МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ  
  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА(САМАРСКИЙ УНИВЕРСИТЕТ)  
  
Институт информатики и кибернетики  
Кафедра программных систем

Дисциплина  
**Объектно-ориентированное программирование**

**ОТЧЕТ**по лабораторной работе № 4  
 **Объектно-ориентированное программирование обработки строк с использованием функций стандартной библиотеки языка С++ в стиле языка С.**  
Вариант № 12

Студент: Мананников М.А.  
Группа: 6103-020302D  
  
Преподаватель: Баландин А.В.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2022

**Текст задания**

Написать программу управления информацией о книгах, находящихся в библиотеке,

состоящей из двух списков:

1) Динамически формируемый список записей о книгах вида:

{<шифр\_автора>;

<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:

{<индекс\_книги>;

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

<индекс\_книги> - порядковый номер 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, имеющиеся в библиотеке

на полках.

**Объектно-ориентированный анализ (ООА)**

Информационная модель предметной области

Информационный объект «Запись в список книг»

Свойства:

• Шифр автора;

• Название книги;

• Количество экземпляров.

Информационный объект «Запись в список авторов»

Свойства:

• Шифр автора;

• Имя автора.

• Информационный объект «Запись в список индексов»

Свойства:

• Индекс записи в списке книг;

• Индекс записи в списке авторов;

• Количество экземпляров.

Процедуры:

• Добавление книг в список книг;

• Добавление возвращенной книги;

• Выдача книги на руки;

• Вывод на экран всех книг в библиотеке;

• Вывод на экран всех авторов;

• Вывод на экран всех книг определенного автора;

• Освобождение памяти.

**Объектно-ориентированное проектирование (OOD)**

Информационный объект «Запись в список книг»

Свойства:

• Название книги – символьная строка;

• Шифр автора– положительное целое число.

• Количество экземпляров – положительное целое число.

• Информационный объект «Запись в список авторов»

Свойства:

• Имя автора – символьная строка;

• Шифр автора – положительное целое число.

• Информационный объект “Запись в списке индексов”.

Свойства:

• Индекс записи в списке книг - положительное целое число;

• Индекс записи в списке авторов- указатель записи в списке.

• Количество книг - положительное целое число.

Процедуры:

• Добавление книг в список книг - функция;

• Добавление возвращенной книги - функция;

• Выдача книги на руки - функция;

• Вывод на экран всех книг в библиотеке - функция;

• Вывод на экран всех авторов - функция;

• Вывод на экран всех книг определенного автора - функция;

• Освобождение памяти - функция.

**Структура алгоритма программы**

• Добавление книг в список книг (AddBook);

• Выдача книги из библиотеки, удаляя из списка (GiveBook);

• Возвращение книги, добавление в список (GiveBackBook);

• Вывод на экран всех книг (PrintAll);

• Вывод на экран всех авторов (PrintAutors);

• Вывод на экран всех книг определённого автора (PrintBooksofAutor).

**Текст программы**

**Класс Program**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <string.h>//библиотека для работы со си-строками

#include <stdio.h> // стандартная библиотека ввода\вывода

#include <stdlib.h>// поиск, сортировка и тд

#include <vector> // подключаем библиотеку для работы с векторами

class DataBase

{

struct BookStruct//структура книга

{

char\* shifrautor = (char\*)malloc(20);

char\* namebook = (char\*)malloc(20);

int numbofbook;

};

struct AutorStruct//структура автора

{

char\* shifrautor = (char\*)malloc(20);

char\* nameautor = (char\*)malloc(20);

};

struct IndexStruct//структура индекса

{

int indexofbook;

int numbofbook;

};

std::vector<BookStruct\*> books;//вектор с элеменатами структуры книга

std::vector<AutorStruct\*> autors;//вектор с элементами структуры автора

std::vector<IndexStruct\*> indexes;//вектор с элементами структуры индекс

public: void AddBook() //добавление книг

{

int n = 0;

puts("Введите количество поступивших книг");

try

{

START:

fflush(stdin);

scanf("\n%d", &n);

if (n == 0)

{

throw (char\*)"Неверный ввод количества!";

}

}

catch (const char\* exception)

{

while (getchar() != '\n');

puts(exception);

goto START;

}

for (int i = 0; i < n; i++)

{

char\* namebook = (char\*)malloc(20);//буфер записи названи

char\* nameautor = (char\*)malloc(20);//буфер записи имени автора

char\* shifrautor = (char\*)malloc(20);//буфер записи шифра

int k = 1;

puts("Введите название книги");

try //проверка исключения

{

BOOK\_1:

fflush(stdin);

scanf("%s", namebook);

if (namebook[0] <= '9' && namebook[0] >= '0')

{

throw(char\*)"Вы ввели цифру(ы), введите строку!";

}

}

catch (const char\* exception)

{

while (getchar() != '\n');

puts(exception);

goto BOOK\_1;

}

puts("Введите автора книги");

try //проверка исключения

{

AUTOR:

fflush(stdin);

scanf("%s", nameautor);

if (nameautor[0] <= '9' && nameautor[0] >= '0')

{

throw(char\*)"Вы ввели цифру(ы), введите строку!";

}

}

catch (const char\* exception)

{

while (getchar() != '\n');

puts(exception);

goto AUTOR;

}

puts("Введите шифр автора книги");

try//проверка исключения

{

fflush(stdin);

scanf("%s", shifrautor);

if (shifrautor == "")

{

throw(char\*)"Вы ввели некоректные данные!";

}

}

catch (const char\* exception)

{

puts(exception);

}

BookStruct\* bs = new BookStruct; //создание экземпляра структуры

AutorStruct\* as = new AutorStruct; //создание экземпляра структуры

IndexStruct\* is = new IndexStruct; //создание экземпляра структуры

int check = 0;

for (size\_t j = 0; j < books.size(); j++)

{

if (strcmp(books[j]->namebook, namebook) == 0 && strcmp(books[j]->shifrautor, shifrautor) == 0)//проверка на существование такой книги

{

books[j]->numbofbook++;

check++;

for (int b = 0; b < indexes.size(); b++)

{

if (indexes[b]->indexofbook == j)

{

indexes[b]->numbofbook = books[j]->numbofbook;

}

}

}

}

if (check == 0)//добавление книги, если ее не было

{

bs->namebook = namebook;

bs->numbofbook = k;

bs->shifrautor = shifrautor;

is->indexofbook = books.size();

is->numbofbook = k;

indexes.push\_back(is);//добавление в конец списка

books.push\_back(bs);// добавление в конец списка

}

check = 0;

for (int j = 0; j < autors.size(); j++)//проверка на существование автора

{

if (strcmp(autors[j]->nameautor, nameautor) == 0 && strcmp(autors[j]->shifrautor, shifrautor) == 0)

{

check++;

}

}

if (check == 0)//добвление автора

{

as->nameautor = nameautor;

as->shifrautor = shifrautor;

autors.push\_back(as);

}

}

}

public: void GiveBackBook()//возврат книги

{

char\* namebook = (char\*)malloc(20);//буфер записи названия книги

char\* nameautor = (char\*)malloc(20);//буфер записи имени автора

char\* shifrautor = (char\*)malloc(20);//буфер записи шифра

int k = 1;

puts("Введите название книги");

try//проверка исключения

{

BOOK\_2:

fflush(stdin);

scanf("%s", namebook);

if (namebook[0] <= '9' && namebook[0] >= '0')

{

throw(char\*)"Вы ввели цифру(ы), введите строку!";

}

}

catch (const char\* exception)

{

while (getchar() != '\n');

puts(exception);

goto BOOK\_2;

}

puts("Введите автора книги");

try//проверка исключения

{

AUTOR\_1:

fflush(stdin);

scanf("%s", nameautor);

if (nameautor[0] <= '9' && nameautor[0] >= '0')

{

throw(char\*)"Вы ввели цифру(ы), введите строку!";

}

}

catch (const char\* exception)

{

while (getchar() != '\n');

puts(exception);

goto AUTOR\_1;

}

puts("Введите шифр автора книги");

try//проверка исключения

{

fflush(stdin);

scanf("%s", shifrautor);

if (shifrautor == "")

{

throw(char\*)"Вы ввели некоректные данные!";

}

}

catch (const char\* exception)

{

puts(exception);

}

BookStruct\* bs = new BookStruct;

AutorStruct\* as = new AutorStruct;

IndexStruct\* is = new IndexStruct;

int check = 0;

for (size\_t j = 0; j < books.size(); j++)

{

if (strcmp(books[j]->namebook, namebook) == 0 && strcmp(books[j]->shifrautor, shifrautor) == 0)//проверка на существование такой книги

{

books[j]->numbofbook++;

check++;

for (int b = 0; b < indexes.size(); b++)

{

if (indexes[b]->indexofbook == j)

{

indexes[b]->numbofbook = books[j]->numbofbook;

}

}

}

}

if (check == 0)//добавление книги

{

bs->namebook = namebook;

bs->numbofbook = k;

bs->shifrautor = shifrautor;

is->indexofbook = books.size();

is->numbofbook = k;

indexes.push\_back(is);

books.push\_back(bs);

}

check = 0;

for (int j = 0; j < autors.size(); j++)//проверка на существование автора

{

if (strcmp(autors[j]->nameautor, nameautor) == 0 && strcmp(autors[j]->shifrautor, shifrautor) == 0)

{

check++;

}

}

if (check == 0)//добавление автора

{

as->nameautor = nameautor;

as->shifrautor = shifrautor;

autors.push\_back(as);

}

}

public: void GiveBook()//выдача книги на руки

{

char\* namebook = (char\*)malloc(20);//буфер записи названия книги

char\* nameautor = (char\*)malloc(20);//буфер записи имени автора

char\* shifrautor = (char\*)malloc(20);//буфер записи шифра

puts("Введите название книги");

try//проверка исключения

{

BOOK\_3:

fflush(stdin);

scanf("%s", namebook);

if (namebook[0] <= '9' && namebook[0] >= '0')

{

throw(char\*)"Вы ввели цифру(ы), введите строку!";

}

}

catch (const char\* exception)

{

while (getchar() != '\n');

puts(exception);

goto BOOK\_3;

}

puts("Введите автора книги");

try//проверка исключения

{

AUTOR\_2:

fflush(stdin);

scanf("%s", nameautor);

if (nameautor[0] <= '9' && nameautor[0] >= '0')

{

throw(char\*)"Вы ввели цифру(ы), введите строку!";

}

}

catch (const char\* exception)

{

while (getchar() != '\n');

puts(exception);

goto AUTOR\_2;

}

puts("Введите шифр автора книги");

try//проверка исключения

{

fflush(stdin);

scanf("%s", shifrautor);

if (shifrautor == "")

{

throw(char\*)"Вы ввели некоректные данные!";

}

}

catch (const char\* exception)

{

puts(exception);

}

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

{

if (strcmp(books[i]->namebook, namebook) == 0 && strcmp(books[i]->shifrautor, shifrautor) == 0)//поиск книги в списке

{

for (int j = 0; j < indexes.size(); j++)

{

if (indexes[j]->indexofbook == i)

{

indexes[j]->numbofbook--;

if (indexes[j]->numbofbook == 0)//удаление книги из списка при нулевом количестве

{

indexes.erase(indexes.begin() + j);

for (int t = j; t < indexes.size(); t++)

{

indexes[t]->indexofbook--;

}

}

}

}

books[i]->numbofbook--;

if (books[i]->numbofbook == 0)

{

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

}

}

}

}

public: void printAll()//печать всех книг

{

puts("Название книги\tКоличество\tИмя автора\t");

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

{

printf("%s\t\t%d\t", books[i]->namebook, books[i]->numbofbook);

for (int j = 0; j < autors.size(); j++)

{

if (strcmp(books[i]->shifrautor, autors[j]->shifrautor) == 0)

{

printf("%s\n", autors[j]->nameautor);

}

}

}

}

public: void printAutors()//печать всех авторов

{

for (int j = 0; j < autors.size(); j++)

{

printf("%s\t", autors[j]->nameautor);

}

puts("");

}

public: void printBooksofAutor()//печать всех книг определенного автора

{

char\* nameautor = (char\*)malloc(20);

puts("Введите автора книги");

scanf("%s", nameautor);

if (nameautor[0] <= '9' && nameautor[0] >= '0')

throw (char\*)"Вы не ввели имя автора!";

for (int j = 0; j < autors.size(); j++)

{

if (strcmp(autors[j]->nameautor, nameautor) == 0)

{

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

{

if (strcmp(books[i]->shifrautor, autors[j]->shifrautor) == 0)

{

printf("%s\t", books[i]->namebook);

}

}

}

}

}

~DataBase()//деструктор

{

books.clear();

autors.clear();

indexes.clear();

}

};

int main()

{

system("chcp 1251");//Разрешить кириллицу для консольного ввода/вывода

printf("Лабораторная работа 4 (12 вариант) подготовил студент группы 6103 Мананников Максим\n");

DataBase db;//создание экземпляра класса

puts("\nДобавление книги");

db.AddBook();

puts("\nВыдача книги");

db.GiveBook();

puts("\nВозвращение книги");

db.GiveBackBook();

puts("\n\nПечать всех книг");

db.printAll();

puts("\nПечать всех авторов");

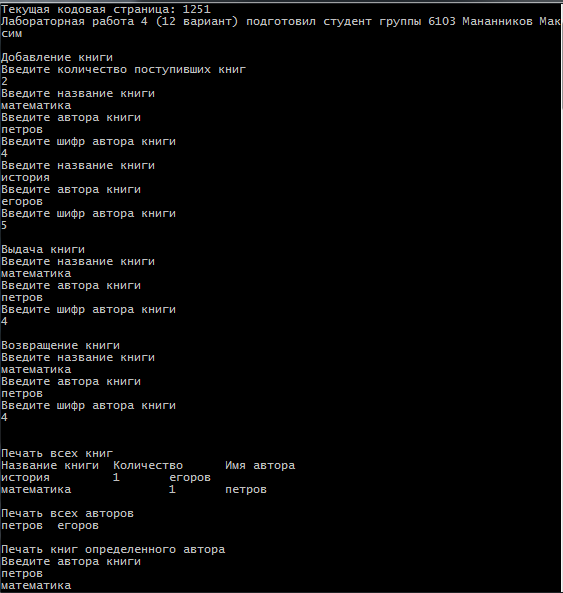
db.printAutors();

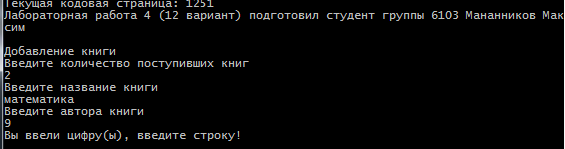
puts("\nПечать книг определенного автора");

db.printBooksofAutor();

}

**Пример работы программы**

Рисунок 1 – Пример работы программы

Рисунок 2 - Пример работы исключительных ситуаций