```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

/************************************************************
 * GUI for a MarketPlace Simulator
 *
 * @author Scott Grissom, Silas Agnew
 * @version November 21, 2017
 ************************************************************/
public class MarketGUI extends JFrame implements ActionListener{

    /** MarketPlace */
    MarketPlace store;

    /** Buttons */
    JButton simulateBtn;

    /** Text Fields */
    JTextField cashierField;
    JTextField avgArrivalField;
    JTextField avgServiceField;

    /** Checkbox */
    JCheckBox displayCheck;

    /** Results text area */
    JTextArea resultsArea;

    /** Menu items */
    JMenuBar  menus;
    JMenu     fileMenu;
    JMenuItem quitItem;
    JMenuItem clearItem;

    // Parameters
    int cashiers;
    double avgServiceTime;
    double avgArrivalTime;
    boolean displayCheckout;

    /****************************************************************
     * Main Method
     ****************************************************************/
    public static void main(String args[]){
        MarketGUI gui = new MarketGUI();
        gui.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        gui.setTitle("MarketPlace Simulator");
        gui.pack();
        gui.setVisible(true);
    }

    /****************************************************************
     * constructor installs all of the GUI components
     ****************************************************************/
```

```
 57     public MarketGUI(){
 58         // Database
 59         store = new MarketPlace();
 60 
 61         // set the layout to GridBag
 62         setLayout(new GridBagLayout());
 63         GridBagConstraints loc = new GridBagConstraints();
 64 
 65         // create results area to span one column and 10 rows
 66         resultsArea = new JTextArea(20,20);
 67         JScrollPane scrollPane = new JScrollPane(resultsArea);
 68         loc.gridx = 0;
 69         loc.gridy = 1;
 70         loc.gridheight = 10;
 71         loc.insets.left = 20;
 72         loc.insets.right = 20;
 73         loc.insets.bottom = 20;
 74         add(scrollPane, loc);
 75 
 76         // create Results label
 77         loc = new GridBagConstraints();
 78         loc.gridx = 0;
 79         loc.gridy = 0;
 80         loc.insets.bottom = 20;
 81         loc.insets.top = 20;
 82         add(new JLabel("Results"), loc);
 83 
 84         // create Parameters label
 85         loc = new GridBagConstraints();
 86         loc.gridx = 1;
 87         loc.gridy = 0;
 88         loc.gridwidth = 2;
 89         add(new JLabel("Parameters"), loc);
 90 
 91         // create Cashier label
 92         loc = new GridBagConstraints();
 93         loc.gridx = 1;
 94         loc.gridy = 1;
 95         add(new JLabel("Cashier"), loc);
 96 
 97         // create Avg Arrival time label
 98         loc = new GridBagConstraints();
 99         loc.gridx = 1;
100         loc.gridy = 2;
101         loc.insets.top = 5;
102         add(new JLabel("Avg Arrival Time"), loc);
103 
104         // create Avg Service time label
105         loc = new GridBagConstraints();
106         loc.gridx = 1;
107         loc.gridy = 3;
108         loc.insets.top = 5;
109         add(new JLabel("Avg Service Time"), loc);
110 
111         // create display checkbox
112         displayCheck = new JCheckBox("Display");
```

```
113         loc = new GridBagConstraints();
114         loc.gridx = 2;
115         loc.gridy = 4;
116         add(displayCheck, loc);
117 
118         // create simulate sort button
119         simulateBtn = new JButton("Simulate");
120         loc = new GridBagConstraints();
121         loc.gridx = 2;
122         loc.gridy = 5;
123         loc.anchor = GridBagConstraints.WEST;
124         add(simulateBtn, loc);
125 
126         // create Cashier text field
127         cashierField = new JTextField(5);
128         loc = new GridBagConstraints();
129         loc.gridx = 2;
130         loc.gridy = 1;
131         loc.anchor = GridBagConstraints.WEST;
132         loc.insets.left = 5;
133         add(cashierField, loc);
134 
135         // create Average Arrival time text field
136         avgArrivalField = new JTextField(10);
137         loc = new GridBagConstraints();
138         loc.gridx = 2;
139         loc.gridy = 2;
140         loc.anchor = GridBagConstraints.WEST;
141         loc.insets.top = 5;
142         loc.insets.left = 5;
143         add(avgArrivalField, loc);
144 
145         // create Average Service time text field
146         avgServiceField = new JTextField(10);
147         loc = new GridBagConstraints();
148         loc.gridx = 2;
149         loc.gridy = 3;
150         loc.anchor = GridBagConstraints.WEST;
151         loc.insets.top = 5;
152         loc.insets.left = 5;
153         add(avgServiceField, loc);
154 
155         simulateBtn.addActionListener(this);
156 
157         // hide details of creating menus
158         setupMenus();
159     }
160 
161     /*****************************************************************
162      * This method is called when any button is clicked.  The proper
163      * internal method is called as needed.
164      *
165      * @param e the event that was fired
166      *****************************************************************/
167     public void actionPerformed(ActionEvent e){
168 
```

```
169         // extract the button that was clicked
170         JComponent buttonPressed = (JComponent) e.getSource();
171 
172         // Allow user to load baby names from a file
173         if (buttonPressed == clearItem)
174         {
175             resultsArea.setText("");
176         }
177         else if (buttonPressed == quitItem)
178         {
179             System.exit(0);
180         }
181         else if (buttonPressed == simulateBtn)
182         {
183             if (parseParameters())
184             {
185                 resultsArea.setText("");
186                 store.setParameters(cashiers, avgServiceTime,
187                         avgArrivalTime, displayCheckout);
188                 store.startSimulation();
189                 resultsArea.append(store.getReport());
190             } else return;
191         }
192     }
193 
194     /*******************************************************
195     Creates the menu items
196      *******************************************************/
197     private void setupMenus(){
198         fileMenu = new JMenu("File");
199         quitItem = new JMenuItem("Quit");
200         clearItem = new JMenuItem("Clear");
201         fileMenu.add(clearItem);
202         fileMenu.add(quitItem);
203         menus = new JMenuBar();
204         setJMenuBar(menus);
205         menus.add(fileMenu);
206 
207         clearItem.addActionListener(this);
208         quitItem.addActionListener(this);
209     }
210 
211     //-Private Helper Methods-----------------------------------//
212 
213     /**
214      * Parses simulation parameters from the associated text fields.
215      * @return If all parameters were successfully parsed
216      */
217     private boolean parseParameters()
218     {
219         cashiers = 0;
220         avgServiceTime = 0;
221         avgArrivalTime = 0;
222         try {
223             cashiers = Integer.parseInt(cashierField.getText());
224             if (cashiers < 1) throw new IllegalArgumentException();
```

```
225         } catch (Exception ex)
226         {
227             JOptionPane.showMessageDialog(this,
228                     "Text in the cashier field is ill-formatted.");
229             return false;
230         }
231         try {
232             avgArrivalTime = Double.parseDouble(avgArrivalField.getText());
233             if (avgArrivalTime < .001) throw new IllegalArgumentException();
234         } catch (Exception ex)
235         {
236             JOptionPane.showMessageDialog(this,
237                     "Text in the Avg Arrival Time field is ill-formatted.");
238             return false;
239         }
240         try {
241             avgServiceTime = Double.parseDouble(avgServiceField.getText());
242             if (avgServiceTime < .001) throw new IllegalArgumentException();
243         } catch (Exception ex)
244         {
245             JOptionPane.showMessageDialog(this,
246                     "Text in the Avg Service Time field is ill-formatted.");
247             return false;
248         }
249 
250         displayCheckout = displayCheck.isSelected();
251         return true;
252     }
253 }
```