```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.util.*;


/**
 * Observer35 is the main file and handles the UI for the application
 *
 * @author Akhila Diddi
 * @version 1.0
 * @since 2019-10-25, number of hours = 5  hours - 1 1/2hr meeting 3 1/2 hour
coding
 */
public class ObserverTable35 implements Observer {

    private JPanel panel;
    private JTable jt;
    private DefaultTableModel model;

    ObserverTable35() {
        panel = new JPanel ( );
        DefaultTableModel model1 = new DefaultTableModel ( );
        model1.addColumn ( "ID" );
        model1.addColumn ( "FIRST NAME" );
        model1.addColumn ( "LAST NAME" );
        model1.addColumn ( "PROGRAM AND PLAN" );
        model1.addColumn ( "ASURITE" );
        model1.addColumn ( "ATTENDANCE" );
        model1.addColumn ( "DATE LAST UPDATED" );
        jt = new JTable ( model1 );


        jt.setBounds ( 20, 20, 1400, 300 );
        JScrollPane sp = new JScrollPane ( jt );
        sp.setBounds ( 20, 20, 1400, 300 );
        panel.setLayout ( null );
        panel.add ( sp );
        model = (DefaultTableModel) jt.getModel ( );

    }

    public JPanel getPanel() {
        return this.panel;
    }

    @Override
    public void update(Observable o, Object arg) {


        Map<String, StudentDecorator32> copy = new HashMap<> ( );
        copy.putAll ( (Map<String, StudentDecorator32>) arg );


        Iterator it = copy.entrySet ( ).iterator ( );


        while (it.hasNext ( )) {
            String[] temp = new String[7];
            Map.Entry pair = (Map.Entry) it.next ( );
            temp[4] = ((CoreStudent32) pair.getValue ( )).getAsurite ( );
            temp[1] = ((CoreStudent32) pair.getValue ( )).getFirstName ( );
            temp[2] = ((CoreStudent32) pair.getValue ( )).getLastName ( );
            temp[3] = ((CoreStudent32) pair.getValue ( )).getProgram ( );

            temp[0] = ((CoreStudent32) pair.getValue ( )).getId ( );
            try {
                boolean found = false;
                int rows = model.getRowCount ( );

                HashMap<String, Integer> map = (HashMap<String, Integer>)
((AttendanceStudent32) ((StudentDecorator32) pair.getValue ( )).getStudent
( )).getAttendance ( );
                for (Map.Entry<String, Integer> entry : map.entrySet ( )) {
                    temp[5] = entry.getValue ( ).toString ( );
                    temp[6] = entry.getKey ( );
                }


                for (int i = 0; i < rows; i++) {
                    if (model.getValueAt ( i, 4 ).toString ( ).equals ( temp[4] ))
{
                        model.setValueAt ( temp[0], i, 0 );
                        model.setValueAt ( temp[1], i, 1 );
                        model.setValueAt ( temp[2], i, 2 );
                        model.setValueAt ( temp[3], i, 3 );
                        model.setValueAt ( temp[5], i, 5 );
                        model.setValueAt ( temp[6], i, 6 );
                        found = true;
                    }

                }
                if (found == false || rows == 0) {
                    model.addRow ( new Object[]{temp[0], temp[1], temp[2], temp[3],
temp[4], "N/A", "N/A"} );
                }

                it.remove ( );

            } catch (Exception e) {
                boolean found = false;
                int rows = model.getRowCount ( );
                int cols = model.getColumnCount ( );


                for (int i = 0; i < rows; i++) {
                    if (model.getValueAt ( i, 4 ).toString ( ).equals ( temp[4] ))
{
                        model.setValueAt ( temp[0], i, 0 );
                        model.setValueAt ( temp[1], i, 1 );
                        model.setValueAt ( temp[2], i, 2 );
                        model.setValueAt ( temp[3], i, 3 );
                        model.setValueAt ( temp[5], i, 5 );
                        model.setValueAt ( temp[6], i, 6 );
                        found = true;
                    }
                }
                if (found == false || rows == 0) {
                    model.addRow ( new Object[]{temp[0], temp[1], temp[2], temp[3],
temp[4], "N/A", "N/A"} );
                }


            }


        }

    }
}
```