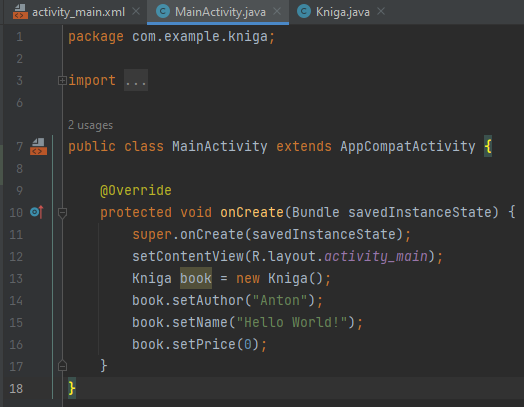
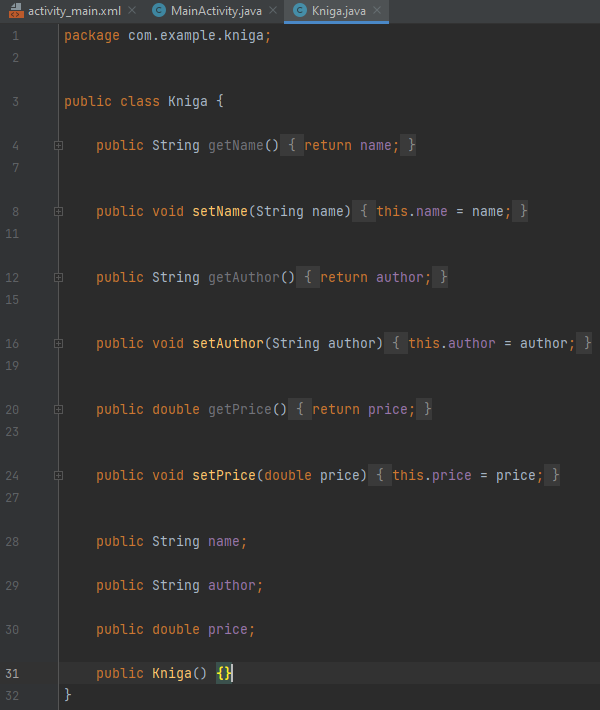
Лаба3 практическое задание варианта номер 10:





Контрольные ответы на контрольные вопросы лабы номер 3:

1. Встроенные типы данных в Java:

- Целочисленные типы: byte, short, int, long

- Типы с плавающей точкой: float, double

- Символьный тип: char

- Типы логических значений: boolean

2. Для объявления одномерного массива в Java используется следующий синтаксис: тип\_элементов[] имя\_массива = new тип\_элементов[размер];

Например:

int[] numbers = new int[5];

3. Для объявления двумерного массива в Java используется следующий синтаксис: тип\_элементов[][] имя\_массива = new тип\_элементов[размер\_первого\_измерения][размер\_второго\_измерения];

Например:

int[][] matrix = new int[3][3];

4. В Java для работы со строками доступны различные методы из класса String:

- length(): возвращает длину строки.

- charAt(index): возвращает символ по указанному индексу.

- substring(beginIndex, endIndex): возвращает подстроку, начиная с указанного индекса и до индекса endIndex - 1.

- equals(otherString): сравнивает текущую строку с другой строкой на равенство.

- toUpperCase(): преобразует строку к верхнему регистру.

- toLowerCase(): преобразует строку к нижнему регистру.

- split(delimiter): разделяет строку на подстроки с помощью разделителя и возвращает массив подстрок.

- и другие.

5. В Java для создания списка используется интерфейс List и его реализации, такие как ArrayList или LinkedList. Пример создания списка:

import java.util.List;

import java.util.ArrayList;

public class Main {

public static void main(String[] args) {

List<String> list = new ArrayList<>();

list.add("Элемент 1");

list.add("Элемент 2");

list.add("Элемент 3");

// Доступ к элементам списка

System.out.println(list.get(0)); // Вывод: Элемент 1

System.out.println(list.get(1)); // Вывод: Элемент 2

System.out.println(list.get(2)); // Вывод: Элемент 3

}

}

6. Модификатор доступа в классе определяет, каким образом другие классы или объекты могут получить доступ к полям и методам этого класса. Модификатор доступа ограничивает видимость элементов класса.

7. В Java доступны следующие модификаторы доступа в классе:

- public: элемент доступен всем классам и пакетам.

- protected: элемент доступен внутри пакета и в производных классах.

- без модификатора (default, package-private): элемент доступен только внутри пакета.

- private: элемент доступен только внутри самого класса.

8. Поле (переменная) класса — это переменная, объявленная внутри класса, которая хранит данные для использования в методах этого класса. Поля класса хранят состояние объекта и могут иметь разные типы данных.

9. Метод класса — это блок кода, который выполняет определенные действия или операции. Он объявляется внутри класса и может принимать параметры и возвращать значение. Методы класса используются для выполнения различных операций и взаимодействия с полями класса.

10. Наследование в Java — это механизм, позволяющий классу наследовать свойства и методы другого класса. Класс, от которого наследуется другой класс, называется базовым классом (или суперклассом), а класс, который наследует свойства, называется производным классом (или подклассом). Подкласс наследует все неприватные поля и методы базового класса.

11. Иерархия классов — это организация классов в виде иерархической древовидной структуры, где каждый класс наследует свойства и методы от одного или нескольких родительских классов. Иерархия классов позволяет организовать код более структурированно и переиспользовать функционал базовых классов.

12. Поля и методы классов наследуются в подклассе, если он наследует от другого класса. Класс-потомок получает все доступные поля и методы класса-родителя, которые не являются приватными.

При наследовании полей, они становятся доступными в подклассе без необходимости их повторного определения. Подкласс может использовать, изменять и добавлять новые поля.

При наследовании методов, они также становятся доступными в подклассе. Подкласс может использовать и изменять эти методы с помощью переопределения (override). При переопределении подкласс может изменить реализацию метода, заменив его собственной версией.

Класс-потомок может также добавлять свои собственные поля и методы, которые не существуют в классе-родителе. Это позволяет подклассу иметь свои собственные специфичные для него свойства и поведение.

Кроме того, класс-потомок может обращаться к полям и методам класса-родителя с помощью ключевого слова super.