**Лабораторная работа №15**

**Основы программирования на Java: проектирование классов**

**Цель работы:** Освоить основные способы создания Java-программ либо с помощью обычного редактора, либо с помощью среды разработки. Приобрести навыки проектирования классов в Java.

**Задание 1**

Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setTun(), getTun(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль. В каждом классе, обладающем информацией, должно быть объявлено несколько конструкторов.

Book: id, Название, Автор (ы), Издательство, Год издания, Количество страниц, Цена, Тип переплета:

Создать массив объектов. Вывести:

а) список книг заданного автора;

Ь) список книг, выпущенных заданным издательством;

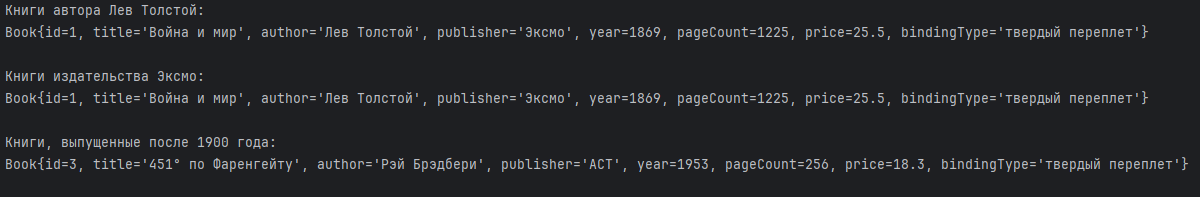
с) список книг, выпущенных после заданного года.

**Первый класс:**

import java.util.ArrayList;  
  
class Book {  
 private int id;  
 private String title;  
 private String author;  
 private String publisher;  
 private int year;  
 private int pageCount;  
 private double price;  
 private String bindingType;  
  
 public Book(int id, String title, String author, String publisher, int year, int pageCount, double price, String bindingType) {  
 this.id = id;  
 this.title = title;  
 this.author = author;  
 this.publisher = publisher;  
 this.year = year;  
 this.pageCount = pageCount;  
 this.price = price;  
 this.bindingType = bindingType;  
 }  
  
 // Геттеры и сеттеры  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public void setTitle(String title) {  
 this.title = title;  
 }  
  
 public String getAuthor() {  
 return author;  
 }  
  
 public void setAuthor(String author) {  
 this.author = author;  
 }  
  
 public String getPublisher() {  
 return publisher;  
 }  
  
 public void setPublisher(String publisher) {  
 this.publisher = publisher;  
 }  
  
 public int getYear() {  
 return year;  
 }  
  
 public void setYear(int year) {  
 this.year = year;  
 }  
  
 public int getPageCount() {  
 return pageCount;  
 }  
  
 public void setPageCount(int pageCount) {  
 this.pageCount = pageCount;  
 }  
  
 public double getPrice() {  
 return price;  
 }  
  
 public void setPrice(double price) {  
 this.price = price;  
 }  
  
 public String getBindingType() {  
 return bindingType;  
 }  
  
 public void setBindingType(String bindingType) {  
 this.bindingType = bindingType;  
 }  
  
 @Override  
 public String toString() {  
 return "Book{" +  
 "id=" + id +  
 ", title='" + title + '\'' +  
 ", author='" + author + '\'' +  
 ", publisher='" + publisher + '\'' +  
 ", year=" + year +  
 ", pageCount=" + pageCount +  
 ", price=" + price +  
 ", bindingType='" + bindingType + '\'' +  
 '}';  
 }  
}

**Второй класс:**

java.util.ArrayList;  
  
public class Main {  
 public static void main(String[] args) {  
 ArrayList<Book> library = new ArrayList<>();  
  
 // Добавляем книги в библиотеку  
 library.add(new Book(1, "Война и мир", "Лев Толстой", "Эксмо", 1869, 1225, 25.5, "твердый переплет"));  
 library.add(new Book(2, "Преступление и наказание", "Ф.М. Достоевский", "Азбука", 1866, 331, 12.8, "мягкий переплет"));  
 library.add(new Book(3, "451° по Фаренгейту", "Рэй Брэдбери", "ACT", 1953, 256, 18.3, "твердый переплет"));  
  
 // Вывод списка книг заданного автора  
 String author = "Лев Толстой";  
 System.*out*.println("Книги автора " + author + ":");  
 for (Book book : library) {  
 if (book.getAuthor().equals(author)) {  
 System.*out*.println(book);  
 }  
 }  
  
 // Вывод списка книг, выпущенных заданным издательством  
 String publisher = "Эксмо";  
 System.*out*.println("\nКниги издательства " + publisher + ":");  
 for (Book book : library) {  
 if (book.getPublisher().equals(publisher)) {  
 System.*out*.println(book);  
 }  
 }  
  
 // Вывод списка книг, выпущенных после заданного года  
 int year = 1900;  
 System.*out*.println("\nКниги, выпущенные после " + year + " года:");  
 for (Book book : library) {  
 if (book.getYear() > year) {  
 System.*out*.println(book);  
 }  
 }  
 }  
}

**Вывод**

**Задание 2**

определить класс Комплекс, действительная и мнимая часть которой представлены в виде Рациональной Дроби. Создать массив/список/множество размерности n из комплексных координат. Передать его в метод, который выполнит сложение/умножение его элементов.

**Первый класс:**

class Rational {  
 private int num; // числитель  
 private int den; // знаменатель  
  
 public Rational(int num, int den) {  
 this.num = num;  
 this.den = den;  
 }  
  
 public int getNum() {  
 return num;  
 }  
  
 public int getDen() {  
 return den;  
 }  
  
 public Rational add(Rational other) {  
 int newNum = this.num \* other.den + other.num \* this.den;  
 int newDen = this.den \* other.den;  
 return simplify(newNum, newDen);  
 }  
  
 public Rational multiply(Rational other) {  
 int newNum = this.num \* other.num;  
 int newDen = this.den \* other.den;  
 return simplify(newNum, newDen);  
 }  
  
 private int gcd(int a, int b) {  
 return b == 0 ? a : gcd(b, a % b);  
 }  
  
 private Rational simplify(int num, int den) {  
 int divisor = gcd(num, den);  
 return new Rational(num / divisor, den / divisor);  
 }  
  
 @Override  
 public String toString() {  
 return num + "/" + den;  
 }  
}

**Второй класс:**

class Complex {  
 private Rational realPart; // действительная часть  
 private Rational imaginaryPart; // мнимая часть  
  
 public Complex(Rational realPart, Rational imaginaryPart) {  
 this.realPart = realPart;  
 this.imaginaryPart = imaginaryPart;  
 }  
  
 public Complex add(Complex other) {  
 Rational newRealPart = realPart.add(other.realPart);  
 Rational newImaginaryPart = imaginaryPart.add(other.imaginaryPart);  
 return new Complex(newRealPart, newImaginaryPart);  
 }  
  
 public Complex multiply(Complex other) {  
 Rational newRealPart = realPart.multiply(other.realPart).add(imaginaryPart.multiply(other.imaginaryPart).multiply(new Rational(-1, 1)));  
 Rational newImaginaryPart = realPart.multiply(other.imaginaryPart).add(imaginaryPart.multiply(other.realPart));  
 return new Complex(newRealPart, newImaginaryPart);  
 }  
  
 @Override  
 public String toString() {  
 return realPart.toString() + " + " + imaginaryPart.toString() + "i";  
 }  
}

**Третий класс:**

public class Main {  
 public static void main(String[] args) {  
 int n = 5; // размерность массива/списка/множества  
  
 // создание массива комплексных чисел  
 Complex[] complexArray = new Complex[n];  
 for (int i = 0; i < n; i++) {  
 Complex complexNumber = new Complex(new Rational(i, 1), new Rational(i + 1, 1));  
 complexArray[i] = complexNumber;  
 }  
  
 // выполнение сложения элементов массива  
 Complex sum = *sumComplexArray*(complexArray);  
 System.*out*.println("Сумма: " + sum);  
  
 // выполнение умножения элементов массива  
 Complex product = *multiplyComplexArray*(complexArray);  
 System.*out*.println("Произведение: " + product);  
 }  
  
 public static Complex sumComplexArray(Complex[] array) {  
 Complex sum = new Complex(new Rational(0, 1), new Rational(0, 1));  
 for (Complex complexNumber : array) {  
 sum = sum.add(complexNumber);  
 }  
 return sum;  
 }  
  
 public static Complex multiplyComplexArray(Complex[] array) {  
 Complex product = new Complex(new Rational(1, 1), new Rational(0, 1));  
 for (Complex complexNumber : array) {  
 product = product.multiply(complexNumber);  
 }  
 return product;  
 }  
}

**Вывод**

****