Each functionthe execution process:

1. `void displayMenu()`

   - This function simply displays the main menu of the application with three options: Register, Login, and Exit.

2. `void displayUserMenu(const char* username)`

   - This function displays the user menu after a successful login. It presents four options: Write an Entry, View Previous Entries, Search Entries, and Logout.

3. `void registerUser()`

   - This function allows a user to register by providing a username and password. The username and password are read from the user and stored in a `struct User` variable. The password is then hashed using the `simpleHash` function. The user information is then written to the "users.txt" file.

4. `int loginUser(char* currentUser)`

   - This function allows a user to login by providing a username and password. The username and password are read from the user and stored in a `struct User` variable. The password is hashed using the `simpleHash` function. The function then reads the "users.txt" file to compare the entered credentials with the stored user information. If a match is found, the username is copied to the `currentUser` parameter, and the function returns 1 (indicating a successful login); otherwise, it returns 0.

5. `void writeEntry(const char* username)`

   - This function allows a user to write a new diary entry. The user is prompted to enter the date and content of the entry. The date is validated using the `validateDate` function. The content is then encrypted using the `encrypt` function. The encrypted entry, along with the username and date, is written to the "diary.txt" file.

6. `void viewEntries(const char* username)`

   - This function displays all previous diary entries associated with the logged-in user. It reads the "diary.txt" file, and for each entry, it decrypts the content using the `decrypt` function and displays the date and content on the console.

7. `void searchEntries(const char* username)`

- This function allows the user to search for previous diary entries based on a search term (either the date or the content). The user is prompted to enter the search term. The function reads the "diary.txt" file, decrypts the content of each entry using the `decrypt` function, and checks if the search term matches either the date or the content of the entry. If a match is found, the entry's date and decrypted content are displayed on the console.

8. `void clearBuffer()`

- This function clears the input buffer to ensure no unwanted characters are left behind in the buffer after reading input.

9. `void encrypt(char* text, int key)`

- This function encrypts the provided text using a simple encryption algorithm. It takes each character in the text and adds the `key` value to it.

10. `void decrypt(char* text, int key)`

- This function decrypts the provided text using the same simple encryption algorithm used in the `encrypt` function. It takes each character in the text and subtracts the `key` value from it to reverse the encryption process.

11. `bool validateDate(const char* date)`

- This function validates the format of the date provided by the user. It checks if the date can be parsed correctly in the format DD-MM-YYYY and if the day, month, and year fall within valid ranges.

Execution Process:

1. When the program starts, it enters the main loop, displaying the main menu and prompting the user for a choice.

2. Based on the user's choice, the appropriate function is called. If the user chooses to register, the `registerUser` function is called, allowing the user to create an account.

3. If the user chooses to login, the `loginUser` function is called, and the user is prompted to enter their credentials. If the login is successful, the `displayUserMenu` function is called, showing the user menu.

4. In the user menu, the user can select various options. Depending on the option chosen, the corresponding function is called.

5. When the user writes a new entry, the `writeEntry` function is called, and the user is prompted to enter the date and content. The entry is then encrypted and saved to the "diary.txt" file.

6. When the user views previous entries, the `viewEntries` function is called. It reads the "diary.txt" file and displays all previous entries associated with the logged-in user, decrypting the content before display.

7. When the user searches for entries, the `searchEntries` function is called. It reads the "diary.txt" file, decrypts the content of each entry, and checks if the search term matches either the date or the content.

8. The loop continues until the user selects the "Logout

" option from the user menu, at which point they are logged out and returned to the main menu.

9. The application continues running until the user selects the "Exit" option from the main menu, at which point the program terminates.