**Introduction:**

The Basic Library Management System implemented in C provides a foundation for managing a library's administrative and user-related tasks. The system is designed to register and log in, manage books, and facilitate user registrations. This project serves as an illustrative example, demonstrating fundamental concepts in C programming, file handling, and basic data structure usage.

**Features:**

* **Admin Registration and Login:**
  + Admins can register with unique usernames and passwords, which are stored in the 'admins.txt' file.
  + The login functionality verifies the entered credentials against the stored admin data.
* **User Registration:**
  + Users can register with unique usernames and passwords, which are stored in the 'users.txt' file.
* **Book Management:**
  + Admins can add new books, delete existing books, issue books to users, and return books to the library.
  + The book information, including availability status and user ID for issued books, is stored in the 'books.txt' file
* **Display Functions:**
  + The system provides options to display available books, issued books with user information, and search for books based on the title.

**Implementation:**The project is organized into separate files for admin functions ('admin.c' and 'admin.h'), user functions ('user.c' and 'user.h'), book-related functions ('book.c' and 'book.h'), and the main program ('main.c'). A simple file-based approach is used for data storage, and each user, admin, and book is represented by a structure.

**Limitations:** The project has some limitations that should be considered for real-world scenarios. These include security concerns related to password storage, minimal error handling, and the absence of advanced features commonly found in fully-fledged library management system

Conclusion:

The Basic Library Management System in C provides a foundation for understanding key programming concepts, file handling, and data structures. It is suitable for educational purposes and as a starting point for developing more complex library management systems with enhanced security, scalability, and functionality. Developers are encouraged to build upon this project, addressing its limitations and incorporating additional features to meet specific requirements.

Code:

// admin.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "admin.h"

struct Admin {

    char username[50];

    char password[50];

};

void saveAdmin(struct Admin admin) {

    FILE \*file = fopen("admins.txt", "a");

    if (file != NULL) {

        fprintf(file, "%s %s\n", admin.username, admin.password);

        fclose(file);

    } else {

        printf("Error: Unable to save admin data.\n");

    }

}

void adminRegistration() {

    struct Admin admin;

    printf("Enter admin username: ");

    scanf("%s", admin.username);

    printf("Enter admin password: ");

    scanf("%s", admin.password);

    saveAdmin(admin);

    printf("Admin registration successful!\n");

}

int verifyAdmin(struct Admin inputAdmin) {

    struct Admin storedAdmin;

    FILE \*file = fopen("admins.txt", "r");

    if (file != NULL) {

        while (fscanf(file, "%s %s", storedAdmin.username, storedAdmin.password) != EOF) {

            if (strcmp(storedAdmin.username, inputAdmin.username) == 0 &&

                strcmp(storedAdmin.password, inputAdmin.password) == 0) {

                fclose(file);

                return 1; // Credentials match

            }

        }

        fclose(file);

    }

    return 0; // Credentials do not match

}

int adminLogin() {

    struct Admin admin;

    printf("Enter admin username: ");

    scanf("%s", admin.username);

    printf("Enter admin password: ");

    scanf("%s", admin.password);

    if (verifyAdmin(admin)) {

        printf("Admin login successful!\n");

        return 1;

    } else {

        printf("Invalid credentials. Admin login failed.\n");

        return 0;

    }

}

// admin.h

#ifndef ADMIN\_H

#define ADMIN\_H

void adminRegistration();

int adminLogin();

#endif

// book.c

#include <stdio.h>

#include <string.h>

#include "book.h"

struct Book library[100];

int bookCount = 0;

void saveBook(struct Book book) {

    FILE \*file = fopen("books.txt", "a");

    if (file != NULL) {

        fprintf(file, "%s %s %d %d\n", book.title, book.author, book.isAvailable, book.userID);

        fclose(file);

    } else {

        printf("Error: Unable to save book data.\n");

    }

}

void loadBooks() {

    FILE \*file = fopen("books.txt", "r");

    if (file != NULL) {

        while (fscanf(file, "%s %s %d %d", library[bookCount].title, library[bookCount].author,

                      &library[bookCount].isAvailable, &library[bookCount].userID) != EOF) {

            library[bookCount].bookID = bookCount;

            bookCount++;

        }

        fclose(file);

    }

}

// void addBook() {

//     struct Book newBook;

//     printf("Enter book title: ");

//     scanf("%s", newBook.title);

//     printf("Enter book author: ");

//     scanf("%s", newBook.author);

//     newBook.isAvailable = 1;

//     newBook.userID = -1;

//      newBook.bookID = bookCount;

//     library[bookCount++] = newBook;

//     saveBook(newBook);

//     printf("Book added successfully!\n");

// }

void addBook() {

    struct Book newBook;

    int numCopies;

    printf("Enter book title: ");

    scanf("%s", newBook.title);

    printf("Enter book author: ");

    scanf("%s", newBook.author);

    printf("Enter the number of copies: ");

    scanf("%d", &numCopies);

    for (int i = 0; i < numCopies; ++i) {

        newBook.isAvailable = 1;

        newBook.userID = -1;

        newBook.bookID = bookCount;  // Assign a unique book ID

        library[bookCount++] = newBook;

        saveBook(newBook);

    }

    printf("%d copies of the book added successfully!\n", numCopies);}

void deleteBook() {

    int bookID;

    printf("Enter book ID to delete: ");

    scanf("%d", &bookID);

    if (bookID >= 0 && bookID < bookCount) {

        // Move all subsequent books one position back

        for (int i = bookID; i < bookCount - 1; ++i) {

            library[i] = library[i + 1];

        }

        bookCount--;

        // Update the file with the modified book list

        FILE \*file = fopen("books.txt", "w");

        if (file != NULL) {

            for (int i = 0; i < bookCount; ++i) {

                fprintf(file, "%s %s %d %d\n", library[i].title, library[i].author,

                        library[i].isAvailable, library[i].userID);

            }

            fclose(file);

        } else {

            printf("Error: Unable to update book data.\n");

        }

        printf("Book deleted successfully!\n");

    } else {

        printf("Invalid book ID.\n");

    }

}

void issueBook() {

    int bookID, userID;

    printf("Enter book ID to issue: ");

    scanf("%d", &bookID);

    if (bookID >= 0 && bookID < bookCount) {

        if (library[bookID].isAvailable) {

            printf("Enter user ID: ");

            scanf("%d", &userID);

            library[bookID].isAvailable = 0;

            library[bookID].userID = userID;

            // Update the file with the modified book list

            FILE \*file = fopen("books.txt", "w");

            if (file != NULL) {

                for (int i = 0; i < bookCount; ++i) {

                    fprintf(file, "%s %s %d %d\n", library[i].title, library[i].author,

                            library[i].isAvailable, library[i].userID);

                }

                fclose(file);

            } else {

                printf("Error: Unable to update book data.\n");

            }

            printf("Book issued successfully!\n");

        } else {

            printf("Book is not available for issuing.\n");

        }

    } else {

        printf("Invalid book ID.\n");

    }

}

void returnBook() {

    int bookID;

    printf("Enter book ID to return: ");

    scanf("%d", &bookID);

    if (bookID >= 0 && bookID < bookCount) {

        if (!library[bookID].isAvailable) {

            library[bookID].isAvailable = 1;

            library[bookID].userID = -1;

            // Update the file with the modified book list

            FILE \*file = fopen("books.txt", "w");

            if (file != NULL) {

                for (int i = 0; i < bookCount; ++i) {

                    fprintf(file, "%s %s %d %d\n", library[i].title, library[i].author,

                            library[i].isAvailable, library[i].userID);

                }

                fclose(file);

            } else {

                printf("Error: Unable to update book data.\n");

            }

            printf("Book returned successfully!\n");

        } else {

            printf("Book is already available.\n");

        }

    } else {

        printf("Invalid book ID.\n");

    }

}

void displayBookAvailability() {

    printf("Available Books:\n");

    for (int i = 0; i < bookCount; ++i) {

        if (library[i].isAvailable) {

            printf("ID: %d, Title: %s, Author: %s\n", library[i].bookID, library[i].title, library[i].author);

        }

    }

}

void displayIssuedBooks() {

    printf("Issued Books:\n");

    for (int i = 0; i < bookCount; ++i) {

        if (!library[i].isAvailable) {

            printf("ID: %d, Title: %s, Author: %s, Issued to User ID: %d\n", library[i].bookID,

                   library[i].title, library[i].author, library[i].userID);

        }

    }

}

void searchBook() {

    char searchTitle[100];

    printf("Enter book title to search: ");

    scanf("%s", searchTitle);

    printf("Search Results:\n");

    for (int i = 0; i < bookCount; ++i) {

        if (strstr(library[i].title, searchTitle) != NULL) {

            printf("ID: %d, Title: %s, Author: %s, Available: %s\n", library[i].bookID,

                   library[i].title, library[i].author, library[i].isAvailable ? "Yes" : "No");

        }

    }

}

// book.h

#ifndef BOOK\_H

#define BOOK\_H

struct Book {

    int bookID;

    char title[100];

    char author[100];

    int isAvailable;

    int userID;

};

void addBook();

void deleteBook();

void issueBook();

void returnBook();

void displayBookAvailability();

void displayIssuedBooks();

void searchBook();

void loadBooks();

#endif

// main.c

#include <stdio.h>

#include "admin.h"

#include "user.h"

#include "book.h"

int main() {

    int choice;

    int adminLoggedIn = 0;

    loadBooks();

    do {

        if(adminLoggedIn){

        printf("\nLibrary Management System Menu:\n");

        printf("1. Admin Logout\n");

        printf("2. User Registration\n");

        printf("3. User Login\n");

        printf("4. Add Book\n");

        printf("5. Delete Book\n");

        printf("6. Issue Book\n");

        printf("7. Return Book\n");

        printf("8. Display Available Books\n");

        printf("9. Display Issued Books\n");

        printf("10. Search Book\n");

        printf("0. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                adminLoggedIn = 0; // Admin Logout

                    printf("Admin Logout successful!\n");

                break;

            case 2:

                userRegistration();

                break;

            case 3:

                userLogin();

                break;

            case 4:

                addBook();

                break;

            case 5:

                deleteBook();

                break;

            case 6:

                issueBook();

                break;

            case 7:

                returnBook();

                break;

            case 8:

                displayBookAvailability();

                break;

            case 9:

                displayIssuedBooks();

                break;

            case 10:

                searchBook();

                break;

            case 0:

                printf("Exiting Library Management System. Goodbye!\n");

                break;

            default:

                printf("Invalid choice. Please try again.\n");

                break;

        }

        } else {

              printf("\nAdmin Login Menu:\n");

            printf("1. Admin registration\n");

            printf("2. Admin Login\n");

            printf("0. Exit\n");

            printf("Enter your choice: ");

            scanf("%d", &choice);

            switch (choice) {

                case 1:

                    adminRegistration();

                    break;

                case 2:

                    adminLoggedIn = adminLogin();

                    break;

                case 0:

                    printf("Exiting Library Management System. Goodbye!\n");

                    break;

                default:

                    printf("Invalid choice. Please try again.\n");

                    break;

        }

        }

    } while (choice != 0);

    return 0;

}

// user.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "user.h"

struct User {

    char username[50];

    char password[50];

};

void saveUser(struct User user) {

    FILE \*file = fopen("users.txt", "a");

    if (file != NULL) {

        fprintf(file, "%s %s\n", user.username, user.password);

        fclose(file);

    } else {

        printf("Error: Unable to save user data.\n");

    }

}

void userRegistration() {

    struct User user;

    printf("Enter user username: ");

    scanf("%s", user.username);

    printf("Enter user password: ");

    scanf("%s", user.password);

    saveUser(user);

    printf("User registration successful!\n");

}

int verifyUser(struct User inputUser) {

    struct User storedUser;

    FILE \*file = fopen("users.txt", "r");

    if (file != NULL) {

        while (fscanf(file, "%s %s", storedUser.username, storedUser.password) != EOF) {

            if (strcmp(storedUser.username, inputUser.username) == 0 &&

                strcmp(storedUser.password, inputUser.password) == 0) {

                fclose(file);

                return 1; // Credentials match

            }

        }

        fclose(file);

    }

    return 0; // Credentials do not match

}

int userLogin() {

    struct User user;

    printf("Enter user username: ");

    scanf("%s", user.username);

    printf("Enter user password: ");

    scanf("%s", user.password);

    if (verifyUser(user)) {

        printf("User login successful!\n");

        return 1;

    } else {

        printf("Invalid credentials. User login failed.\n");

        return 0;

    }

}

// user.h

#ifndef USER\_H

#define USER\_H

void userRegistration();

int userLogin();

#endif

**Output:**

