


```

import java.nio.file.*;
import java.io.*;
import java.util.Arrays;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileNameExtensionFilter;

public class UI_Design extends javax.swing.JFrame {

    /**
     * Creates new form UI_Design
     */
    public UI_Design() {
        initComponents();
    }

    class clsGlobals{
        /**
         * Variables (respectively):
         * - A Multi-Dimensional Array to eventually contain the data records
         * - An Integer to store how long the records
         *   file is based on the number of lines it contains
         * - A Path datatype to eventually store the path to the records file
         * - A Boolean to record if there is any unsaved data that has been
         *   added to the record
         */
        private static String[][] masRecords;
        private static int intFileLineCount = 0;
        private static Path pthDataFile;
        private static boolean blnIsSaved = true;

        /**
         * A method used to set 'intFileLineCount' based on the parsed file
         * path. Returns an Integer
         */
        public static int mthGetFileLength(Path pthFile) {
            intFileLineCount = 0;
            try {
                InputStream ismFileInput = new
BufferedInputStream(Files.newInputStream(pthFile));
                BufferedReader bfrFileReader = new BufferedReader(new
InputStreamReader(ismFileInput));

                while (bfrFileReader.readLine() != null) intFileLineCount++;

                ismFileInput.close();
                bfrFileReader.close();
            }
            catch (IOException ioeExcept) {
                System.err.println("IO Exception in
'clsGlobals.getFileLength()': " + ioeExcept);
            }
            return intFileLineCount;
        }

        /**
         * A method to create and return a new array based on the parsed
         * length. Returns a Multi-Dimensional Array with a size of
         * ("parsed length" x "3")
         */
        public static String[][] mthCreateNewArray(int intLength){
            masRecords = new String[intLength][3];
        }
    }
}

```

```

        return masRecords;
    }
}

```

SKIP

```

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */

@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code"> //GEN-BEGIN: initComponents
private void initComponents() {

    lblHeading = new javax.swing.JLabel();
    cmbRecordIndex = new javax.swing.JComboBox<>();
    txtName = new javax.swing.JTextField();
    txtClass = new javax.swing.JTextField();
    txtYear = new javax.swing.JTextField();
    btnLoad = new javax.swing.JButton();
    btnAddRecord = new javax.swing.JButton();
    btnSave = new javax.swing.JButton();
    btnExit = new javax.swing.JButton();
    lblRecordNumber = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_ON_CLOSE);
    addWindowListener(new java.awt.event.WindowAdapter() {
        public void windowClosing(java.awt.event.WindowEvent evt) {
            lsnWindowClosing(evt);
        }
    });

    lblHeading.setText("Heading: Name, Class, Year");

    cmbRecordIndex.setModel(new javax.swing.DefaultComboBoxModel<>(new
String[] { "Empty" }));
    cmbRecordIndex.setToolTipText("Record Number");
    cmbRecordIndex.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            cmbRecordIndexActionPerformed(evt);
        }
    });

    txtName.setEditable(false);
    txtName.setText("Empty");
    txtName.setToolTipText("Name");

    txtClass.setEditable(false);
    txtClass.setText("Empty");
    txtClass.setToolTipText("Class");

    txtYear.setEditable(false);
    txtYear.setText("Empty");
    txtYear.setToolTipText("Year");

    btnLoad.setText("Load");
    btnLoad.setToolTipText("Load Data");
    btnLoad.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            btnLoadActionPerformed(evt);

```

```
}  
});  
  
btnAddRecord.setText("Add Record");  
btnAddRecord.setToolTipText("Add Record");  
btnAddRecord.setEnabled(false);  
btnAddRecord.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
        btnAddRecordActionPerformed(evt);  
    }  
});  
  
btnSave.setText("Save");  
btnSave.setToolTipText("Save Records");  
btnSave.setEnabled(false);  
btnSave.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
        btnSaveActionPerformed(evt);  
    }  
});  
  
btnExit.setText("Exit");  
btnExit.setToolTipText("Exit");  
btnExit.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
        btnExitActionPerformed(evt);  
    }  
});  
  
lblRecordNumber.setText("Record Number");  
  
javax.swing.GroupLayout layout = new  
javax.swing.GroupLayout(getContentPane());  
getContentPane().setLayout(layout);  
layout.setHorizontalGroup(  
  
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
    .addGroup(layout.createSequentialGroup()  
        .addGap(10, 10, 10)  
        .addComponent(btnLoad)  
        .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)  
        .addComponent(lblHeading)  
        .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)  
        .addComponent(btnAddRecord)  
        .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)  
        .addComponent(btnSave)  
        .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)  
        .addComponent(btnExit))  
    .addContainerGap())  
    .addGroup(layout.createSequentialGroup()  
        .addGap(49, 49, 49)  
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```

        .addComponent(lblRecordNumber)
        .addGap(0, 0, Short.MAX_VALUE))
    .addGroup(layout.createSequentialGroup())
        .addComponent(cmbRecordIndex,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    .addGroup(layout.createSequentialGroup())
        .addComponent(txtName,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(40, 40, 40)
        .addComponent(txtClass,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacem
ent.RELATED, 53, Short.MAX_VALUE)
        .addComponent(txtYear,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(45, 45, 45)))
    );
    layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup())
        .addGap(18, 18, 18)
        .addComponent(lblHeading)
        .addGap(24, 24, 24)
        .addComponent(lblRecordNumber)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRE
LATED)
        .addComponent(cmbRecordIndex,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(33, 33, 33)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.BASELINE)
            .addComponent(txtYear,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(txtClass,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(txtName,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELA
TED, 73, Short.MAX_VALUE)
            .addComponent(btnLoad)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRE
LATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.BASELINE)
                .addComponent(btnAddRecord)
                .addComponent(btnExit)
                .addComponent(btnSave))
            .addContainerGap())
        );
    pack();
} // </editor-fold> // GEN-END: initComponents

```

END SKIP

```
private void btnLoadActionPerformed(java.awt.event.ActionEvent evt)
{
    //GEN-FIRST:event_btnLoadActionPerformed
    /**
     * - Create a file browser to allow the user to select the records file
     * - Store the path of the file a file is selected and cancel the
     *   operation if not
     */
    JFileChooser jfcFileBrowser = new
    JFileChooser(System.getProperty("user.dir"));
    jfcFileBrowser.setSelectedFile(new File("records.txt"));
    jfcFileBrowser.addChoosableFileFilter(new FileNameExtensionFilter("Text
Documents", "txt"));
    jfcFileBrowser.setAcceptAllFileFilterUsed(false);
    int intResponse = jfcFileBrowser.showOpenDialog(null);
    Path pthRecordFilePath = null;

    if (intResponse == JFileChooser.APPROVE_OPTION) {
        pthRecordFilePath =
        Paths.get(jfcFileBrowser.getSelectedFile().getAbsolutePath());
        clsGlobals.pthDataFile = pthRecordFilePath;
    } else {
        return;
    }

    String strCurrentLine;
    String strDelimiter = ", ";

    /**
     * (With error handling):
     * - Load the selected file and create a new array based on the length
     *   of the loaded file
     * - Loop through the line of the file to set the new array to the data
     *   within the file
     * - Close the file and clear the record index combo box before adding
     *   indexes to it and setting the global array to the locally created
     *   array of loaded data
     * - Catch any IO Exceptions and enable the add record button
     */
    try {
        BufferedInputStream bisFileInput = new
        BufferedInputStream(Files.newInputStream(pthRecordFilePath));
        BufferedReader bfrFileReader = new BufferedReader(new
        InputStreamReader(bisFileInput));

        String[][] masLoadedRecords =
        clsGlobals.mthCreateNewArray(clsGlobals.mthGetFileLength(pthRecordFilePath));
        int intRecordIndex = 0;

        System.out.println();
        strCurrentLine = bfrFileReader.readLine();

        while (strCurrentLine != null) {
            masLoadedRecords[intRecordIndex] =
            strCurrentLine.split(strDelimiter);
            strCurrentLine = bfrFileReader.readLine();
            intRecordIndex += 1;
        }
        bisFileInput.close();
        bfrFileReader.close();
    }
}
```

```

        cmbRecordIndex.removeAllItems();

        for (int i = 0; i < masLoadedRecords.length; i++) {
            cmbRecordIndex.addItem(Integer.toString(i + 1));
            clsGlobals.masRecords[i] = masLoadedRecords[i];
            // DEBUG:
            System.out.println(Arrays.toString(masLoadedRecords[i]));
        }
    }
    catch (IOException ioeExcept) {
        System.err.println("IO Exception in 'btnLoadActionPerformed': " +
ioeExcept);
    }

    btnAddRecord.setEnabled(true);
} // GEN-LAST:event_btnLoadActionPerformed

private void btnAddRecordActionPerformed(java.awt.event.ActionEvent evt)
{ // GEN-FIRST:event_btnAddRecordActionPerformed
    /**
     * - Prompt user for new data and load into separate variables
     * - Load old and new data into arrays and combine them into a new,
     *   larger array
     * - Update global array and set the saved state to false and enable the
     *   save button
     */
    String strNewName = (String)JOptionPane.showInputDialog(null, "Name:");
    if ((strNewName == null)) return;

    String strNewClass = (String)JOptionPane.showInputDialog(null,
"Class:");
    if ((strNewClass == null)) return;

    String strNewYear = (String)JOptionPane.showInputDialog(null, "Year:");
    if ((strNewYear == null)) return;

    String[][] masExistingRecords = clsGlobals.masRecords;

    String[][] masNewRecord = {{strNewName, strNewClass, strNewYear}};

    String[][] masUpdatedRecords =
clsGlobals.mthCreateNewArray(clsGlobals.intFileLineCount + 1);

    for (int i = 0; i < masUpdatedRecords.length; i++) {
        if (i < masUpdatedRecords.length - 1) {
            masUpdatedRecords[i] = masExistingRecords[i];
        }
        else if (i == masUpdatedRecords.length - 1) {
            masUpdatedRecords[i] = masNewRecord[0];
        }
    }

    clsGlobals.masRecords =
clsGlobals.mthCreateNewArray(clsGlobals.intFileLineCount + 1);
    clsGlobals.masRecords = masUpdatedRecords;

    System.out.println();

    cmbRecordIndex.removeAllItems();

    for (int i = 0; i < clsGlobals.masRecords.length; i++) {
        cmbRecordIndex.addItem(Integer.toString(i + 1));
        // DEBUG:

```

```

System.out.println(Arrays.toString(clsGlobals.masRecords[i]));
    }

    clsGlobals.blnIsSaved = false;
    btnSave.setEnabled(true);
} //GEN-LAST:event_btnAddRecordActionPerformed

private void btnExitActionPerformed(java.awt.event.ActionEvent evt)
{ //GEN-FIRST:event_btnExitActionPerformed
    /**
     * - Warn the user of any unsaved record when the exit button is pressed
     */
    if (clsGlobals.blnIsSaved == false) {
        int intConfirmExit = (int) JOptionPane.showConfirmDialog(null,
"Unsaved records!\nDo you wish to proceed?", "Unsaved Records",
JOptionPane.YES_NO_OPTION, JOptionPane.ERROR_MESSAGE);
        if (intConfirmExit == 0) System.exit(0);
    } else {
        System.exit(0);
    }
} //GEN-LAST:event_btnExitActionPerformed

private void cmbRecordIndexActionPerformed(java.awt.event.ActionEvent
evt) { //GEN-FIRST:event_cmbRecordIndexActionPerformed
    /**
     * - If the combo box has a selection, set the text fields to the
     * appropriate data based on the combo box selection
     */
    if (cmbRecordIndex.getSelectedItem() != null) {

txtName.setText(clsGlobals.masRecords[cmbRecordIndex.getSelectedIndex()][0]);

txtClass.setText(clsGlobals.masRecords[cmbRecordIndex.getSelectedIndex()][1]);

txtYear.setText(clsGlobals.masRecords[cmbRecordIndex.getSelectedIndex()][2]);
    }
} //GEN-LAST:event_cmbRecordIndexActionPerformed

private void btnSaveActionPerformed(java.awt.event.ActionEvent evt)
{ //GEN-FIRST:event_btnSaveActionPerformed
    /**
     * - Loop through the global array and write the data to new lines in
the original records file
     * - Notify the user of success and print any errors
     * - Set the saved state to true and disable the save button
     */
    try {
        FileWriter fwrFileWriter = new
FileWriter(clsGlobals.pthDataFile.toString());
        String strData;

        for (int i = 0; i < clsGlobals.masRecords.length; i++) {
            strData = Arrays.toString(clsGlobals.masRecords[i]);
            strData = strData.substring(1, strData.length() - 1);

            fwrFileWriter.write(strData);
            fwrFileWriter.write("\n");
        }
        fwrFileWriter.close();

        JOptionPane.showMessageDialog(null, "File Saved Successfully");
    }
    catch (IOException ioeExcept) {

```



```

        System.err.println("IO Exception in 'btnSaveActionPerformed: " +
ioeExcept);
    }

    clsGlobals.blnIsSaved = true;
    btnSave.setEnabled(false);
} //GEN-LAST:event_btnSaveActionPerformed

private void lsnWindowClosing(java.awt.event.WindowEvent evt) { //GEN-
FIRST:event_lsnWindowClosing
/**
 * - Warn the user of any unsaved record when the "X" in the title bar
 *   is pressed.
 */
    if (clsGlobals.blnIsSaved == false) {
        int intConfirmExit = (int) JOptionPane.showConfirmDialog(null,
"Unsaved records!\nDo you wish to proceed?", "Unsaved Records",
JOptionPane.YES_NO_OPTION, JOptionPane.ERROR_MESSAGE);
        if (intConfirmExit == 0) System.exit(0);
    } else {
        System.exit(0);
    }
} //GEN-LAST:event_lsnWindowClosing

```

SKIP

```

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting
code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the
default look and feel.
    * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(UI_Design.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(UI_Design.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(UI_Design.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(UI_Design.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    }
}
//</editor-fold>

```

```

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new UI_Design().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton btnAddRecord;
private javax.swing.JButton btnExit;
private javax.swing.JButton btnLoad;
private javax.swing.JButton btnSave;
private javax.swing.JComboBox<String> cmbRecordIndex;
private javax.swing.JLabel lblHeading;
private javax.swing.JLabel lblRecordNumber;
private javax.swing.JTextField txtClass;
public javax.swing.JTextField txtName;
private javax.swing.JTextField txtYear;
// End of variables declaration//GEN-END:variables
END SKIP
}

```

Heading: Name, Class, Year lblHeading

Record Number lblRecordNumber

Empty ▼ cmbRecordIndex

Empty txtName Empty txtClass Empty txtYear

Load btnLoad btnSave btnExit

Add Record btnAddRecord Save Exit

records.txt (at top of project folder):

```

Frank, Maths, 10
Shelby, English, 7
Bob, Chemistry, 11
Dylan, Biology, 9
Logan, Science, 5

```