#include <stdio.h>

#include <stdlib.h>

#include <string.h>

// Define a structure for a book (a node in the linked list)

struct Book {

int id;

char title[100];

char author[100];

struct Book\* next;

};

// Function to create a new book node

struct Book\* createBook(int id, char title[], char author[]) {

struct Book\* newBook = (struct Book\*)malloc(sizeof(struct Book));

newBook->id = id;

strcpy(newBook->title, title);

strcpy(newBook->author, author);

newBook->next = NULL;

return newBook;

}

// Function to add a book to the list

void addBook(struct Book\*\* head, int id, char title[], char author[]) {

struct Book\* newBook = createBook(id, title, author);

newBook->next = \*head;

\*head = newBook;

printf("Book added: %s by %s\n", title, author);

}

// Function to delete a book by its ID

void deleteBook(struct Book\*\* head, int id) {

struct Book\* temp = \*head;

struct Book\* prev = NULL;

// If head node itself holds the book to be deleted

if (temp != NULL && temp->id == id) {

\*head = temp->next;

free(temp);

printf("Book with ID %d deleted.\n", id);

return;

}

// Search for the book to be deleted

while (temp != NULL && temp->id != id) {

prev = temp;

temp = temp->next;

}

// If book not found

if (temp == NULL) {

printf("Book with ID %d not found.\n", id);

return;

}

// Unlink the book from the linked list

prev->next = temp->next;

free(temp);

printf("Book with ID %d deleted.\n", id);

}

// Function to search for a book by title or author

void searchBook(struct Book\* head, char searchTerm[]) {

struct Book\* temp = head;

int found = 0;

while (temp != NULL) {

if (strstr(temp->title, searchTerm) || strstr(temp->author, searchTerm)) {

printf("Book found: %s by %s (ID: %d)\n", temp->title, temp->author, temp->id);

found = 1;

}

temp = temp->next;

}

if (!found) {

printf("No book found with the search term: %s\n", searchTerm);

}

}

// Function to display all books in the list

void displayBooks(struct Book\* head) {

struct Book\* temp = head;

if (head == NULL) {

printf("No books in the library.\n");

return;

}

while (temp != NULL) {

printf("ID: %d, Title: %s, Author: %s\n", temp->id, temp->title, temp->author);

temp = temp->next;

}

}

int main() {

struct Book\* head = NULL;

int choice, id;

char title[100], author[100], searchTerm[100];

while (1) {

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

printf("1. Add Book\n");

printf("2. Delete Book\n");

printf("3. Search Book\n");

printf("4. Display All Books\n");

printf("5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter book ID: ");

scanf("%d", &id);

printf("Enter book title: ");

scanf(" %[^\n]", title); // This allows spaces in input

printf("Enter book author: ");

scanf(" %[^\n]", author);

addBook(&head, id, title, author);

break;

case 2:

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

scanf("%d", &id);

deleteBook(&head, id);

break;

case 3:

printf("Enter title or author to search: ");

scanf(" %[^\n]", searchTerm);

searchBook(head, searchTerm);

break;

case 4:

displayBooks(head);

break;

case 5:

printf("Exiting the program.\n");

exit(0);

default:

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

}

}

return 0;

}

**Features:**

1. **Add new books**: You can add books by providing an ID, title, and author.
2. **Delete books**: You can delete a book by its ID.
3. **Search books**: You can search for a book by title or author.
4. **Display all books**: Lists all the books currently in the library.

**How it works:**

* Books are represented as nodes in a **singly linked list**.
* The linked list is dynamically updated as books are added or deleted.
* Searching checks the book titles and authors for the given term.

This is a simple, easy-to-understand project that demonstrates basic linked list operations.