A black background with a black square

Description automatically generated with medium confidence

**COMP1028 Programming and Algorithm**

**Session: Autumn 2024**

**Group Coursework**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Group Name** | **Gcc world** | | | | | |
| **Group Members** | | | | | | |
| **Name 1 (Leader)** | **Ahmed Gobran** | | | **ID 1** | **20580127** | |
| **Name 2** | **Umer Imran** | | | **ID 2** | **20511126** | |
| **Name 3** | **Kazuki Ichikawa** | | | **ID 3** | **20605542** | |
| **Name 4** | **Karam Allaham** | | | **ID 4** | **20702950** | |
| **Name 5** |  | | | **ID 5** |  | |
| **Marks**  **Total (100)** | Program functionality and code quality (50) | Usability (10) | Comments(5) | Documentation (5) | Presentation & Demo (Recording) (15) | Bonus Feature (15) |
|  |  |  |  |  |  |  |
| **Leader Sign** | **ahmed** | | | | | |
| **Date** | **5/12/2024** | | | | | |

**Contact Management System**

**1.0 Introduction**

The ***Contact Management System*** is a simple, command-line program written in C to organize and manage contact information. It allows users to add, search, edit, delete, search them in various methods, and sort contacts, as well as filter them by date. Contact details are validated and stored in a file for persistence, ensuring data is retained between sessions. The program functionality includes encryption and decryption of data when needed. The program uses a linked list for dynamic contact management and includes basic error handling and input validation.

**Purpose:**

This project is a command-line-based **Contact Management System** designed to allow users to manage a list of personal or professional contacts efficiently. It provides features to store, retrieve, update, delete, and search contact information in a structured and user-friendly manner.

**2.0 Installation**

**Installation and Setup Guide**

Follow these step-by-step instructions to set up and run the Contact Management System.

**Prerequisites**

Before starting, make sure you have the following installed on your system:

1. **C Compiler (e.g., GCC):**
   * Ensure that a C compiler is installed. For most systems:
     + On **Linux** or **MacOS**, GCC is commonly pre-installed or can be installed via package managers.
     + On **Windows**, install [MinGW](https://www.mingw-w64.org/) or use the [Windows Subsystem for Linux (WSL)](https://learn.microsoft.com/en-us/windows/wsl/) for Linux-like development.

To verify, run in your command terminal:

gcc --version

**Repository:**

1. Clone the repository:

A black screen with white text

Description automatically generated

1. **Navigate to the Project Directory:**

Open your terminal and go to the folder where the project files are located.

A black screen with white text

Description automatically generated

1. **Compile the program:**

**A black screen with white text

Description automatically generated**

**3. Run the program:**

A black screen with white text

Description automatically generated

**3.0 Features**

**Key Features:**

1. **Add Contacts:**
   * Add new contacts to the list with details such as name, phone number, address, and email.
   * Automatically saves the date when the contact was added.
2. **Search Contacts:**
   * Search contacts by name, phone number, address, or email.
   * Provides partial matching functionality for more flexible searches.
3. **Edit Contacts:**
   * Modify existing contact details (e.g., phone number, address, or email) while preserving other information.
4. **Delete Contacts:**
   * Remove specific contacts from the list by providing their name.
5. **Display All Contacts:**
   * View all saved contacts in the list along with their details.
6. **Sort Contacts:**
   * Sort the contacts alphabetically by their names.
7. **Filter Contacts by Date:**
   * Filter contacts added within a specific date range.
8. **File Storage:**
   * Contacts are saved in a file (contact.txt), ensuring data persistence across sessions.
   * Automatically loads contacts from the file upon startup.
9. **Validation:**
   * Validates email addresses and phone numbers during the addition or update process to ensure data accuracy.
10. **Advanced Search:**
    * Ability to search contacts and their components dynamically, via an advanced search option.

**Technical Details:**

* Written in **C**, using standard libraries for file handling, memory management, and string processing.
* Implements a **singly linked list** to dynamically manage contacts.
* Utilizes date handling to track and filter contacts.
* **Graphical User Interface (GUI):**
  + Implemented the GUI using **GTK** for the user interface.
  + Designed the interface with **Glade**, a GTK interface designer.
* **Environment Setup:**
  + Used **MSYS2**, a terminal emulator and development environment, to set up the required tools for GTK development.
* **Contact Storage:**
  + Contacts are saved in a persistent text file (contact.txt), ensuring data is retained between sessions.
* **Use of Header Files:**
  + Implemented .h files to declare functions, types, and constants for modularity and reuse across .c files.
* **Makefile Usage:**
  + Created a **Makefile** to automate the build process, which includes:
    - make clean command to remove previous build artifacts.
    - make command to compile and link the program efficiently.
* **User Interaction:**
  + Allowed users to add, edit, delete, and search for contacts through the GUI.
  + Contact details, including name, phone, address, and email, are validated and stored.
* **Application Execution:**
  + The application was executed in MSYS2 with the ./contact\_manager command, launching a user-friendly interface for managing contacts.

**Encryption/Decryption:**

Utilizing encryption/decryption functions to encrypt data being inputted by the user, and when needed decrypting as well.

The encrypt and decrypt functions implement a simple substitution cipher to secure text data. The encrypt function shifts each alphanumeric character in the input string by a defined key (ENCRYPTION\_KEY), binding around within their respective ranges (uppercase letters, lowercase letters, or digits).

For example, an uppercase letter 'A' becomes 'D' if the key is 3. The decrypt function reverses this process by shifting characters back by the same key, ensuring data integrity when decrypted. This encryption is applied when saving contact information to the file, protecting sensitive data. Conversely, decryption is used when loading contacts from the file to ensure the original text is restored for display.

**4.0 Contact**

* Ahmed Gobran
* [Efyag11@nottingham.edu.my](mailto:Efyag11@nottingham.edu.my)