МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНФОРМАЦИОННЫХ

ТЕХНОЛОГИЙ И ТЕЛЕКОММУНИКАЦИЙ

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

**Лабораторная работа №6**

по дисциплине

«Объектно-ориентированное программирование»

**Выполнил:**

Иванов Даниил Евгеньевич

Студент 2 курса группы \_ПИН-б-о-22-1

Направления подготовки

09.03.03 Прикладная информатика

очной формы обучения

Ставрополь, 2023 г.

Тема: Стандартная библиотека шаблонов

Цель работы: изучить стандартные библиотеки шаблонов и научится реализовать их.

Выполнение работы:

Вариант -16

Листинг:

#include <iostream>

#include "Book.h"

#include "Library.h"

#include "Test.h"

#include <Windows.h>

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1551);

setlocale(LC\_ALL, "Russian");

Library library;

int choice;

do {

std::cout << "1. Добавить книгу\n";

std::cout << "2. Удалить книгу\n";

std::cout << "3. Отобразить все книги по авторам\n";

std::cout << "4. Отобразить все книги по годам\n";

std::cout << "5. Выход\n";

std::cout << "Введите свой выбор: ";

std::cin >> choice;

switch (choice) {

case 1: {

std::string author, title;

int year, quantity;

std::cout << "Введите автора: ";

std::cin.ignore();

std::getline(std::cin, author);

std::cout << "Введите название: ";

std::getline(std::cin, title);

std::cout << "Введите год: ";

std::cin >> year;

std::cout << "Введите количество: ";

std::cin >> quantity;

Book newBook(author, title, year, quantity);

library.addBook(newBook);

break;

}

case 2: {

std::string title;

std::cout << "Введите название книги, которую нужно удалить: ";

std::cin.ignore();

std::getline(std::cin, title);

library.removeBook(title);

break;

}

case 3:

library.displayByAuthor();

break;

case 4:

library.displayByYear();

break;

case 5:

std::cout << "Существующая программа...\n";

break;

default:

std::cout << "Неверный выбор. Пожалуйста, введите допустимый вариант.\n";

}

} while (choice != 5);

runTests();

return 0;

}

#ifndef BOOK\_H

#define BOOK\_H

#include <string>

class Book {

private:

std::string author;

std::string title;

int year;

int quantity;

public:

Book(std::string author, std::string title, int year, int quantity);

// Средства получения и настройки свойств книги

std::string getAuthor() const;

std::string getTitle() const;

int getYear() const;

int getQuantity() const;

void setQuantity(int quantity);

// Функция отображения сведений о книге

void display() const;

};

#endif // BOOK\_H

#include "Book.h"

#include <iostream>

Book::Book(std::string author, std::string title, int year, int quantity)

: author(std::move(author)), title(std::move(title)), year(year), quantity(quantity) {}

std::string Book::getAuthor() const {

return author;

}

std::string Book::getTitle() const {

return title;

}

int Book::getYear() const {

return year;

}

int Book::getQuantity() const {

return quantity;

}

void Book::setQuantity(int quantity) {

this->quantity = quantity;

}

void Book::display() const {

std::cout << "Автор: " << author << "\tНазвание: " << title << "\tГод: " << year << "\tКоличество: " << quantity << std::endl;

}

#ifndef LIBRARY\_H

#define LIBRARY\_H

#include "Book.h"

#include <vector>

class Library {

private:

std::vector<Book> books;

public:

// Функция добавления новой книги

void addBook(const Book& book);

// Функция удаления книги по названию

void removeBook(const std::string& title);

// Функция отображения всех книг, отсортированных по авторам

void displayByAuthor() const;

// Функция отображения всех книг, отсортированных по годам

void displayByYear() const;

// Метод для получения списка книг

const std::vector<Book>& getBooks() const {

return books;

}

};

#endif // LIBRARY\_H

#include "Library.h"

#include <algorithm>

void Library::addBook(const Book& book) {

books.push\_back(book);

}

void Library::removeBook(const std::string& title) {

books.erase(std::remove\_if(books.begin(), books.end(),

[&title](const Book& book) { return book.getTitle() == title; }),

books.end());

}

void Library::displayByAuthor() const {

std::vector<Book> sortedBooks = books;

std::sort(sortedBooks.begin(), sortedBooks.end(),

[](const Book& book1, const Book& book2) { return book1.getAuthor() < book2.getAuthor(); });

for (const auto& book : sortedBooks) {

book.display();

}

}

void Library::displayByYear() const {

std::vector<Book> sortedBooks = books;

std::sort(sortedBooks.begin(), sortedBooks.end(),

[](const Book& book1, const Book& book2) { return book1.getYear() < book2.getYear(); });

for (const auto& book : sortedBooks) {

book.display();

}

}

#ifndef TEST\_H

#define TEST\_H

void runTests();

#endif // TEST\_H

#include "Test.h"

#include "Book.h"

#include "Library.h"

#include <cassert>

void testBook() {

// Тест класса Book

Book book("Автор", "Название", 2020, 5);

assert(book.getAuthor() == "Автор");

assert(book.getTitle() == "Название");

assert(book.getYear() == 2020);

assert(book.getQuantity() == 5);

// Тест метода setQuantity

book.setQuantity(10);

assert(book.getQuantity() == 10);

}

void testLibrary() {

// Тест класса Library

Library library;

// Тест метода addBook

Book book1("Author1", "Title1", 2020, 5);

library.addBook(book1);

assert(library.getBooks().size() == 1);

// Тест метода removeBook

library.removeBook("Title1");

assert(library.getBooks().empty());

// Тест методов displayByAuthor и displayByYear

Book book2("Author2", "Title2", 2018, 8);

Book book3("Author3", "Title3", 2022, 3);

library.addBook(book1);

library.addBook(book2);

library.addBook(book3);

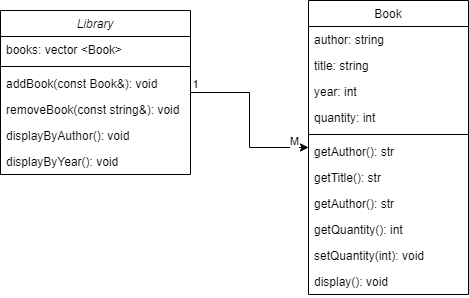
}

void runTests() {

testBook();

testLibrary();

}



Вывод: изучил стандартные библиотеки шаблонов и научился реализовать их.