

**Faculty of Computing & Informatics Department of Software Engineering**

**DATA STRUCTURE AND ALGORITHMS ASSIGNMENT**

**Group Project Member’s Names**

1. Rott Elikkan 223124192
2. Marion Sande 220090025
3. Jaden Kilian 222071486
4. Gotfried Namandje 216003067
5. Landuleni Kanime 221008055
6. Jonas Mwakonga 219102090
7. Winston Tsabalala 213008726

**DSA PSEUDOCODE**

**Insert Contact**

FUNCTION InsertContact(name, phone)

IF name IS NOT EMPTY AND phone IS NOT EMPTY THEN

contactString = "Name: " + name + ", Phone: " + phone

Add contactString TO contactListModel

ELSE

DISPLAY "Please enter both name and phone number."

END IF

END FUNCTION

**Search Contact**

FUNCTION SearchContact(searchTerm)

results = EMPTY LIST

FOR EACH contact IN contactListModel

IF contact CONTAINS searchTerm THEN

ADD contact TO results

END IF

END FOR

IF results IS NOT EMPTY THEN

DISPLAY results

ELSE

DISPLAY "Contact not found."

END IF

END FUNCTION

**Display All Contacts**

FUNCTION DisplayAllContacts()

contacts = COPY OF contactListModel AS ARRAY

SORT contacts IN ALPHABETICAL ORDER

IF contacts IS NOT EMPTY THEN

FOR EACH contact IN contacts

DISPLAY contact

END FOR

ELSE

DISPLAY "No contacts to display."

END IF

END FUNCTION

**Contact Delete**

FUNCTION DeleteContact(selectedIndex)

IF selectedIndex IS NOT -1 THEN

CONFIRM deletion

IF user CONFIRMS THEN

REMOVE contact AT selectedIndex FROM contactListModel

DISPLAY "Contact deleted successfully."

ELSE

DISPLAY "Delete action cancelled."

END IF

ELSE

DISPLAY "Please select a contact to delete."

END IF

END FUNCTION

**Update Contact**

FUNCTION UpdateContact(selectedIndex)

IF selectedIndex IS NOT -1 THEN

currentContact = contactListModel[selectedIndex]

SPLIT currentContact INTO currentName AND currentPhone

newName = PROMPT "Edit Name:" WITH DEFAULT currentName

newPhone = PROMPT "Edit Phone:" WITH DEFAULT currentPhone

IF newName IS NOT EMPTY OR newPhone IS NOT EMPTY THEN

updatedName = IF newName IS EMPTY THEN currentName ELSE newName

updatedPhone = IF newPhone IS EMPTY THEN currentPhone ELSE newPhone

contactListModel[selectedIndex] = "Name: " + updatedName + ", Phone: " + updatedPhone

DISPLAY "Contact updated successfully."

END IF

ELSE

DISPLAY "Please select a contact to update."

END IF

END FUNCTION

**Sort Contacts (optional)**

FUNCTION SortContacts()

contacts = COPY OF contactListModel AS ARRAY

SORT contacts IN ALPHABETICAL ORDER

CLEAR contactListModel

FOR EACH contact IN contacts

ADD contact TO contactListModel

END FOR

END FUNCTION

**DSA Code**

import java.awt.\*;  
import java.awt.event.\*;  
import javax.swing.\*;  
import java.util.Arrays;  
  
public class Phonebook implements ActionListener {  
 JFrame frame;  
 JTextField nameField, phoneField, searchField;  
 JList<String> contactList;  
 DefaultListModel<String> contactListModel;  
 JButton addButton, viewButton, searchButton, deleteButton, updateButton; // Added updateButton  
 JPanel inputPanel, outputPanel;  
  
 public Phonebook() {  
 // Create JFrame  
 frame = new JFrame("Contact Book");  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
  
 // Create text fields and buttons  
 nameField = new JTextField();  
 phoneField = new JTextField();  
 searchField = new JTextField();  
  
 addButton = new JButton("Add Contact");  
 viewButton = new JButton("View Contacts");  
 searchButton = new JButton("Search");  
 deleteButton = new JButton("Delete Contact");  
 updateButton = new JButton("Update Contact"); // Initialize update button  
  
 // Create panels  
 inputPanel = new JPanel(new GridLayout(3, 2));  
 outputPanel = new JPanel(new BorderLayout());  
  
 // Initialize the contact list  
 contactListModel = new DefaultListModel<>();  
 contactList = new JList<>(contactListModel);  
 contactList.setSelectionMode(ListSelectionModel.*SINGLE\_SELECTION*);  
  
 // Initialize JScrollPane for the contact list  
 JScrollPane scrollPane = new JScrollPane(contactList);  
  
 // Add components to input panel  
 inputPanel.add(new JLabel("Name:"));  
 inputPanel.add(nameField);  
 inputPanel.add(new JLabel("Phone:"));  
 inputPanel.add(phoneField);  
 inputPanel.add(new JLabel("Search:"));  
 inputPanel.add(searchField);  
  
 // Add action listeners  
 addButton.addActionListener(this);  
 viewButton.addActionListener(this);  
 searchButton.addActionListener(this);  
 deleteButton.addActionListener(this);  
 updateButton.addActionListener(this); // Add action listener for update button  
  
 // Add components to output panel  
 outputPanel.add(scrollPane, BorderLayout.*CENTER*);  
 outputPanel.add(viewButton, BorderLayout.*SOUTH*);  
  
 // Add panels to frame  
 frame.add(inputPanel, BorderLayout.*NORTH*);  
 frame.add(outputPanel, BorderLayout.*CENTER*);  
  
 JPanel buttonPanel = new JPanel(new FlowLayout());  
 buttonPanel.add(addButton);  
 buttonPanel.add(searchButton);  
 buttonPanel.add(deleteButton); // Add the delete button  
 buttonPanel.add(updateButton); // Add the update button  
 frame.add(buttonPanel, BorderLayout.*SOUTH*);  
  
 // Frame settings  
 frame.setSize(600,  
 600);  
 frame.setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 new Phonebook();  
 }  
  
 public void actionPerformed(ActionEvent e) {  
 // Implement functionality for buttons here  
 if (e.getSource() == addButton) {  
 String name = nameField.getText();  
 String phone = phoneField.getText();  
 if (!name.isEmpty() && !phone.isEmpty()) {  
 contactListModel.addElement("Name: " + name + ", Phone: " + phone);  
 nameField.setText("");  
 phoneField.setText("");  
 } else {  
 JOptionPane.*showMessageDialog*(frame, "Please enter both name and phone number.");  
 }  
 } else if (e.getSource() == viewButton) {  
 String[] contacts = new String[contactListModel.size()];  
 for (int i = 0; i < contactListModel.size(); i++) {  
 contacts[i] = contactListModel.get(i);  
 }  
 Arrays.*sort*(contacts); // Sort contacts alphabetically  
  
 StringBuilder contactListDisplay = new StringBuilder();  
 for (String contact : contacts) {  
 contactListDisplay.append(contact).append("\n");  
 }  
  
 if (contactListModel.size() > 0) {  
 JOptionPane.*showMessageDialog*(frame, contactListDisplay.toString(), "All Contacts", JOptionPane.*INFORMATION\_MESSAGE*);  
 } else {  
 JOptionPane.*showMessageDialog*(frame, "No contacts to display.");  
 }  
 } else if (e.getSource() == searchButton) {  
 String search = searchField.getText().trim();  
 StringBuilder results = new StringBuilder();  
 boolean found = false;  
  
 for (int i = 0; i < contactListModel.size(); i++) {  
 String contact = contactListModel.get(i);  
 if (contact.toLowerCase().contains(search.toLowerCase())) {  
 results.append(contact).append("\n");  
 found = true;  
 }  
 }  
  
 if (found) {  
 JOptionPane.*showMessageDialog*(frame, results.toString(), "Search Results", JOptionPane.*INFORMATION\_MESSAGE*);  
 } else {  
 JOptionPane.*showMessageDialog*(frame, "Contact not found.");  
 }  
 } else if (e.getSource() == deleteButton) {  
 int selectedIndex = contactList.getSelectedIndex();  
 if (selectedIndex != -1) {  
 int response = JOptionPane.*showConfirmDialog*(frame,  
 "Are you sure you want to delete the selected contact?",  
 "Confirm Delete",  
 JOptionPane.*YES\_NO\_OPTION*);  
  
 if (response == JOptionPane.*YES\_OPTION*) {  
 contactListModel.remove(selectedIndex);  
 JOptionPane.*showMessageDialog*(frame, "Contact deleted successfully.");  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(frame, "Please select a contact to delete.");  
 }  
 } else if (e.getSource() == updateButton) {  
 int selectedIndex = contactList.getSelectedIndex();  
 if (selectedIndex != -1) {  
 String currentContact = contactListModel.get(selectedIndex);  
 String[] parts = currentContact.split(", "); // Split into name and phone  
  
 String currentName = parts[0].substring(6); // Remove "Name: "  
 String currentPhone = parts[1].substring(7); // Remove "Phone: "  
  
 // Show dialog for updating  
 String newName = JOptionPane.*showInputDialog*(frame, "Edit Name:", currentName);  
 String newPhone = JOptionPane.*showInputDialog*(frame, "Edit Phone:", currentPhone);  
  
 // Update if fields are not null (user pressed Cancel)  
 if (newName != null || newPhone != null) {  
 String updatedName = (newName != null && !newName.isEmpty()) ? newName : currentName;  
 String updatedPhone = (newPhone != null && !newPhone.isEmpty()) ? newPhone : currentPhone;  
  
 contactListModel.set(selectedIndex, "Name: " + updatedName + ", Phone: " + updatedPhone);  
 JOptionPane.*showMessageDialog*(frame, "Contact updated successfully.");  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(frame, "Please select a contact to update.");  
 }  
 }  
 }  
}