Proposal for Simple Calendar Application

Summary of Project Goals

The goal of this project is to create a Simple Calendar Application that allows users to view a calendar for any specified month and year, as well as manage events associated with that calendar. The program will allow users to:

- Add events (e.g., meetings or reminders) to specific dates.
- Delete events by name.
- Print all events for a specific month.

This program is intended for individual users who want to keep track of their schedules and appointments in an easy-to-use calendar interface.

Alignment with Project Guidelines

This project meets the guidelines in the following ways:

- **Key Algorithms and Data Structures**: A linked list will be used to store and manage all events associated with the calendar. This data structure supports efficient insertion, deletion, and traversal of events. Sorting and searching functionalities will be implemented to organize and retrieve events efficiently.

- **Object-Oriented Programming (OOP)**: The program will use:
  - **Inheritance**: An abstract base class 'Event' will define common functionality for different event types (e.g., meetings and reminders), with subclasses adding specific behavior.
  - **Composition**: The 'Calendar' class will contain a linked list of 'Event' objects, demonstrating a has-a relationship.

- **Communication**: Classes will interact through clear interfaces, ensuring modularity and maintainability.
- **Polymorphism**: The use of virtual methods will enable handling different event types uniformly.

UML Diagram of Classes/Relationships

The UML diagram will illustrate the relationships among the following classes:

- **Calendar**: Manages the linked list of events and provides functionality for adding, deleting, and printing events.
- **Event**: An abstract base class with subclasses 'Meeting' and 'Reminder', defining the structure and behavior of events.
- **LinkedList<T>**: A template class that manages a list of events, supporting sorting and searching.

The diagram will highlight inheritance (between Event and its subclasses), composition (Calendar contains LinkedList<Event*>), and communication (Calendar interacting with LinkedList and Event).

Stages of Development

The development process will follow these stages:

1. **Initial Setup**:

- Set up the project structure and create basic class definitions (Calendar, Event, LinkedList).
- Implement the LinkedList template class with add, remove, and traversal functionality.

2. **Basic Event Management**:

- Define the abstract Event class and implement concrete subclasses (Meeting and Reminder).
- Integrate the LinkedList into the Calendar class and test adding/removing events.

3. **Search and Sort**:

- Add sorting functionality to organize events by date.

- Implement search functionality to find events by name.

4. **User Interaction**:

- Create a text-based interface to handle user input for adding, deleting, and printing events.

- Ensure that the calendar display updates dynamically with user actions.

5. **Final Touches**:

- Test and debug the program for edge cases (e.g., empty lists, invalid input).

- Refine the interface and prepare for deployment.