Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ



**Звіт**

до лабораторної роботи №2

З дисципліни: «Кросплатформні засоби програмування»

На тему: «Класи та пакети»

**Варіант 3**

Виконав: ст. гр. КІ-305

Гнідець В.М.

Прийняв:

Іванов Ю.С.

Львів 2023

**Мета роботи:** ознайомитися з процесом розробки класів та пакетів мовою Java.

**Завдання:**

1. Написати та налагодити програму на мові Java, що реалізує у вигляді класу предметну область згідно варіанту. Програма має задовольняти наступним вимогам:

* програма має розміщуватися в пакеті Група.Прізвище.Lab2;
* клас має містити мінімум 3 поля, що є об’єктами класів, які описують складові частини предметної області;
* клас має містити кілька конструкторів та мінімум 10 методів;
* для тестування і демонстрації роботи розробленого класу розробити клас-драйвер;
* методи класу мають вести протокол своєї діяльності, що записується у файл;
* розробити механізм коректного завершення роботи з файлом (не надіятися на метод finalize());
* програма має володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

1. Автоматично згенерувати документацію до розробленої програми.
2. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
4. Дати відповідь на контрольні запитання.

**Варіант 3:** *Пес*

**Код програми**

**Class Dog**

package KI305\_Gnidec\_Lab2;  
  
import java.io.FileWriter;  
import java.io.IOException;  
import java.io.PrintWriter;  
  
*/\*\*  
 \* Клас, що представляє собаку та зберігає інформацію про неї.  
 \*/*public class Dog {  
 //Поля класу  
 private Breed breed;  
 private Color color;  
 private Age age;  
  
 //Поле для запису протоколу  
 private PrintWriter logWriter;  
  
 */\*\*  
 \* Пустий конструктор без аргументів.  
 \* Ініціалізує об'єкт Dog та створює файл протоколу.  
 \*/* //Пустий конструктор без аргументів  
 public Dog() {  
 try {  
 logWriter = new PrintWriter(new FileWriter("D:\\NULP\\KZP\\Lab2\\KI305\_Gnidec\_Lab2\\doc\\Dog.txt"));  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 //Конструктор зі всіма аргументами  
 public Dog(Breed breed, Color color, Age age) {  
 this.breed = breed;  
 this.color = color;  
 this.age = age;  
 try {  
 logWriter = new PrintWriter(new FileWriter("D:\\NULP\\KZP\\Lab2\\KI305\_Gnidec\_Lab2\\doc\\Dog.txt"));  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 //Гетери і сетери для об'єктів класу  
 public Breed getBreed() {  
 return breed;  
 }  
  
 public void setBreed(Breed breed) {  
 this.breed = breed;  
 }  
  
 public Color getColor() {  
 return color;  
 }  
  
 public void setColor(Color color) {  
 this.color = color;  
 }  
  
 public Age getAge() {  
 return age;  
 }  
  
 public void setAge(Age age) {  
 this.age = age;  
 }  
  
 */\*\*  
 \* Метод для запису дій в файл протоколу.  
 \*  
 \* @param activity Дія, яку потрібно зареєструвати в протоколі.  
 \*/* private void logActivity(String activity) {  
 if (logWriter != null) {  
 logWriter.println(activity);  
 logWriter.flush();  
 }  
 }  
  
 //Методи для роботи з автомобілем  
 public void feed() {  
 System.*out*.println("Feed the dog.");  
 logActivity("You fed the dog.");//Запис у протокол  
 }  
  
 public void pet() {  
 System.*out*.println("Pet the dog.");  
 logActivity("You petted the dog.");//Запис у протокол  
 }  
  
 public void spinAroundThroughLeft() {  
 System.*out*.println("Dog spins around through the left side.");  
 logActivity("You told the dog to spin around through the left side.");//Запис у протокол  
 }  
  
 public void spinAroundThroughRight() {  
 System.*out*.println("Dog spins around through the right side.");  
 logActivity("You told the dog to spin around through the right side.");//Запис у протокол  
 }  
  
 public void goForAWalk() {  
 System.*out*.println("Go for a walk with dog.");  
 logActivity("You went for a walk with dog\"");//Запис у протокол  
 }  
  
 public void call() {  
 System.*out*.println("Call the dog.");  
 logActivity("You called the dog.");//Запис у протокол  
 }  
  
 public void tellToWait() {  
 System.*out*.println("Tell the dog to wait patiently.");  
 logActivity("You told the dog to wait patiently.");//Запис у протокол  
 }  
  
 public void tellToSit() {  
 System.*out*.println("Tell the dog to sit.");  
 logActivity("You told the dog to sit.");//Запис у протокол  
 }  
  
 public void tellToSpeak() {  
 System.*out*.println("Tell the dog to speak loudly and clearly.");  
 logActivity("You told the dog to speak loudly and clearly.");//Запис у протокол  
 closeLogFile();//Закриваємо файл при завершенні роботи  
 }  
  
 // Метод для закриття файлу протоколу  
 private void closeLogFile() {  
 if (logWriter != null) {  
 logWriter.close();  
 }  
 }  
 //Метод toString()  
 @Override  
 public String toString() {  
 return "Dog:\nBreed - " + breed + "\nColor - " + color + "\nAge - " + age;  
 }  
  
}

**Class Breed**

package KI305\_Gnidec\_Lab2;  
  
*/\*\*  
 \* Клас, що представляє інформацію про породу собаки  
 \*/*public class Breed {  
 // Поля класу  
 private String nameBreed; // Назва компанії  
 private String infoBreed; // Інформація про компанію  
  
 */\*\*  
 \* Пустий конструктор без аргументів.  
 \* Ініціалізує об'єкт Breed з пустими значеннями.  
 \*/* public Breed(){  
  
 }  
  
 */\*\*  
 \* Конструктор з усіма аргументами.  
 \* Ініціалізує об'єкт Breed з вказаними параметрами.  
 \*  
 \* @param nameBreed Назва породи.  
 \* @param infoBreed Інформація про породу.  
 \*/* public Breed(String nameBreed, String infoBreed){  
 this.nameBreed = nameBreed;  
 this.infoBreed = infoBreed;  
 }  
  
 */\*\*  
 \* Перевизначений метод toString().  
 \*  
 \* @return Рядок, що представляє об'єкт Breed.  
 \*/* @Override  
 public String toString() {  
 return "Breed " +  
 "name: " + nameBreed +  
 ", info: " + infoBreed;  
 }  
}

**Class Color**

package KI305\_Gnidec\_Lab2;  
  
*/\*\*  
 \* Клас, що представляє колір собаки.  
 \*/*public class Color {  
 // Поле класу  
 private String nameColor; // Забарвлення собаки  
  
 */\*\*  
 \* Пустий конструктор без аргументів.  
 \* Ініціалізує об'єкт Color з пустою назвою моделі.  
 \*/* public Color(){  
  
 }  
  
 */\*\*  
 \* Конструктор з усіма аргументами.  
 \* Ініціалізує об'єкт Color з вказаною назвою моделі.  
 \*  
 \* @param nameColor Назва забарвлення собаки.  
 \*/* public Color(String nameColor){  
 this.nameColor = nameColor;  
 }  
  
 */\*\*  
 \* Перевизначений метод toString().  
 \*  
 \* @return Рядок, що представляє об'єкт Color.  
 \*/* @Override  
 public String toString() {  
 return "Color " +  
 "name: " + nameColor;  
 }  
}

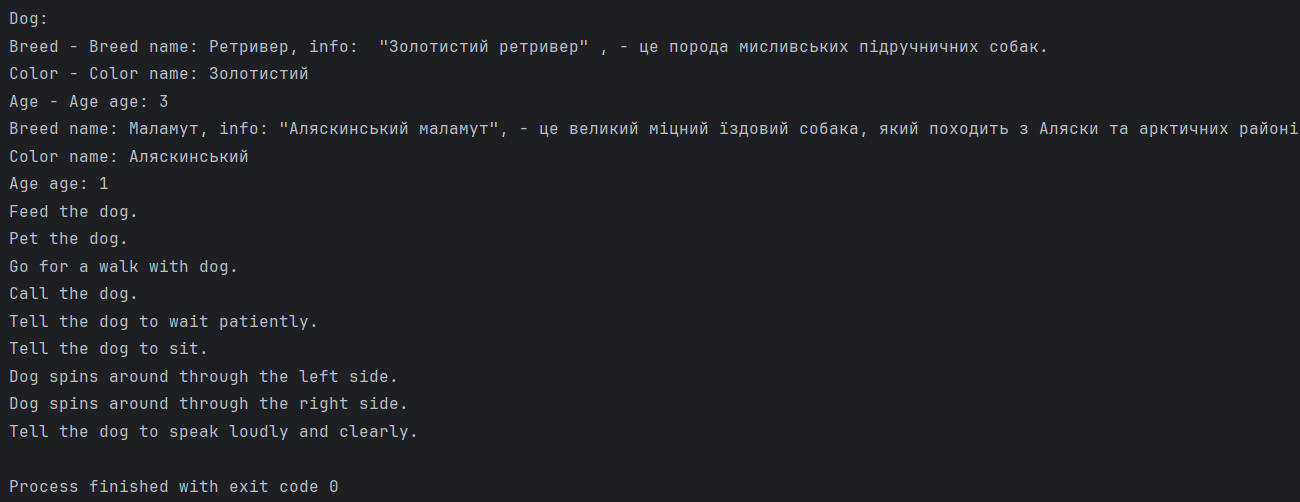
**Class Age**

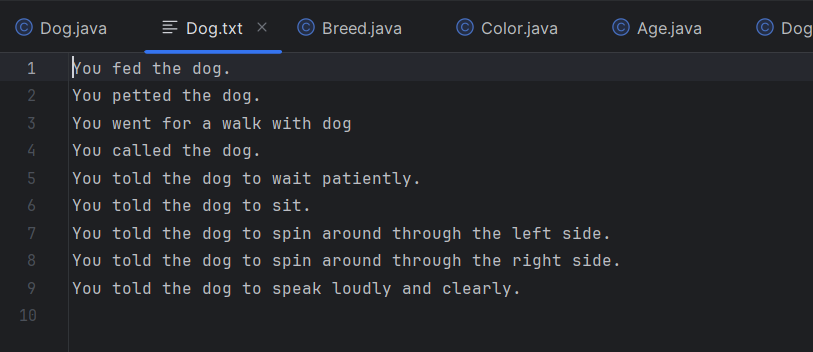
package KI305\_Gnidec\_Lab2;  
  
*/\*\*  
 \* Клас, що представляє вік собаки  
 \*/*public class Age {  
 // Поле класу  
 private int age; // Вік собаки  
  
 */\*\*  
 \* Пустий конструктор без аргументів.  
 \* Ініціалізує об'єкт Age з нульовою сумою.  
 \*/* public Age() {  
  
 }  
  
 */\*\*  
 \* Конструктор з усіма аргументами.  
 \* Ініціалізує об'єкт Age з вказаним віком собаки.  
 \*  
 \* @param age вік собаки.  
 \*/* public Age(int age) {  
 this.age = age;  
 }  
  
 */\*\*  
 \* Перевизначений метод toString().  
 \*  
 \* @return Рядок, що представляє об'єкт Age.  
 \*/* @Override  
 public String toString() {  
 return "Age " + "age: " + age;  
 }  
}

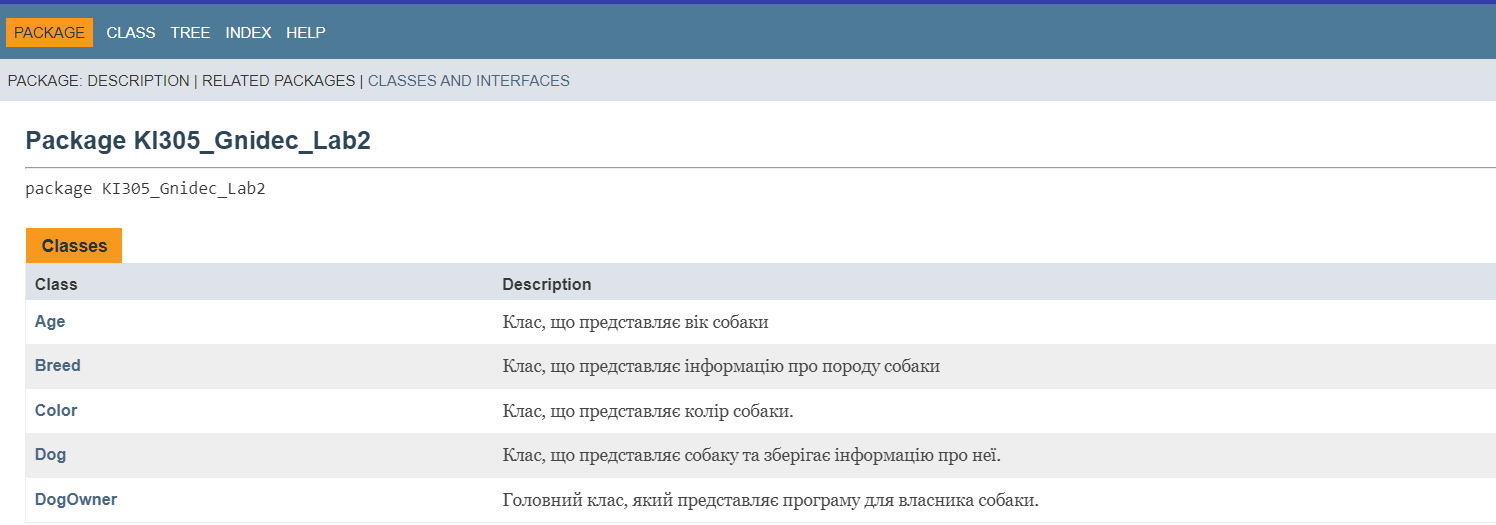
**Class DogOwner**

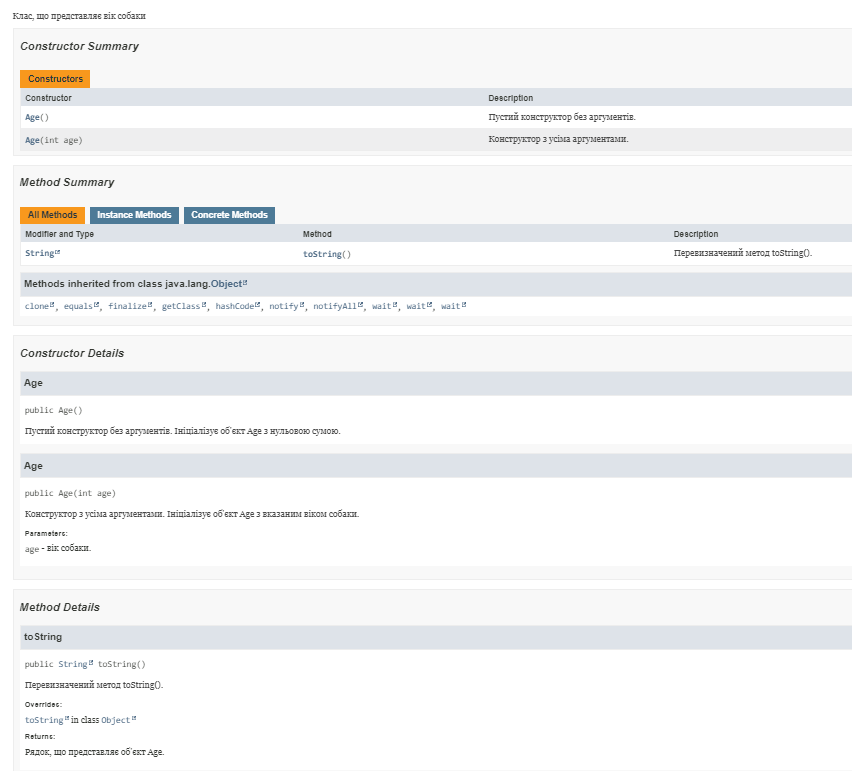
package KI305\_Gnidec\_Lab2;  
  
*/\*\*  
 \* Головний клас, який представляє програму для власника собаки.  
 \*/*public class DogOwner {  
 */\*\*  
 \* Головний метод програми.  
 \*  
 \* @param args Масив рядків аргументів командного рядка.  
 \*/* public static void main(String[] args) {  
  
 // Ініціалізація полів  
 Breed breed = new Breed("Ретривер", " \"Золотистий ретривер\" , - це порода мисливських підручничних собак.");  
 Color color = new Color("Золотистий");  
 Age age = new Age(3);  
  
 // Зміна полів за допомогою метода Сет і відображення зміненого за допомогою метода Гет  
 Dog dog = new Dog(breed, color, age);  
 System.*out*.println(dog);  
  
 dog.setBreed(new Breed("Маламут", "\"Аляскинський маламут\", - це великий міцний їздовий собака, який походить з Аляски та арктичних районів."));  
 System.*out*.println(dog.getBreed());  
  
 dog.setColor(new Color("Аляскинський"));  
 System.*out*.println(dog.getColor());  
  
 dog.setAge(new Age(1));  
 System.*out*.println(dog.getAge());  
  
 // Виклик усіх методів класу  
 dog.feed();  
 dog.pet();  
 dog.goForAWalk();  
 dog.call();  
 dog.tellToWait();  
 dog.tellToSit();  
 dog.tellToSpeak();  
 dog.spinAroundThroughLeft();  
 dog.spinAroundThroughRight();  
 }  
}

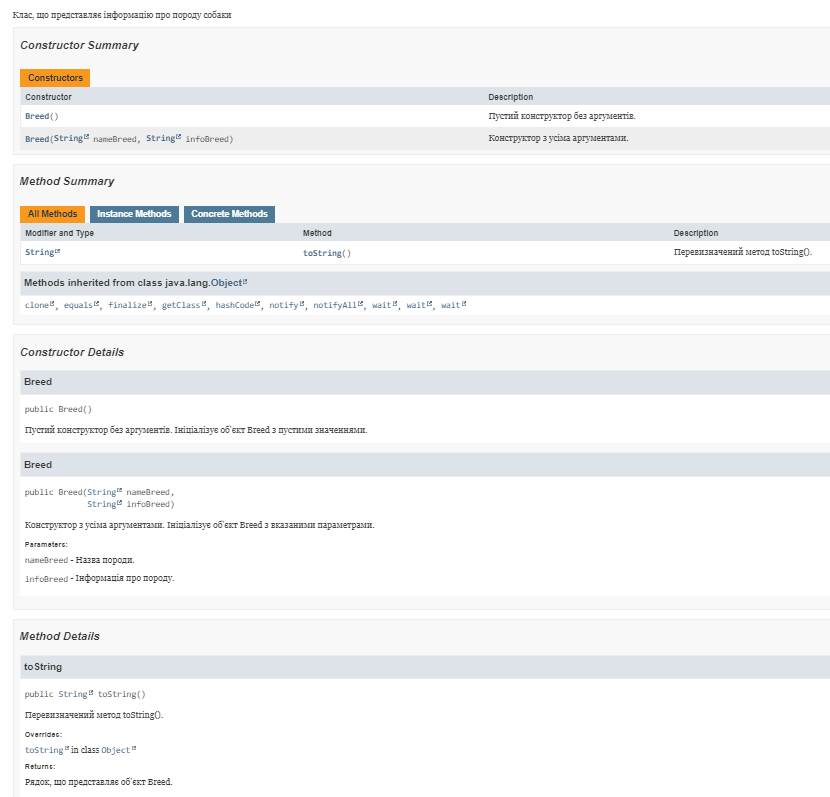
**Результат роботи програми:**

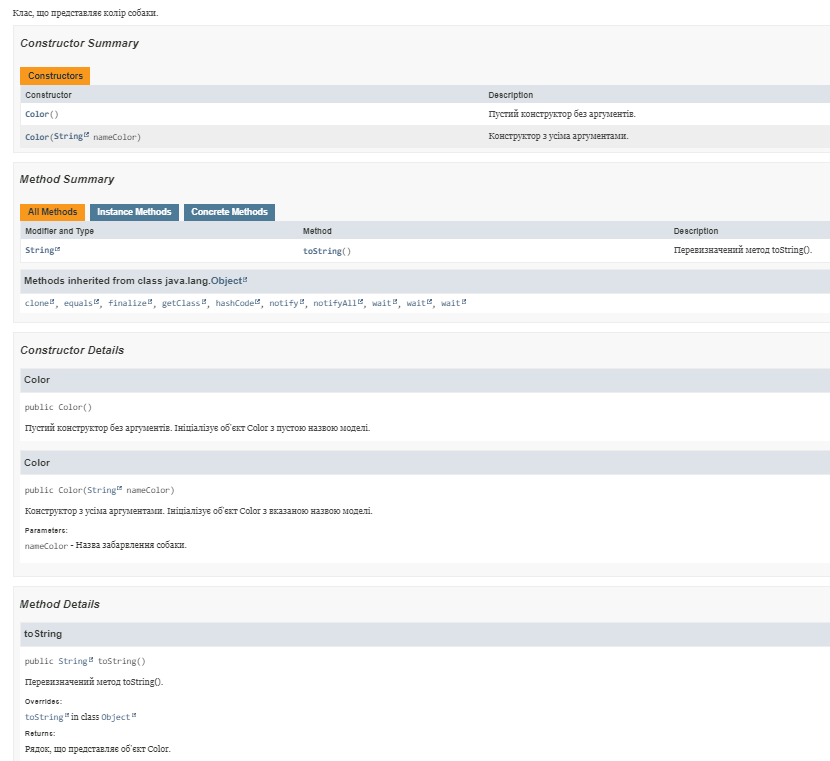
****

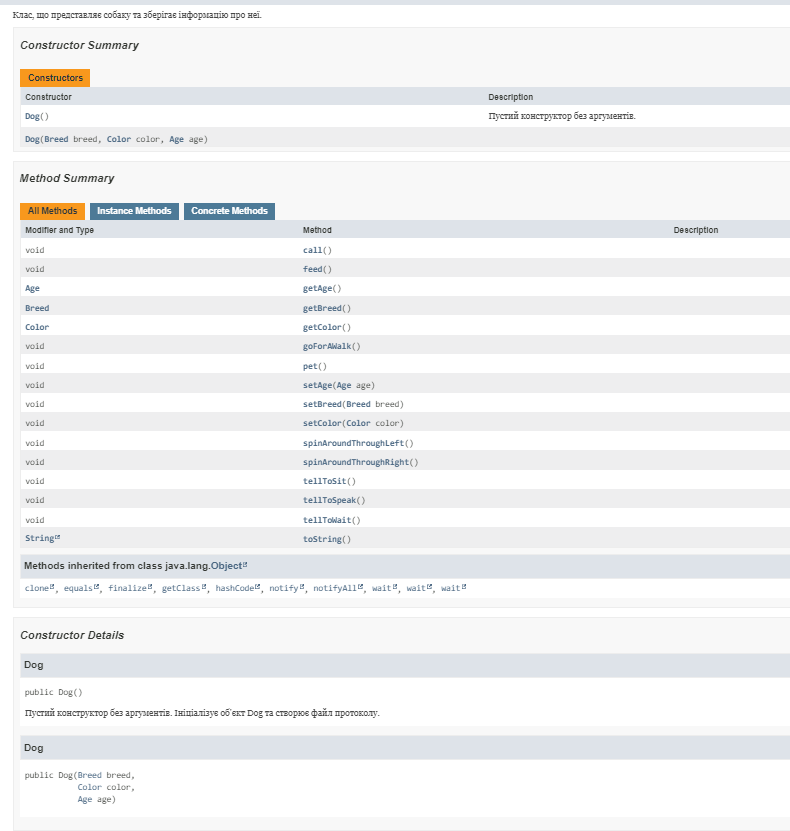
****



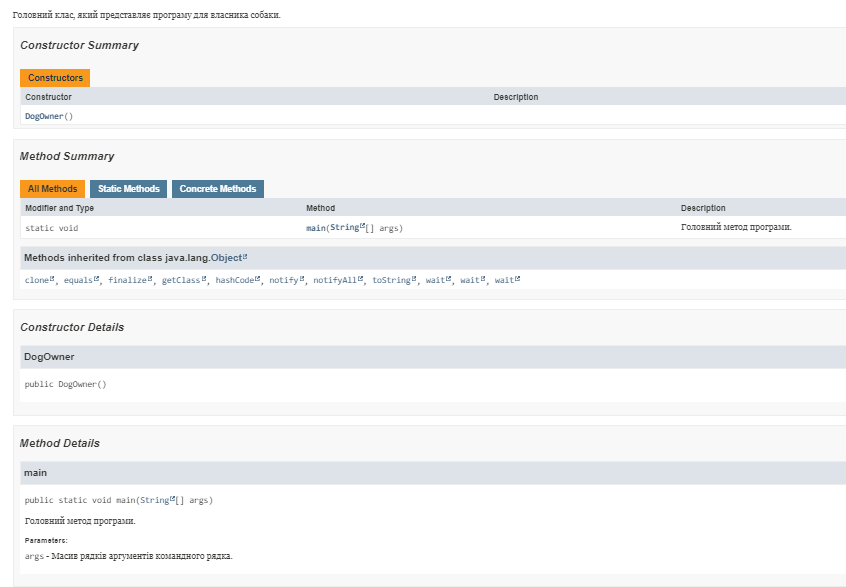






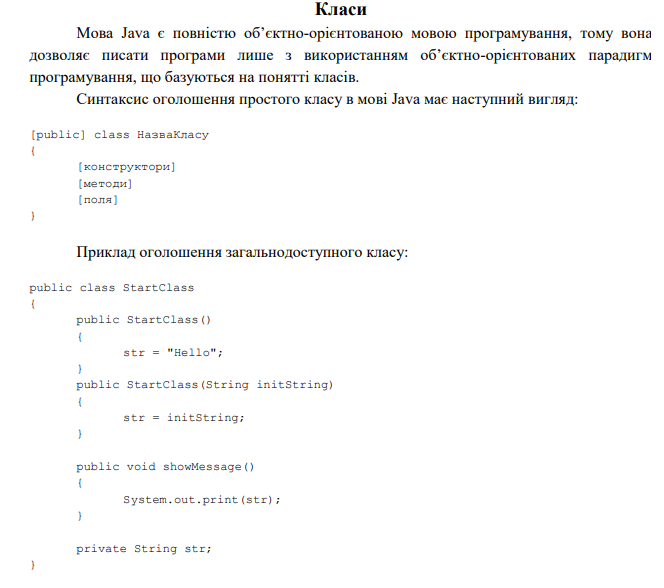




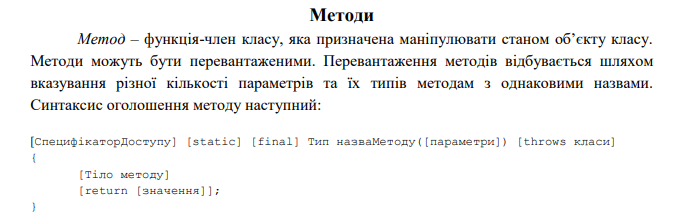


**Контрольні запитання**

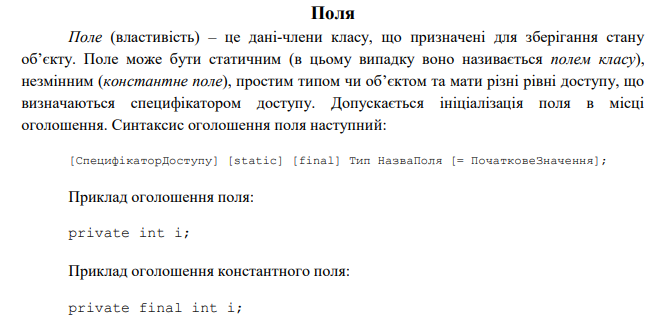
1. ***Синтаксис визначення класу.***

******

***2. Синтаксис визначення методу.***



***3.Синтаксис оголошення поля.***

******

***4. Як оголосити та ініціалізувати константне поле?***

public class MyClass {

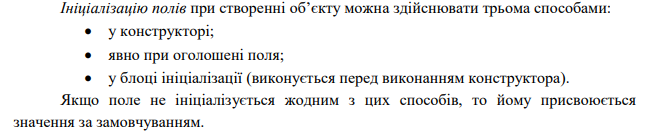
// Константне поле, ініціалізоване одразу

public final int MY\_CONSTANT\_FIELD = 42;

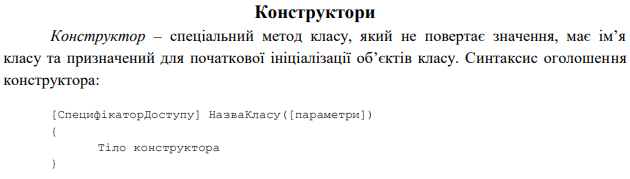
// Решта коду класу

}

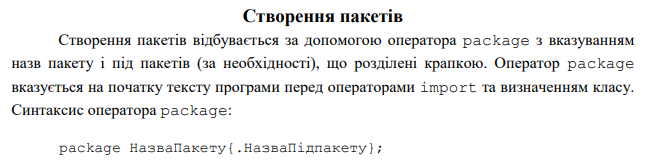
***5. Які є способи ініціалізації полів?***

******

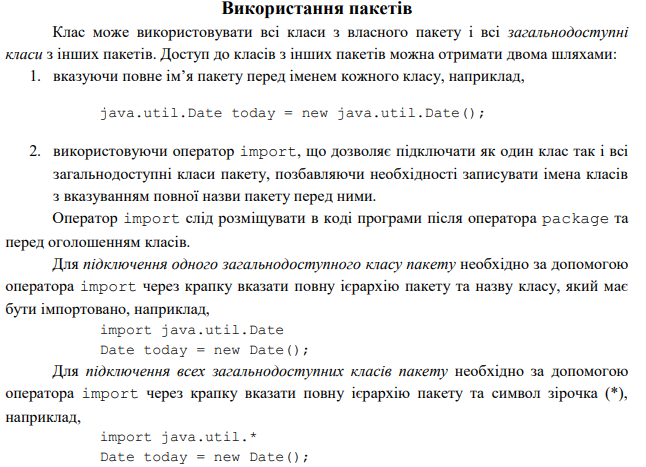
1. ***Синтаксис визначення конструктора.***

******

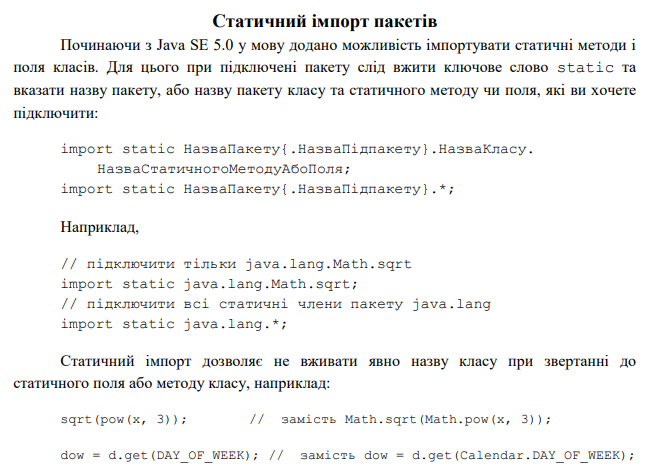
1. ***Синтаксис оголошення пакету.***

******

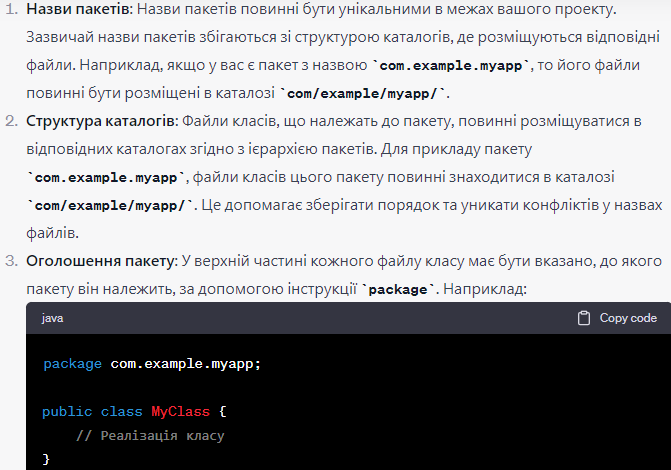
1. ***Як підключити до програми класи, що визначені в зовнішніх пакетах?***

******

1. ***В чому суть статичного імпорту пакетів?***

******

***10. Які вимоги ставляться до файлів і каталогів при використанні пакетів?***

****

**Висновок:** я ознайомився з процесом розробки класів та пакетів мовою Java.