***Library Management System***

***Group Member:***

M.Asghar Ali

Fahad Ali Akbar

Dilawar Khalid

***Code :***

#include <iostream>

#include <string>

using namespace std;

// Define Book class

class Book {

private:

string title;

string author;

bool available;

int id;

Book\* next;

public:

Book(string title, string author, int id) : title(title), author(author), available(true), id(id), next(nullptr) {}

string getTitle() const {

return title;

}

string getAuthor() const {

return author;

}

bool isAvailable() const {

return available;

}

int getID() const {

return id;

}

void borrowBook() {

available = false;

}

void returnBook() {

available = true;

}

friend class Library;

};

// Define Library class

class Library {

private:

Book\* head;

int lastBookID;

// Function to find the previous node of a given node

Book\* findPreviousNode(Book\* current) const {

Book\* temp = head;

while (temp && temp->next != current) {

temp = temp->next;

}

return temp;

}

public:

Library() : head(nullptr), lastBookID(0) {}

void addBook() {

string title, author;

cout << "Enter book title: ";

cin.ignore();

getline(cin, title);

cout << "Enter book author: ";

getline(cin, author);

Book\* newBook = new Book(title, author, ++lastBookID);

if (!head || head->getID() > newBook->getID()) {

newBook->next = head;

head = newBook;

} else {

Book\* current = head;

while (current->next && current->next->getID() < newBook->getID()) {

current = current->next;

}

newBook->next = current->next;

current->next = newBook;

}

cout << "Book added successfully with ID: " << newBook->getID() << endl;

}

void deleteBook(int bookID) {

Book\* current = head;

Book\* prev = nullptr;

while (current && current->getID() != bookID) {

prev = current;

current = current->next;

}

if (current) {

if (prev)

prev->next = current->next;

else

head = current->next;

delete current;

cout << "Book with ID " << bookID << " deleted successfully." << endl;

} else {

cout << "Book with ID " << bookID << " not found." << endl;

}

}

void displayBooks() const {

cout << "Library Catalog:\n";

Book\* current = head;

while (current) {

cout << "ID: " << current->getID() << ", Title: " << current->getTitle() << ", Author: " << current->getAuthor() << ", Available: " << (current->isAvailable() ? "Yes" : "No") << endl;

current = current->next;

}

}

void borrowBook(int bookID) {

Book\* current = head;

while (current) {

if (current->getID() == bookID && current->isAvailable()) {

current->borrowBook();

cout << "You have borrowed the book: " << current->getTitle() << endl;

return;

}

current = current->next;

}

cout << "Sorry, the book with ID " << bookID << " is not available for borrowing." << endl;

}

void returnBook(int bookID) {

Book\* current = head;

while (current) {

if (current->getID() == bookID && !current->isAvailable()) {

current->returnBook();

cout << "You have returned the book: " << current->getTitle() << endl;

return;

}

current = current->next;

}

cout << "Invalid return: The book with ID " << bookID << " is not borrowed or not found." << endl;

}

void searchBook(const string& title) const {

bool found = false;

Book\* current = head;

while (current) {

if (current->getTitle() == title) {

found = true;

cout << "Book found:\n";

cout << "ID: " << current->getID() << ", Title: " << current->getTitle() << ", Author: " << current->getAuthor() << ", Available: " << (current->isAvailable() ? "Yes" : "No") << endl;

}

current = current->next;

}

if (!found) {

cout << "Book with title \"" << title << "\" not found." << endl;

}

}

bool authenticateAdmin() const {

string password;

cout << "Enter the admin password: ";

cin >> password;

return password == "aaa";

}

};

int main() {

Library library;

// Prompt user for role (admin or student)

string role;

cout << "Enter your role (admin or student): ";

cin >> role;

// Authenticate admin

if (role == "admin") {

if (library.authenticateAdmin()) {

cout << "Admin login successful.\n";

while (true) {

cout << "\nOptions:\n1. Add Book\n2. Delete Book\n3. Display Books\n4. Search Book\n5. Exit\nEnter option: ";

int option;

cin >> option;

switch (option) {

case 1:

library.addBook();

break;

case 2: {

int bookID;

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

cin >> bookID;

library.deleteBook(bookID);

break;

}

case 3:

library.displayBooks();

break;

case 4: {

string title;

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

cin.ignore();

getline(cin, title);

library.searchBook(title);

break;

}

case 5:

cout << "Exiting program...\n";

exit(0);

default:

cout << "Invalid option. Please try again.\n";

}

}

} else {

cout << "Invalid password. Access denied." << endl;

}

}

// For students

else if (role == "student") {

string option;

while (true) {

cout << "\nOptions:\n1. Display Books\n2. Borrow Book\n3. Return Book\n4. Search Book\n5. Exit\nEnter option: ";

cin >> option;

if (option == "1") {

library.displayBooks();

} else if (option == "2") {

int bookID;

cout << "Enter the ID of the book you want to borrow: ";

cin >> bookID;

library.borrowBook(bookID);

} else if (option == "3") {

int bookID;

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

cin >> bookID;

library.returnBook(bookID);

} else if (option == "4") {

string title;

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

cin.ignore();

getline(cin, title);

library.searchBook(title);

} else if (option == "5") {

cout << "Exiting program...\n";

break;

} else {

cout << "Invalid option. Please try again.\n";

}

}

} else {

cout << "Invalid role. Please enter 'admin' or 'student'.\n";

}

return 0;

}