Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Классы и объекты. Использование конструкторов.

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Токарев Павел Аркадьевич

Проверил

Доцент кафедры ИТАС

Полякова Ольга Андреевна

Г. Пермь-2023

**Постановка задачи**

1.Реализовать определение нового класса.

2.Поля класса: Название-string, Автор-string, Год издания-int.

3.Определить в классе конструктор без параметров, с параметрами, копированием и деструктор.

4.Определить в классе селекторы и модификаторы.

4.В главной функции продемонстрировать вызов всех трёх конструкторов.

**Анализ**

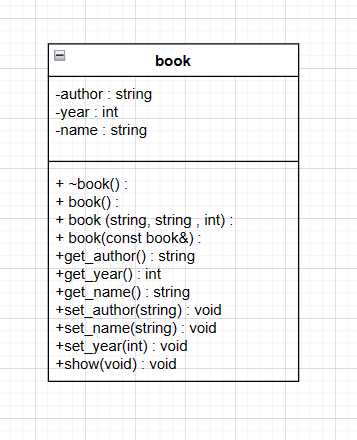
1.Определим класс book и его методы в заголовочном файле Book.h

2.Конструктор без параметров создаёт объект со всеми полями, равными 0.

3.Конструктор с параметрами при создании объекта присваивает его полям значения, переданные ему в качестве аргументов.

4.Конструктор копированием присваивает полям объекта поля класса, ссылка на который ему переданна.

**Диаграмма UML**

****

**Код программы**

main.cpp

#include <iostream>

#include"book.h"

using namespace std;

book createbook() {

string a, b;

int c;

cout << "Введите название книги:";

cin >> a;

cout << "Введите авторы книги:";

cin >> b;

cout << "Введите год издания:";

cin >> c;

book tmp(a, b, c);

return tmp;

}

void printbook(book tmp) {

tmp.show();

}

int main()

{

setlocale(LC\_ALL, "rus");

book ex1;

ex1.show();

book ex2("Преступление и наказание", "Ф. М. Достоевский", 1866);

ex2.show();

book ex3 = ex2;

ex3.show();

return 0;

}

book.h

#pragma once

#include <iostream>

using namespace std;

class book {

string name;

string author;

int year;

public:

book();

book(string, string, int);

book(const book&);

~book();

string getname();

string getauthor();

int getyear();

void setname(string);

void setauthor(string);

void setyear(int);

void show();

};

book.cpp

#include<iostream>

#include"book.h"

using namespace std;

book::book() {

name = "";

author = "";

year = 0;

cout << "Вызван конструктор без параметров" << endl;

}

book::book(string nam, string autho, int yer) {

name = nam;

author = autho;

year = yer;

cout << "Вызван конструктор с параметрами" << endl;

}

book::book(const book& t) {

this->name = t.name;

this->author = t.author;

this->year = t.year;

cout << "Вызван конструктор копированием" << endl;

}

book::~book() {

cout << "Вызван деструктор" << endl;

};

string book::getname() {

return name;

}

string book::getauthor() {

return author;

}

int book::getyear() {

return year;

}

void book::setname(string tmp) {

name = tmp;

}

void book::setauthor(string tmp) {

author = tmp;

}

void book::setyear(int tmp) {

year = tmp;

}

void book::show() {

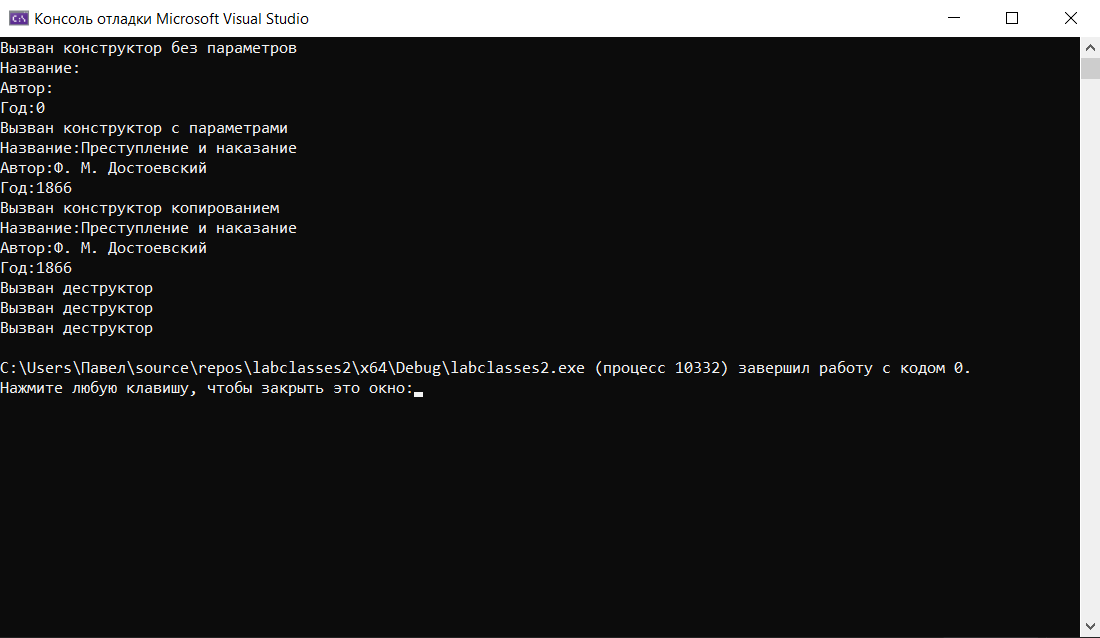
cout << "Название:" << name << endl;

cout << "Автор:" << author << endl;

cout << "Год:" << year << endl;

}

**Результаты работы программы**

****

**Анализ результатов**

Как видно из результата работы программы, без ошибок выполняется создание объектов класса всеми конструкторами.

**Контрольные вопросы**

1.Конструктор-метод, использующийся для инициализации объекта.

2. В C++ существует 3 типа конструкторов (без параметров, с параметрами, копированием).

3.Деструктор-методы, применяющийся для освобождения памяти, выделенной созданному объекту. Описывать деструктор явно требуется, если объект содержит указатели на память, выделяемую динамически.

4. Конструктор без параметров используется для создания объекта с пустыми полями.

Конструктор с параметрами используется для создания объекта с заданными полями.

Конструктор копирования используется для создания объекта с полями, идентичными полям уже существующего объекта.

5.Конструктор вызывается, когда объект создаётся путём копирования существующего.

6.Конструктор не возвращает значения. Класс может иметь несколько конструкторов. Если конструктор не указан, то компилятор создаёт его автоматически. Конструкторы не наследуются.

7.Деструктор удаляется автоматически, когда объект удаляется из памяти. Имя деструктора начинается с ~. Если деструктор не определён, то компилятор автоматически создаёт пустой деструктор.

8.Методы класса имеют доступ ко всем элементам класса вне зависимости от спецификатора доступа.

9.Указатель this указывает на текущий объект данного класса.

10.Методы имеют неограниченный доступ к всем элементам класса.

11.Конструктор не возвращает значения. Даже типа void.

12.По умолчанию компилятор создаёт конструктор и деструктор.

13.Деструктор не возвращает значения. Даже типа void.

14.Отсутствует конструктор без параметров.

15.Будет вызван конструктор без параметров.

16.Будет вызван конструктор с параметрами.

17.Сначала при создании объекта s1 будет вызван конструктор с параметрами (параметры в аргументах). После этого для определения объекта s2 будет вызван конструктор копированием.

18.При создании объекта s1 будет вызван конструктор с параметрами. Далее создаётся объект s2 вызывается конструктор без параметров. Для объекта s3 вызывается конструктор копированием.

19.При передаче объекта как аргумента функции используется конструктор копированием.

20. Объекту p можно изменить значения поля name с помощью модификатора set\_name.

p.setname(“name”);