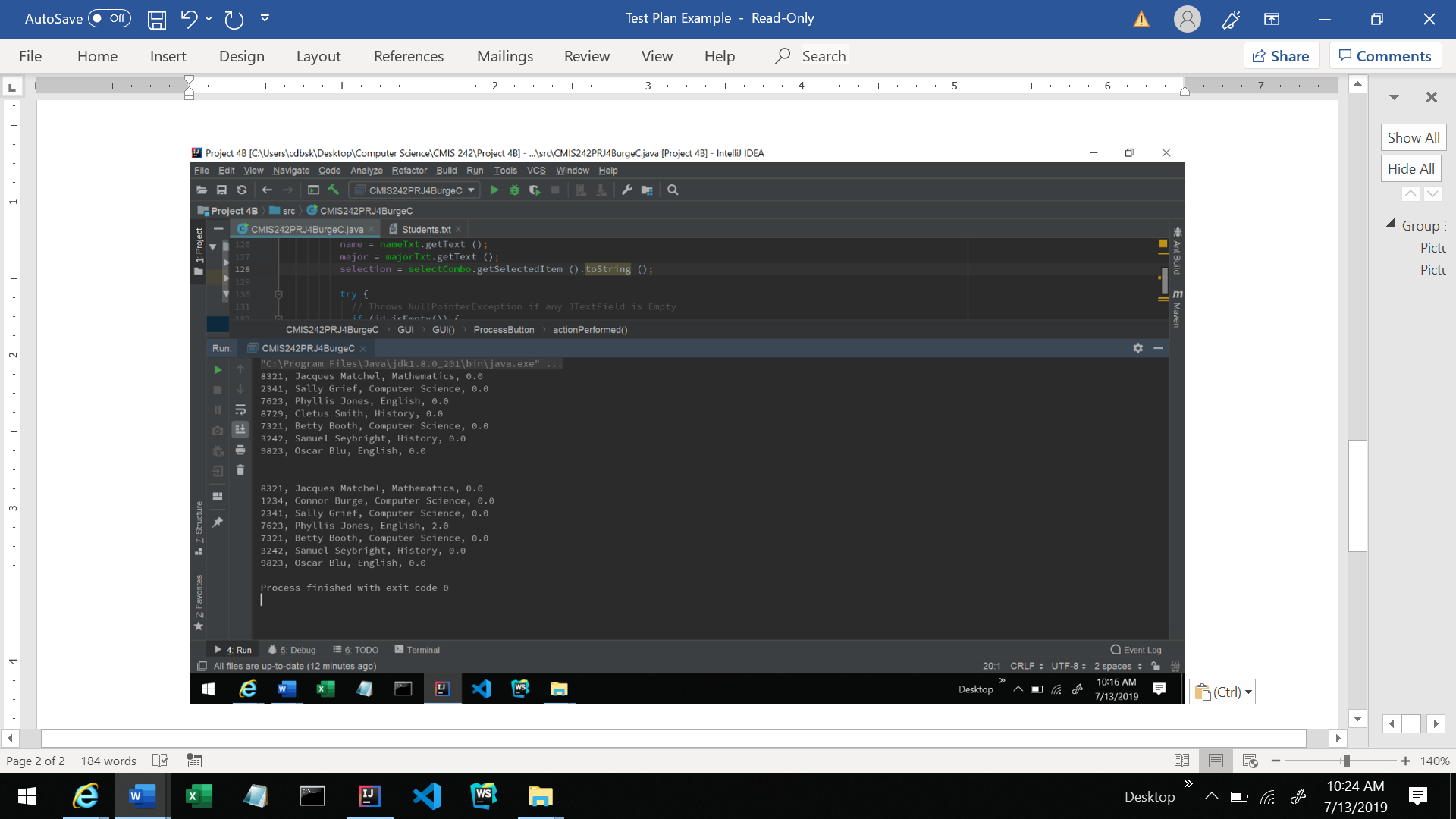
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Name** | **Major** | **Insert/Delete/**  **Find/Update** | **Choose Grade** | **Choose Credits** | **Display** |
| ***Test Case #1: Updating credits*** | | | | | | | |
| 7623 |  |  | Find |  |  | Name: Phyllis Jones  Major: English  GPA: 4.0 |
| 7623 |  |  | Update | C | 6 | Student record was updated |
| 7623 |  |  | Find |  |  | Name: Phyllis Jones  Major: English  GPA: 2.0 |
| ***Test Case #2: Deleting record*** | | | | | | | |
| 8729 |  |  | Find |  |  | Name: Cletus Smith  Major: History  GPA: 4.0 |
| 8729 |  |  | Delete |  |  | Student was removed from the Database |
| 8729 |  |  | Find |  |  | ID does not exist in the database |
| ***Test Case #3: Inserting record*** | | | | | | | |
| 1234 |  |  | Find |  |  | ID does not exist in the database |
| 1234 | Connor Burge | Computer Science | Insert |  |  | Student added to the database |
| 1234 |  |  | Find |  |  | Name: Connor Burge  Major: Computer Science  GPA: 4.0 |
| 1234 | Connor Burge | Computer Science | Insert |  |  | ID already exist in the database |
| ***Test Case #4: Trying to manipulate a record that is not in the database*** | | | | | | | |
| 1239 |  |  | Delete |  |  | ID does not exist in the database |
| 1239 |  |  | Update | B | 3 | ID does not exist in the database |
| 1239 |  |  | Find |  |  | ID does not exist in the database |
| ***Test Case #5: No student data included*** | | | | | | | |
| 0000 |  |  | Insert |  |  | Student added to database |
| 0000 |  |  | Find |  |  | Name:  Major:  GPA: 4.0 |

**Screenshots of program:**



**A screenshot of a social media post

Description automatically generatedA screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**CMIS242PRJ4BurgeC.java**

import javax.swing.\*;  
import java.awt.GridBagConstraints;  
import java.awt.GridBagLayout;  
import java.awt.Insets;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.text.DecimalFormat;  
import java.util.HashMap;  
import java.util.Map;  
  
public class CMIS242PRJ4BurgeC {  
  
 // Creates the HashMap to hold the students information  
 static HashMap<Integer, Student> *students* = new HashMap<>();  
  
 public static class GUI extends JFrame {  
 // Variables  
 private String id;  
 private String name;  
 private String major;  
 private String selection;  
  
 */\*\*  
 \* Constructs GUI for the program and handles ActionListeners  
 \*/* private GUI() {  
  
 String[] gradeStr = {"A", "B", "C", "D", "F"};  
 String[] creditStr = {"3", "6"};  
  
 setLayout (new GridBagLayout ());  
 GridBagConstraints window = new GridBagConstraints ();  
 window.insets = new Insets (5, 5, 5, 5);  
  
 JLabel idLabel = new JLabel ("Id:");  
 idLabel.setHorizontalAlignment (JLabel.*RIGHT*);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 0;  
 window.gridy = 0;  
 window.gridwidth = 1;  
 add (idLabel, window);  
  
 JTextField idTxt = new JTextField (10);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 1;  
 window.gridy = 0;  
 window.gridwidth = 3;  
 add (idTxt, window);  
  
 JLabel nameLabel = new JLabel ("Name:");  
 nameLabel.setHorizontalAlignment (JLabel.*RIGHT*);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 0;  
 window.gridy = 2;  
 window.gridwidth = 1;  
 add (nameLabel, window);  
  
 JTextField nameTxt = new JTextField (null);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 1;  
 window.gridy = 2;  
 window.gridwidth = 3;  
 add (nameTxt, window);  
  
 JLabel majorLabel = new JLabel ("Major:");  
 majorLabel.setHorizontalAlignment (JLabel.*RIGHT*);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 0;  
 window.gridy = 3;  
 window.gridwidth = 1;  
 add (majorLabel, window);  
  
 JTextField majorTxt = new JTextField (null);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 1;  
 window.gridy = 3;  
 window.gridwidth = 3;  
 add (majorTxt, window);  
  
 JLabel selectLabel = new JLabel ("Choose Selection:");  
 selectLabel.setHorizontalAlignment (JLabel.*RIGHT*);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 0;  
 window.gridy = 4;  
 window.gridwidth = 1;  
 add (selectLabel, window);  
  
 String[] selectStr = {"Insert", "Delete", "Find", "Update"};  
 JComboBox<String> selectCombo = new JComboBox<> (selectStr);  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 1;  
 window.gridy = 4;  
 window.gridwidth = 3;  
 add (selectCombo, window);  
  
 JButton processBtn = new JButton ("Process Request");  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 0;  
 window.gridy = 5;  
 window.gridwidth = 6;  
 add (processBtn, window);  
  
 JButton exitBtn = new JButton ("Exit");  
 window.fill = GridBagConstraints.*HORIZONTAL*;  
 window.gridx = 0;  
 window.gridy = 6;  
 window.gridwidth = 6;  
 add (exitBtn, window);  
  
 *students*.put (7623, new Student ("Phyllis Jones","English",0));  
 *students*.put (8729, new Student ("Cletus Smith","History",0));  
 *students*.put (7321, new Student ("Betty Booth","Computer Science",0));  
 *students*.put (3242, new Student ("Samuel Seybright","History",0));  
 *students*.put (9823, new Student ("Oscar Blu","English",0));  
 *students*.put (2341, new Student ("Sally Grief","Computer Science",0));  
 *students*.put (8321, new Student ("Jacques Matchel","Mathematics",0));  
  
 displayHashmap();  
  
 //Action listener for the Deposit button  
 class ProcessButton implements ActionListener {  
  
 public void actionPerformed(ActionEvent e) {  
 // Assigns JTextFields to variables  
 id = idTxt.getText ();  
 name = nameTxt.getText ();  
 major = majorTxt.getText ();  
 selection = selectCombo.getSelectedItem ().toString ();  
  
 try {  
 // Throws NullPointerException if any JTextField is Empty  
 if (id.isEmpty()) {  
 throw new NullPointerException();  
 } else {  
 // ComboBox Selection Switch (Determines action performed on studentDb)  
 switch (selection) {  
  
 // Insert Function  
 case "Insert":  
 // Shows error if key exists  
 if (*students*.containsKey(Integer.*valueOf* (id))) {  
 JOptionPane.*showMessageDialog*(null, "ID already exists in database", "Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 } else {  
 // Adds to studentDb, Shows Message  
 *students*.put(Integer.*valueOf* (id), new Student(name, major, 0));  
 JOptionPane.*showMessageDialog*(null, "Student added to Database", "Success",  
 JOptionPane.*INFORMATION\_MESSAGE*);  
 }  
 break;  
  
 // Delete Function  
 case "Delete":  
 // Shows error if key does not exist  
 if (!*students*.containsKey(Integer.*valueOf* (id))) {  
 JOptionPane.*showMessageDialog*(null, "ID does not exist in database", "Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 } else {  
 // Deletes from studentDb, Shows Message  
 *students*.remove(Integer.*valueOf* (id));  
 JOptionPane.*showMessageDialog*(null, "Student removed from Database", "Success",  
 JOptionPane.*INFORMATION\_MESSAGE*);  
 }  
 break;  
  
 // Find Function  
 case "Find":  
 if (!*students*.containsKey(Integer.*valueOf* (id))) {  
 JOptionPane.*showMessageDialog*(null, "ID does not exist in database", "Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 } else {  
 *students*.get(Integer.*valueOf* (id));  
 String toString = *students*.get(Integer.*valueOf* (id)).toString();  
 JOptionPane.*showMessageDialog*(null, "Student found in Database\n" + toString, "Success",  
 JOptionPane.*INFORMATION\_MESSAGE*);  
 }  
 break;  
  
 // Update Function  
 case "Update":  
 if (*students*.containsKey(Integer.*valueOf* (id))) {  
 String grade = (String) JOptionPane.*showInputDialog*(null, "Choose grade:", "",  
 JOptionPane.*QUESTION\_MESSAGE*, null, gradeStr, gradeStr[0]);  
 if (grade != null) {  
 String creditHours = (String) JOptionPane.*showInputDialog*(null, "Choose credits:", "",  
 JOptionPane.*QUESTION\_MESSAGE*, null, creditStr, creditStr[0]);  
 if (creditHours != null) {  
 *students*.get(Integer.*valueOf* (id)).courseCompleted(grade, Integer.*parseInt*(creditHours));  
 JOptionPane.*showMessageDialog*(null, "Student record was updated", "Success",  
 JOptionPane.*INFORMATION\_MESSAGE*);  
 } else {  
 JOptionPane.*showMessageDialog*(null, "Credits were not entered", "Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(null, "Grade was not entered", "Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(null, "ID does not exists in database", "Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 }  
 break;  
 }  
 }  
 } catch (NullPointerException e1) {  
 JOptionPane.*showMessageDialog*(null, "The ID text field is required", "Error",  
 JOptionPane.*ERROR\_MESSAGE*);  
 }  
 } // End of actionPerformed()  
 } // End of ProcessButton  
 processBtn.addActionListener (new ProcessButton ());  
  
 //Action listener for the Deposit button  
 class ExitButton implements ActionListener {  
  
 public void actionPerformed(ActionEvent e) {  
 System.*out*.println (" ");  
 displayHashmap ();  
 System.*exit* (0);  
 }  
 }  
 exitBtn.addActionListener (new ExitButton ());  
 }  
  
 public void displayHashmap(){  
 for(Map.Entry<Integer, Student> entry : *students*.entrySet()) {  
 System.*out*.println (entry.getKey () + ", " + entry.getValue ().name + ", " + entry.getValue ().major + ", " + entry.getValue().gradePoints);  
 }  
 }  
 } // End of GUI  
  
 public static class Student {  
  
 // Variables  
 private String name;  
 private String major;  
 private double credits;  
 private double qualityPoints;  
 private double gradePoints;  
 private double gpa = 4.0;  
 private DecimalFormat df = new DecimalFormat("#0.00");  
  
 Student(String name, String major, int gradePoints) {  
 this.name = name;  
 this.major = major;  
 this.gradePoints = gradePoints;  
 credits = 0;  
 qualityPoints = 0;  
 }  
  
 void courseCompleted(String grade, int creditHours) {  
 // Calculates points based on grade  
 switch (grade) {  
 case "A":  
 gradePoints = 4;  
 break;  
 case "B":  
 gradePoints = 3;  
 break;  
 case "C":  
 gradePoints = 2;  
 break;  
 case "D":  
 gradePoints = 1;  
 break;  
 case "F":  
 gradePoints = 0;  
 break;  
 }  
 // Calculate gradePoints total  
 gradePoints = gradePoints\*creditHours;  
  
 // Calculate qualityPoints and Credits  
 qualityPoints += gradePoints;  
 credits += creditHours;  
  
 // Calculate GPA  
 gpa = qualityPoints/credits;  
 gradePoints = gpa;  
 }  
  
 public String toString() {  
 return "\nName: \t" + name + "\nMajor: \t" + major + "\nGPA: \t" + df.format(gpa);  
 }  
 } // End of Student  
  
  
 public static void main(String[] args) {  
  
 GUI frame = new GUI();  
 frame.setDefaultCloseOperation (JFrame.*EXIT\_ON\_CLOSE*);  
 frame.setSize (400, 275);  
 frame.setTitle ("Students");  
 frame.setLocationRelativeTo (null);  
 frame.setVisible (true);  
 } // End of main()  
}// End of CMIS242PRJ4BurgeC