TrojanBook - Contact Management System

Phase 2 Enhancements

- **Friend Connections**: Added ability to connect people as friends
- **Friend Display**: Enhanced person display to show their connections
- **Sorted Friend List**: Implementation of a function to display friends in alphabetical order
- **Data Persistence**: Updated file I/O to store and load friendship information
- **Unique ID System**: Added codeName function to generate unique IDs for people

Classes

Contact Class (contact.h, contact.cpp)

- Abstract base class with two derived classes: Email and Phone
- Implements virtual methods for handling contact information
- Email class manages email addresses
- Phone class handles phone numbers with proper formatting (XXX-XXX-XXXX)

Date Class (date.h, date.cpp)

- Handles date information with different format options
- Implements parsing of M/D/YYYY format
- Provides date comparison operators

Person Class (person.h, person.cpp)

- Stores personal information: first name, last name, birthdate, email, and phone
- Maintains a vector of friends (added in Phase 2)
- Provides methods for creating and managing friendships
- Implements printing functions for person details and friend lists

Network Class (network.h, network.cpp)

- Manages a doubly linked list of Person objects
- Implements load and save functionality for database files
- Provides search methods to find people by name
- Includes methods for adding and removing people
- Features an interactive menu interface
- Implements "Connect" functionality for creating friendships (Phase 2)

Misc Functions (misc.h, misc.cpp)

- Utility functions used across the application
- printMe: Displays the application banner
- codeName: Generates unique IDs by concatenating names, removing spaces, and converting to lowercase

Menu Options

1. **Save network database**: Save the current network to a file
2. **Load network database**: Load a network from a file
3. **Add a new person**: Add a person to the network
4. **Remove a person**: Remove a person from the network
5. **Print people with last name**: Search for people by last name
6. **Connect**: Make a connection between two people (added in Phase 2)
7. **Display sorted friends**: Show the friends of a person in a sorted order (added in Phase 2)

Friend Functionality

- Each Person object maintains a vector of pointers to their friends
- Friends are displayed in the person's information output
- The `print_friends` method displays a sorted list of a person's friends
- Friend connections are bidirectional (if A is friends with B, then B is also friends with A)
- Friend relationships are persisted when saving and loading the network

Compilation Instructions

```
# Compile the Network Test
g++ -o test_network date.cpp contact.cpp person.cpp network.cpp misc.cpp test_network.cpp

# Compile the Person Equality Test
g++ -o test_person date.cpp contact.cpp person.cpp test_person_eq.cpp
```

Testing Phase 2 Features

To test the Phase 2 features of the TrojanBook application, you have two options:

Option 1: Automated Test Program

We've included a special test program that demonstrates all Phase 2 features:

1. **Compile the program**:

   ```
   make test_phase2
   ```

or simply `make` to build all executables.

2. **Run the test program**:

   `./test_phase2.o`

   This program will:

   - Create a network with 5 people
   - Make friend connections between them
   - Display a person's information with their friends
   - Show the sorted friends list
   - Save the network to a file
   - Load the network from the file
   - Verify that friend relationships were preserved

   The output will clearly show each step of the testing process and verify that all Phase 2 functionality works correctly.

Option 2: Interactive Testing

You can also test the features interactively through the menu system:

1. **Compile the project**:

   `make`

   This will create the executable "test_network.o"

2. **Run the application**:

   `./test_network.o`

3. **Testing Friend Connections**:

   a. First, add some people to the network (option 3) or load an existing database (option 2).

   b. To connect two people as friends, select option 6 from the main menu.

   c. Enter the first and last name of the first person.

   d. Enter the first and last name of the second person.

   e. The application will display both people's information and then make them friends with each other.

   f. After creating the connection, the application will automatically display the sorted friends list for both people.

4. **Testing Sorted Friends Display**:

   a. To view a person's friends in a sorted order at any time, select option 7 from the main menu.

b. Enter the first and last name of the person whose friends you want to view.

c. The application will display the person's friends sorted by their code names according to the sorting criteria specified in Part 5.

5. **Testing Data Persistence**:

   a. After creating some friend connections, save the network to a file (option 1).

   b. Exit the application and run it again.

   c. Load the saved database file (option 2).

   d. View a person's information (option 5 or 7) to verify that the friend connections have been preserved.

6. **Expected Outputs**:

   - When using option 6 (Connect), after connecting two people, you should see the information of both people and then the sorted friends list for each.
   - When using option 7 (Display sorted friends), you should see the person's name followed by a list of their friends sorted by code name.
   - When viewing a person's information after making connections, the friends should be listed with their code and name in parentheses.

File Format

The network database files store person information in the following format: - First name - Last name - Birthdate (MM/DD/YYYY) - Phone information - Email information - Friend codes (added in Phase 2) - Separator line (———————) between people

Future Enhancements

- Enhanced search capabilities
- Group management
- Profile pictures
- Message exchange system
- Privacy settings