**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,
СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

Ордена Трудового Красного Знамени
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский технический университет связи и информатики»

Факультет «Информационные технологии»
Кафедра «Математическая кибернетика и информационные технологии»
Дисциплина: «Информационные технологии и программирование»

Лабораторная работа №2

Объектно-ориентированное программирование в Java

Выполнил:
студент группы БВТ2402
Деминова Виктория

Москва
2025

Цель работы

Ознакомление с базовыми принципами работы ООП в Java.

Задачи

Создайте иерархию классов. Ваша иерархия должна содержать:

• абстрактный класс;

• два уровня наследуемых классов (классы должны содержать в себе минимум 3 поля и 2 метода, описывающих поведение объекта);

• демонстрацию реализации всех принципов ООП;

• наличие конструкторов (в том числе по умолчанию);

• наличие геттеров и сеттеров;

• ввод/вывод информации о создаваемых объектах;

• предусмотрите в одном из классов создание счетчика созданных объектов с использованием статической переменной, продемонстрируйте работу.

Базовый класс: Книга. Дочерние классы: Аудиокнига, Фильм, Мюзикл.

Ход работы

Импорт и абстрактный класс Книга.

```
import java.util.Scanner;

abstract class Book {
    private String title;
    private String author;
    private int year;

    public Book(String title, String author, int year) {
        this.title = title;
        this.author = author;
        this.year = year;
    }

    public Book() {
        this.title = "Неизвестно";
        this.author = "Неизвестен";
        this.year = 0;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }

    public int getYear() {
        return year;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public abstract void playback();

    public void available() {
        System.out.println("Книга в продаже");
    }

    public void getInfo() {
        System.out.println("Название: " + title + ", Автор: " + author + ", Год:
" + year);
    }
}
```

Класс Аудиокнига - наследник абстрактного класса Книга.

```
class Audiobook extends Book {
    private int minutes;

    public Audiobook(String title, String author, int year, int minutes) {
        super(title, author, year);
        this.minutes = minutes;
    }

    public Audiobook() {
        super();
        this.minutes = 0;
    }

    public int getMinutes() {
        return minutes;
    }

    public void setMinutes(int minutes) {
        this.minutes = minutes;
    }

    @Override
    public void playback() {
        System.out.println("Аудиокнига воспроизводится");
    }

    @Override
    public void available() {
        System.out.println("Аудиокнига в продаже");
    }

    @Override
    public void getInfo() {
        super.getInfo();
        System.out.println("Длительность: " + minutes + " минут");
    }
}
```

Класс Фильм - наследник класса Аудиокнига.

```
class Movie extends Audiobook {
    private static int counter = 0;

    public Movie(String title, String author, int year, int minutes) {
        super(title, author, year, minutes);
        counter++;
    }

    public Movie() {
        super();
        counter++;
    }

    public static int getCounter() {
        return counter;
    }

    @Override
    public void playback() {
        System.out.println("Фильм идёт");
    }

    @Override
    public void available() {
        System.out.println("Фильм в прокате");
    }
}
```

Класс Мюзикл - наследник класса Аудиокнига.

```
class Musical extends Audiobook {

    public Musical(String title, String author, int year, int minutes) {
        super(title, author, year, minutes);
    }

    public Musical() {
        super();
    }

    @Override
    public void playback() {
        System.out.println("Мюзикл идёт");
    }

    @Override
    public void available() {
        System.out.println("Мюзикл в прокате");
    }
}
```

Класс Main – главный класс с демонстрацией работы программы.

```
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Введите название: ");
        String aTitle = scanner.nextLine();
        System.out.print("Введите автора: ");
        String anAuthor = scanner.nextLine();
        System.out.print("Введите год выпуска: ");
        int anYear = scanner.nextInt();
        System.out.print("Введите длительность (мин): ");
        int aDuration = scanner.nextInt();
        scanner.close();
        Audiobook audiobook1 = new Audiobook(aTitle, anAuthor, anYear,
aDuration);
        Movie movie1 = new Movie("Оппенгеймер", "Кристофер Нолан", 2023, 180);
        Musical musical1 = new Musical();
        audiobook1.setTitle("Большой Соня");
        Book[] library = {audiobook1, movie1, musical1};
        for (Book item : library) {
            item.getInfo();
            item.playback();
        }
        Movie movie2 = new Movie("Аватар", "Джеймс Кэмерон", 2009, 162);
        System.out.println("Всего фильмов: " + Movie.getCounter());
    }
}
```

Вывод программы:

```
Введите название: НИ СЫ
Введите автора: Джен Синсеро
Введите год выпуска: 2018
Введите длительность (мин): 325
Название: Большой Соня, Автор: Джен Синсеро, Год: 2018
Длительность: 325 минут
Аудиокнига воспроизводится
Название: Оппенгеймер, Автор: Кристофер Нолан, Год: 2023
Длительность: 180 минут
Фильм идёт
Название: Неизвестно, Автор: Неизвестен, Год: 0
Длительность: 0 минут
Мюзикл идёт
Всего фильмов: 2

Process finished with exit code 0
```

Вывод

В результате выполнения лабораторной работы были изучены основные принципы работы ООП в Java и реализованы базовый класс: Книга и дочерние классы: Аудиокнига, Фильм, Мюзикл.

Примечания

Ссылка на репозиторий на GitHub: https://github.com/AnlayKor/IT