Project Summary

You are tasked with designing and implementing a phonebook application for a Namibian telecommunications company. The project is divided into two main sections:

Section A: Pseudocode and Flowcharts (75 marks)
- **Objective**: Design algorithms using pseudocode and flowcharts to handle basic phonebook operations.
- **Operations to Implement**:
  1. Insert Contact
  2. Search Contact
  3. Display All Contacts
  4. Delete Contact
  5. Update Contact
  6. (Optional) Sort Contacts
  7. Analyze the efficiency of your search algorithm

Section B: Practical Implementation (25 marks)
- **Objective**: Implement the designed algorithms using a programming language of your choice.
- **Evaluation Criteria**:
  - **Creativity**: The project should be engaging and well thought out.
  - **Logic**: Use appropriate programming constructs and logical operations.
  - **Effort**: Demonstrate a substantial amount of work and time invested in the project.

What You Need to Do

1. Form a Group
- Your group should consist of 5 to 6 members.
- Ensure everyone contributes equally and is prepared to present their work.

2. Design Algorithms (Section A)
- **Create Modules and Functions**: Break down the application into modules and functions. For example, have distinct modules for inserting, searching, and updating contacts.
- **Write Pseudocode**: Clearly define each function using pseudocode. Make sure the pseudocode is easy to understand and follows logical steps.
- **Draw Flowcharts**: Use flowcharts to visually represent the algorithms. Ensure they use correct symbols and are easy to follow.
- **Document Your Work**: Include a README file in your repository that describes the project, its modules, and contributors.

3. Implement the Project (Section B)
- **Choose a Programming Language**: Use any programming language you are comfortable with (e.g., Python, Java, C++).
- **Implement Data Structures and Operations**: Implement the algorithms for inserting, searching, updating, and deleting contacts.
- **Ensure Quality**: Make sure your code is well-structured, logical, and demonstrates creativity. Utilize good programming practices and commit enough effort.

Platforms and Tools

For Section A (Pseudocode and Flowcharts)

- **Pseudocode**: Use a simple text editor like Notepad++ or online pseudocode tools like [Pseudocode Online](https://www.pseudocode.org/).
- **Flowcharts**: Free tools like [draw.io](https://app.diagrams.net/) or [Lucidchart](https://www.lucidchart.com/) (limited free version available) can be used for creating flowcharts.

For Section B (Practical Implementation)
- **Programming Languages**: Python, Java, C++ (all are free and widely used).
- **Development Environment**:
  - **Python**: Use [PyCharm Community Edition](https://www.jetbrains.com/pycharm/download/) or [VS Code](https://code.visualstudio.com/).
  - **Java**: Use [Eclipse IDE](https://www.eclipse.org/downloads/) or [NetBeans](https://netbeans.apache.org/).
  - **C++**: Use [Code::Blocks](http://www.codeblocks.org/) or [Visual Studio Code](https://code.visualstudio.com/).

Version Control
- **GitHub**: Use GitHub for version control and collaboration. Create a repository for your project and ensure all members commit their code and documentation.

Best Practices

1. **Collaborate Effectively**: Communicate regularly with your group members and divide tasks based on each member's strengths.
2. **Plan Ahead**: Create a timeline for completing each section of the project.
3. **Document Clearly**: Make sure your pseudocode, flowcharts, and code are well-documented and easy to understand.
4. **Test Thoroughly**: Implement and test each function to ensure it works as expected.
5. **Prepare for Presentation**: Be ready to explain your contributions and the project’s functionality during the presentation.

By following these guidelines and using the recommended tools, you can create a robust and well-documented phonebook application that meets the project requirements and earns you a high grade.

Project Overview

You are assigned a group project to develop a phonebook application for a telecommunications company. The project is divided into two sections:

1. Section A: Design and Documentation (75 marks)
- **Objective**: Design the algorithms for a phonebook application using pseudocode and flowcharts.
- **Requirements**:
  1. **Insert Contact**
  2. **Search Contact**
  3. **Display All Contacts**
  4. **Delete Contact**
  5. **Update Contact**
  6. **Sort Contacts** (Optional)
  7. **Analyze Search Efficiency**

2. Section B: Implementation (25 marks)
- **Objective**: Implement the algorithms using Java and create a working phonebook application.
- **Evaluation Criteria**:
  - **Creativity**: The project should be engaging and well-designed.
  - **Logic**: Use appropriate programming constructs.
  - **Effort**: Demonstrate a substantial amount of work and time invested.

How to Approach the Project

Section A: Design and Documentation

1. **Form Your Group**
   - Assemble 5-6 members.
   - Allocate roles and responsibilities to ensure each member contributes.

2. **Design the Algorithm**
   - **Modules and Functions**:
     - Create distinct modules for each operation (Insert, Search, Display, Delete, Update, Sort).
     - Define functions within these modules with clear input/output specifications.

   - **Pseudocode**:
     - Write pseudocode for each function. Ensure it is logical, clear, and precise.
     - Example pseudocode for `Insert Contact`:
       ```
       Function InsertContact(name, phoneNumber):
           Create new Contact object with name and phoneNumber
           Add Contact object to the list
       ```

   - **Flowcharts**:
     - Create flowcharts for each function. Use tools like [draw.io](https://app.diagrams.net/) for free flowchart creation.
     - Ensure symbols are used correctly (e.g., ovals for start/end, rectangles for processes, diamonds for decisions).

3. **Documentation**
  - **README File**:
    - Include a README file in your GitHub repository.
    - Describe the project, the algorithms used, and how to run the code.
    - List contributors and their roles.

Section B: Implementation

1. **Set Up Your Development Environment**
  - **IDE**: Use [Eclipse IDE](https://www.eclipse.org/downloads/) or [IntelliJ IDEA Community Edition](https://www.jetbrains.com/idea/download/) for Java development.
  - **Version Control**: Use [GitHub](https://github.com/) for version control. Create a repository for your project.

2. **Implement the Application**
  - **Java Data Structures**:
    - **Array or ArrayList**: For storing contacts.
    - **LinkedList**: For managing contacts if you need efficient insertions/deletions.

  - **Java Functions**:
    - **Insert Contact**:
      ```java
      public void insertContact(String name, String phoneNumber) {
          Contact newContact = new Contact(name, phoneNumber);
          contacts.add(newContact);
      }
      ```

    - **Search Contact**:
      ```java
      public Contact searchContact(String name) {
          for (Contact contact : contacts) {
              if (contact.getName().equals(name)) {
                  return contact;
              }
          }
          return null;
      }
      ```

    - **Display All Contacts**:
      ```java
      public void displayContacts() {
          for (Contact contact : contacts) {
              System.out.println(contact);
          }
      }
      ```

    - **Delete Contact**:
      ```java
      public void deleteContact(String name) {
      ```

```
         contacts.removeIf(contact -> contact.getName().equals(name));
     }
     ```

    - **Update Contact**:
     ```java
     public void updateContact(String name, String newPhoneNumber) {
         for (Contact contact : contacts) {
             if (contact.getName().equals(name)) {
                 contact.setPhoneNumber(newPhoneNumber);
                 return;
             }
         }
     }
     ```

   - **Sorting Contacts** (Optional):
     - Implement sorting using Java’s built-in sorting methods or libraries.

   - **Analyze Search Efficiency**:
     - Implement and measure the performance of your search algorithm, e.g., linear search vs. binary search.

3. **Testing and Finalization**
   - Test all functions to ensure they work as expected.
   - Make necessary refinements and ensure the application is user-friendly.

4. **Prepare for Presentation**
   - Be ready to present your project and demonstrate each functionality.
   - Ensure all group members can explain their contributions.

Free Tools and Platforms

- **IDE**: [Eclipse IDE](https://www.eclipse.org/downloads/) or [IntelliJ IDEA Community Edition](https://www.jetbrains.com/idea/download/)
- **Flowchart Tool**: [draw.io](https://app.diagrams.net/)
- **Version Control**: [GitHub](https://github.com/)

Best Practices

1. **Collaborate Efficiently**: Communicate regularly with your team and divide tasks effectively.
2. **Follow the Rubric**: Ensure that each part of your project meets the criteria specified in the rubric.
3. **Document Everything**: Keep your documentation and code well-organized and clear.
4. **Test Rigorously**: Ensure thorough testing to catch and fix any issues.
5. **Practice Presentation**: Prepare and rehearse your presentation to ensure a smooth delivery.

By following these guidelines and utilizing the provided tools, you can effectively tackle the project and aim for a high grade.