println("Produce");
            buffer.add(i);
            tableModel.addRow(new Object[] { i });
            buffer.notifyAll();
          }
          try {
            Thread.sleep((long) (Math.random() * 2000));
          } catch (InterruptedException e) {
            System.out.println("Producer interrupted");
            return;
          }
        }
        System.out.println("Producer finished");
      }
    }

    // The Consumer class
    static class Consumer extends Thread {
      private LinkedList buffer;
      private DefaultTableModel tableModel;

      public Consumer(LinkedList buffer, DefaultTableModel tableModel) {
        this.buffer = buffer;
        this.tableModel = tableModel;
      }

      public void run() {
        while (true) {
          synchronized (buffer) {
            while (buffer.isEmpty()) {
              try {
                buffer.wait();
              } catch (InterruptedException e) {
                System.out.println("Consumer interrupted");
                return;
              }
```

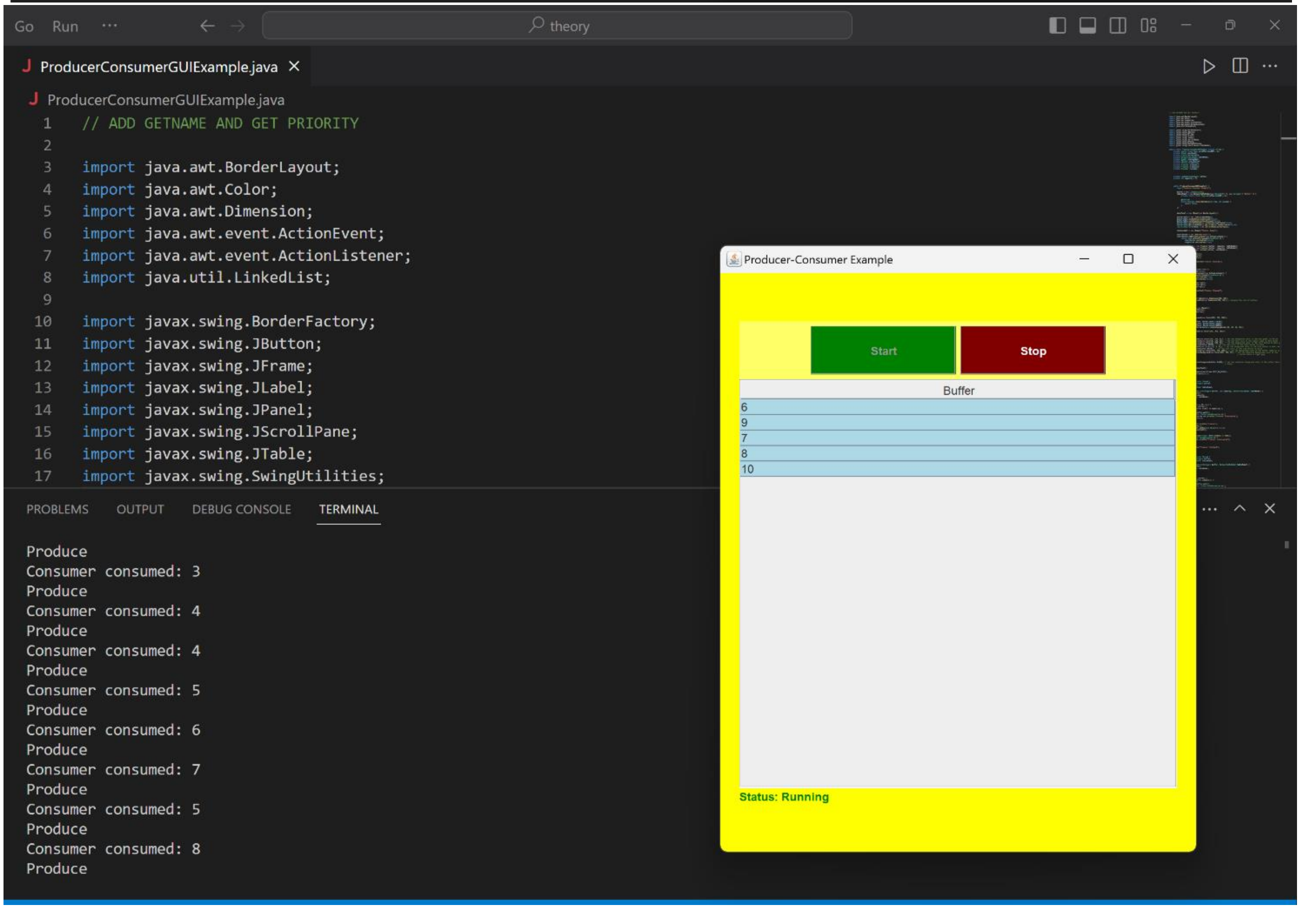
```
                }
                int value = buffer.removeFirst();
                tableModel.removeRow(0);
                buffer.notifyAll();
                System.out.println("Consumer consumed: " + value);
            }
            try {
                Thread.sleep((long) (Math.random() * 2000));
            } catch (InterruptedException e) {
                System.out.println("Consumer interrupted");
                return;
            }
        }
    }
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            ProducerConsumerGUIExample ex = new ProducerConsumerGUIExample();
            ex.setVisible(true);
        }
    });
}
}
```

Screenshots/Results:



```
import java.awt.event.ActionListener;
import java.util.LinkedList;

import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.SwingUtilities;
```

```
PS D:\kartik\SY IT SEM - II\OOPS\theory> cd "d:\kartik\SY IT SEM - II\OOPS\theor ; if ($
?) { java ProducerConsumerGUIExample }
```



```
// ADD GETNAME AND GET PRIORITY

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.LinkedList;

import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.SwingUtilities;
```

```
Produce
Consumer consumed: 3
Produce
Consumer consumed: 4
Produce
Consumer consumed: 4
Produce
Consumer consumed: 5
Produce
Consumer consumed: 6
Produce
Consumer consumed: 7
Produce
Consumer consumed: 5
Produce
Consumer consumed: 8
Produce
```

Buffer
6
9
7
8
10