Министерство образования и науки РФ

Санкт-Петербургский политехнический университет

Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа компьютерных технологий и информационных систем

УДК \_\_\_\_\_\_\_\_\_\_\_\_

УТВЕРЖДАЮ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_ г.

**ОТЧЁТ**

**Лабораторная работа №12 по теме «Fundamentals of Java»**

по дисциплине «Системный подход в разработке программного обеспечения»

**Выполнил:**

студент гр. 5130902/20201 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. И. Сафонов

подпись, дата

**Проверила:**

Доцент к. т. н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_С.А. Нестеров

подпись, дата

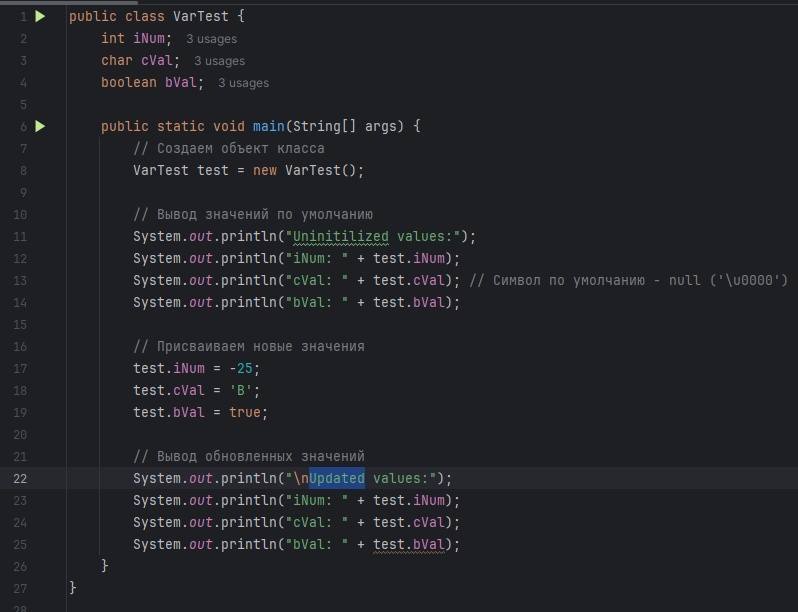
Санкт-Петербург

2024

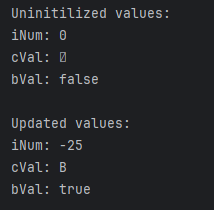
**Занятие 1.1**

**Упражнение 1.**

Код:

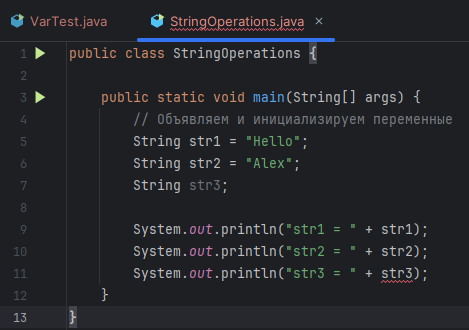


Результат:

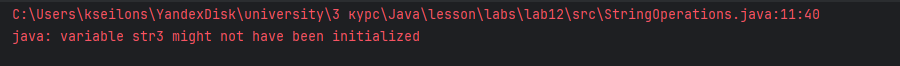


**Упражнение 2.**

Код:

****

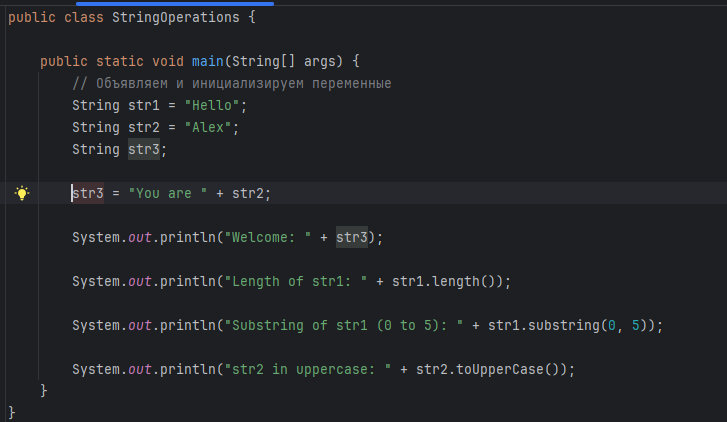
Результат:



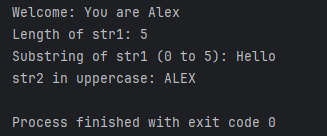
Если попытаться использовать str3 без инициализации, компилятор выдаст ошибку. Например, следующий код: System.out.println("str3: " + str3); вызовет ошибку компиляции, так как str3 не инициализирована.

**Упражнение 3**

Код:

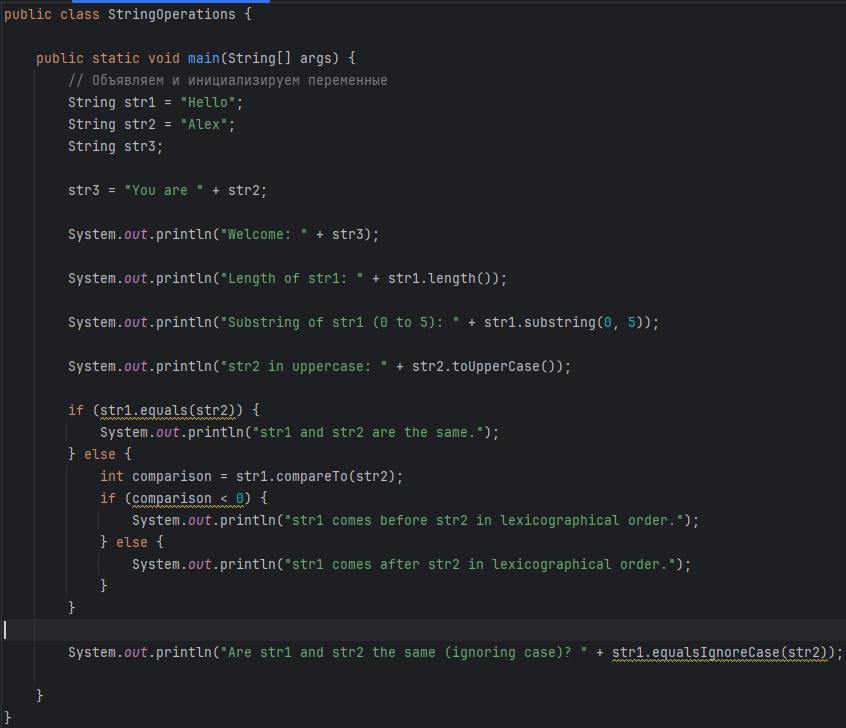


Результат:

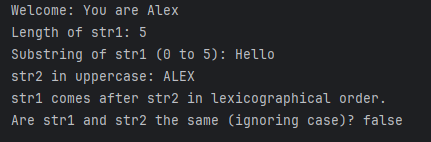


**Упражнение 4.**

Код:

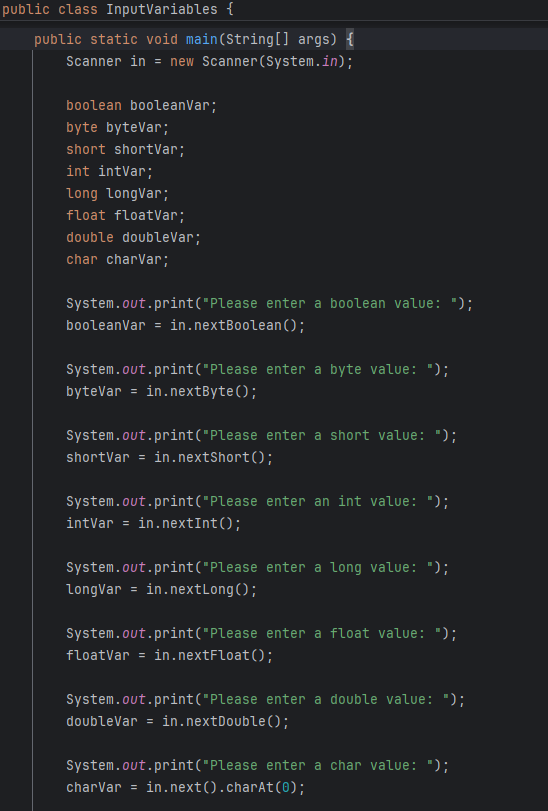


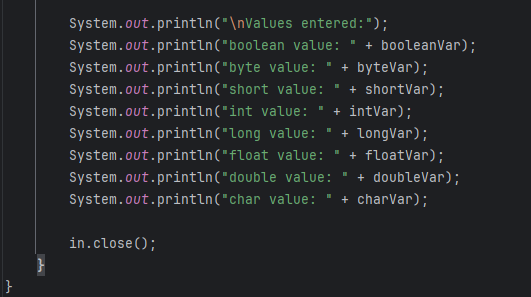
Результат:



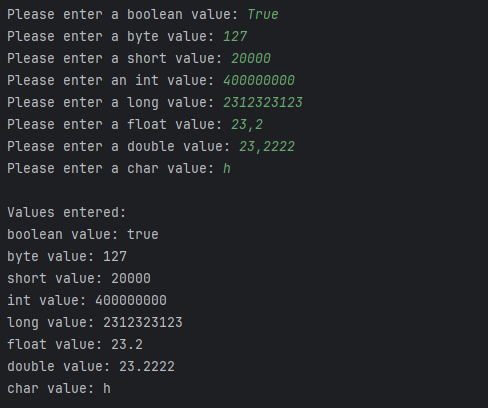
**Упражнение 5.**

Код:



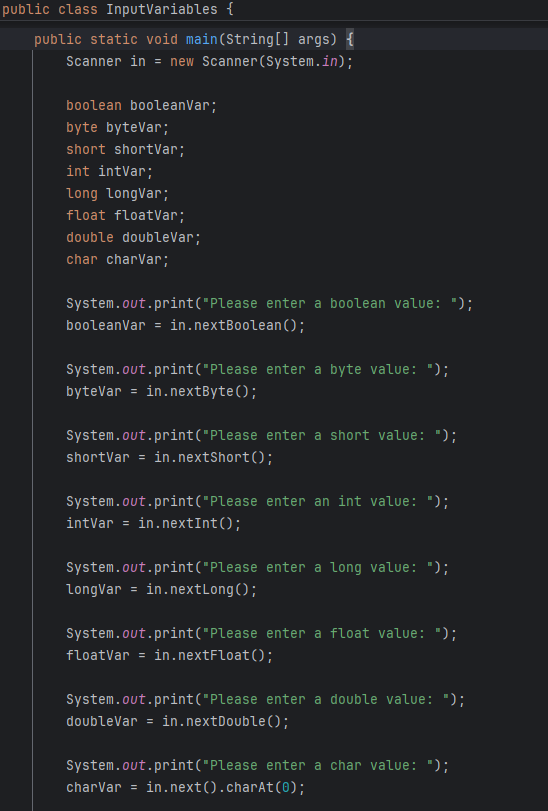


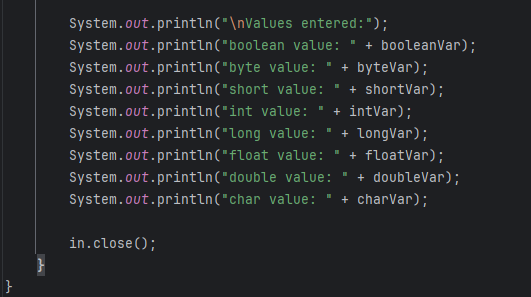
Результат:



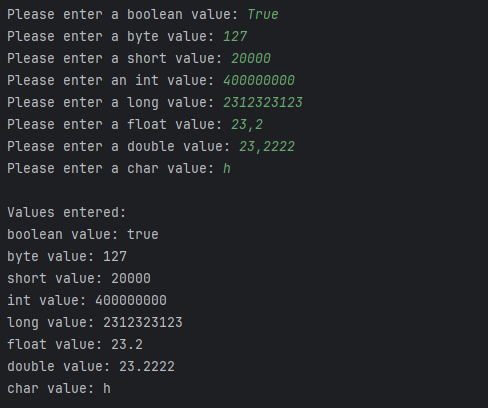
**Упражнение 6.**

Код:



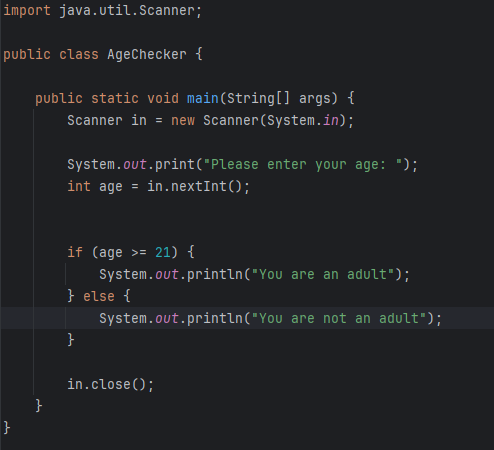


Результат:



**Упражнение 7.**

Код:

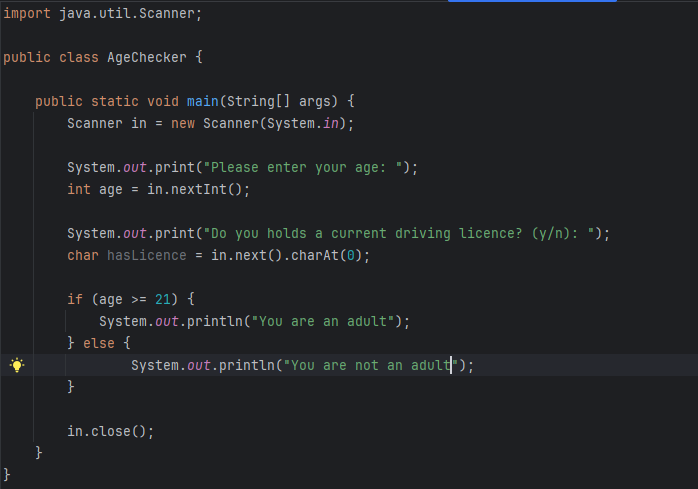


Результат:

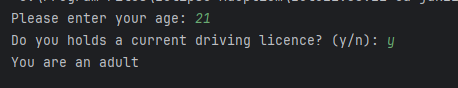


**Упражнение 8.**

Код:

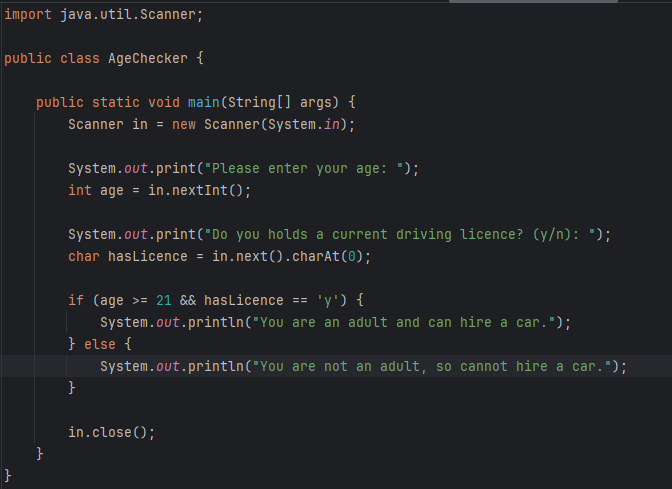


Результат:

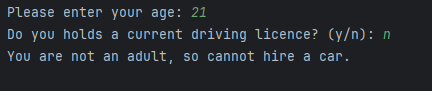


**Упражнение 9.**

Код:

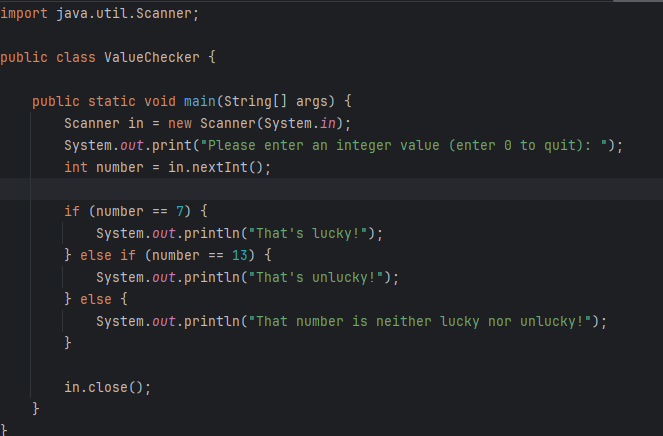


Результаты:

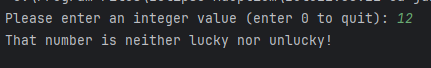


**Упражнение 10.**

Код:

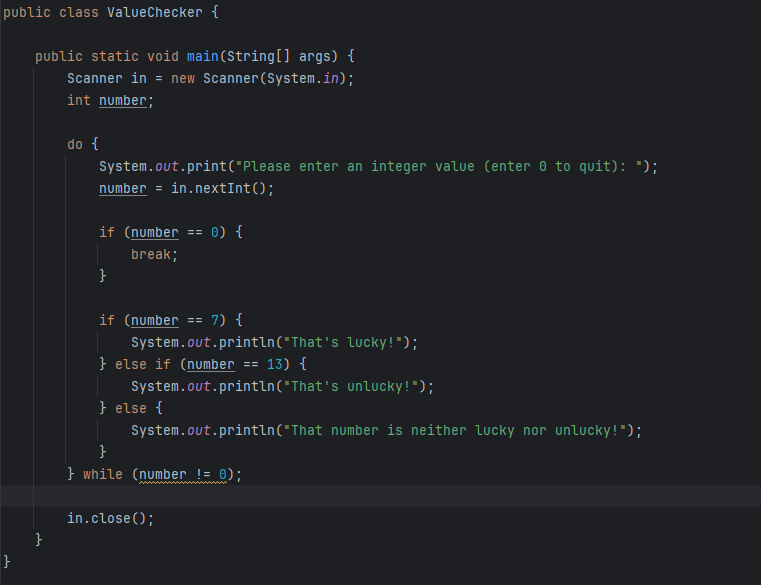


Результаты:

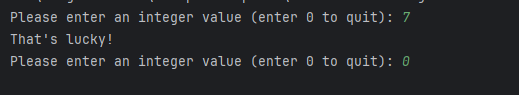


**Упражнение 11.**

Код:

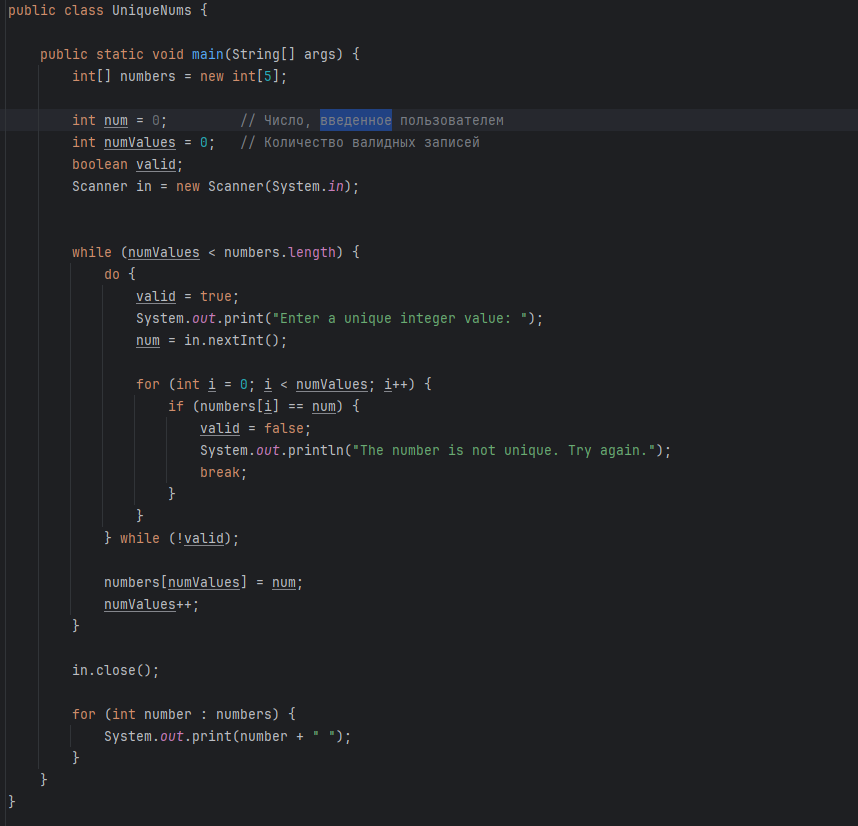


Результаты:

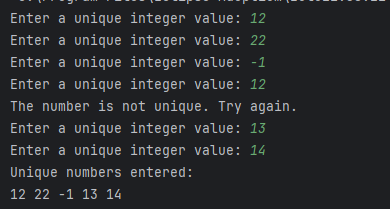


**Упражнение 12.**

Код:

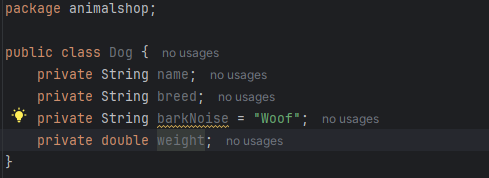


Результаты:



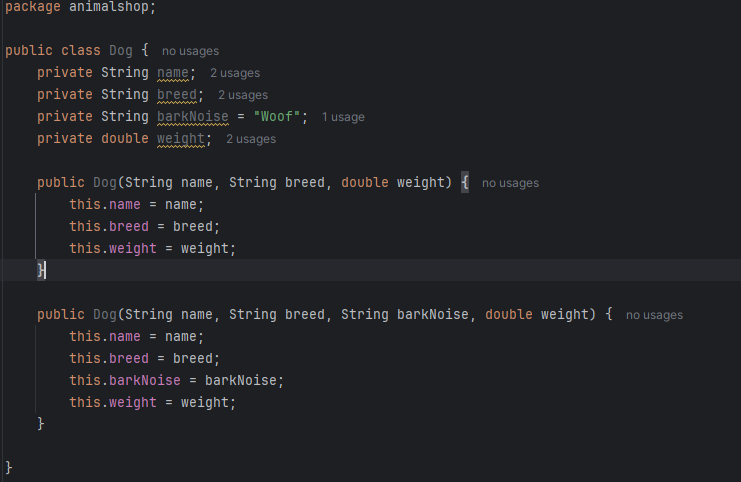
**Упражнение 13.**

Код:



**Упражнение 14.**

Код:



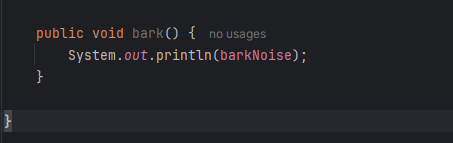
**Упражнение 15.**

Код:



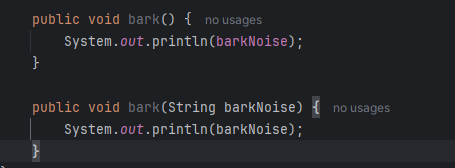
**Упражнение 16.**

Код:

****

**Упражнение 17.**

Код:

****

**Упражнение 18.**

Код:

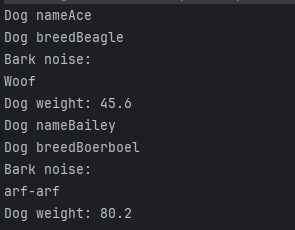
****

**Упражнение 19.**

Код:



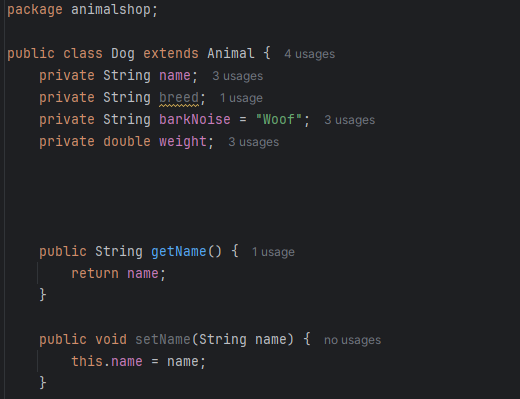
Результаты:



**Упражнение 20.**

Код:

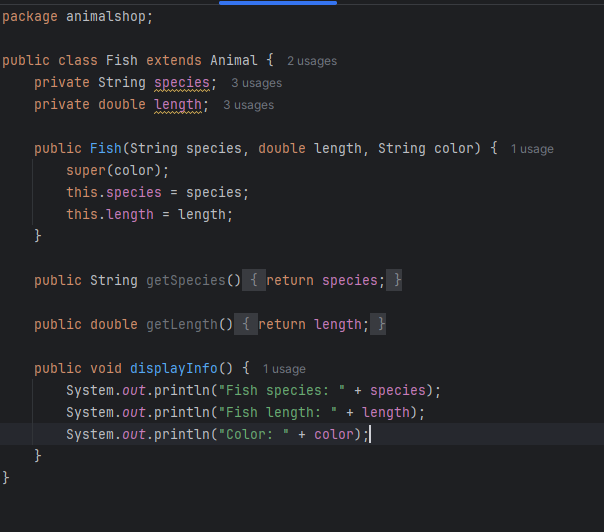
Dog:



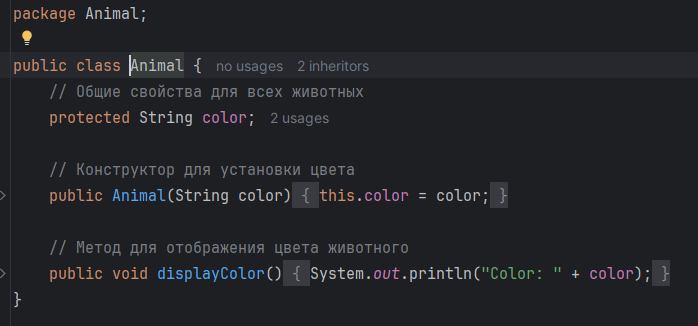
****

AnimalTester:  

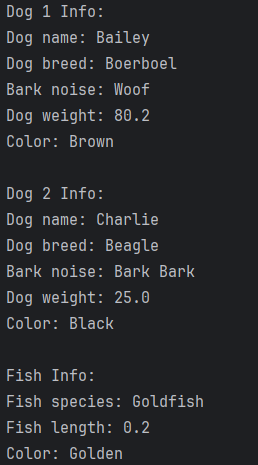

Fish:

****

Animal:



Результаты:

****

**КНИГА ВАСИЛЬЕВА**

### **1.Основы наследования**

* **Наследование** позволяет создавать новые классы (подклассы), которые используют функциональность существующих классов (суперклассов).
* Подклассы наследуют поля и методы суперклассов, кроме закрытых (private).
* Взаимодействие с закрытыми полями осуществляется через открытые или защищённые (protected) методы.

### 2. **Создание подклассов**

* Подкласс создаётся с использованием ключевого слова extends. Пример:

class SuperClass {

// Поля и методы суперкласса

}

class SubClass extends SuperClass {

// Дополнительные члены подкласса

}

* Если конструктор подкласса не вызывает явно конструктор суперкласса, используется его конструктор без параметров.

### 3. **Ключевые слова** super **и** this

* super используется:
  + Для вызова конструктора суперкласса.
  + Для обращения к методам или полям суперкласса, переопределённым в подклассе.
* this используется:
  + Для ссылки на текущий объект.
  + Для вызова другого конструктора текущего класса.

Пример использования super:

class A {

void show() {

System.out.println("Метод суперкласса");

}

}

class B extends A {

void show() {

super.show(); // Вызов метода суперкласса

System.out.println("Метод подкласса");

}

}

### 4. **Переопределение методов**

* Методы суперкласса можно переопределить в подклассе для изменения поведения.
* При вызове метода определяется версия по типу объекта, а не по типу ссылки.

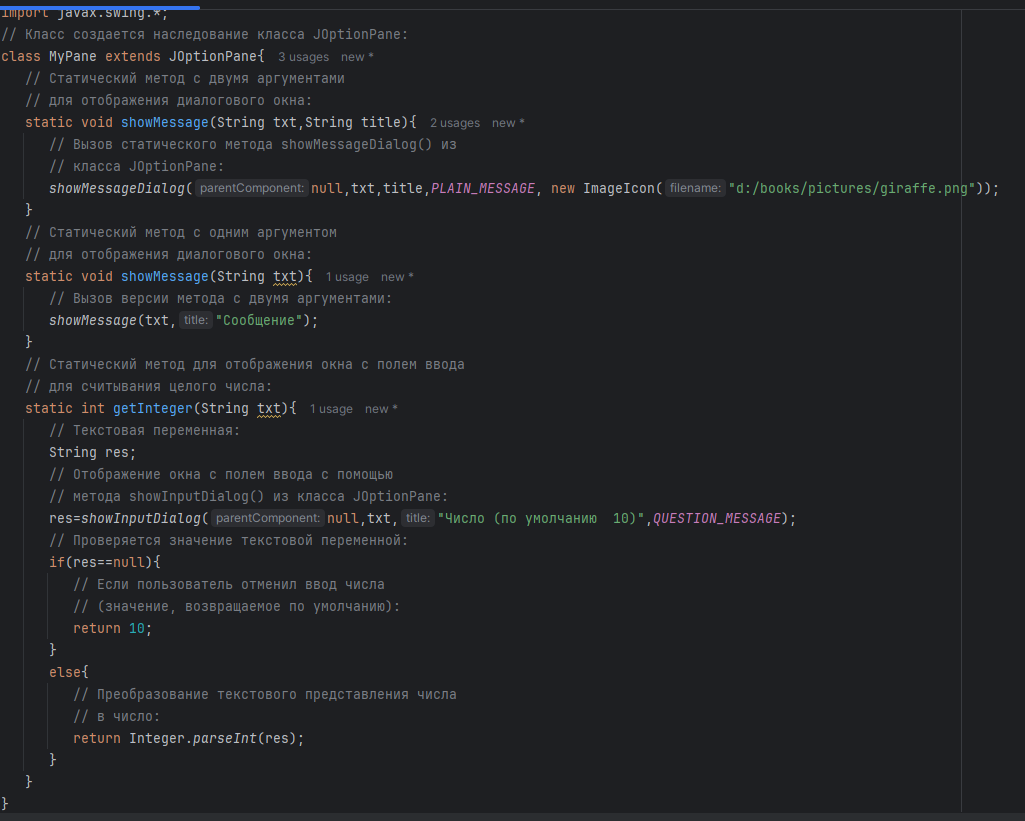
### 5. **Доступность полей и методов**

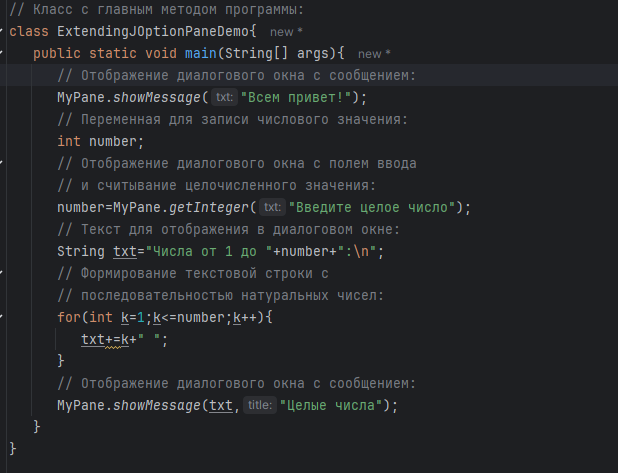
* Уровни доступа:
  + private: доступ только внутри класса.
  + protected: доступен подклассам и в пакете.
  + public: доступен всем.

### 6. **Особенности виртуальных методов**

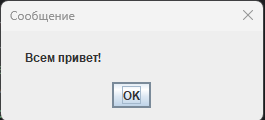
* В Java методы и конструкторы являются виртуальными:
  + Тип объекта определяет, какая версия метода будет вызвана.
  + Поля не являются виртуальными: их версия зависит от типа ссылки.

Код:

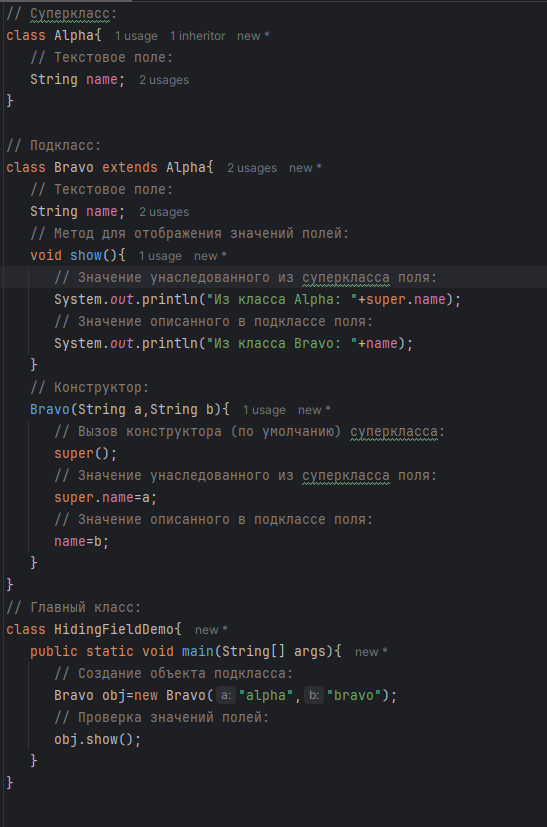




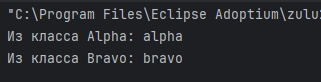
Результат:



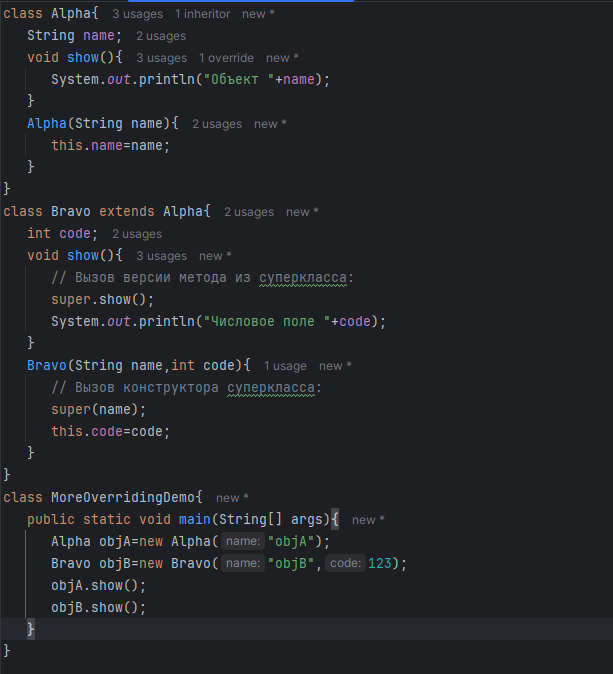
Код:



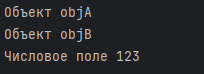
Результат:



Код:



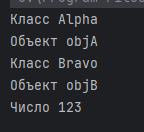
Результат:



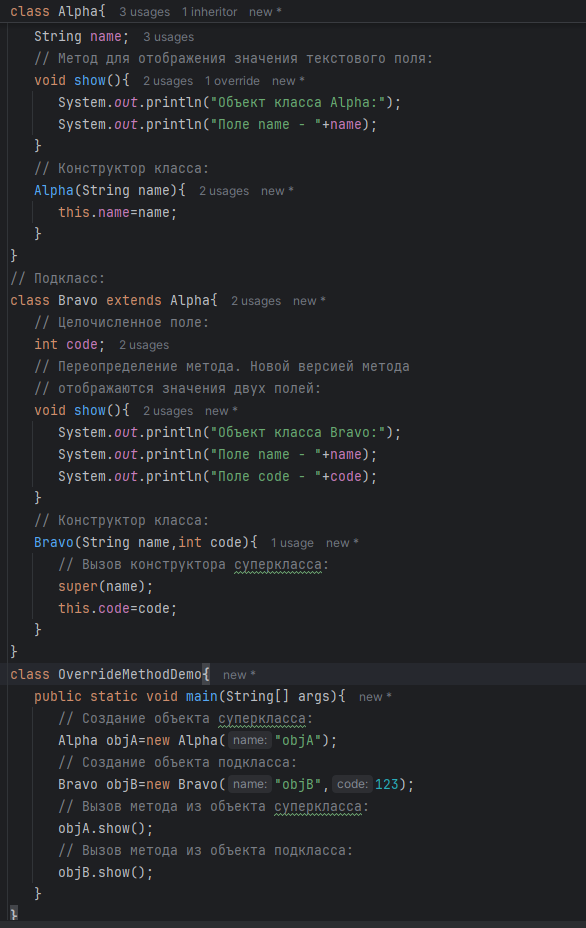
Код:



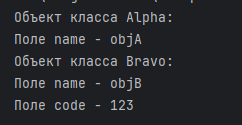
Результат:



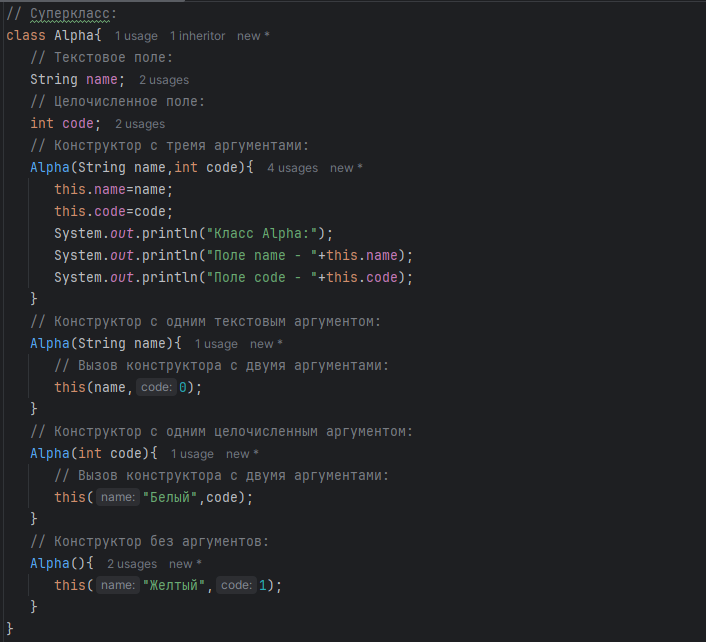
Код:

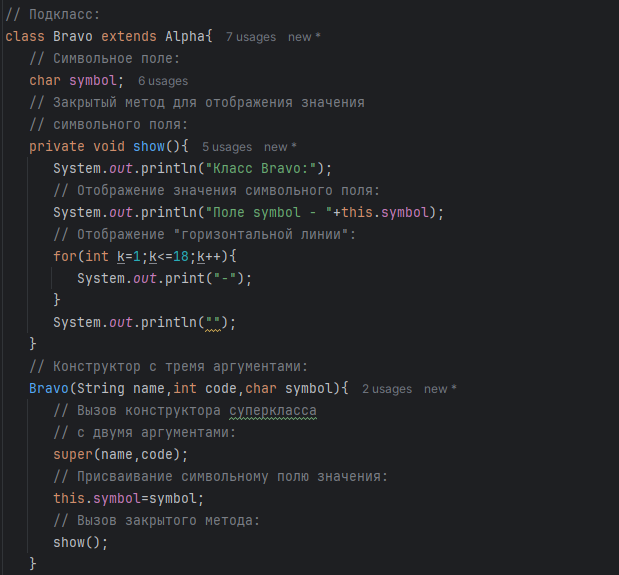


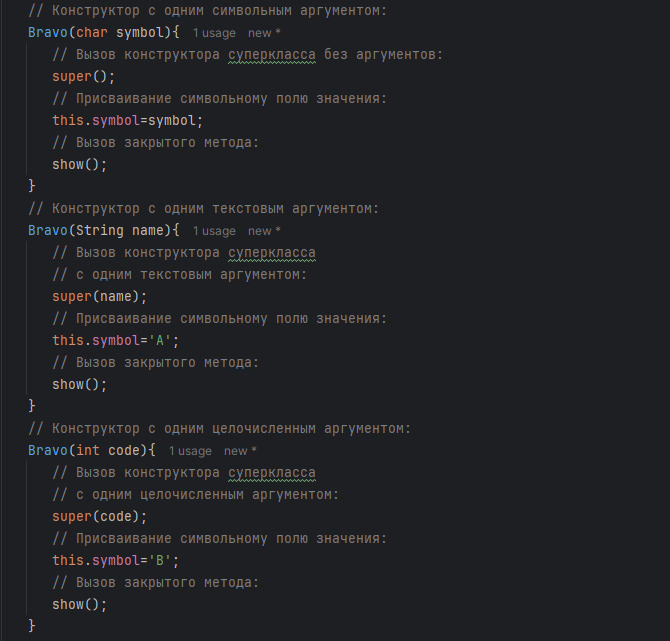
Результат:

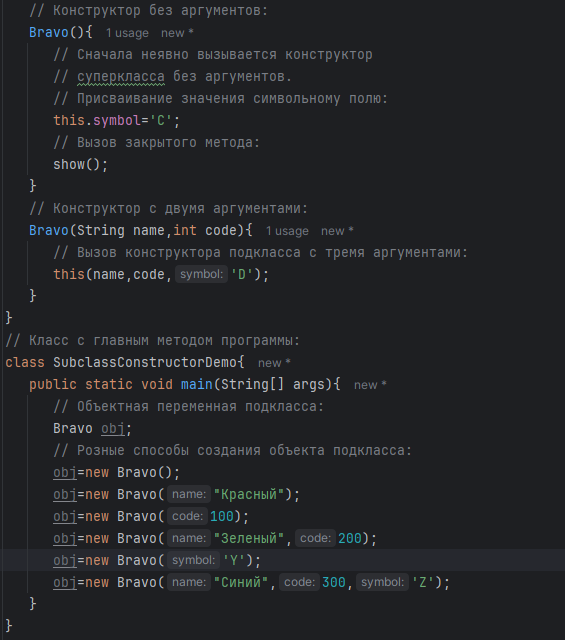


Код:

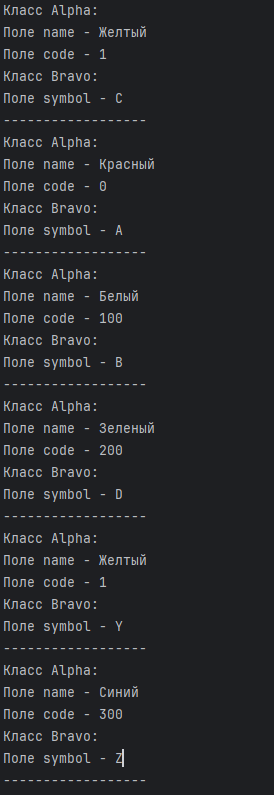




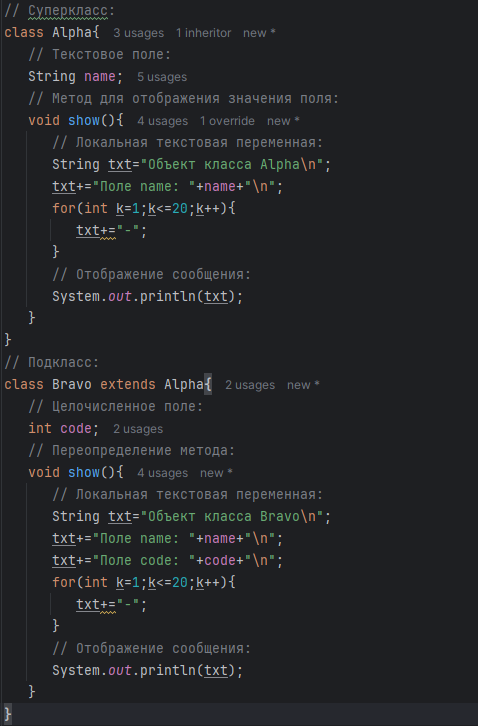


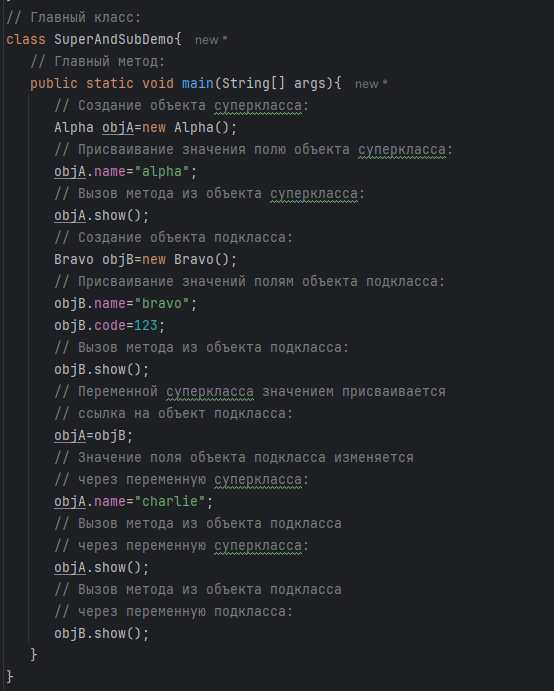


Результат:

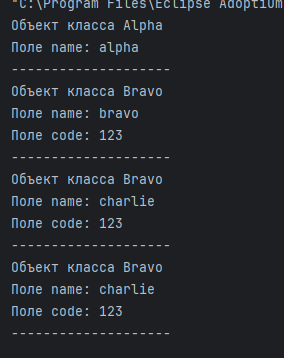


Код:

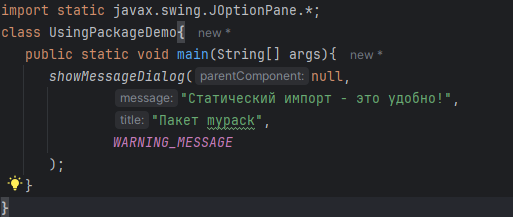




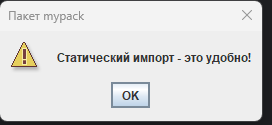
Результат:



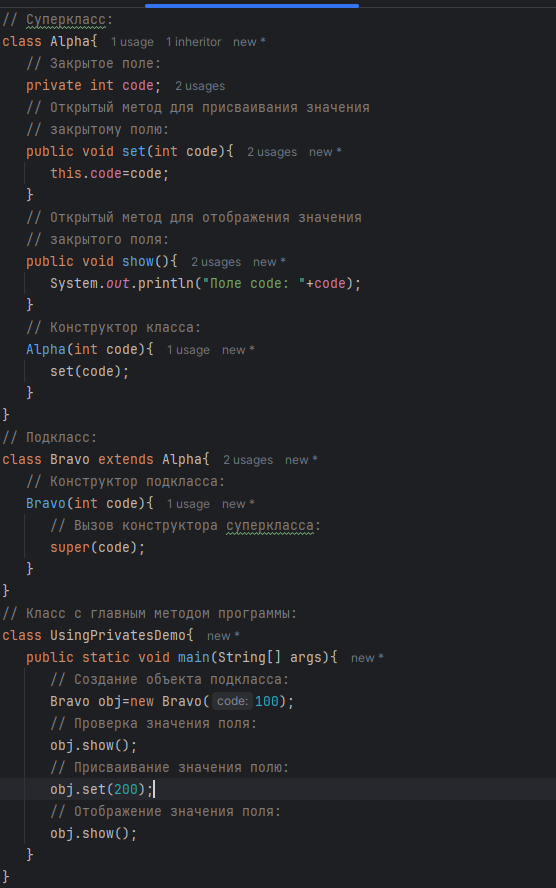
Код:



Результат:



Код:



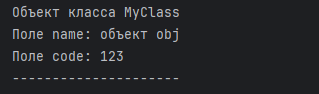
Результат:



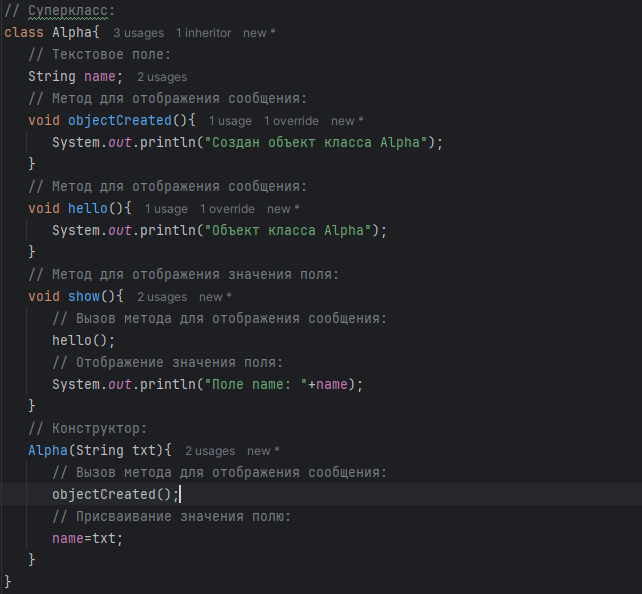
Код:



Результат:



Код:





Результат:

