

資料結構第二次作業

班級:資工二乙

姓名:侯秉辰

學號:4B2G0116

一．說明

圖書管理：使用 `std::list` 儲存圖書資料，每本書有書名、作者和唯一編號。可以新增、刪除、搜索圖書，並依編號排序列出。

借閱者管理：使用 `std::forward_list` 儲存借閱者資料，每位借閱者有姓名及借閱的書籍編號。可以新增、刪除、搜索借閱者，並列出所有借閱者及其借閱書籍。

主程式：包含測試資料，測試各項功能，展示如何添加、搜尋、刪除和列出圖書和借閱者資訊。

二．程式

```
#include <iostream>
#include <list>
#include <forward_list>
#include <vector>
#include <algorithm>
#include <regex>

using namespace std;

struct Book {
    string title;
    string author;
    string id;

    Book(const string& title, const string& author, const string& id) : title(title),
author(author), id(id) {}
};

struct Borrower {
    string name;
    vector<string> borrowedBooks;

    Borrower(const string& name, const vector<string>& borrowedBooks) : name(name),
borrowedBooks(borrowedBooks) {}
};
```

```

class LibrarySystem {
private:
    list<Book> books;
    forward_list<Borrower> borrowers;

public:
    // 添加圖書
    void addBook(const string& title, const string& author, const string& id) {
        if (!isValidBookID(id)) {
            cout << "圖書編號格式錯誤，請重新輸入。" << endl;
            return;
        }
        books.emplace_back(title, author, id);
        cout << "圖書已添加成功。" << endl;
    }

    // 刪除圖書
    void deleteBook(const string& id) {
        auto it = find_if(books.begin(), books.end(), [&](const Book& book) { return
book.id == id; });
        if (it != books.end()) {
            books.erase(it);
            cout << "圖書已刪除成功。" << endl;
        }
        else {
            cout << "找不到指定編號的圖書。" << endl;
        }
    }

    // 搜索圖書
    void searchBook(const string& id) {
        auto it = find_if(books.begin(), books.end(), [&](const Book& book) { return
book.id == id; });
        if (it != books.end()) {
            cout << "找到圖書: " << it->title << " by " << it->author << ", 編號: "
<< it->id << endl;
        }
    }
}

```

```

        else {
            cout << "找不到指定編號的圖書。" << endl;
        }
    }

    // 列出所有圖書（依照編號排序）
    void listBooks() {
        books.sort([](const Book& a, const Book& b) { return a.id < b.id; });
        for (const auto& book : books) {
            cout << "圖書: " << book.title << ", 作者: " << book.author << ", 編號: " << book.id << endl;
        }
    }

    // 添加借閱者
    void addBorrower(const string& name, const vector<string>& borrowedBooks) {
        borrowers.emplace_front(name, borrowedBooks);
        cout << "借閱者已添加成功。" << endl;
    }

    // 刪除借閱者
    void deleteBorrower(const string& name) {
        borrowers.remove_if([&](const Borrower& borrower) { return borrower.name == name; });
        cout << "借閱者已刪除成功（如果存在）。" << endl;
    }

    // 搜索借閱者
    void searchBorrower(const string& name) {
        auto it = find_if(borrowers.begin(), borrowers.end(), [&](const Borrower& borrower) { return borrower.name == name; });
        if (it != borrowers.end()) {
            cout << "找到借閱者: " << it->name << ", 借閱的圖書編號: ";
            for (const auto& bookID : it->borrowedBooks) {
                cout << bookID << " ";
            }
            cout << endl;
        }
    }

```

```

        else {
            cout << "找不到指定姓名的借閱者。" << endl;
        }
    }

// 列出所有借閱者
void listBorrowers() {
    for (const auto& borrower : borrowers) {
        cout << "借閱者: " << borrower.name << "，借閱的圖書編號: ";
        for (const auto& bookID : borrower.borrowedBooks) {
            cout << bookID << " ";
        }
        cout << endl;
    }
}

private:
    // 驗證圖書編號格式
    bool isValidBookID(const string& id) {
        regex pattern("[A-Z][0-9]{4}$");
        return regex_match(id, pattern);
    }
};

int main() {
    LibrarySystem library;

    // 添加圖書資料
    library.addBook("紅樓夢", "曹雪芹", "A1234");
    library.addBook("西遊記", "吳承恩", "B2345");
    library.addBook("水滸傳", "施耐庵", "B3456");
    library.addBook("三國演義", "羅貫中", "C4567");
    library.addBook("金瓶梅", "蘭陵笑笑生", "C5678");
    library.addBook("聊齋志異", "蒲松齡", "D6789");
    library.addBook("儒林外史", "吳敬梓", "D7890");
    library.addBook("封神演義", "許仲琳", "E8901");
    library.addBook("鏡花緣", "李汝珍", "E9012");
    library.addBook("老殘遊記", "劉鶚", "F0123");
}

```

```

// 添加借閱者資料
library.addBorrower("小光", { "E9012", "F0123" });
library.addBorrower("小華", { "C4567" });
library.addBorrower("小美", { "D6789", "E8901" });
library.addBorrower("小強", { "F0123" });
library.addBorrower("小麗", { "B3456", "C5678" });

// 測試功能
cout << "所有圖書：" << endl;
library.listBooks();
cout << "\n所有借閱者：" << endl;
library.listBorrowers();

cout << "\n搜尋圖書編號 B2345：" << endl;
library.searchBook("B2345");
cout << "\n搜尋借閱者 小明：" << endl;
library.searchBorrower("小明");

cout << "\n刪除圖書編號 B2345：" << endl;
library.deleteBook("B2345");
cout << "更新後的圖書清單：" << endl;
library.listBooks();

cout << "\n刪除借閱者 小明：" << endl;
library.deleteBorrower("小明");
cout << "更新後的借閱者清單：" << endl;
library.listBorrowers();

return 0;
}

```

三・執行結果

```
Microsoft Visual Studio 編輯...
圖書已添加成功。
圖書已添加成功。
圖書已添加成功。
圖書已添加成功。
圖書已添加成功。
圖書已添加成功。
圖書已添加成功。
圖書已添加成功。
圖書已添加成功。
借閱者已添加成功。
借閱者已添加成功。
借閱者已添加成功。
借閱者已添加成功。
借閱者已添加成功。
所有圖書：
圖書：紅樓夢，作者：曹雪芹，編號：A1234
圖書：西遊記，作者：吳承恩，編號：B2345
圖書：水滸傳，作者：施耐庵，編號：B3456
圖書：三國演義，作者：羅貫中，編號：C4567
圖書：金瓶梅，作者：蘭陵笑笑生，編號：C5678
圖書：聊齋志異，作者：蒲松齡，編號：D6789
圖書：儒林外史，作者：吳敬梓，編號：D7890
圖書：封神演義，作者：許仲琳，編號：E8901
圖書：鏡花緣，作者：李汝珍，編號：E9012
圖書：老殘遊記，作者：劉鶚，編號：F0123

所有借閱者：
借閱者：小麗，借閱的圖書編號：B3456 C5678
借閱者：小強，借閱的圖書編號：F0123
借閱者：小美，借閱的圖書編號：D6789 E8901
借閱者：小華，借閱的圖書編號：C4567
借閱者：小光，借閱的圖書編號：E9012 F0123

搜尋圖書編號 B2345：
找到圖書：西遊記 by 吳承恩，編號：B2345

搜尋借閱者 小明：
找不到指定姓名的借閱者。

刪除圖書編號 B2345：
圖書已刪除成功。
更新後的圖書清單：
圖書：紅樓夢，作者：曹雪芹，編號：A1234
圖書：水滸傳，作者：施耐庵，編號：B3456
圖書：三國演義，作者：羅貫中，編號：C4567
圖書：金瓶梅，作者：蘭陵笑笑生，編號：C5678
圖書：聊齋志異，作者：蒲松齡，編號：D6789
圖書：儒林外史，作者：吳敬梓，編號：D7890
圖書：封神演義，作者：許仲琳，編號：E8901
圖書：鏡花緣，作者：李汝珍，編號：E9012
圖書：老殘遊記，作者：劉鶚，編號：F0123

刪除借閱者 小明：
借閱者已刪除成功（如果存在）。
更新後的借閱者清單：
借閱者：小麗，借閱的圖書編號：B3456 C5678
借閱者：小強，借閱的圖書編號：F0123
借閱者：小美，借閱的圖書編號：D6789 E8901
借閱者：小華，借閱的圖書編號：C4567
借閱者：小光，借閱的圖書編號：E9012 F0123

D:\11-3\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe (流程 22424) 已結束，代碼為 0 (0x0)。
若要在偵錯停止時自動關閉主控台，請啟用 [工具] -> [選項] -> [偵錯] -> [偵錯停止時，自動關閉主控台]。
按任意鍵關閉此視窗。]
```

<https://github.com/4b2g0116/4B2G0116-.git>