BOARD

Board Infinity's LPU Summer Training Program

Data Structure and Algorithms

Name: Vivek Baghel

Program: Computer Science and Engineering

Training Program name: DSA in C++

University: Lovely Professional University

Project Topic: Library Management System

Project Code:

```
#include <iostream>

#include <string>

#include <stdexcept>

using namespace std;

class Book {
public:
    int id;
    string title;
    string author;
    bool isIssued;
    string issuedTo;

    Book(int id, string title, string author) : id(id), title(title),
author(author), isIssued(false), issuedTo("") {}
};

class Node {
public:
    Book *book;
    Node *next;

    Node(Book *book) : book(book), next(nullptr) {}
};

class BookList {
private:
    Node *head;

public:
    BookList() : head(nullptr) {}

    void addBook(Book *book) {
        Node *newNode = new Node(book);
        newNode->next = head;
        head = newNode;
    }

    void deleteBook(int id) {
        Node *current = head;
        Node *previous = nullptr;

        while (current != nullptr && current->book->id != id) {
```

```
            previous = current;
            current = current->next;
        }

        if (current == nullptr) {
            throw runtime_error("Book with ID " + to_string(id) + " not
found.");
        }

        if (previous == nullptr) {
            head = current->next;
        } else {
            previous->next = current->next;
        }

        delete current->book;
        delete current;
        cout << "Book with ID " << id << " deleted." << endl;
    }

    Book* searchBookById(int id) {
        Node *current = head;
        while (current != nullptr) {
            if (current->book->id == id) {
                return current->book;
            }
            current = current->next;
        }
        throw runtime_error("Book with ID " + to_string(id) + " not found.");
    }

    Book* searchBookByTitle(string title) {
        Node *current = head;
        while (current != nullptr) {
            if (current->book->title == title) {
                return current->book;
            }
            current = current->next;
        }
        throw runtime_error("Book with title '" + title + "' not found.");
    }

    Book* searchBookByAuthor(string author) {
        Node *current = head;
        while (current != nullptr) {
            if (current->book->author == author) {
                return current->book;
            }
```

```
            current = current->next;
        }
        throw runtime_error("Book with author '" + author + "' not found.");
    }

    void listAllBooks() {
        Node *current = head;
        if (!current) {
            cout << "No books in the library." << endl;
            return;
        }
        while (current != nullptr) {
            cout << "ID: " << current->book->id << ", Title: " << current-
>book->title << ", Author: " << current->book->author << ", Status: " <<
(current->book->isIssued ? "Issued to " + current->book->issuedTo :
"Available") << endl;
            current = current->next;
        }
    }

    void updateBook(int id, string newTitle, string newAuthor) {
        Book *book = searchBookById(id);
        book->title = newTitle;
        book->author = newAuthor;
        cout << "Book with ID " << id << " updated." << endl;
    }

    void sortBooksById() {
        if (head == nullptr || head->next == nullptr) {
            return;
        }

        Node *sorted = nullptr;
        Node *current = head;

        while (current != nullptr) {
            Node *next = current->next;
            if (sorted == nullptr || sorted->book->id >= current->book->id) {
                current->next = sorted;
                sorted = current;
            } else {
                Node *temp = sorted;
                while (temp->next != nullptr && temp->next->book->id <
current->book->id) {
                    temp = temp->next;
                }
                current->next = temp->next;
                temp->next = current;
```

```
            }
            current = next;
        }

        head = sorted;
    }
};

class IssuedBooksStack {
private:
    Book **stack;
    int top;
    int capacity;

public:
    IssuedBooksStack(int capacity) : capacity(capacity), top(-1) {
        stack = new Book*[capacity];
    }

    ~IssuedBooksStack() {
        delete[] stack;
    }

    void push(Book *book) {
        if (top == capacity - 1) {
            throw runtime_error("Stack overflow. Cannot issue more books.");
        }
        stack[++top] = book;
    }

    Book* pop() {
        if (top == -1) {
            throw runtime_error("Stack underflow. No issued books to
return.");
        }
        return stack[top--];
    }

    bool isEmpty() {
        return top == -1;
    }
};

class Library {
private:
    BookList bookList;
    IssuedBooksStack issuedBooksStack;
```

BOARD

```
public:
    Library(int stackCapacity) : issuedBooksStack(stackCapacity) {}

    void addNewBook(int id, string title, string author) {
        try {
            Book *newBook = new Book(id, title, author);
            bookList.addBook(newBook);
            cout << "Book added: " << title << " by " << author << endl;
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
    }

    void searchBook(string title) {
        try {
            Book *book = bookList.searchBookByTitle(title);
            cout << "Book found: " << endl;
            cout << "ID: " << book->id << ", Title: " << book->title << ",
Author: " << book->author << ", Status: " << (book->isIssued ? "Issued to " +
book->issuedTo : "Available") << endl;
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
    }

    void searchBook(int id) {
        try {
            Book *book = bookList.searchBookById(id);
            cout << "Book found: " << endl;
            cout << "ID: " << book->id << ", Title: " << book->title << ",
Author: " << book->author << ", Status: " << (book->isIssued ? "Issued to " +
book->issuedTo : "Available") << endl;
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
    }

    void issueBook(int id, string studentName) {
        try {
            Book *book = bookList.searchBookById(id);
            if (book->isIssued) {
                throw runtime_error("Book with ID " + to_string(id) + " is
already issued.");
            }
            book->isIssued = true;
            book->issuedTo = studentName;
            issuedBooksStack.push(book);
```

```
            cout << "Book with ID " << id << " issued to " << studentName <<
endl;
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
    }

    void returnBook() {
        try {
            Book *book = issuedBooksStack.pop();
            book->isIssued = false;
            book->issuedTo = "";
            cout << "Book with ID " << book->id << " returned." << endl;
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
    }

    void listAllBooks() {
        bookList.listAllBooks();
    }

    void deleteBook(int id) {
        try {
            bookList.deleteBook(id);
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
    }

    void updateBook(int id, string newTitle, string newAuthor) {
        try {
            bookList.updateBook(id, newTitle, newAuthor);
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
    }

    void searchBookByAuthor(string author) {
        try {
            Book *book = bookList.searchBookByAuthor(author);
            cout << "Book found: " << endl;
            cout << "ID: " << book->id << ", Title: " << book->title << ",
Author: " << book->author << ", Status: " << (book->isIssued ? "Issued to " +
book->issuedTo : "Available") << endl;
        } catch (const exception &e) {
            cerr << e.what() << endl;
        }
```

```
    }

    void sortBooksById() {
        bookList.sortBooksById();
        cout << "Books sorted by ID." << endl;
    }
};

void displayMenu() {
    cout << "\nLibrary Management System\n";
    cout << "1. Add New Book\n";
    cout << "2. Search Book by Title\n";
    cout << "3. Search Book by ID\n";
    cout << "4. Issue Book\n";
    cout << "5. Return Book\n";
    cout << "6. List All Books\n";
    cout << "7. Delete Book\n";
    cout << "8. Update Book\n";
    cout << "9. Search Book by Author\n";
    cout << "10. Sort Books by ID\n";
    cout << "0. Exit\n";
    cout << "Enter your choice: ";
}

int main() {
    Library library(100);
    int choice, id;
    string title, author, studentName, newTitle, newAuthor;

    do {
        displayMenu();
        cin >> choice;

        switch (choice) {
            case 1:
                cout << "Enter Book ID: ";
                cin >> id;
                cout << "Enter Book Title: ";
                cin.ignore();
                getline(cin, title);
                cout << "Enter Book Author: ";
                getline(cin, author);
                library.addNewBook(id, title, author);
                break;

            case 2:
                cout << "Enter Book Title: ";
                cin.ignore();
```

```
            getline(cin, title);
            library.searchBook(title);
            break;

        case 3:
            cout << "Enter Book ID: ";
            cin >> id;
            library.searchBook(id);
            break;

        case 4:
            cout << "Enter Book ID: ";
            cin >> id;
            cout << "Enter Student Name: ";
            cin.ignore();
            getline(cin, studentName);
            library.issueBook(id, studentName);
            break;

        case 5:
            library.returnBook();
            break;

        case 6:
            library.listAllBooks();
            break;

        case 7:
            cout << "Enter Book ID to delete: ";
            cin >> id;
            library.deleteBook(id);
            break;

        case 8:
            cout << "Enter Book ID to update: ";
            cin >> id;
            cout << "Enter New Title: ";
            cin.ignore();
            getline(cin, newTitle);
            cout << "Enter New Author: ";
            getline(cin, newAuthor);
            library.updateBook(id, newTitle, newAuthor);
            break;

        case 9:
            cout << "Enter Book Author: ";
            cin.ignore();
            getline(cin, author);
```

```
                library.searchBookByAuthor(author);
                break;

            case 10:
                library.sortBooksById();
                break;

            case 0:
                cout << "Exiting the system. Goodbye!" << endl;
                break;

            default:
                cout << "Invalid choice. Please try again." << endl;
        }
    } while (choice != 0);

    return 0;
}
```

Outputs

1. **Add new Book**

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice:

1
Enter Book ID: 123
Enter Book Title: You can win!
Enter Book Author: Shive Khera
Book added: You can win! by Shive Khera
```

2. Search Book by Title

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 2
Enter Book Title: You can win!
Book found:
ID: 123, Title: You can win!, Author: Shive Khera, Status: Available
```

3. Search Book by Id

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 3
Enter Book ID: 123
Book found:
ID: 123, Title: You can win!, Author: Shive Khera, Status: Available
```

4. Issue Book

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 4
Enter Book ID: 123
Enter Student Name: Vivek Baghel
Book with ID 123 issued to Vivek Baghel
```

5. Return Book

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 5
Book with ID 123 returned.
```

BOARD

6. List All Books

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 6
ID: 345, Title: The C++ Programming Language, Author: Bjarne Stroustrup, Status: Available
ID: 234, Title: The Art of Computer Programming, Author: Donald knuth, Status: Available
ID: 123, Title: You can win!, Author: Shive Khera, Status: Available
```

7. Delete Book

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 7
Enter Book ID to delete: 234
Book with ID 234 deleted.
```

8. Update Book

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 8
Enter Book ID to update: 123
Enter New Title: You Can Win!!
Enter New Author: Shri Shiv Khera
Book with ID 123 updated.
```

9. Search Book by Author

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 9
Enter Book Author: Bjarne Stroustrup
Book found:
ID: 345, Title: The C++ Programming Language, Author: Bjarne Stroustrup, Status: Available
```

BOARD

10. Sort Books by ID

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 10
Books sorted by ID.
```

```
Library Management System
1. Add New Book
2. Search Book by Title
3. Search Book by ID
4. Issue Book
5. Return Book
6. List All Books
7. Delete Book
8. Update Book
9. Search Book by Author
10. Sort Books by ID
0. Exit
Enter your choice: 6
ID: 123, Title: You Can Win!!, Author: Shri Shiv Khera, Status: Available
ID: 345, Title: The C++ Programming Language, Author: Bjarne Stroustrup, Status: Available
```

Thank You!