**Phonebook Software**  
  
**Summary**

* With the help of a few simple phonebook functions, users can add, search, display, delete, and update contacts in this project's basic phonebook application. Java is used in its implementation, and IntelliJ IDEA can run it. In the project, contact management in a mobile phonebook system is demonstrated using standard algorithms and fundamental data structures (an ArrayList).

**Qualities**

The phonebook program facilitates the subsequent functions:

* Add Contact: Incorporate a new contact into the phone directory.
* Contact Search: Look up a contact by name or phone number.
* Show Every Contact: The phonebook's whole list of contacts is displayed.
* Delete a contact: To Delete a contact select it from the phonebook.
* Change Contact: Make changes to an existing contact's details (phone number or name).
* Organize Contacts: Sort contacts by name in alphabetical order (optional feature).

**Project Organization**

There are two primary Java classes in the project:

* **Phonebook.java**: This class houses all of the phonebook application's activities (insert, search, show, delete, update, and sort) as well as the menu system.
* **Java Contact**: With fields for the contact's name and phone number, this class represents a contact in a phonebook.

**1. Java Phonebook**  
The communication between the user and the phonebook is controlled by this class. It offers the subsequent techniques:

* insertContact(): Prompts the user for contact details (name and phone number) and adds the contact to the phonebook.
* searchContact(): Prompts the user for a contact name and searches for it in the phonebook.
* displayContacts(): Displays all contacts stored in the phonebook.
* deleteContact(): Deletes a contact from the phonebook based on the name.
* updateContact(): Updates the phone number of a contact based on the name or Updates the name.
* sortContacts(): Sorts all contacts in the phonebook alphabetically by name.

**2.** **Contact.java**

This class models a contact with the following properties:

* Name: the contact’s name.
* Phone: the contact’s phone number

It includes:

* Getters and setters for the contact’s name and phone number.
* An overridden tostring() method to display the contact’s details.

**How to run the project**

**Prerequisites**

* Java development kit (JDK)
* IntelliJ IDEA (or any Java IDE)

**Running the program**

1. Clone the GitHub repository:
2. Open the project in IntelliJ IDEA (or any Java IDE).
3. Compile and run the phonebook.java file. The program will start and display a menu where you can choose from the available operations.
4. Follow the menu prompts to interact with the phonebook

**Analysis of the search algorithm**

The search operation implemented in this project uses a linear search, which iterates through the list of contacts one by one. The time complexity of this search operation is **O(n)**where n is the number of contacts. While this is not the most efficient search method, it is simple and works well for small datasets.

An optional optimization is to use sorting combined with a binary search, which would reduce the search time complexity to **O(log n)**, provided the list is sorted.

**Contributors**

This project was developed by a team of (\_\_) members as part of the DSA521S group project. Below is the list of contributions and their respective roles in the project:

|  |  |  |
| --- | --- | --- |
| **Contributor Name** | **Student ID** | **Role** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |