NAMIBIA UNIVERSITY OF SCIENCE AND TECHNOLOGY

Faculty of Computing & Informatics

Department of Software Engineering

Data Structures and Algorithms (DSA521S)

GROUP PROJECT 2024

GROUP MEMBERS:

Group 37

Group Leader:

NAME & SURNAME	**STUDENT NUMBER**
Endelina Uugwanga	224080008

Group members:

NAME & SURNAME	**STUDENT NUMBER**
Ndati Kafidi	224066765
Asanda Noludwe	223008575
Jedidja Mbinga	224016148
Didilikeni Kronelius	224025791
Petrus Amukogo	224032119

DESCRIPTION OF PROJECT

This PhoneBook program is a simple Java application that allows users to manage their contacts. It provides functionalities to add, search, update, delete, display, and sort contacts.

Key Features:

Contact Management: Users can add new contacts by entering a name and phone number.

Search Functionality: Users can search for a contact by name, and the program will display the contact's details if found.

Update and Delete: Users can update a contact's phone number or delete a contact from the phone book.

Display and Sort: The program can display all contacts and sort them alphabetically by name.

User Interaction: It uses a menu-driven interface, allowing users to choose actions easily.

Data Structure Used:

The program uses an ArrayList to store the contacts, which allows for dynamic resizing as contacts are added or removed. This makes it efficient for managing a variable number of contacts.

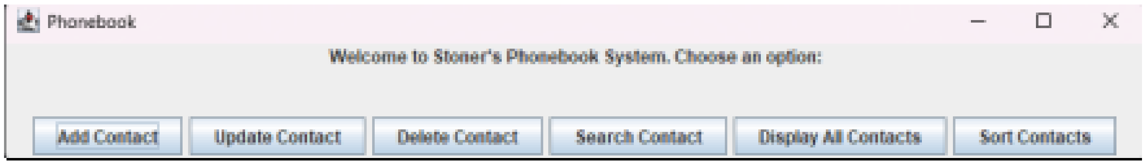
Overall, this phone book application is a practical tool for organizing and accessing contact information.

IDE Used:

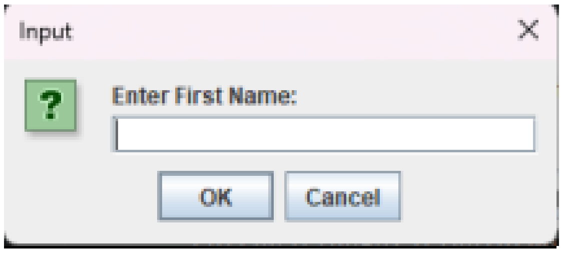
IntelliJ IDEA 2024.1.4(Ultimate Edition)

SCREENSHOTS OF OUR PHONEBOOK GUI SYSTEM

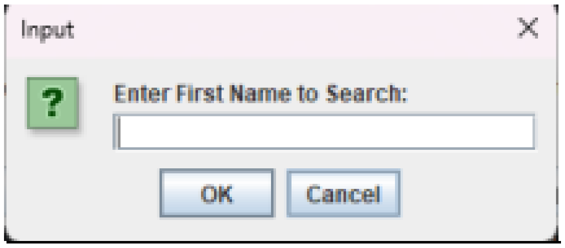
MAIN MENU



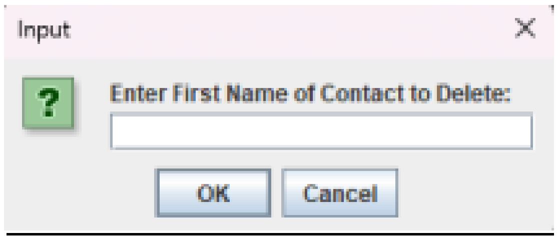
ADD CONTACT



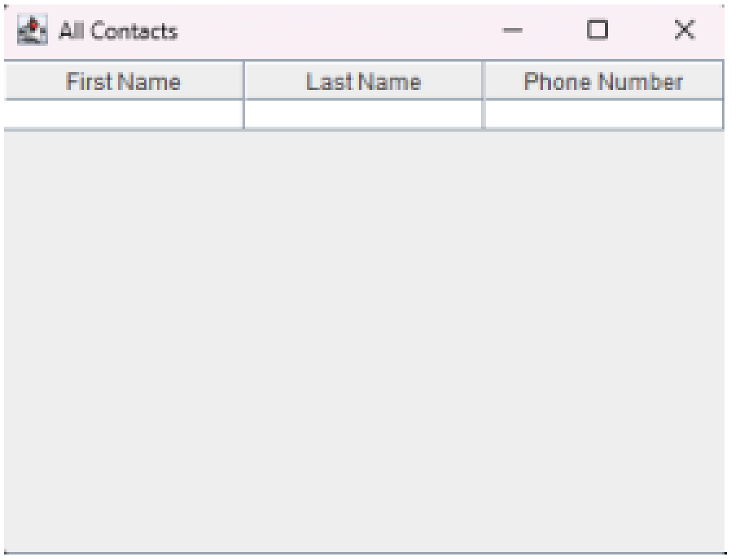
SEARCH CONTACT



DELETE CONTACT

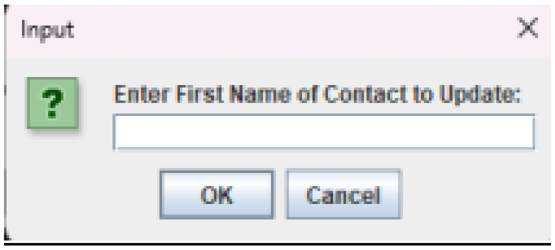


DISPLAY CONTACTS



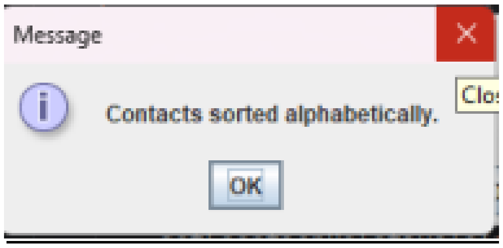
First Name	Last Name	Phone Number

UPDATE CONTACT



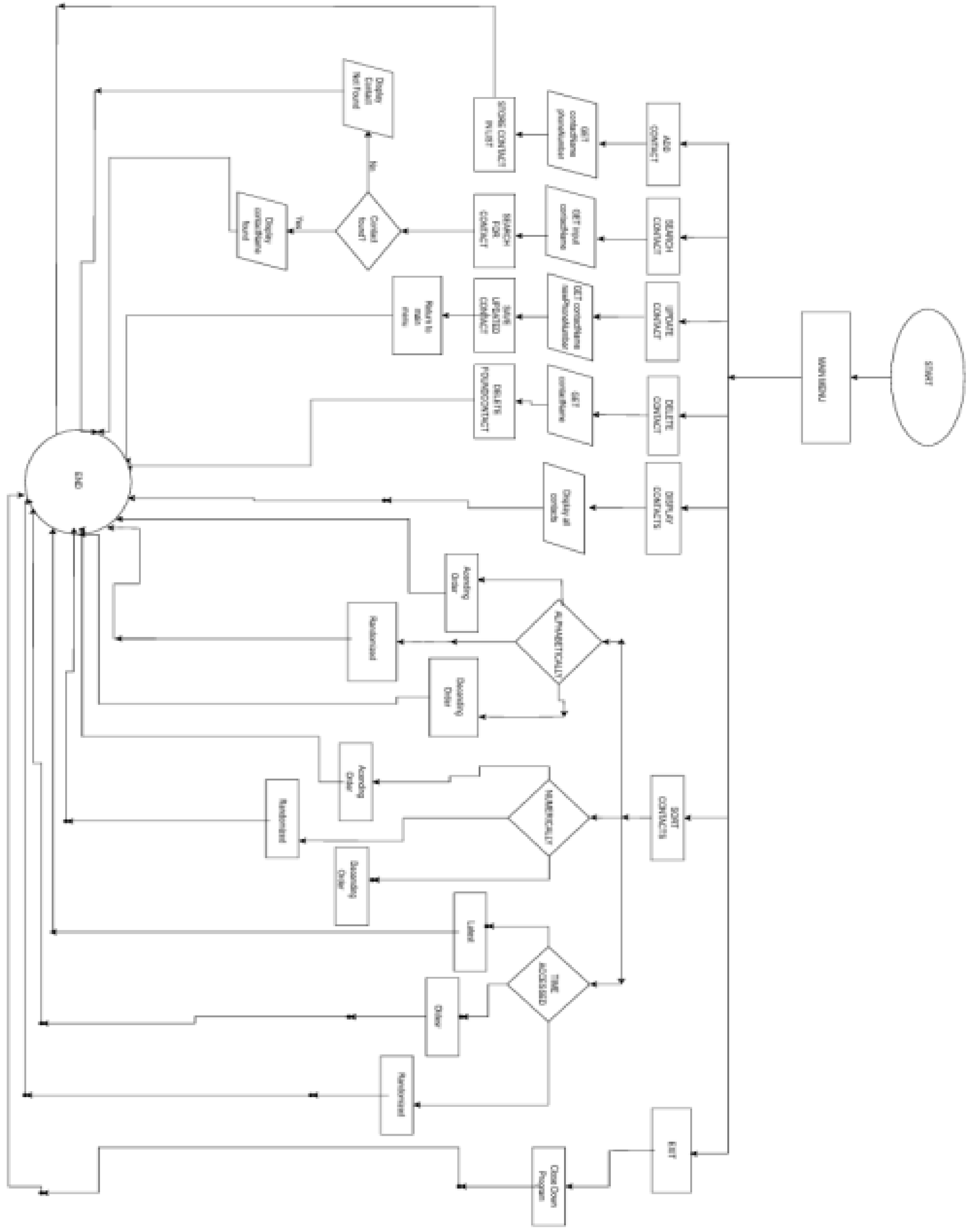
Enter First Name of Contact to Update:

SORT CONTACTS



Contacts sorted alphabetically.

FLOWCHART FOR OUR PHONEBOOK



PSEUDOCODES

INSERTING CONTACT

```
Start
Insert(phoneBook, name, telNumber):
  If name is not in phoneBook:
    phoneBook[name] = telNumber
    Print "Entry added successfully."
  Else:
    Print "Name already exists. Updating  number."
    phoneBook[name] = telNumber
End
```

SEARCHING CONTACT

```
Start
 Search(phoneBook, name):
  If name is available in phoneBook:
    Print "telephone  number for", name, "is", phoneBook[name]
  Else:
    Print "Name not found in phone book."
End
```

UPDATING CONTACT

```
Start
 updateContact(name, newPhoneNumber)
  set index TO -1
  for i from 0 to contactCount - 1 do
    if phonebook[i].name EQUALS name then
```

```
      set index to i
      break
    End if
  End for
  if index not equals -1 then
    set phonebook[index].phoneNumber to newPhoneNumber
    display "Contact updated successfully"
  Else
    display "Contact not found"
  End if
End
```

DISPLAY CONTACTS

```
Start
 DisplayAll(phoneBook):
  If  telnumber available in the phoneBook :
    For each name in phoneBook:
      Print name, ":", phoneBook[name]
  Else:
    Print "Phone book is empty."
End
```

DELETE CONTACT

```
Start
 deleteContact(name)
  set index TO -1
  for i from 0 to contactCount - 1 do
    If phonebook[i].name == name Then
```

```
        set index to i
        break
      End if
    End for
    If index !=  -1 Then
      For i From index To contactCount - 2 Do
        set phonebook[i] to phonebook[i + 1]
      End for
      Set contactCount To contactCount - 1
      display "Contact deleted successfully"
    Else
      display "Contact not found"
    End if
End
```

SORTING CONTACTS

```
Start
 sortContacts()
    for i From 0 To contactCount - 1 Do
      For j From 0 To contactCount - i - 1 Do
        If phonebook[j].name > phonebook[j + 1].name Then
          Set temp To phonebook[j]
          Set  phonebook[j] To phonebook[j + 1]
          Set phonebook[j + 1] To temp
        End if
      End for
    End for
    display "Contacts sorted successfully"
```

End

PHONE BOOK SOURCE CODE

```
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
import java.util.Scanner;

class Contact{
  private String name;
  public String phoneNumber;

  public Contact(String name, String phoneNumber){
    this.name = name;
    this.phoneNumber = phoneNumber;
  }

  public String getName(){
    return name;
  }

  public String getPhoneNumber(){
    return phoneNumber;
  }

  @Override
  public String toString(){
    return "Name Of Contact : "+ name + "Contact number : " + phoneNumber;
  }
}

class PhoneBook{
  private List<Contact>contacts;

  public PhoneBook(){
    contacts = new ArrayList<>();
  }
```

```
//insert contact Fuction
public void insertContact(String name, String phoneNumber){
  Contact contact = new Contact(name, phoneNumber);
  contacts.add(contact);
  System.out.println("Contact" + name + "Added.");
}

//search For contact function
public Contact searchContact(String searchName){
  for (Contact contact : contacts){
    if (contact.getName().equalsIgnoreCase(searchName)){
      System.out.println("Contact found : "+ contact );
      return contact;
    }
  }
  System.out.println("Contact not found.");
  return null;
}

//displaying contacts
public void displayContacts(){
  if (contacts.isEmpty()){
    System.out.println("PhoneBook is Empty");
  }else{
    for (Contact contact:contacts){
      System.out.println(contact);
    }
  }
}

//deleting a contact
public void deleteContact(String deleteName){
  for (int i = 0; i < contacts.size(); i++){
    if (contacts.get(i).getName().equalsIgnoreCase(deleteName)){
      contacts.remove(i);
      System.out.println("Contact " + deleteName + "Deleted.");
      return;
    }
  }
  System.out.println("Contact noy found.");
}
```

```
  //updating a contact's information
  public void updateContact(String updateName, String newPhoneNumber){
    for (Contact contact : contacts){
      if (contact.getName().equalsIgnoreCase(updateName)){
        contact.phoneNumber = newPhoneNumber;
        System.out.println("Contact " + updateName + "Updated.");
        return;
      }
    }
    System.out.println("Contact not found.");
  }

  //sorting contacts
  public void sortContacts(){
    contacts.sort(Comparator.comparing(Contact ::getName));
    System.out.println("Contacts sorted by name.");
  }
}

public class Main{
  public static void main(String[] args){

    PhoneBook phoneBook = new PhoneBook();
    Scanner scanner = new Scanner(System.in);
    String choice;

    do {
      System.out.println("\nPhoneBook Menu:");
      System.out.println("1. Add Contact");
      System.out.println("2. Search Contact");
      System.out.println("3. Update Contact");
      System.out.println("4. Delete Contact");
      System.out.println("5. Display Contacts");
      System.out.println("6. Sort Contacts");
      System.out.println("7. Exit");
      System.out.print("Choose an Option: ");
      choice = scanner.nextLine();

      switch (choice) {
        case "1":
```

```
    System.out.print("Enetr contact name: ");
    String name = scanner.nextLine();
    System.out.print("Enter Phone Number: ");
    String phoneNumber = scanner.nextLine();
    phoneBook.insertContact(name, phoneNumber);
    break;

  case "2":
    System.out.print("Enter contact name to search: ");
    String searchName = scanner.nextLine();
    phoneBook.searchContact(searchName);
    break;

  case "3":
    System.out.print("Enter contact name to update: ");
    String updateName = scanner.nextLine();
    System.out.print("Enter new phone number: ");
    String newPhoneNumber = scanner.nextLine();
    phoneBook.updateContact(updateName, newPhoneNumber);
    break;

  case "4":
    System.out.print("Enetr contact name to delete: ");
    String deleteName = scanner.nextLine();
    phoneBook.deleteContact(deleteName);
    break;

  case "5":
    phoneBook.displayContacts();
    break;

  case "6":
    phoneBook.sortContacts();
    break;

  case "7":
    System.out.println("Exiting...");
    break;

  default:
    System.out.println("Invalid Option. Please try again.");
```

```
        }
      }while(!choice.equals("7"));

      scanner.close();

      }
   }
```

GRAPHICAL USER INTERFACE FOR OUR PHONE BOOK

```
import javax.swing.*;
import java.awt.*;
import java.util.ArrayList;
import java.util.Comparator;

public class Phonebook extends JFrame {
   private ArrayList<Contact> contacts;

   public Phonebook() {
      contacts = new ArrayList<>();
      setTitle("Phonebook");
      setSize(800, 120);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setLayout(new BorderLayout());
      setLocationRelativeTo(null);

      JLabel heading = new JLabel("Welcome to Stoner's Phonebook System. Choose an
option:");
      heading.setHorizontalAlignment(0);
      add((heading), BorderLayout. NORTH);

      JButton addButton = new JButton("Add Contact");
      addButton.addActionListener(_ -> addContact());

      JButton updateButton = new JButton("Update Contact");
      updateButton.addActionListener(_ -> updateContact());
```

```
        JButton deleteButton = new JButton("Delete Contact");
        deleteButton.addActionListener(_ -> deleteContact());

        JButton searchButton = new JButton("Search Contact");
        searchButton.addActionListener(_ -> searchContact());

        JButton displayAllButton = new JButton("Display All Contacts");
        displayAllButton.addActionListener(_ -> displayAllContacts());

        JButton sortButton = new JButton("Sort Contacts");
        sortButton.addActionListener(e -> sortContacts());

        JPanel buttonPanel = new JPanel(new FlowLayout());
        buttonPanel.add(addButton);
        buttonPanel.add(updateButton);
        buttonPanel.add(deleteButton);
        buttonPanel.add(searchButton);
        buttonPanel.add(displayAllButton);
        buttonPanel.add(sortButton);
        add((buttonPanel), BorderLayout.SOUTH);
    }

    private void addContact() {
        String firstName = JOptionPane.showInputDialog("Enter First Name:");
        String lastName = JOptionPane.showInputDialog("Enter Last Name:");
        String phoneNumber = JOptionPane.showInputDialog("Enter Phone Number: ");
        contacts.add(new Contact(firstName, lastName, phoneNumber));
    }

    private void updateContact() {
        String name = JOptionPane.showInputDialog("Enter First Name of Contact to
Update:");
        for (Contact contact : contacts) {
            if (contact.getFirstName().equalsIgnoreCase(name)) {
                String newLastName = JOptionPane.showInputDialog("Enter New Last Name:");
                String newPhoneNumber = JOptionPane.showInputDialog("Enter New Phone
Number:");
                contact.setLastName(newLastName);
                contact.setPhoneNumber(newPhoneNumber);
                return;
            }
        }
        JOptionPane.showMessageDialog(this, "Contact not found.");
    }
    private void deleteContact() {
        String name = JOptionPane.showInputDialog("Enter First Name of Contact to
Delete:");
        contacts.removeIf(contact -> contact.getFirstName().equalsIgnoreCase(name));
    }

    private void searchContact() {
        String name = JOptionPane.showInputDialog("Enter First Name to Search:");
        for (Contact contact : contacts) {
            if (contact.getFirstName().equalsIgnoreCase(name)) {
                JOptionPane.showMessageDialog(this, contact.toString());
                return;
            }
        }
        JOptionPane.showMessageDialog(this, "Contact not found.");
    }

    private void displayAllContacts() {
        JFrame displayFrame = new JFrame("All Contacts");
        displayFrame.setSize(400, 300);
        displayFrame.setLayout(new BorderLayout());

        String[] columnNames = {"First Name", "Last Name", "Phone Number"};
        String[][] data = new String[contacts.size()][3];
        for (int i = 0; i < contacts.size(); i++) {
            data[i][0] = contacts.get(i).getFirstName();
            data[i][1] = contacts.get(i).getLastName();
            data[i][2] = contacts.get(i).getPhoneNumber();
        }
        JTable table = new JTable(data, columnNames);
        displayFrame.add(new JScrollPane(table), BorderLayout.CENTER);
        displayFrame.setVisible(true);
        displayFrame.setLocationRelativeTo(null);
    }

    private void sortContacts() {

contacts.sort(Comparator.comparing(Contact::getFirstName).thenComparing(Contact
::getLastName));
        JOptionPane.showMessageDialog(this, "Contacts sorted alphabetically.");
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            Phonebook phonebook = new Phonebook();
            phonebook.setVisible(true);
        });
    }

    class Contact {
        private String firstName;
        private String lastName;
        private String phoneNumber;

        public Contact(String firstName, String lastName, String phoneNumber) {
            this.firstName = firstName;
            this.lastName = lastName;
            this.phoneNumber = phoneNumber;
        }

        public String getFirstName() {
            return firstName;
        }

        public String getLastName() {
            return lastName;
        }

        public String getPhoneNumber() {
            return phoneNumber;
        }

        public void setLastName(String lastName) {
            this.lastName = lastName;
        }

        public void setPhoneNumber(String phoneNumber) {
            this.phoneNumber = phoneNumber;
        }
        @Override
        public String toString() {
            return firstName + " " + lastName + " - " + phoneNumber;
        }
    }
}
```

GITHUB LINK

https://github.com/Lina-nyanyu/PhonebookApp

READMETEXT FILE

README.md

README.md

SOFTWARES USED

IntelliJ IDEA 2024.1.4(Ultimate Edition)