1. Write a program to present an address book app that enables you to browse existing entries, add newentries and search for entries with a specific last name using prepared statements.



**Person.java**

// Fig. 24.30: Person.java  
// Person class that represents an entry in an address book.  
public class Person {  
 private int addressID;  
 private String firstName;  
 private String lastName;  
 private String email;  
 private String phoneNumber;  
  
 // constructor  
 public Person(int addressID, String firstName, String lastName,  
 String email, String phoneNumber) {  
 setAddressID(addressID);  
 setFirstName(firstName);  
 setLastName(lastName);  
 setEmail(email);  
 setPhoneNumber(phoneNumber);  
 }  
  
 // sets the addressID  
 public void setAddressID(int addressID) {  
 this.addressID = addressID;  
 }  
  
 // returns the addressID  
 public int getAddressID() {  
 return addressID;  
 }  
  
 // sets the firstName  
 public void setFirstName(String firstName) {  
 this.firstName = firstName;  
 }  
  
 // returns the first name  
 public String getFirstName() {  
 return firstName;  
 }  
  
 // sets the lastName  
 public void setLastName(String lastName) {  
 this.lastName = lastName;  
 }  
  
 // returns the last name  
 public String getLastName() {  
 return lastName;  
 }  
  
 // sets the email address  
 public void setEmail(String email) {  
 this.email = email;  
 }  
  
 // returns the email address  
 public String getEmail() {  
 return email;  
 }  
  
 // sets the phone number  
 public void setPhoneNumber(String phone) {  
 this.phoneNumber = phone;  
 }  
  
 // returns the phone number  
 public String getPhoneNumber() {  
 return phoneNumber;  
 }  
}

**PersonQueries.java**

import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.List;  
import java.util.ArrayList;  
  
public class PersonQueries {  
 private static final String URL = "jdbc:mysql://localhost:3306/AddressBook";  
 private static final String USERNAME = "root";  
 private static final String PASSWORD = "";  
  
 private Connection connection; // manages connection  
 private PreparedStatement selectAllPeople;  
 private PreparedStatement selectPeopleByLastName;  
 private PreparedStatement insertNewPerson;  
  
 public PersonQueries() {  
 try {  
 connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);  
// create query that selects all entries in the AddressBook  
 selectAllPeople = connection.prepareStatement("SELECT \* FROM Addresses");  
// create query that selects entries with a specific last name  
 selectPeopleByLastName = connection.prepareStatement("SELECT \* FROM Addresses WHERE LastName = ?");  
// create insert that adds a new entry into the database  
 insertNewPerson = connection.prepareStatement("INSERT INTO Addresses (FirstName, LastName, Email, PhoneNumber) VALUES (?, ?, ?, ?)");  
 } catch (SQLException sqlException) {  
 sqlException.printStackTrace();  
 System.exit(1);  
 }  
 }  
  
 // select all the addresses in the database  
 public List<Person> getAllPeople() {  
 List<Person> results = null;  
 ResultSet resultSet = null;  
 try {  
// executeQuery returns ResultSet containing matching entries  
 resultSet = selectAllPeople.executeQuery();  
 results = new ArrayList<>();  
  
 while (resultSet.next()) {  
 results.add(new Person(  
 resultSet.getInt("addressID"),  
 resultSet.getString("FirstName"),  
 resultSet.getString("LastName"),  
 resultSet.getString("Email"),  
 resultSet.getString("PhoneNumber")));  
 }  
 } catch (SQLException sqlException) {  
 sqlException.printStackTrace();  
 } finally {  
 try {  
 assert resultSet != null;  
 resultSet.close();  
 } catch (SQLException sqlException) {  
 sqlException.printStackTrace();  
 close();  
 }  
 }  
  
 return results;  
 }  
  
 // select person by last name  
 public List<Person> getPeopleByLastName(String name) {  
 List<Person> results = null;  
 ResultSet resultSet = null;  
  
 try {  
 selectPeopleByLastName.setString(1, name); // specify last name  
// executeQuery returns ResultSet containing matching entries  
 resultSet = selectPeopleByLastName.executeQuery();  
 results = new ArrayList<>();  
  
 while (resultSet.next()) {  
 results.add(new Person(resultSet.getInt("addressID"),  
 resultSet.getString("FirstName"),  
 resultSet.getString("LastName"),  
 resultSet.getString("Email"),  
 resultSet.getString("PhoneNumber")));  
 }  
 } catch (SQLException sqlException) {  
 sqlException.printStackTrace();  
 } finally {  
 try {  
 assert resultSet != null;  
 resultSet.close();  
 } catch (SQLException sqlException) {  
 sqlException.printStackTrace();  
 close();  
 }  
 }  
  
 return results;  
 }  
  
 // add an entry  
 public int addPerson(  
 String fname, String lname, String email, String num) {  
 int result = 0;  
  
 // set parameters, then execute insertNewPerson  
 try {  
 insertNewPerson.setString(1, fname);  
 insertNewPerson.setString(2, lname);  
 insertNewPerson.setString(3, email);  
 insertNewPerson.setString(4, num);  
// insert the new entry; returns # of rows updated  
 result = insertNewPerson.executeUpdate();  
 } catch (SQLException sqlException) {  
 sqlException.printStackTrace();  
 close();  
 }  
  
 return result;  
 }  
  
 public void close() {  
 try {  
 connection.close();  
 } catch (SQLException sqlException) {  
 sqlException.printStackTrace();  
 }  
 }  
}

**app.java**

// Fig. 24.32: AddressBookDisplay.java  
// A simple address book  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.WindowAdapter;  
import java.awt.event.WindowEvent;  
import java.util.List;  
  
public class app extends JFrame {  
 private Person currentEntry;  
 private PersonQueries personQueries;  
 private List<Person> results;  
 private int numberOfEntries = 0;  
 private int currentEntryIndex;  
  
 private final JTextField emailTextField;  
 private final JTextField firstNameTextField;  
 private final JTextField idTextField;  
 private final JTextField indexTextField;  
 private final JTextField lastNameTextField;  
 private final JTextField maxTextField;  
 private final JButton nextButton;  
 private final JTextField phoneTextField;  
 private final JButton previousButton;  
 private final JTextField queryTextField;  
  
 public app() {  
 super("Address Book");  
  
 personQueries = new PersonQueries();  
  
 JPanel navigatePanel = new JPanel();  
 previousButton = new JButton();  
 indexTextField = new JTextField(2);  
 JLabel ofLabel = new JLabel();  
 maxTextField = new JTextField(2);  
 nextButton = new JButton();  
 JPanel displayPanel = new JPanel();  
 JLabel idLabel = new JLabel();  
 idTextField = new JTextField(10);  
 JLabel firstNameLabel = new JLabel();  
 firstNameTextField = new JTextField(10);  
 JLabel lastNameLabel = new JLabel();  
 lastNameTextField = new JTextField(10);  
 JLabel emailLabel = new JLabel();  
 emailTextField = new JTextField(10);  
 JLabel phoneLabel = new JLabel();  
 phoneTextField = new JTextField(10);  
 JPanel queryPanel = new JPanel();  
 JLabel queryLabel = new JLabel();  
 queryTextField = new JTextField(10);  
 JButton queryButton = new JButton();  
 JButton browseButton = new JButton();  
 JButton insertButton = new JButton();  
  
 setLayout(new FlowLayout(FlowLayout.CENTER, 10, 10));  
 setSize(400, 300);  
 setResizable(false);  
  
 navigatePanel.setLayout(new BoxLayout(navigatePanel, BoxLayout.X\_AXIS));  
  
 previousButton.setText("Previous");  
 previousButton.setEnabled(false);  
 previousButton.addActionListener(this::previousButtonActionPerformed);  
  
 navigatePanel.add(previousButton);  
 navigatePanel.add(Box.createHorizontalStrut(10));  
  
 indexTextField.setHorizontalAlignment(JTextField.CENTER);  
 indexTextField.addActionListener(this::indexTextFieldActionPerformed  
 );  
  
 navigatePanel.add(indexTextField);  
 navigatePanel.add(Box.createHorizontalStrut(10));  
  
 ofLabel.setText("of");  
 navigatePanel.add(ofLabel);  
 navigatePanel.add(Box.createHorizontalStrut(10));  
  
 maxTextField.setHorizontalAlignment(  
 JTextField.CENTER);  
 maxTextField.setEditable(false);  
 navigatePanel.add(maxTextField);  
 navigatePanel.add(Box.createHorizontalStrut(10));  
  
 nextButton.setText("Next");  
 nextButton.setEnabled(false);  
 nextButton.addActionListener(this::nextButtonActionPerformed);  
  
 navigatePanel.add(nextButton);  
 add(navigatePanel);  
  
 displayPanel.setLayout(new GridLayout(5, 2, 4, 4));  
  
 idLabel.setText("Address ID:");  
 displayPanel.add(idLabel);  
 idTextField.setEditable(false);  
 displayPanel.add(idTextField);  
  
 firstNameLabel.setText("First Name:");  
 displayPanel.add(firstNameLabel);  
 displayPanel.add(firstNameTextField);  
  
 lastNameLabel.setText("Last Name:");  
 displayPanel.add(lastNameLabel);  
 displayPanel.add(lastNameTextField);  
  
 emailLabel.setText("Email:");  
 displayPanel.add(emailLabel);  
 displayPanel.add(emailTextField);  
  
 phoneLabel.setText("Phone Number:");  
 displayPanel.add(phoneLabel);  
 displayPanel.add(phoneTextField);  
 add(displayPanel);  
  
 queryPanel.setLayout(new BoxLayout(queryPanel, BoxLayout.X\_AXIS));  
  
 queryPanel.setBorder(BorderFactory.createTitledBorder("Find an entry by last name"));  
 queryLabel.setText("Last Name:");  
 queryPanel.add(Box.createHorizontalStrut(5));  
 queryPanel.add(queryLabel);  
 queryPanel.add(Box.createHorizontalStrut(10));  
 queryPanel.add(queryTextField);  
 queryPanel.add(Box.createHorizontalStrut(10));  
  
 queryButton.setText("Find");  
 queryButton.addActionListener(this::queryButtonActionPerformed);  
  
 queryPanel.add(queryButton);  
 queryPanel.add(Box.createHorizontalStrut(5));  
 add(queryPanel);  
  
 browseButton.setText("Browse All Entries");  
 browseButton.addActionListener(this::browseButtonActionPerformed);  
  
 add(browseButton);  
  
 insertButton.setText("Insert New Entry");  
 insertButton.addActionListener(this::insertButtonActionPerformed);  
  
 add(insertButton);  
  
 addWindowListener(  
 new WindowAdapter() {  
 public void windowClosing(WindowEvent evt) {  
 personQueries.close(); // close database connection  
 System.exit(0);  
 }  
 }  
 );  
  
 setVisible(true);  
 } // end constructor  
  
 // handles call when previousButton is clicked  
 private void previousButtonActionPerformed(ActionEvent evt) {  
 currentEntryIndex--;  
  
 if (currentEntryIndex < 0)  
 currentEntryIndex = numberOfEntries - 1;  
  
 indexTextField.setText("" + (currentEntryIndex + 1));  
 indexTextFieldActionPerformed(evt);  
 }  
  
 // handles call when nextButton is clicked  
 private void nextButtonActionPerformed(ActionEvent evt) {  
 currentEntryIndex++;  
  
 if (currentEntryIndex >= numberOfEntries)  
 currentEntryIndex = 0;  
 indexTextField.setText("" + (currentEntryIndex + 1));  
 indexTextFieldActionPerformed(evt);  
 }  
  
 // handles call when queryButton is clicked  
 private void queryButtonActionPerformed(ActionEvent evt) {  
 results =personQueries.getPeopleByLastName(queryTextField.getText());  
 numberOfEntries = results.size();  
  
 if (numberOfEntries != 0) {  
 currentEntryIndex = 0;  
 currentEntry = results.get(currentEntryIndex);  
 idTextField.setText("" + currentEntry.getAddressID());  
 firstNameTextField.setText(currentEntry.getFirstName());  
 lastNameTextField.setText(currentEntry.getLastName());  
 emailTextField.setText(currentEntry.getEmail());  
 phoneTextField.setText(currentEntry.getPhoneNumber());  
 maxTextField.setText("" + numberOfEntries);  
 indexTextField.setText("" + (currentEntryIndex + 1));  
 nextButton.setEnabled(true);  
 previousButton.setEnabled(true);  
 } else  
 browseButtonActionPerformed(evt);  
 }  
  
 // handles call when a new value is entered in indexTextField  
 private void indexTextFieldActionPerformed(ActionEvent evt) {  
 currentEntryIndex = (Integer.parseInt(indexTextField.getText()) - 1);  
  
 if (numberOfEntries != 0 && currentEntryIndex < numberOfEntries) {  
 currentEntry = results.get(currentEntryIndex);  
 idTextField.setText("" + currentEntry.getAddressID());  
 firstNameTextField.setText(currentEntry.getFirstName());  
 lastNameTextField.setText(currentEntry.getLastName());  
 emailTextField.setText(currentEntry.getEmail());  
 phoneTextField.setText(currentEntry.getPhoneNumber());  
 maxTextField.setText("" + numberOfEntries);  
 indexTextField.setText("" + (currentEntryIndex + 1));  
 }  
 }  
  
 // handles call when browseButton is clicked  
 private void browseButtonActionPerformed(ActionEvent evt) {  
 try {  
 results = personQueries.getAllPeople();  
 numberOfEntries = results.size();  
  
 if (numberOfEntries != 0) {  
 currentEntryIndex = 0;  
 currentEntry = results.get(currentEntryIndex);  
 idTextField.setText("" + currentEntry.getAddressID());  
 firstNameTextField.setText(currentEntry.getFirstName());  
 lastNameTextField.setText(currentEntry.getLastName());  
 emailTextField.setText(currentEntry.getEmail());  
 phoneTextField.setText(currentEntry.getPhoneNumber());  
 maxTextField.setText("" + numberOfEntries);  
 indexTextField.setText("" + (currentEntryIndex + 1));  
 nextButton.setEnabled(true);  
 previousButton.setEnabled(true);  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 // handles call when insertButton is clicked  
 private void insertButtonActionPerformed(ActionEvent evt) {  
 int result = personQueries.addPerson(  
 firstNameTextField.getText(),  
 lastNameTextField.getText(),  
 emailTextField.getText(),  
 phoneTextField.getText()  
 );  
 if (result == 1)  
 JOptionPane.showMessageDialog(this, "Person added!",  
 "Person added", JOptionPane.PLAIN\_MESSAGE);  
 else  
 JOptionPane.showMessageDialog(this, "Person not added!",  
 "Error", JOptionPane.PLAIN\_MESSAGE);  
  
 browseButtonActionPerformed(evt);  
 }  
  
 // main method  
 public static void main(String[] args) {  
 new app();  
 }  
}