МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Самарский государственный аэрокосмический университет имени академика С.П. Королёва (национальный исследовательский университет)» (СГАУ)

Факультет информатики

Кафедра программных систем

Дисциплина

**Объектно-ориентированное программирование**

**ОТЧЕТ**

по лабораторной работе № 4

**<** **Объектно-ориентированное программирование**

**на языке С++.>**

Вариант № 9

Студент: Гижевская В.Д.

Группа: 6213-020302D

Преподаватель: Баландин А.В.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2019

**Задание**

Написать программу управления информацией о книгах, находящихся в библиотеке, состоящей из двух списков:

1) Динамически формируемый список записей о книгах вида:

{<название\_книги>;

<количество\_экземпляров\_в\_библиотеке>;}

2) Динамически формируемый список авторов с записями вида:

{<шифр\_автора>;

<Фамилия\_И. О.>;}

3) Динамически формируемый список индексных записей, связывающих между собой авторов и их книг:

{<шифр\_автора>;

<индекс\_книги>;

<количество\_экземпляров\_на\_полке>;}.

<шифр\_автора> - трёхсимвольный идентификатор.

<индекс\_книги> - указатель записи в списке книг.

Программа должна обеспечивать выполнение следующих функций:

1) Поступление новой книги в библиотеку.

2) Выдачу книги читателям на руки.

3) Возврат книги в библиотеку.

4) Вывод на экран всех зарегистрированных в библиотеке произведений (список всех книг с их авторами и количеством экземпляров).

5) Вывод на экран авторов всех зарегистрированных в библиотеке книг.

6) По фамилии автора выводить на экран все его произведения, стоящие в библиотеке на полках.

**Объектная модель**

Информационный объект – библиотека

* Свойства:
  + Список книг
  + Список авторов
  + Список, связывающий авторы и книги
* Процедуры:
  + Создание объекта библиотеки
  + Добавление книги в библиотеку
  + Выдача книги
  + Возврат книги
  + Вывод всех книг
  + Вывод всех авторов
  + Вывод книг заданного автора, стоящих на полке
  + Удаление объекта библиотеки
  + Поиск книги (указателя в списке) по названию
  + Поиск автора (указателя в списке) по имени
  + Получение индексной записи по индексу книги и шифру

Информационный объект – запись о книге

* Свойства:
  + Название книги
  + Количество экземпляров в библиотеке

Информационный объект – запись с авторами

* Свойства:
  + Шифр автора
  + Имя автора

Информационный объект – индексная запись

* Свойства:
  + Шифр автора
  + Индекс книги
  + Количество экземпляров на полке

**Исходный текст программы**

**Library.h**

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma once

#include <list>

#include <string>

#include <iostream>

using namespace std;

class Library

{

public:

~Library();

void add\_book(string name, int num, string author, char\* code\_author);//добавление книги в библиотеку

void give\_book(string name, string author);//выдача книга

void return\_book(string name, string author);//возврат книги

void list\_of\_books();//вывод всех книг

void list\_of\_authors();//вывод всех авторов

void available\_books(string author);//вывод книг заданного автора на полках

private:

//запись о книге

typedef struct{

string name\_book;//название книги

int num\_of\_books\_all;//количество экземпляров в библиотеке

} Book;

//запись с авторами

typedef struct{

char code\_author[3];//шифр автора

string name\_author;//имя автора

} Author;

//индексная запись

typedef struct{

char code\_author[3];//шифр автора

Book\* index\_book;//индекс книги

int num\_of\_books;//колиесство экземпляров на полке

} Index;

list <Book\*> listBook;//список книг

list <Author\*> listAuthor;//список авторов

list <Index\*> listIndex;//список, связывающий книги и авторов

Book\* get\_book(string name);//ищем книгу(указатель в списке) по названию

Author\* get\_author(string author);//ищем автора(указатель в списке) по имени

Index\* get\_index(Book\* book, char\* code\_author);//получение индексной записи по индексу книги и шифру

};

**Library.cpp**

#include "Library.h"

Library::~Library(){

for (Book\*& book : listBook) delete book;

for (Author\*& author : listAuthor) delete author;

for (Index\*& index : listIndex) delete index;

}

//добавление книги в библиотеку

void Library::add\_book(string name, int num, string author, char\* code\_author){

Book\* book = new Book;

book->name\_book = name;

book->num\_of\_books\_all = num;

listBook.push\_back(book);

Author\* book\_author = get\_author(author);

if (!book\_author) {

book\_author = new Author;

book\_author->name\_author = author;

strcpy(book\_author->code\_author, code\_author);

listAuthor.push\_back(book\_author);

}

Index\* index = new Index;

strcpy(index->code\_author, code\_author);

index->num\_of\_books = num;

index->index\_book = book;

listIndex.push\_back(index);

}

//выдача книги

void Library::give\_book(string name, string author){

Book\* book\_ptr = get\_book(name);

if (!book\_ptr) {

throw "Такой книги в библиотеке нет";

}

Author\* author\_ptr = get\_author(author);

if (!author\_ptr) {

throw "Такого автора в библиотеке нет";

}

Index\* index = get\_index(book\_ptr, author\_ptr->code\_author);

if (index->num\_of\_books > 0) {

index->num\_of\_books--;

}

else {

throw "Все книги уже выданы";

}

}

//возврат книги

void Library::return\_book(string name, string author){

Book\* book\_ptr = get\_book(name);

if (!book\_ptr) {

throw "Такой книги в библиотеке нет";

}

Author\* author\_ptr = get\_author(author);

if (!author\_ptr) {

throw "Такого автора в библиотеке нет";

}

Index\* index = get\_index(book\_ptr, author\_ptr->code\_author);

if (index->num\_of\_books < book\_ptr->num\_of\_books\_all) {

index->num\_of\_books++;

}

else {

throw "Все книги на месте!";

}

}

//вывод списка книг

void Library::list\_of\_books(){

for (Index\*& index : listIndex){

string name\_author = "";

for (Author\*& author : listAuthor){

if (!strcmp(index->code\_author, author->code\_author)){

name\_author = author->name\_author;

}

}

cout << index->index\_book->name\_book << ", " << name\_author << " - " << index->index\_book->num\_of\_books\_all << " штук" << endl;

}

}

//вывод списка авторов

void Library::list\_of\_authors(){

for (Author\*& author : listAuthor)

{

cout << author->name\_author << endl;

}

}

//вывод списка книг заданного автора, стоящих на полке

void Library::available\_books(string author){

Author\* author\_ptr = get\_author(author);

if (!author\_ptr) {

throw "Такого автора в библиотеке нет";

}

for (Index\*& index : listIndex) {

if ((index->num\_of\_books > 0) && (!strcmp(index->code\_author,author\_ptr->code\_author))) {

cout << index->index\_book->name\_book << " - " << index->num\_of\_books << " штук" << endl;

}

}

}

//ищем книгу(указатель в списке) по названию

Library::Book\* Library::get\_book(string name){

Book\* book\_ptr = NULL;

for (Book\*& book : listBook) {

if (book->name\_book == name) {

book\_ptr = book;

break;

}

}

return book\_ptr;

}

//ищем автора(указатель в списке) по имени

Library::Author\* Library::get\_author(string author){

Author\* author\_ptr = NULL;

for (Author\*& \_author : listAuthor) {

if (\_author->name\_author == author) {

author\_ptr = \_author;

}

}

return author\_ptr;

}

//получение индексной записи по индексу книги и шифру

Library::Index\* Library::get\_index(Book\* book, char\* code\_author){

Index\* index = NULL;

for (Index\*& idx : listIndex) {

if (idx->index\_book == book && !strcmp(idx->code\_author, code\_author)) {

index = idx;

}

}

return index;

}

**OOP\_lab3.cpp**

#pragma once

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <Windows.h>

#include "Library.h"

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

setlocale(LC\_ALL, "");

Library library;

char s;

do {

system("cls");

cout << "Лабораторная работа № 4\nГижевская Валерия\nВариант 9" << endl;

cout << "Что вы хотите сделать?" << endl;

cout << "1 - Добавить книгу в библиотеку" << endl

<< "2 - Выдать книгу читателю на руки" << endl

<< "3 - Вернуть книгу в библиотеку" << endl

<< "4 - Вывести на экран всех зарегистрированные в библиотеке произведения" << endl

<< "5 - Вывести на экран авторов всех зарегистрированных в библиотеке книг" << endl

<< "6 - По фамилии автора вывести на экран все его произведения, стоящие в библиотеке на полках" << endl

<< "0 - Выход" << endl;

cin >> s;

try {

switch (s){

case '1': {

cout << "Название книги: ";

string name;

cin >> name;

cout << "Имя автора: ";

string author;

cin >> author;

cout << "Количество экземпляров: ";

int num;

cin >> num;

cout << "Трёхзначный шифр автора: ";

char code[3];

cin >> code;

library.add\_book(name, num, author, code);

break;

}

case '2': {

cout << "Название книги: ";

string name;

cin >> name;

cout << "Имя автора: ";

string author;

cin >> author;

library.give\_book(name, author);

break;

}

case '3': {

cout << "Название книги: ";

string name;

cin >> name;

cout << "Имя автора: ";

string author;

cin >> author;

library.return\_book(name, author);

break;

}

case '4': {

library.list\_of\_books();

break;

}

case '5': {

library.list\_of\_authors();

break;

}

case '6': {

cout << "Имя автора: ";

string author;

cin >> author;

library.available\_books(author);

break;

}

case '0': {

break;

}

default: {

cout << "Нет такого пункта" << endl;

break;

}

}

}

catch (const char\* exc) {

cout << exc << endl;

}

system("pause");

} while (s != '0');

return 0;

}