**Simple Library Management System**

**Name:**

**Email:**

**Code:**

#include <iostream>

#include <string>

#include <vector>

class Book {

private:

int id;

std::string title;

std::string author;

bool available;

public:

Book(int id, const std::string& title, const std::string& author)

: id(id), title(title), author(author), available(true) {

}

int getID() const {

return id;

}

std::string getTitle() const {

return title;

}

std::string getAuthor() const {

return author;

}

bool isAvailable() const {

return available;

}

void setAvailable(bool status) {

available = status;

}

};

class Library {

private:

std::vector<Book> books;

public:

void addNewBook(const Book& book) {

books.push\_back(book);

}

Book\* searchBookById(int id) {

for (auto& book : books) {

if (book.getID() == id) {

return &book;

}

}

return nullptr;

}

Book\* searchBookByTitle(const std::string& title) {

for (auto& book : books) {

if (book.getTitle() == title) {

return &book;

}

}

return nullptr;

}

void issueBook(int bookId, int memberId) {

Book\* book = searchBookById(bookId);

if (book != nullptr && book->isAvailable()) {

std::cout << "Book details:" << std::endl;

std::cout << "ID: " << book->getID() << ", Title: " << book->getTitle() << ", Author: " << book->getAuthor() << std::endl;

book->setAvailable(false);

std::cout << "Book with ID " << bookId << " issued to member with ID " << memberId << "." << std::endl;

} else {

std::cout << "Book with ID " << bookId << " not found or already issued." << std::endl;

}

}

void returnBook(int bookId) {

Book\* book = searchBookById(bookId);

if (book != nullptr && !book->isAvailable()) {

book->setAvailable(true);

std::cout << "Book with ID " << bookId << " returned." << std::endl;

} else {

std::cout << "Book with ID " << bookId << " not found or already available." << std::endl;

}

}

void listAllBooks() {

if (books.empty()) {

std::cout << "No books available in the library." << std::endl;

} else {

for (const auto& book : books) {

std::cout << "ID: " << book.getID() << ", Title: " << book.getTitle() << ", Author: " << book.getAuthor();

if (book.isAvailable()) {

std::cout << ", Status: Available" << std::endl;

} else {

std::cout << ", Status: Issued" << std::endl;

}

}

}

}

void deleteBook(int bookId) {

for (auto it = books.begin(); it != books.end(); ++it) {

if (it->getID() == bookId) {

books.erase(it);

std::cout << "Book with ID " << bookId << " deleted." << std::endl;

return;

}

}

std::cout << "Book with ID " << bookId << " not found." << std::endl;

}

};

int main() {

Library library;

// Adding new books

library.addNewBook(Book(1, "The Great Gatsby", "F. Scott Fitzgerald"));

library.addNewBook(Book(2, "To Kill a Mockingbird", "Harper Lee"));

library.addNewBook(Book(3, "The Lord of the Rings", "J. R. R. Tolkien"));

library.addNewBook(Book(4, "War and Peace", "Leo Tolstoy"));

library.addNewBook(Book(5, "Anna Karenina", "Leo Tolstoy"));

int choice;

do {

std::cout << "\nLibrary Management System Menu:" << std::endl;

std::cout << "1. Add a New Book" << std::endl;

std::cout << "2. Issue a Book" << std::endl;

std::cout << "3. Return a Book" << std::endl;

std::cout << "4. Search for a Book" << std::endl;

std::cout << "5. List All Books" << std::endl;

std::cout << "6. Delete a Book" << std::endl;

std::cout << "0. Exit" << std::endl;

std::cout << "Enter your choice: ";

std::cin >> choice;

switch (choice) {

case 1: {

int id;

std::string title, author;

std::cout << "Enter the ID of the book: ";

std::cin >> id;

std::cout << "Enter the title of the book: ";

std::cin.ignore();

std::getline(std::cin, title);

std::cout << "Enter the author of the book: ";

std::getline(std::cin, author);

library.addNewBook(Book(id, title, author));

std::cout << "Book added successfully." << std::endl;

break;

}

case 2: {

int bookId, memberId;

std::cout << "Enter the ID of the book to issue: ";

std::cin >> bookId;

std::cout << "Enter the ID of the member: ";

std::cin >> memberId;

library.issueBook(bookId, memberId);

break;

}

case 3: {

int bookId;

std::cout << "Enter the ID of the book to return: ";

std::cin >> bookId;

library.returnBook(bookId);

break;

}

case 4: {

int searchChoice;

std::cout << "Search by:" << std::endl;

std::cout << "1. Title" << std::endl;

std::cout << "2. ID" << std::endl;

std::cout << "Enter your choice: ";

std::cin >> searchChoice;

if (searchChoice == 1) {

std::string title;

std::cout << "Enter the title of the book to search: ";

std::cin.ignore();

std::getline(std::cin, title);

Book\* book = library.searchBookByTitle(title);

if (book != nullptr) {

std::cout << "Book found. Title: " << book->getTitle() << ", Author: " << book->getAuthor() << std::endl;

} else {

std::cout << "Book not found." << std::endl;

}

} else if (searchChoice == 2) {

int id;

std::cout << "Enter the ID of the book to search: ";

std::cin >> id;

Book\* book = library.searchBookById(id);

if (book != nullptr) {

std::cout << "Book found. Title: " << book->getTitle() << ", Author: " << book->getAuthor() << std::endl;

} else {

std::cout << "Book not found." << std::endl;

}

} else {

std::cout << "Invalid choice." << std::endl;

}

break;

}

case 5:

library.listAllBooks();

break;

case 6: {

int id;

std::cout << "Enter the ID of the book to delete: ";

std::cin >> id;

library.deleteBook(id);

break;

}

case 0:

std::cout << "Exiting the program." << std::endl;

break;

default:

std::cout << "Invalid choice. Please try again." << std::endl;

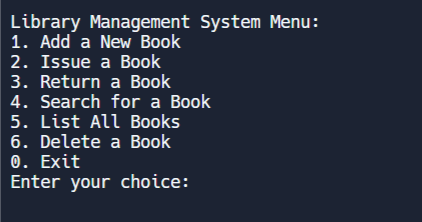
}

} while (choice != 0);

return 0;

}

**Output :**



All Menu

A screen shot of a computer

Description automatically generated

Add New Book to Library

A screenshot of a computer program

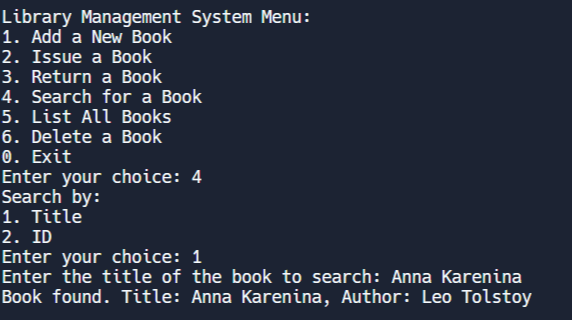
Description automatically generated

Issue a Book

A screen shot of a computer

Description automatically generated

Return a Book

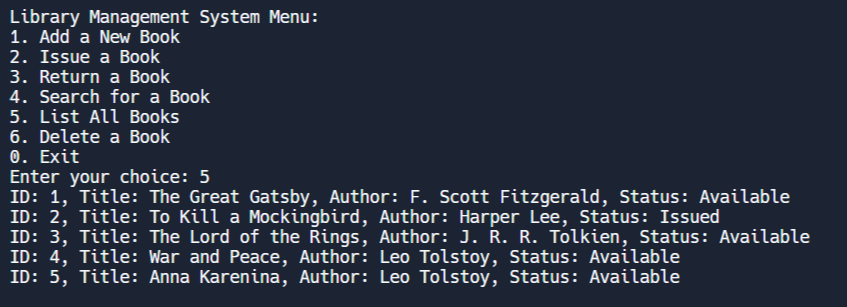


Search for a Book using the Title

A screenshot of a computer

Description automatically generated

Search for a Book using the ID



List All Books

A screenshot of a computer

Description automatically generated

Delete a Book

**High-level algorithm for the Library Management System code:**

1. Start the program.
2. Create an empty Library object.
3. Display the menu options to the user.
4. Read the user's choice from the input.
5. Execute the corresponding functionality based on the user's choice.
6. If the user chooses to add a new book:
   * Read the book details (ID, title, author) from the input.
   * Create a Book object with the provided details.
   * Call the **addNewBook()** function of the Library object, passing the Book object.
   * Display a success message.
7. If the user chooses to issue a book:
   * Read the book ID and member ID from the input.
   * Call the **issueBook()** function of the Library object, passing the book ID and member ID.
   * If the book is available:
     + Display the book details (ID, title, author).
     + Mark the book as issued.
     + Display a success message.
   * If the book is not available or not found:
     + Display an appropriate error message.
8. If the user chooses to return a book:
   * Read the book ID from the input.
   * Call the **returnBook()** function of the Library object, passing the book ID.
   * If the book is issued:
     + Mark the book as available.
     + Display a success message.
   * If the book is not issued or not found:
     + Display an appropriate error message.
9. If the user chooses to search for a book:
   * Display the search options (by title or ID) to the user.
   * Read the search choice from the input.
   * If the user chooses to search by title:
     + Read the book title from the input.
     + Call the **searchBookByTitle()** function of the Library object, passing the title.
     + If the book is found:
       - Display the book details (ID, title, author).
     + If the book is not found:
       - Display an appropriate message.
   * If the user chooses to search by ID:
     + Read the book ID from the input.
     + Call the **searchBookById()** function of the Library object, passing the ID.
     + If the book is found:
       - Display the book details (ID, title, author).
     + If the book is not found:
       - Display an appropriate message.
   * If the user enters an invalid choice:
     + Display an appropriate error message.
10. If the user chooses to list all books:
    * Call the **listAllBooks()** function of the Library object.
    * If there are no books:
      + Display a message indicating no books are available.
    * If there are books:
      + Iterate through each book in the Library object.
      + Display the book details (ID, title, author, status: available or issued).
11. If the user chooses to delete a book:
    * Read the book ID from the input.
    * Call the **deleteBook()** function of the Library object, passing the book ID.
    * If the book is found:
      + Remove the book from the Library object.
      + Display a success message.
    * If the book is not found:
      + Display an appropriate error message.
12. Repeat steps 3-11 until the user chooses to exit (enter 0).
13. End the program.