Personal Diary Application

File handlings in C

Prateek Koirala

Pratik Dahal

Rajesh Kumar Vishwakarma

Sumit Joshi

Intro

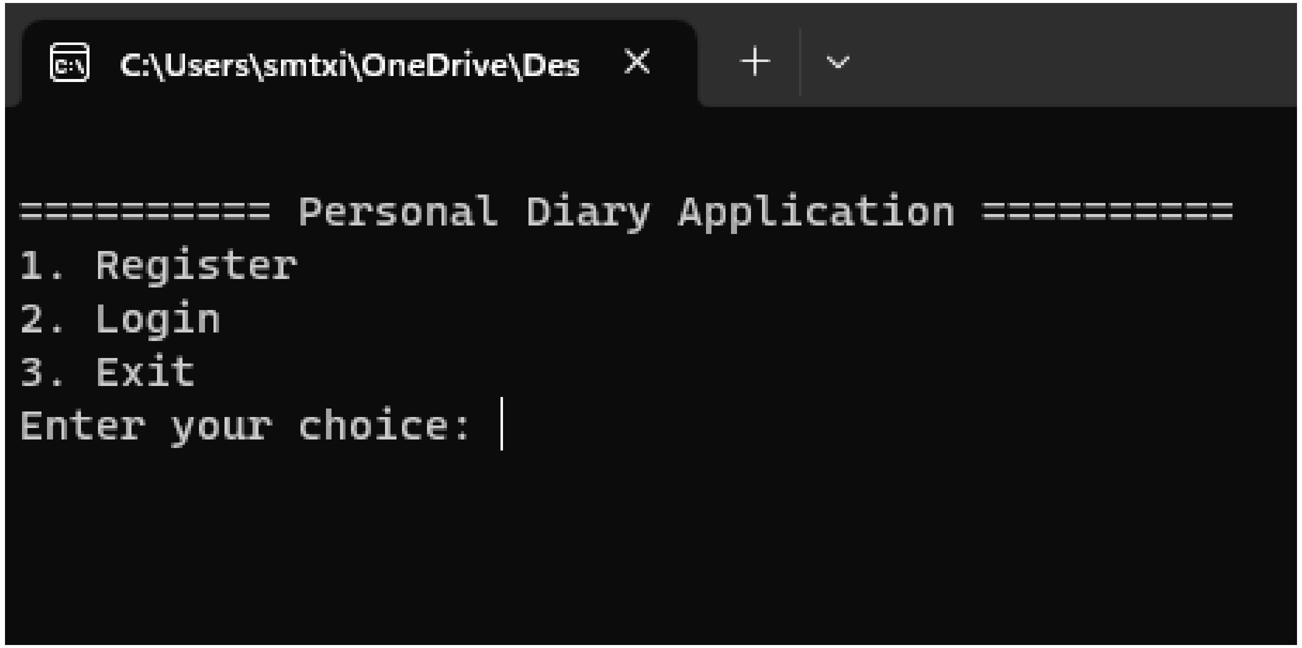
The Personal Diary Application is a console-based program that allows users to register, login, and maintain their diary entries.

It got multi options command interface so as to navigate through said system.

The application is created using C programming language. Required the basics of the programming the application employs the basic C features including functions, pointers, arrays, structs, file-handling, etc

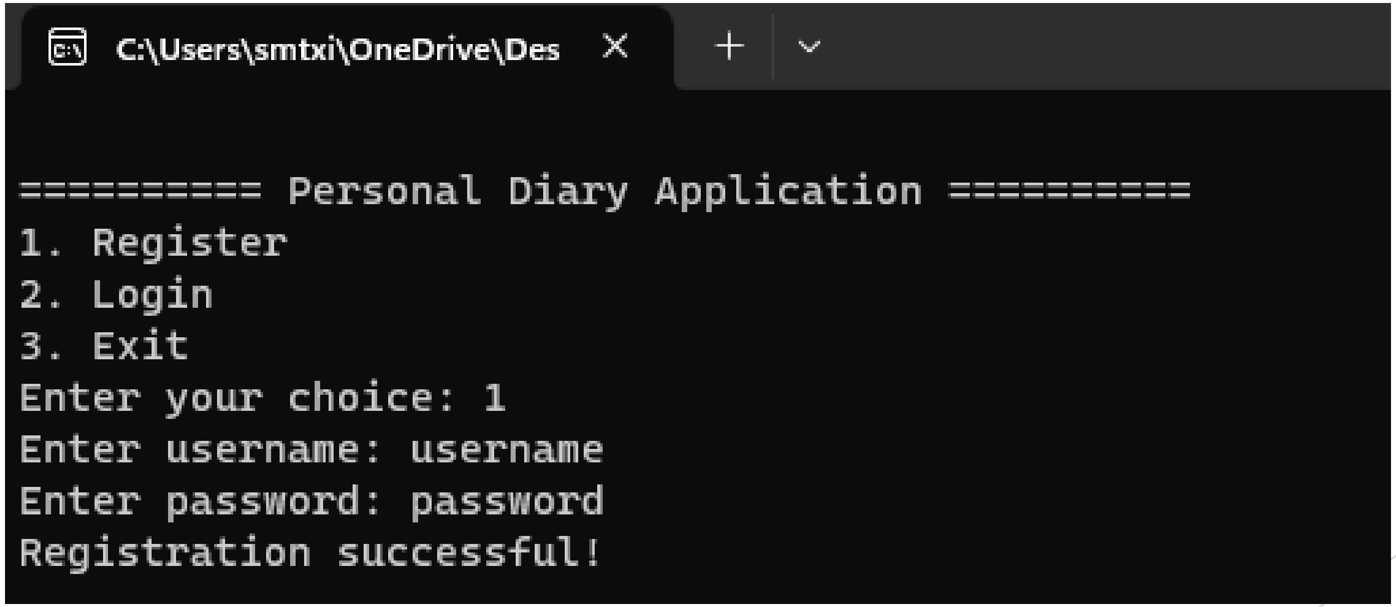
The application with necessary functions allows user to store required data and maintain privacy too

- 1. The application starts by displaying the main menu with three options: Register, Login, and Exit



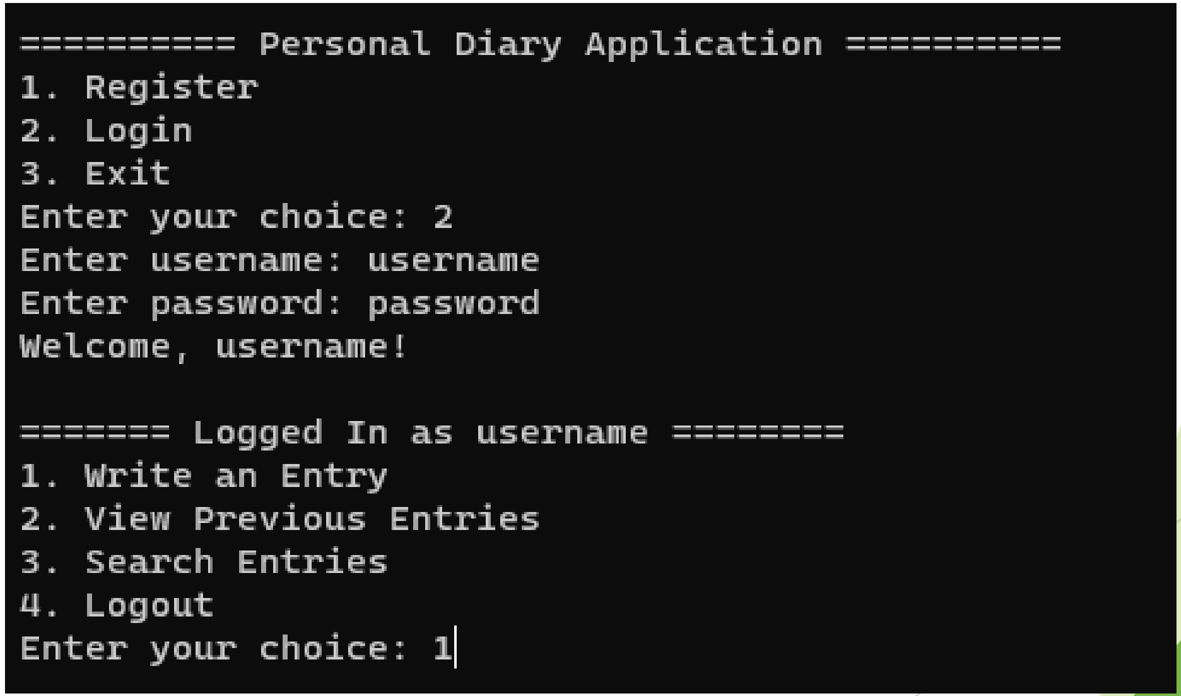
```
========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
Enter your choice:
```

If the user selects the "Register" option, they can create a new account by providing a username and password. The password is hashed using a simple hashing algorithm, and the user's information is stored in a file named "users.txt"



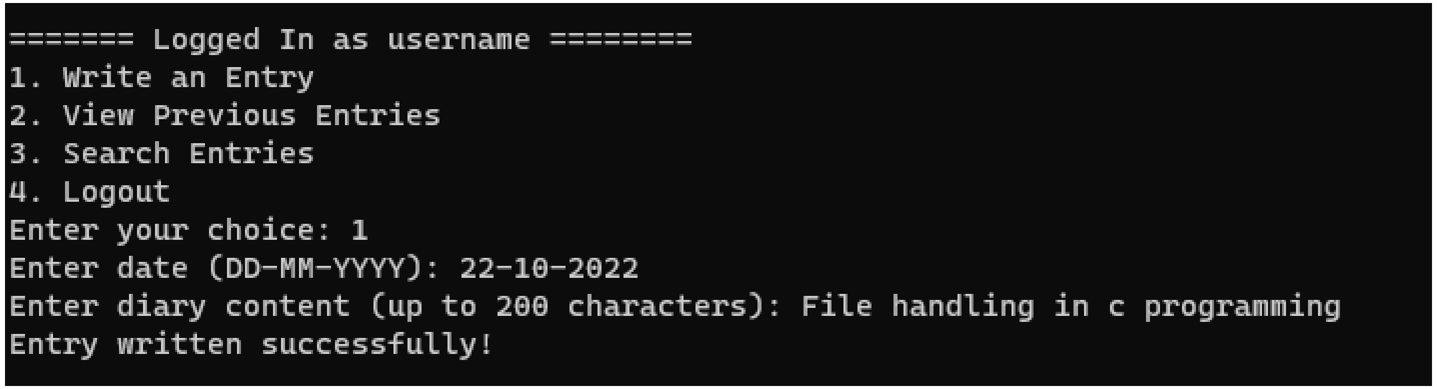
```
========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
Enter your choice: 1
Enter username: username
Enter password: password
Registration successful!
```

- If the user selects the "Login" option, they are prompted to enter their username and password. The application reads the "users.txt" file, hashes the provided password, and checks if the credentials match any of the registered users. If the login is successful, the user is redirected to the user menu



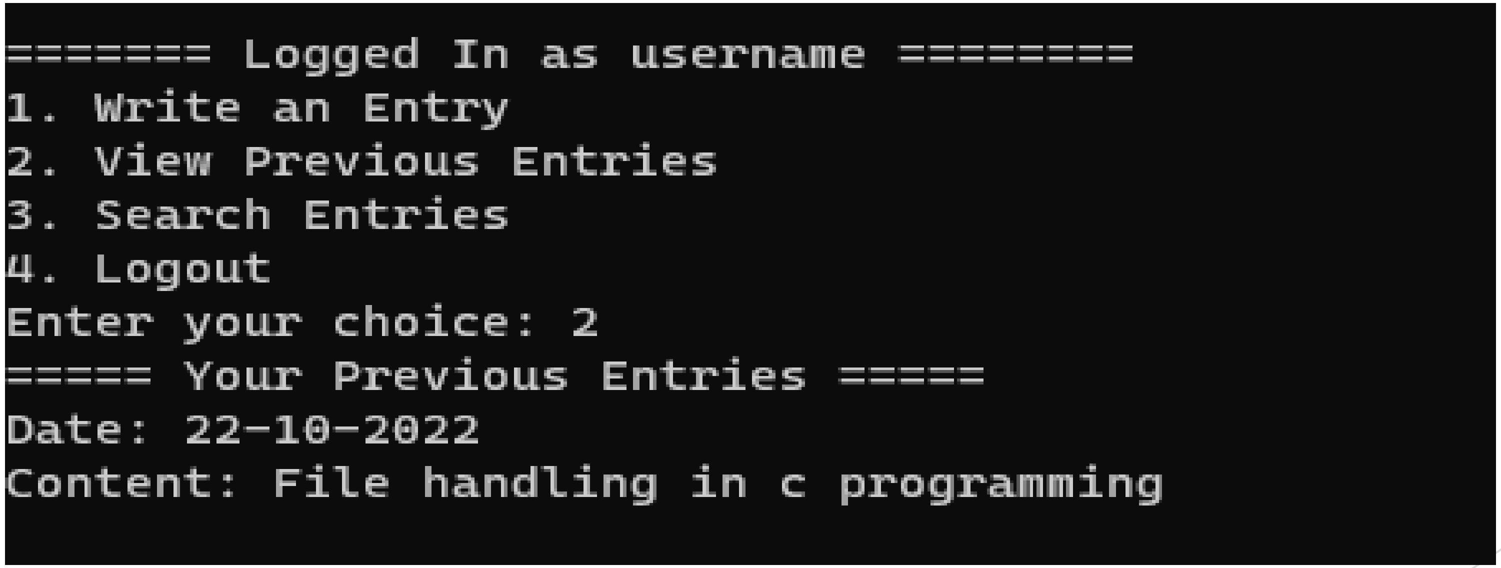
```
========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
Enter your choice: 2
Enter username: username
Enter password: password
Welcome, username!

======= Logged In as username ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 1
```

- If the user selects "Write an Entry", they are prompted to enter the date (in the format DD-MM-YYYY) and the content of the diary entry (up to 200 characters). The application encrypts the content using a simple encryption algorithm and then saves the entry to the "diary.txt" file along with the username and date



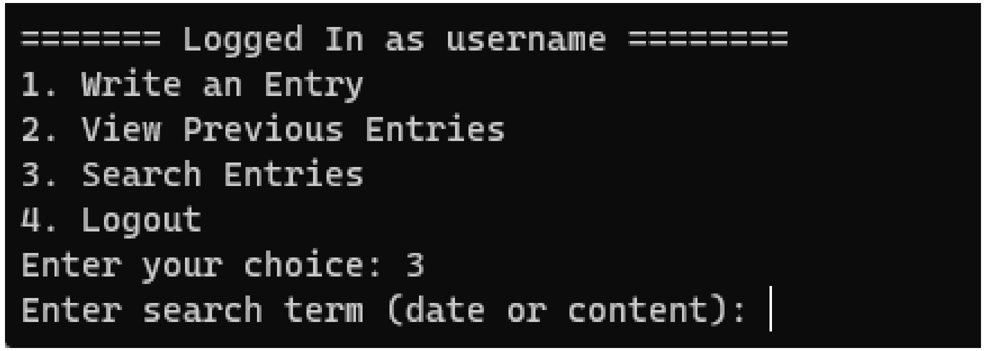
```
======= Logged In as username ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 1
Enter date (DD-MM-YYYY): 22-10-2022
Enter diary content (up to 200 characters): File handling in c programming
Entry written successfully!
```

- If the user selects "View Previous Entries", the application reads the "diary.txt" file and displays all previous diary entries associated with the logged-in user. The encrypted content is decrypted before displaying it to the user.

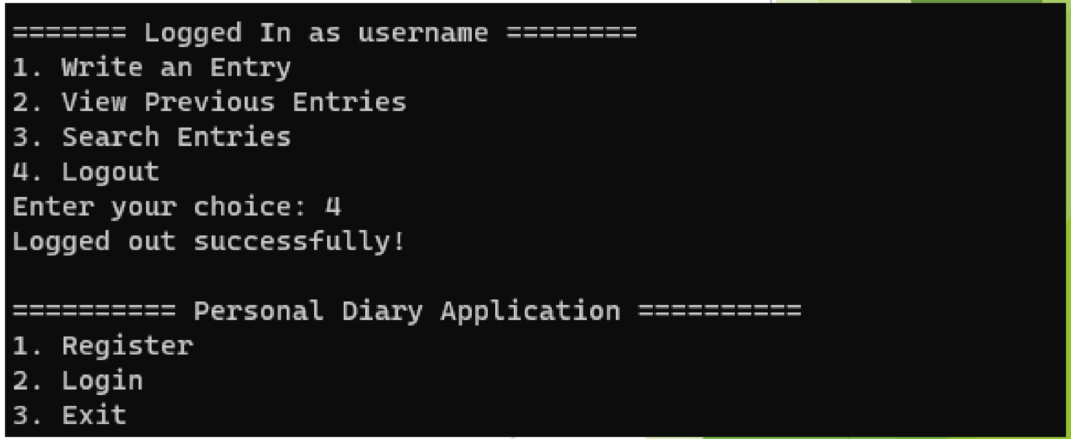


```
======== Logged In as username =========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 2
===== Your Previous Entries =====
Date: 22-10-2022
Content: File handling in c programming
```

- if the user selects "Search Entries", they can search for previous diary entries based on a search term (either the date or the content). The application reads the "diary.txt" file, decrypts the content, and checks for a match with the search term. If a match is found, the corresponding entry is displayed to the user.
- if the user selects "Logout", they are logged out of the current session, and the application returns to the main menu.



```
======= Logged In as username ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 3
Enter search term (date or content):
```



```
======= Logged In as username ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 4
Logged out successfully!

========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
```

Features of the code

- **Data Structures**: The application defines two data structures: struct DiaryEntry and struct User. struct DiaryEntry stores information about a diary entry, including the username, date, and content. struct User stores information about a registered user, including the username and hashed password.
- **Function Prototypes**: The application declares several function prototypes that will be defined later in the code. These function prototypes provide a structure to the code and allow functions to be used before their actual implementation.
- **Constants**: The application defines two constant arrays: usersFileName and diaryFileName. These arrays store the file names where user data and diary entries will be stored, respectively. encryptionKey is also defined as a constant, which is used to encrypt and decrypt diary content.

Code Execution and Mechanism

Functions

- 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

main() function and execution process

main() function

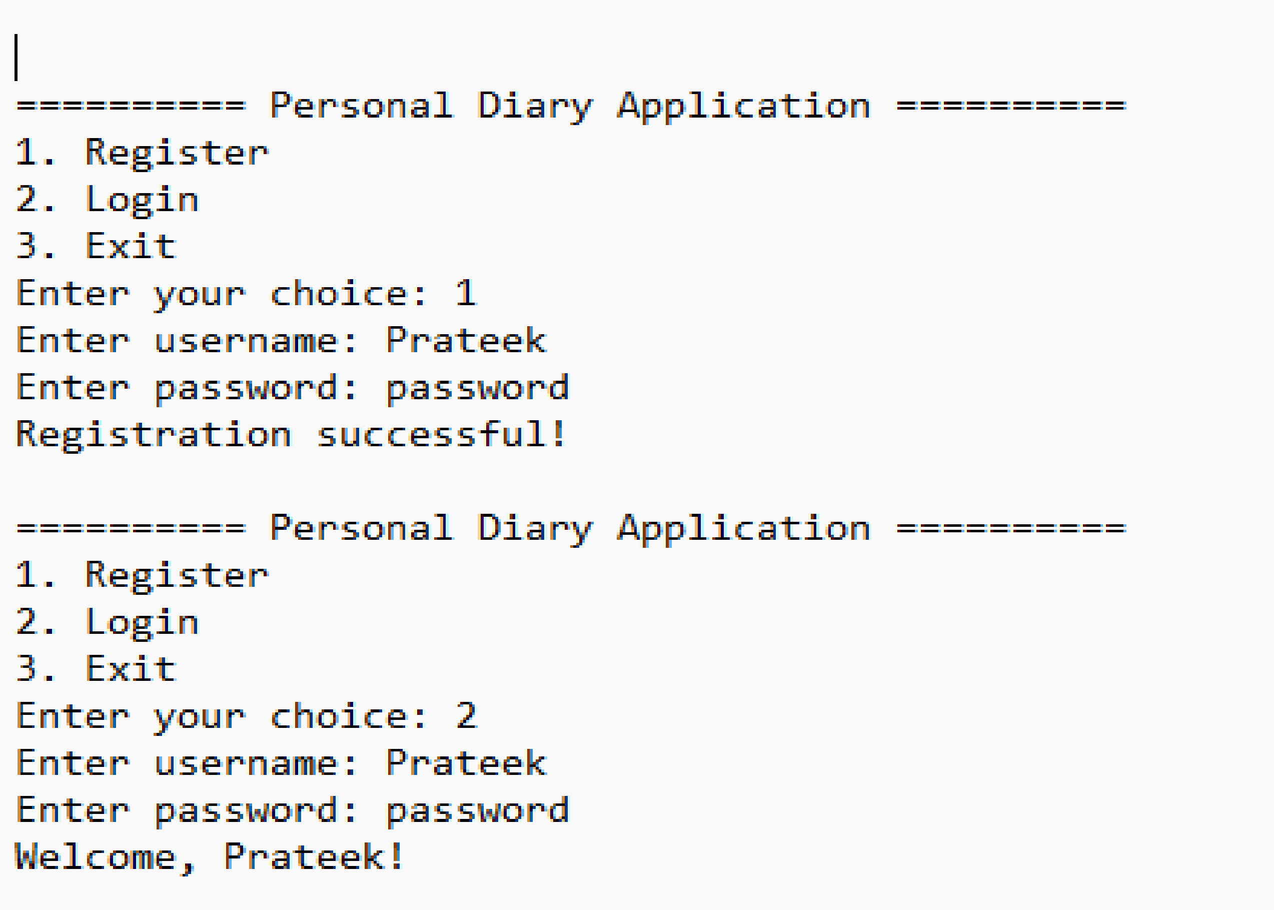
- **main() Function**: The main function is the entry point of the program. It displays the main menu options using displayMenu() and takes user input for the choice using fgets() and sscanf(). Depending on the user's choice, it calls relevant functions such as registerUser(), loginUser(), and displayUserMenu() and is where the flow of the program is defined.

Execution Process

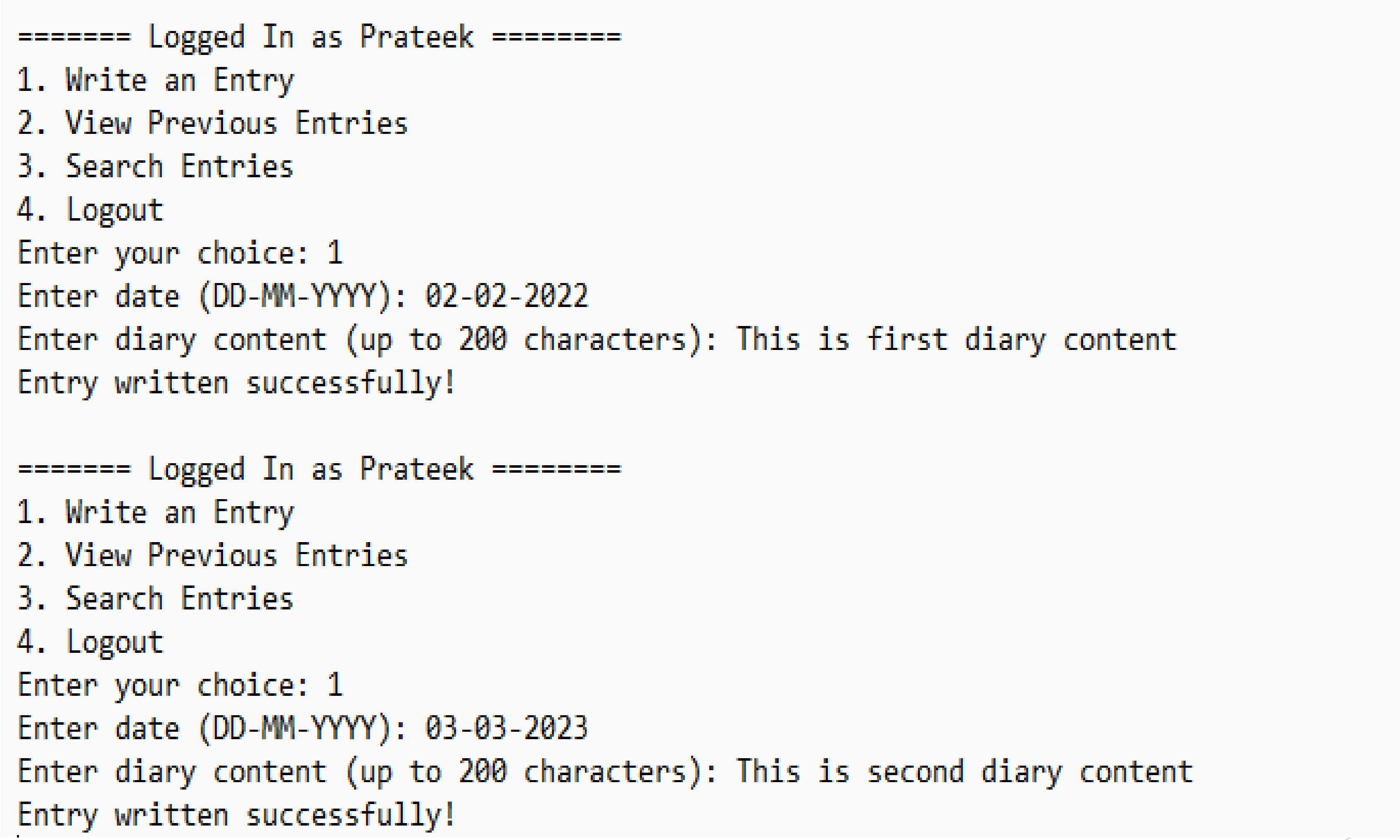
- 1. When the program starts, it enters the main loop, displaying the main menu and prompting the user for a choice.
- 2. Based on the user's choice, the appropriate function is called. If the user chooses to register, the `registerUser` function is called, allowing the user to create an account.
- 3. 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

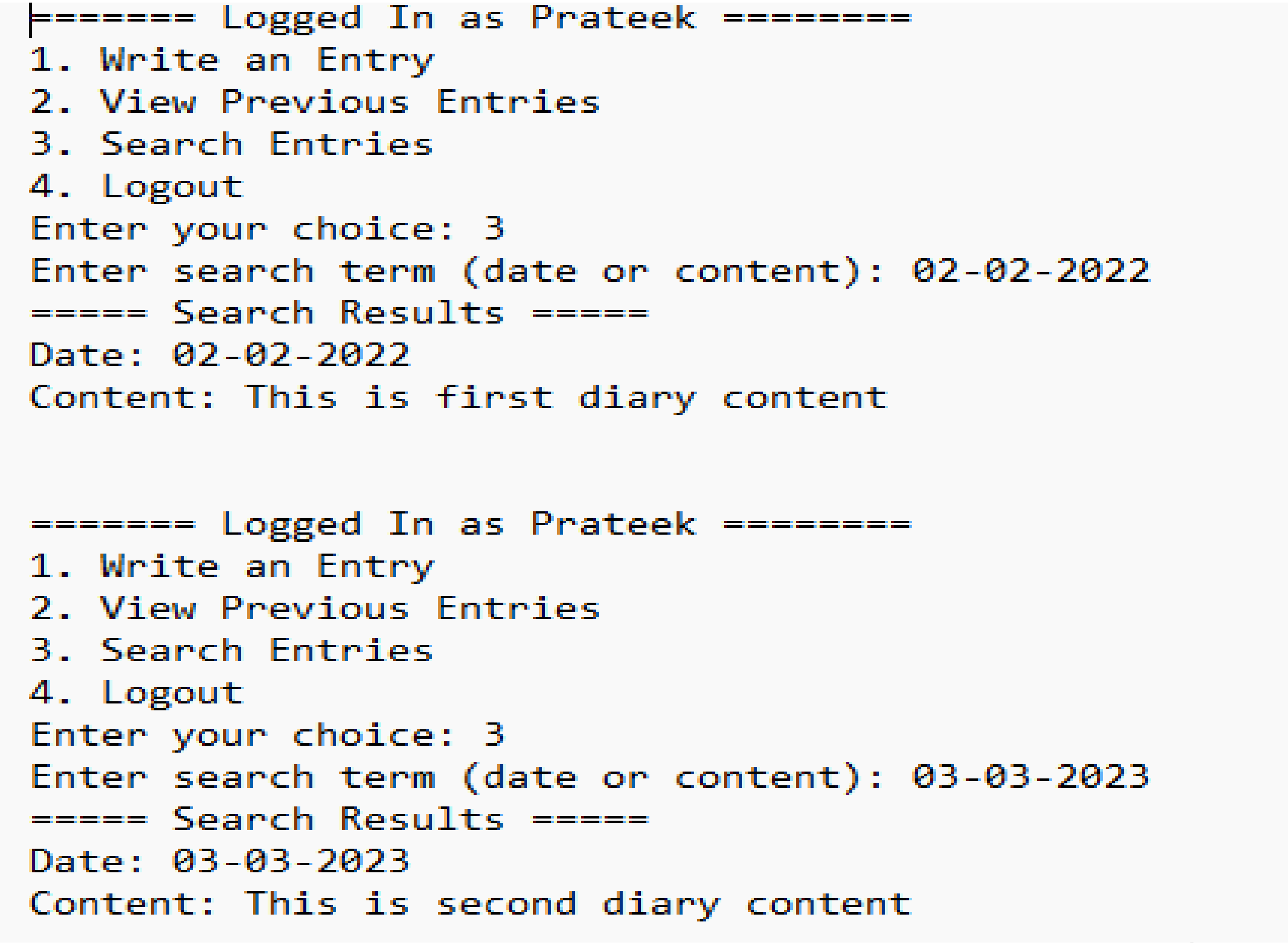
Console Screen:



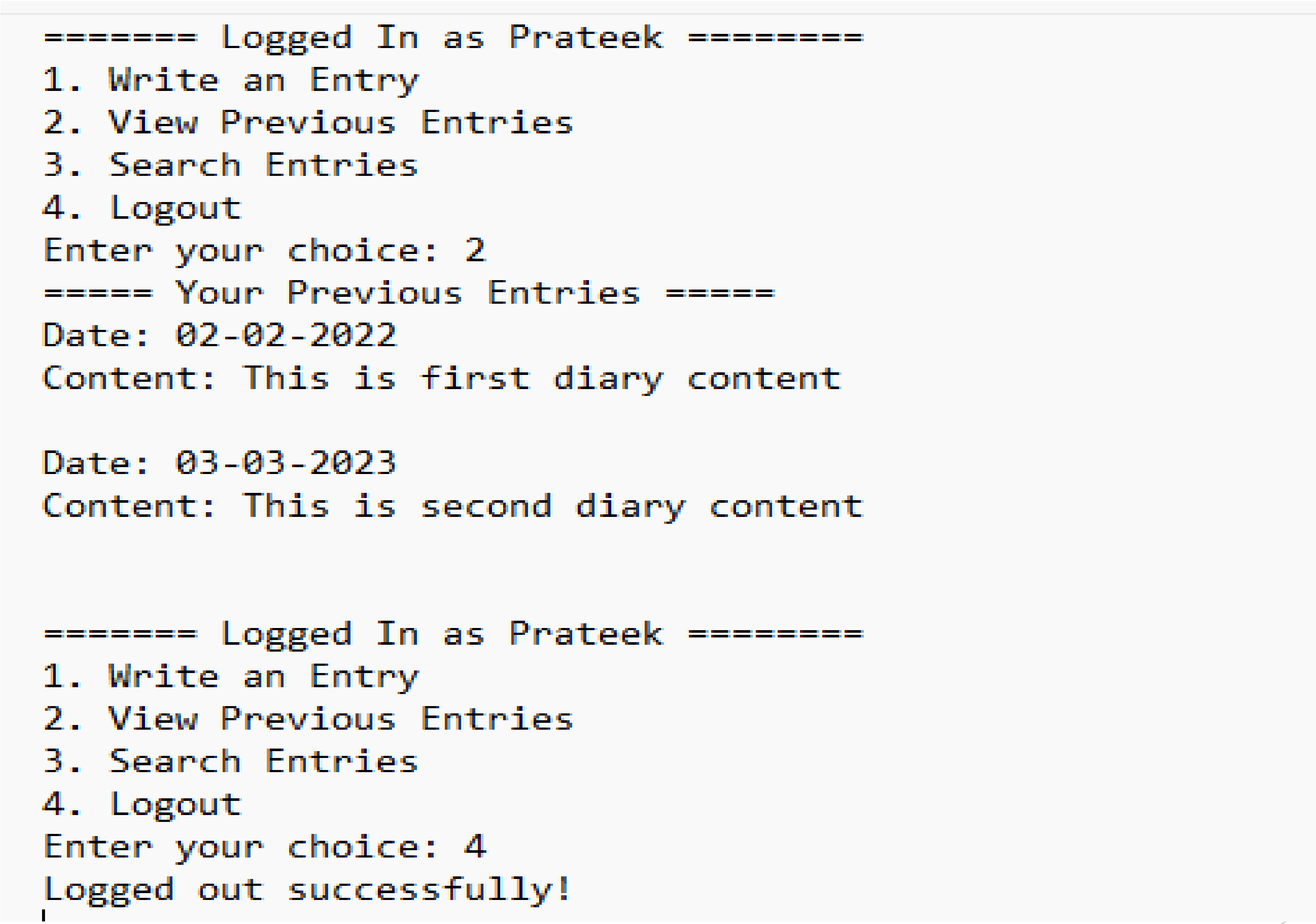
```
========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
Enter your choice: 1
Enter username: Prateek
Enter password: password
Registration successful!

========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
Enter your choice: 2
Enter username: Prateek
Enter password: password
Welcome, Prateek!
```



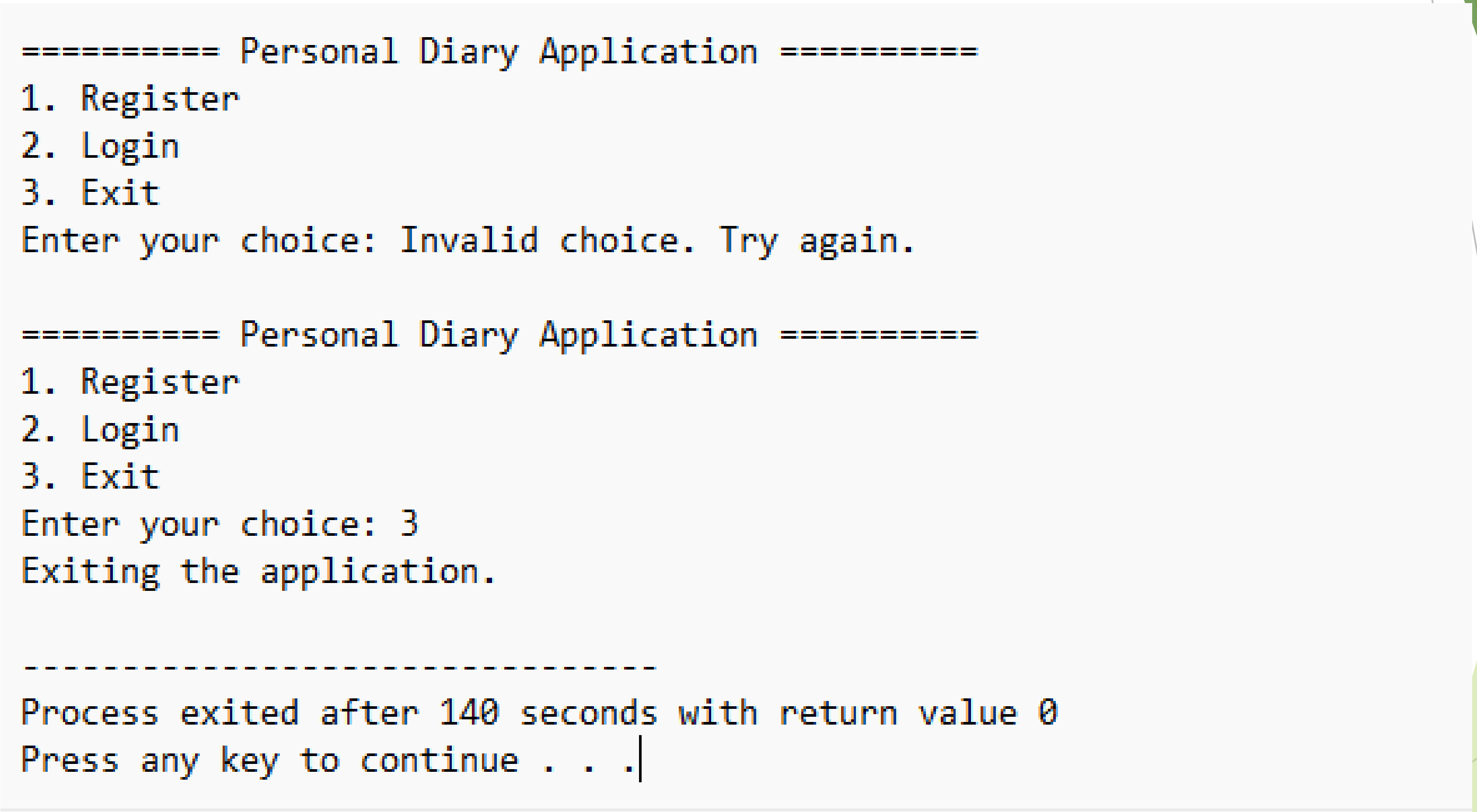
```
======= Logged In as Prateek ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 1
Enter date (DD-MM-YYYY): 02-02-2022
Enter diary content (up to 200 characters): This is first diary content
Entry written successfully!

======= Logged In as Prateek ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 1
Enter date (DD-MM-YYYY): 03-03-2023
Enter diary content (up to 200 characters): This is second diary content
Entry written successfully!
```



```
======= Logged In as Prateek ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 3
Enter search term (date or content): 02-02-2022
===== Search Results =====
Date: 02-02-2022
Content: This is first diary content


======= Logged In as Prateek ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 3
Enter search term (date or content): 03-03-2023
===== Search Results =====
Date: 03-03-2023
Content: This is second diary content
```



```
======= Logged In as Prateek ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 2
===== Your Previous Entries =====
Date: 02-02-2022
Content: This is first diary content

Date: 03-03-2023
Content: This is second diary content



======= Logged In as Prateek ========
1. Write an Entry
2. View Previous Entries
3. Search Entries
4. Logout
Enter your choice: 4
Logged out successfully!
```



```
========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
Enter your choice: Invalid choice. Try again.

========== Personal Diary Application ==========
1. Register
2. Login
3. Exit
Enter your choice: 3
Exiting the application.

--------------------------------
Process exited after 140 seconds with return value 0
Press any key to continue . . .
```

References

- C Programming – RD Bhatta
- The C programming Language- Brian Kernighan
- Lecture notes – Pravin Sangroula
- W3schools