#include <iostream>

#include <string>

using namespace std;

class LibraryItem

{

private:

string title;

string author;

int year;

public:

LibraryItem(string title, string author, int year)

{

this->title = title;

this->author = author;

this->year = year;

}

string getTitle()

{

return title;

}

string getAuthor()

{

return author;

}

void display()

{

cout << "Title: " << title << endl;

cout << "Author: " << author << endl;

cout << "Year of Publication: " << year << endl;

}

};

class Book : public LibraryItem

{

private:

int pages;

public:

Book(string title, string author, int year, int pages) : LibraryItem(title, author, year)

{

this->pages = pages;

}

void display()

{

LibraryItem::display();

cout << "Number of Pages: " << pages << endl;

}

};

class Magazine : public LibraryItem

{

private:

int issueNumber;

public:

Magazine(string title, string author, int year, int issueNumber) : LibraryItem(title, author, year)

{

this->issueNumber = issueNumber;

}

void display()

{

LibraryItem::display();

cout << "Issue Number: " << issueNumber << endl;

}

};

class LibraryNode

{

public:

LibraryItem \*item;

LibraryNode \*next;

LibraryNode(LibraryItem \*item)

{

this->item = item;

next = NULL;

}

};

class LibraryLinkedList

{

private:

LibraryNode \*head;

LibraryNode \*tail;

public:

LibraryLinkedList()

{

head = NULL;

tail = NULL;

}

void addItem(LibraryItem \*item)

{

LibraryNode \*newNode = new LibraryNode(item);

if (!head)

{

head = newNode;

tail = newNode;

}

else

{

tail->next = newNode;

tail = newNode;

}

}

void displayItems()

{

LibraryNode \*current = head;

while (current)

{

current->item->display();

cout << "--------------------------\n";

current = current->next;

}

}

LibraryNode \*getHead()

{

return head;

}

~LibraryLinkedList()

{

while (head)

{

LibraryNode \*temp = head;

head = head->next;

delete temp->item;

delete temp;

}

}

};

class Library

{

private:

LibraryLinkedList collection;

public:

void addItem(LibraryItem \*item)

{

collection.addItem(item);

}

void displayItems()

{

collection.displayItems();

}

void searchItemsByTitle(string searchTerm)

{

bool found = false;

// Convert the search term to lowercase for case-insensitive search

for (int i = 0; i < searchTerm.length(); i++)

{

if (searchTerm[i] >= 'A' && searchTerm[i] <= 'Z')

{

searchTerm[i] = searchTerm[i] - 'A' + 'a'; // Convert uppercase to lowercase

}

}

// Get the head of the linked list

LibraryNode \*current = collection.getHead();

// Iterate through the linked list

while (current)

{

string title = current->item->getTitle();

// Convert the title to lowercase for case-insensitive comparison

for (int j = 0; j < title.length(); j++)

{

if (title[j] >= 'A' && title[j] <= 'Z')

{

title[j] = title[j] - 'A' + 'a'; // Convert uppercase to lowercase

}

}

// Check if the search term is found in the lowercase title

if (title.find(searchTerm) != string::npos)

{

// Display the item and a separator

current->item->display();

cout << "--------------------------" << endl;

found = true;

}

// Move to the next node in the linked list

current = current->next;

}

// If no items matching the search criteria were found, display a message

if (!found)

{

cout << "No items found matching the search criteria." << endl;

}

}

};

class Person

{

protected:

string name, address, phone\_number;

double age;

public:

Person(string name = "unknown", string address = "unknown", string phone\_number = "unknown", double age = 0)

{

this->name = name;

this->address = address;

this->phone\_number = phone\_number;

this->age = age;

}

void setage(double age)

{

this->age = age;

}

double getage()

{

return age;

}

string getName()

{

return name;

}

void Display\_person()

{

cout << "Your name is " << name << endl

<< "Your address is " << address << endl

<< "Your phone number is " << phone\_number << endl

<< "Your age is " << age << " years " << endl;

}

};

class Staff : public Person

{

public:

Staff(string name) : Person(name) {}

void addItemToLibrary(LibraryItem \*item, Library &library)

{

library.addItem(item);

cout << getName() << " added an item to the library." << endl;

}

void displayLibraryCollection(Library &library)

{

cout << getName() << " is displaying the library's collection:" << endl;

library.displayItems();

}

};

class Patron : public Person

{

private:

string typeofbook;

public:

Patron(string name = "unknown", string address = "unknown", string phone\_number = "unknown", double age = 0, string typeofbook = "unknown") : Person(name, address, phone\_number, age)

{

this->typeofbook = typeofbook;

}

void exploreLibraryCollection(Library &library)

{

cout << getName() << " is exploring the library's collection:" << endl;

library.displayItems();

}

void Display\_Patron()

{

Person::Display\_person();

cout << "the type of book " << typeofbook << endl;

}

};

int main()

{

cout << "=====================================================\n";

cout << "======== Welcome to the library's collection ========\n";

cout << "=====================================================\n";

Patron patron("Library Patron 1");

Library library;

Staff staff("Library Staff");

cout << "\nBooks\n"

<< endl;

Book book1("The Catcher in the Rye", "J.D. Salinger", 1951, 224);

staff.addItemToLibrary(&book1, library);

Book book2("To Kill a Mockingbird", "Harper Lee", 1960, 281);

staff.addItemToLibrary(&book2, library);

Book book3("The Great Gatsby", "F. Scott Fitzgerald", 1925, 180);

staff.addItemToLibrary(&book3, library);

cout << "\nMagazines\n"

<< endl;

Magazine magazine1("National Geographic", "Susan Goldberg", 2023, 150);

staff.addItemToLibrary(&magazine1, library);

Magazine magazine2("The New Yorker", "David Remnick", 2023, 80);

staff.addItemToLibrary(&magazine2, library);

cout << "--------------------------\n";

while (true)

{

cout << "\nMenu Options:\n";

cout << "1. Explore Library Collection\n";

cout << "2. Search for an Item\n";

cout << "3. Borrow a book\n";

cout << "4. Exit\n";

int choice;

cout << "\nEnter your choice: ";

cin >> choice;

cin.ignore();

switch (choice)

{

case 1:

patron.exploreLibraryCollection(library);

break;

case 2:

{

cout << "Enter the search term (title): ";

string searchTerm;

cin.ignore();

getline(cin, searchTerm);

library.searchItemsByTitle(searchTerm);

}

break;

case 3:

{

int chosee;

cout << "1. Do you want to read in the library" << endl;

cout << "2. Borrow a book" << endl;

cin >> chosee;

cin.ignore();

if (chosee == 1)

{

string name, typeofbook;

cout << "What's your name? ";

cin.ignore();

getline(cin, name);

cout << "Which book do you want to read? ";

getline(cin, typeofbook);

cout << "Your name is " << name << endl

<< "You are reading the book " << typeofbook << endl;

}

else if (chosee == 2)

{

string name, address, phone\_number, typeofbook;

double age;

cout << "Enter your name: ";

cin.ignore();

getline(cin, name);

cout << "Enter your address: ";

getline(cin, address);

cout << "Enter your phone number: ";

getline(cin, phone\_number);

cout << "Enter your age: ";

cin >> age;

cin.ignore();

cout << "What is the type of book you need? ";

cin.ignore();

getline(cin, typeofbook);

Patron c(name, address, phone\_number, age, typeofbook);

c.Display\_Patron();

}

}

break;

case 4:

cout << "Thanks!!" << endl;

return 0;

default:

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

break;

}

}

return 0;

}