面向对象程序设计实战

实训报告

课程： 面向对象程序设计

学院： 人工智能学院

专业：

班级：

学号：

姓名：

时间： 2023年6月26日-2022年6月30日

1.课题概述

1.1 课题简述（200-300字）

**图书管理系统是一个用于管理图书馆或其他类似机构图书借阅和归还等操作的软件系统。该系统可以提供图书的管理、读者的注册、借阅与归还等功能，并能够方便地进行查询和统计。图书管理系统可以实现以下功能：**

**1. 创建图书：用户可以输入图书的标题、作者、索引代码和条形码，系统将创建一个图书项目并保存到系统中。**

**2. 注册读者：用户可以输入读者的名称、职位和代码，系统将创建一个读者并保存到系统中。**

**3. 借阅图书：用户输入读者代码和图书条形码，系统会检查读者和图书是否存在，并将图书借给该读者。**

**4. 归还图书：用户输入读者代码和图书条形码，系统会检查读者和图书是否存在，并将图书归还给图书馆。**

**5. 查询借阅图书：用户输入读者代码，系统会查询该读者借阅的图书，并将结果显示出来。**

**图书管理系统可以提高图书馆的管理效率，方便读者借阅图书，减少人工操作和纸质记录，提供更便捷和高效的图书借阅服务。同时，图书管理系统还可以进行数据备份和安全管理，确保图书和读者信息的安全性和可靠性。**

1.2课程设计目的

**图书管理系统的设计目的主要是为了提高图书馆的管理效率，方便读者进行图书的借阅和归还，并提供便捷的查询和统计功能。具体设计目的包括：**

**1. 提供便捷的图书管理：通过图书管理系统，图书馆管理员可以方便地进行图书的添加、删除、修改和查询等操作，实现对图书的分类和管理。管理员可以轻松地跟踪和管理图书的借阅情况，减少了繁琐的人工操作和纸质记录。**

**2. 方便读者借阅图书：读者可以通过图书管理系统进行图书的借阅和归还操作。通过简单的输入和确认，读者可以快速借阅所需图书，无需排队等待，提高了借阅效率。同时，系统还提供了图书的查询功能，方便读者查找和选择图书。**

**3. 提供便捷的查询和统计功能：图书管理系统可以根据图书的标题、作者、分类等信息进行查询，使读者能够快速找到所需图书。同时，系统还可以进行图书的借阅统计，包括借阅次数、借阅率、借阅排行等，方便图书馆进行资源管理和决策。**

**4. 提高管理和数据安全性：图书管理系统可以进行数据备份和安全管理，确保图书和读者信息的安全性和可靠性。系统能够对数据进行加密和权限管理，防止非法访问和数据丢失。**

**通过图书管理系统的应用，可以提高图书馆的服务质量，提升读者满意度。**

1.3课程设计要求

**通过面向对象的设计实现，使用了类和对象的概念，将图书和读者作为类的实例进行管理。主要类包括Item（图书）、Reader（读者）和CMain（主类）。**

**项目功能：**

**1. 创建图书：用户可以输入图书的标题、作者、索引代码和条形码，系统将创建一个图书项目并保存到系统中。**

**2. 注册读者：用户可以输入读者的名称、职位和代码，系统将创建一个读者并保存到系统中。**

**3. 借阅图书：用户输入读者代码和图书条形码，系统会检查读者和图书是否存在，并将图书借给该读者。**

**4. 归还图书：用户输入读者代码和图书条形码，系统会检查读者和图书是否存在，并将图书归还给图书馆。**

**5. 查询借阅图书：用户输入读者代码，系统会查询该读者借阅的图书，并将结果显示出来。**

2. 课题主要功能模块的设计与实现

**1. `Item`类的函数：**

**`Item(string title, string author, int indexCode, string barcode)`：通过传入的参数创建一个具有特定信息的图书对象。**

**`Item()`：默认构造函数，创建一个空的图书对象。**

**`string getTitle()`：获取图书的标题。**

**`string getAuthor()`：获取图书的作者。**

**`int getIndexCode()`：获取图书的索引代码。**

**`string getBarcode()`：获取图书的条形码。**

**2. `Reader`类的函数：**

**`Reader(string name, string position, int code)`：通过传入的参数创建一个具有特定信息的读者对象。**

**`Reader()`：默认构造函数，创建一个空的读者对象。**

**`string getName()`：获取读者的姓名。**

**`string getPosition()`：获取读者的职位。**

**`int getCode()`：获取读者的代码。**

**`void borrowItem(Item item)`：将一本图书加入到借阅列表中。**

**`void returnItem(Item item)`：将一本图书从借阅列表中移除。**

**`vector<Item> getBorrowedItems()`：获取读者借阅的图书列表。**

**3. `CMain`类的函数：**

**`void createItem()`：创建一个图书对象，并将其添加到图书集合中。**

**`void createReader()`：创建一个读者对象，并将其添加到读者集合中。**

**`void borrowItem()`：通过读者代码和图书条形码，借阅一本图书。**

**`void returnItem()`：通过读者代码和图书条形码，归还一本图书。**

**`void searchBorrowedBooks()`：通过读者代码，查询该读者借阅的图书列表。**

**4. `main`函数的功能：**

**提供一个菜单，允许用户选择不同的功能模块。**

**根据用户的选择调用`CMain`类中相应的函数来执行相应的操作。**

**循环，直到用户选择退出程序为止。**

**同时本程序的**

**1. Item类：封装了图书的信息，包括标题、作者、索引代码和条形码。可以通过构造函数创建一个具有特定信息的图书对象，也可以通过各个getter函数获取图书的信息。**

**2. Reader类：封装了读者的信息，包括姓名、职位和代码。可以通过构造函数创建一个具有特定信息的读者对象，也可以通过各个getter函数获取读者的信息。读者对象还具有借阅图书和归还图书的功能，通过borrowItem函数将一本图书加入到借阅列表中，通过returnItem函数将一本图书从借阅列表中移除。**

**3. CMain类：管理图书和读者对象的集合。可以通过createItem函数创建一个图书对象并添加到图书集合中，通过createReader函数创建一个读者对象并添加到读者集合中。还可以通过borrowItem函数和returnItem函数实现借阅和归还图书的功能。通过searchBorrowedBooks函数可以查询某个读者借阅的图书列表。**

**4. main函数：以交互方式提供了一个菜单，允许用户选择不同的功能模块。根据用户的选择调用CMain类中相应的函数来执行相应的操作。**

3. [课题程序性能分析与评价](#_Toc73253606)

对以上的代码进行性能分析，如果要提升程序性能可以考虑以下几个方面：

时间空间复杂度方面：

**1. 图书和读者的集合：**

**- 目前的实现中，图书和读者的集合是使用容器（可能是数组或向量）来存储的。如果图书和读者的数量很大，而且需要频繁进行搜索、插入和删除操作，使用容器可能会导致性能下降。**

**- 为了提升性能，可以考虑使用更高效的数据结构，如哈希表或平衡二叉搜索树，来存储图书和读者的集合。这样可以在常量时间内进行搜索、插入和删除操作。**

**2. 搜索借阅的图书：**

**- 目前的实现中，搜索借阅的图书是通过遍历读者的借阅列表来实现的。如果一个读者借阅了大量的图书，遍历列表的操作可能会很耗时。**

**- 为了提升性能，可以在读者的借阅列表中使用更高效的数据结构，如哈希表或平衡二叉搜索树，来存储图书。这样可以在常量时间内进行搜索、插入和删除操作。**

程序源代码如下：

#include <iostream>

#include <string>

#include <vector>

using namespace std;

class Item

{

private:

string title;

string author;

int indexCode;

string barcode;

public:

Item(string title, string author, int indexCode, string barcode)

{

this->title = title;

this->author = author;

this->indexCode = indexCode;

this->barcode = barcode;

}

// 添加默认构造函数

Item()

{

this->title = "";

this->author = "";

this->indexCode = 0;

this->barcode = "";

}

string getTitle()

{

return title;

}

string getAuthor()

{

return author;

}

int getIndexCode()

{

return indexCode;

}

string getBarcode()

{

return barcode;

}

};

class Reader

{

private:

string name;

string position;

int code;

vector<Item> borrowedItems;

public:

Reader(string name, string position, int code)

{

this->name = name;

this->position = position;

this->code = code;

}

// 添加默认构造函数

Reader()

{

this->name = "";

this->position = "";

this->code = 0;

}

string getName()

{

return name;

}

string getPosition()

{

return position;

}

int getCode()

{

return code;

}

void borrowItem(Item item)

{

borrowedItems.push\_back(item);

}

void returnItem(Item item)

{

for (int i = 0; i < borrowedItems.size(); i++)

{

if (borrowedItems[i].getBarcode() == item.getBarcode())

{

borrowedItems.erase(borrowedItems.begin() + i);

break;

}

}

}

vector<Item> getBorrowedItems()

{

return borrowedItems;

}

};

class CMain

{

private:

vector<Item> items;

vector<Reader> readers;

public:

void createItem()

{

string title, author, barcode;

int indexCode;

cout << "请输入图书标题： ";

cin >> title;

cout << "请输入作者姓名： ";

cin >> author;

cout << "请输入索引代码： ";

cin >> indexCode;

cout << "请输入条形码： ";

cin >> barcode;

Item item(title, author, indexCode, barcode);

items.push\_back(item);

cout << "图书项目创建成功。" << endl;

}

void createReader()

{

string name, position;

int code;

cout << "输入读者名称： ";

cin >> name;

cout << "输入读者职位： ";

cin >> position;

cout << "输入读者代码：";

cin >> code;

Reader reader(name, position, code);

readers.push\_back(reader);

cout << "读者注册成功." << endl;

}

void borrowItem()

{

int readerCode;

string barcode;

cout << "请输入读者代码：";

cin >> readerCode;

cout << "请输入图书条形码： ";

cin >> barcode;

Item item;

Reader reader;

for (int i = 0; i < items.size(); i++)

{

if (items[i].getBarcode() == barcode)

{

item = items[i];

break;

}

}

for (int i = 0; i < readers.size(); i++)

{

if (readers[i].getCode() == readerCode)

{

reader = readers[i];

break;

}

}

reader.borrowItem(item);

cout << "图书借阅成功." << endl;

}

void returnItem()

{

int readerCode;

string barcode;

cout << "请输入读者代码： ";

cin >> readerCode;

cout << "请输入图书条形码：";

cin >> barcode;

Item item;

Reader reader;

for (int i = 0; i < items.size(); i++)

{

if (items[i].getBarcode() == barcode)

{

item = items[i];

break;

}

}

for (int i = 0; i < readers.size(); i++)

{

if (readers[i].getCode() == readerCode)

{

reader = readers[i];

break;

}

}

reader.returnItem(item);

cout << "图书归还成功." << endl;

}

void searchBorrowedBooks()

{

int readerCode;

cout << "请输入读者代码：";

cin >> readerCode;

Reader reader;

for (int i = 0; i < readers.size(); i++)

{

if (readers[i].getCode() == readerCode)

{

reader = readers[i];

break;

}

}

vector<Item> borrowedItems = reader.getBorrowedItems();

cout << "借阅中的图书：" << endl;

for (int i = 0; i < borrowedItems.size(); i++)

{

cout << "标题： " << borrowedItems[i].getTitle() << endl;

cout << "作者： " << borrowedItems[i].getAuthor() << endl;

cout << "索引码：" << borrowedItems[i].getIndexCode() << endl;

cout << "条形码： " << borrowedItems[i].getBarcode() << endl;

cout << endl;

}

}

};

int main()

{

CMain main;

int choice;

do

{

cout << "1. 创建图书" << endl;

cout << "2. 注册读者" << endl;

cout << "3. 借阅图书" << endl;

cout << "4. 归还图书" << endl;

cout << "5. 查询借阅图书" << endl;

cout << "6. 退出" << endl;

cout << "请输入您的选择: ";

cin >> choice;

switch (choice)

{

case 1:

main.createItem();

break;

case 2:

main.createReader();

break;

case 3:

main.borrowItem();

break;

case 4:

main.returnItem();

break;

case 5:

main.searchBorrowedBooks();

break;

case 6:

cout << "退出程序..." << endl;

break;

default:

cout << "无效的选择！请再试一次。" << endl;

}

cout << endl;

} while (choice != 6);

return 0;

}

[3.1 实验环境](#_Toc73253607)

版本 Windows 10 专业版

版本号 22H2

安装日期 ‎2021/‎8/‎28

操作系统内部版本 19045.3086

体验 Windows Feature Experience Pack 1000.19041.1000.0

工具：vscode

gcc version 8.1.0 (x86\_64-posix-seh-rev0, Built by MinGW-W64 project)

4. 小结 （200-300字）

针对性能分析伪代码改进方案如下：

以下是对以上性能分析的伪代码改进思路：

**1. 图书和读者的集合：**

**- 使用哈希表或平衡二叉搜索树来存储图书和读者的集合。**

**- 在图书集合中，使用图书的条形码作为键，将图书对象作为值存储。**

**- 在读者集合中，使用读者的代码作为键，将读者对象作为值存储。**

**2. 搜索借阅的图书：**

**- 在读者对象中维护一个哈希表或平衡二叉搜索树，用于存储借阅的图书。**

**- 在图书对象中维护一个标志，表示该图书是否被借阅。**

**- 当读者借阅一本图书时，将图书添加到读者的借阅图书集合中，并将图书的借阅状态设置为已借阅。**

**- 当读者归还一本图书时，将图书从读者的借阅图书集合中移除，并将图书的借阅状态设置为未借阅。**

**通过以上的改进思路，可以提升程序的性能，减少搜索、插入和删除操作的时间复杂度。**

**以下是伪代码示例：**

**plaintext**

**class CMain {**

**HashTable items; // 使用哈希表存储图书集合**

**HashTable readers; // 使用哈希表存储读者集合**

**//**

**void createItem() {**

**// 读取图书信息**

**string title, author;**

**int indexCode;**

**string barcode;**

**// 创建图书对象**

**Item item(title, author, indexCode, barcode);**

**// 添加图书到集合中**

**items.insert(barcode, item);**

**}**

**void createReader() {**

**// 读取读者信息**

**string name, position;**

**int code;**

**// 创建读者对象**

**Reader reader(name, position, code);**

**// 添加读者到集合中**

**readers.insert(code, reader);**

**}**

**void borrowItem() {**

**// 读取读者代码和图书条形码**

**int readerCode;**

**string barcode;**

**// 检查读者和图书是否存在**

**if (!readers.contains(readerCode) || !items.contains(barcode)) {**

**// 读者或图书不存在，输出错误信息**

**print("Reader or item not found.");**

**return;**

**}**

**// 获取读者和图书对象**

**Reader& reader = readers.get(readerCode);**

**Item& item = items.get(barcode);**

**// 借阅图书**

**reader.borrowItem(item);**

**}**

**void returnItem() {**

**// 读取读者代码和图书条形码**

**int readerCode;**

**string barcode;**

**// 检查读者和图书是否存在**

**if (!readers.contains(readerCode) || !items.contains(barcode)) {**

**// 读者或图书不存在，输出错误信息**

**print("Reader or item not found.");**

**return;**

**}**

**// 获取读者和图书对象**

**Reader& reader = readers.get(readerCode);**

**Item& item = items.get(barcode);**

**// 归还图书**

**reader.returnItem(item);**

**}**

**void searchBorrowedBooks() {**

**// 读取读者代码**

**int readerCode;**

**// 检查读者是否存在**

**if (!readers.contains(readerCode)) {**

**// 读者不存在，输出错误信息**

**print("Reader not found.");**

**return;**

**}**

**// 获取读者对象**

**Reader& reader = readers.get(readerCode);**

**// 获取读者借阅的图书集合**

**HashTable borrowedBooks = reader.getBorrowedItems();**

**// 输出借阅的图书信息**

**for (const auto& item : borrowedBooks) {**

**// 输出图书的标题、作者、索引代码和条形码**

**//**

**}**

**}**

**};**

**通过以上的改进思路，可以提高程序的性能，减少搜索、插入和删除操作的时间复杂度，并更有效地处理借阅的图书列表。**

目前针对以下模块中的输入条形码，在输入条形码为字符型变量会产生不断刷新输出控制台的bug，改为int型则不会报错

 public:

    void createItem()

    {

        string title, author, barcode;

        int indexCode;

        cout << "请输入图书标题： ";

        cin >> title;

        cout << "请输入作者姓名： ";

        cin >> author;

        cout << "请输入索引代码： ";

        cin >> indexCode;

        cout << "请输入条形码： ";

        cin >> barcode;

        Item item(title, author, indexCode, barcode);

        items.push\_back(item);

        cout << "图书项目创建成功。" << endl;

    }

5. 参考文献

《代码大全2》

《c++ primer plus》

https://github.com/

https://www.csdn.net/

格式要求：

1. 一级标题：黑体，小三号

二级标题：楷体，四号

三级标题：楷体，小四号

2.正文：小四号 宋体 页面边距设置为左边距：3cm；右边距：2.5cm；上边距：3cm；下边距：2.5cm。行间距为1.5倍行距，（代码可以是一倍行距）；图、表标题采用小五号黑体；表格中文字、图例说明采用五号号宋体；表注采用小五号宋体。

3.页眉、页脚文字均采用小五号宋体，页眉统一为“面向对象程序设计实训报告”，页码排在页脚居中位置。（**首页不加页眉、页码）**