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 5**

Виконала: ст. гр. КІ-305

Гринь С.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 5:** *Машина*

**Код програми**

Class ***Automobile***:

package KI305.Hryn.Lab2;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

/\*\*

\* Клас, що представляє автомобіль та зберігає інформацію про нього.

\*/

public class Automobile {

//Поля класу

private Company company;

private Model model;

private Price price;

//Поле для запису протоколу

private PrintWriter logWriter;

/\*\*

\* Пустий конструктор без аргументів.

\* Ініціалізує об'єкт Automobile та створює файл протоколу.

\*/

//Пустий конструктор без аргументів

public Automobile() {

try {

logWriter = new PrintWriter(new FileWriter("C:\\Users\\User\\IdeaProjects\\java\_lab\\src\\KI305\\Hryn\\Lab2\\Automobile.txt"));

} catch (IOException e) {

e.printStackTrace();

}

}

//Конструктор зі всіма аргументами

public Automobile(Company company, Model model, Price price) {

this.company = company;

this.model = model;

this.price = price;

try {

logWriter = new PrintWriter(new FileWriter("C:\\Users\\User\\IdeaProjects\\java\_lab\\src\\KI305\\Hryn\\Lab2\\Automobile.txt"));

} catch (IOException e) {

e.printStackTrace();

}

}

//Гетери і сетери для об'єктів класу

public Company getCompany() {

return company;

}

public void setCompany(Company company) {

this.company = company;

}

public Model getModel() {

return model;

}

public void setModel(Model model) {

this.model = model;

}

public Price getPrice() {

return price;

}

public void setPrice(Price price) {

this.price = price;

}

/\*\*

\* 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 startEngine() {

System.out.println("Starting the car engine.");

logActivity("We started the car engine.");//Запис у протокол

}

public void brake() {

System.out.println("Turning on the brakes.");

logActivity("We turned on the brakes.");//Запис у протокол

}

public void turnLeft() {

System.out.println("Turning to the left.");

logActivity("We turned to the left.");//Запис у протокол

}

public void turnRight() {

System.out.println("Turning to the right.");

logActivity("We turned to the right.");//Запис у протокол

}

public void checkEngineStatus() {

System.out.println("Сhecking the condition of the engine.");

logActivity("We checked the condition of the engine.");//Запис у протокол

}

public void turnOnHeadlights() {

System.out.println("Turning on the headlights.");

logActivity("We turned on the headlights.");//Запис у протокол

}

public void checkFuelLevel() {

System.out.println("Сhecking the level of the fuel tank in the car.");

logActivity("We checked the level of the fuel tank in the car.");//Запис у протокол

}

public void lockDoors() {

System.out.println("Locking the doors.");

logActivity("We locked the doors.");//Запис у протокол

}

public void stopEngine() {

System.out.println("Stopping the car engine.");

logActivity("We stopped the car engine.");//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 "Automobile: Company - " + company + ", Model - " + model + ", Price - " + price;

}

}

Class ***Company***:

package KI305.Hryn.Lab2;

/\*\*

\* Клас, що представляє інформацію про компанію.

\*/

public class Company {

// Поля класу

private String nameCompany; // Назва компанії

private String infoCompany; // Інформація про компанію

/\*\*

\* Пустий конструктор без аргументів.

\* Ініціалізує об'єкт Company з пустими значеннями.

\*/

public Company(){

}

/\*\*

\* Конструктор з усіма аргументами.

\* Ініціалізує об'єкт Company з вказаними параметрами.

\*

\* @param nameCompany Назва компанії.

\* @param infoCompany Інформація про компанію.

\*/

public Company(String nameCompany, String infoCompany){

this.nameCompany = nameCompany;

this.infoCompany = infoCompany;

}

/\*\*

\* Перевизначений метод toString().

\*

\* @return Рядок, що представляє об'єкт Company.

\*/

@Override

public String toString() {

return "Company{" +

"name='" + nameCompany + '\'' +

", info='" + infoCompany + '\'' +

'}';

}

}

Class ***Model***:

package KI305.Hryn.Lab2;

/\*\*

\* Клас, що представляє модель автомобіля.

\*/

public class Model {

// Поле класу

private String nameModel; // Назва моделі автомобіля

/\*\*

\* Пустий конструктор без аргументів.

\* Ініціалізує об'єкт Model з пустою назвою моделі.

\*/

public Model(){

}

/\*\*

\* Конструктор з усіма аргументами.

\* Ініціалізує об'єкт Model з вказаною назвою моделі.

\*

\* @param nameModel Назва моделі автомобіля.

\*/

public Model(String nameModel){

this.nameModel = nameModel;

}

/\*\*

\* Перевизначений метод toString().

\*

\* @return Рядок, що представляє об'єкт Model.

\*/

@Override

public String toString() {

return "Model{" +

"name:'" + nameModel + '\'' +

'}';

}

}

Class **Price**:

package KI305.Hryn.Lab2;

/\*\*

\* Клас, що представляє ціну.

\*/

public class Price {

// Поле класу

private int sum; // Сума ціни

/\*\*

\* Пустий конструктор без аргументів.

\* Ініціалізує об'єкт Price з нульовою сумою.

\*/

public Price() {

}

/\*\*

\* Конструктор з усіма аргументами.

\* Ініціалізує об'єкт Price з вказаною сумою ціни.

\*

\* @param sum Сума ціни.

\*/

public Price(int sum) {

this.sum = sum;

}

/\*\*

\* Перевизначений метод toString().

\*

\* @return Рядок, що представляє об'єкт Price.

\*/

@Override

public String toString() {

return "Price{" + "sum=" + sum + "}";

}

}

Class ***AutomobileDriver***:

package KI305.Hryn.Lab2;

/\*\*

\* Головний клас, який представляє програму для водія автомобіля.

\*/

public class AutomobileDriver {

/\*\*

\* Головний метод програми.

\*

\* @param args Масив рядків аргументів командного рядка.

\*/

public static void main(String[] args) {

// Ініціалізація полів

Company company = new Company("Ford", "Ford Motor Company, \"Ford Motor Company\" is an American automobile company.");

Model model = new Model("Mustang");

Price price = new Price(34000);

// Зміна полів за допомогою метода Сет і відображення зміненого за допомогою метода Гет

Automobile automobile = new Automobile(company, model, price);

System.out.println(automobile);

automobile.setCompany(new Company("Mercedes", "Mercedes-Benz Group AG (Mercedes-Benz), formerly Daimler AG, is an automobile company."));

System.out.println(automobile.getCompany());

automobile.setModel(new Model("CLS"));

System.out.println(automobile.getModel());

automobile.setPrice(new Price(55000));

System.out.println(automobile.getPrice());

// Виклик усіх методів класу

automobile.startEngine();

automobile.brake();

automobile.turnLeft();

automobile.turnRight();

automobile.checkEngineStatus();

automobile.turnOnHeadlights();

automobile.checkFuelLevel();

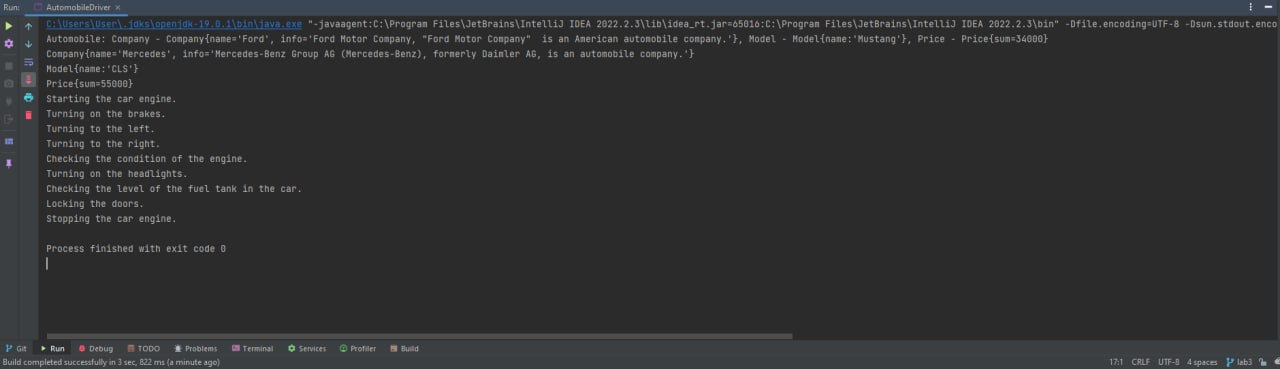
automobile.lockDoors();

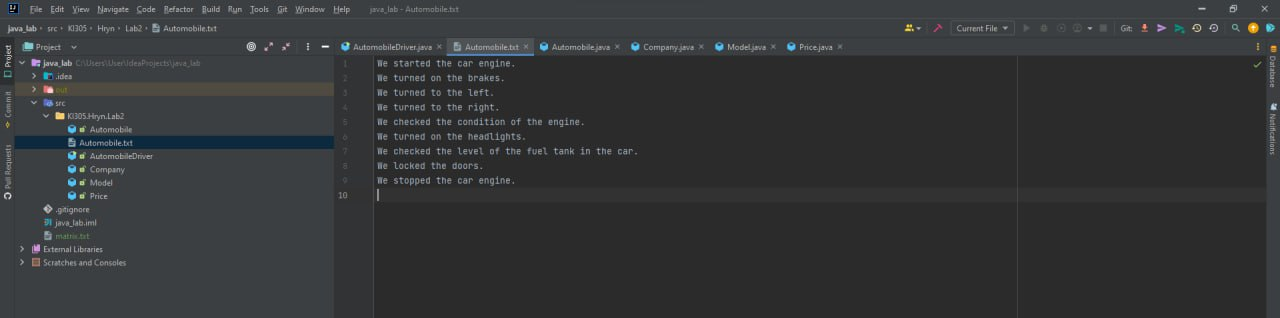
automobile.stopEngine();

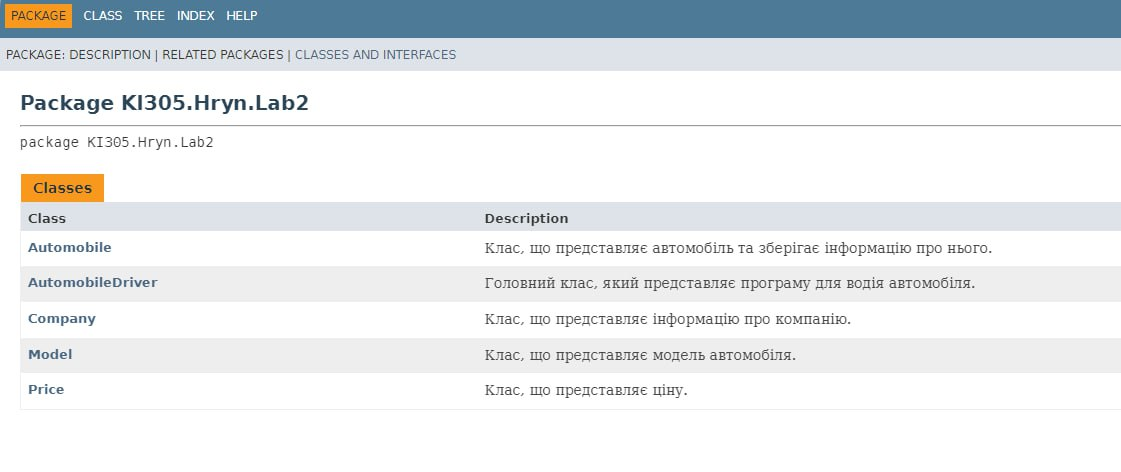
}

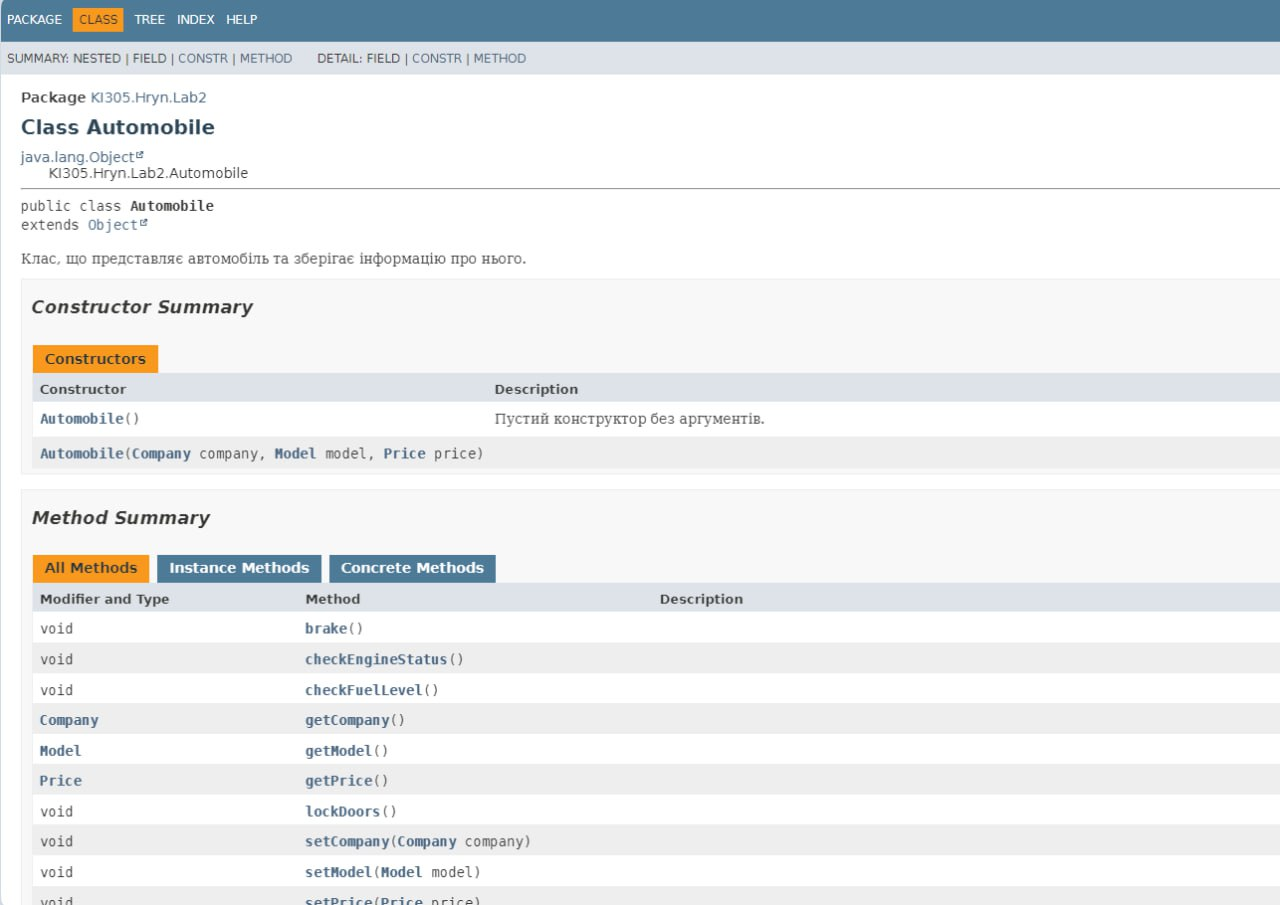
}

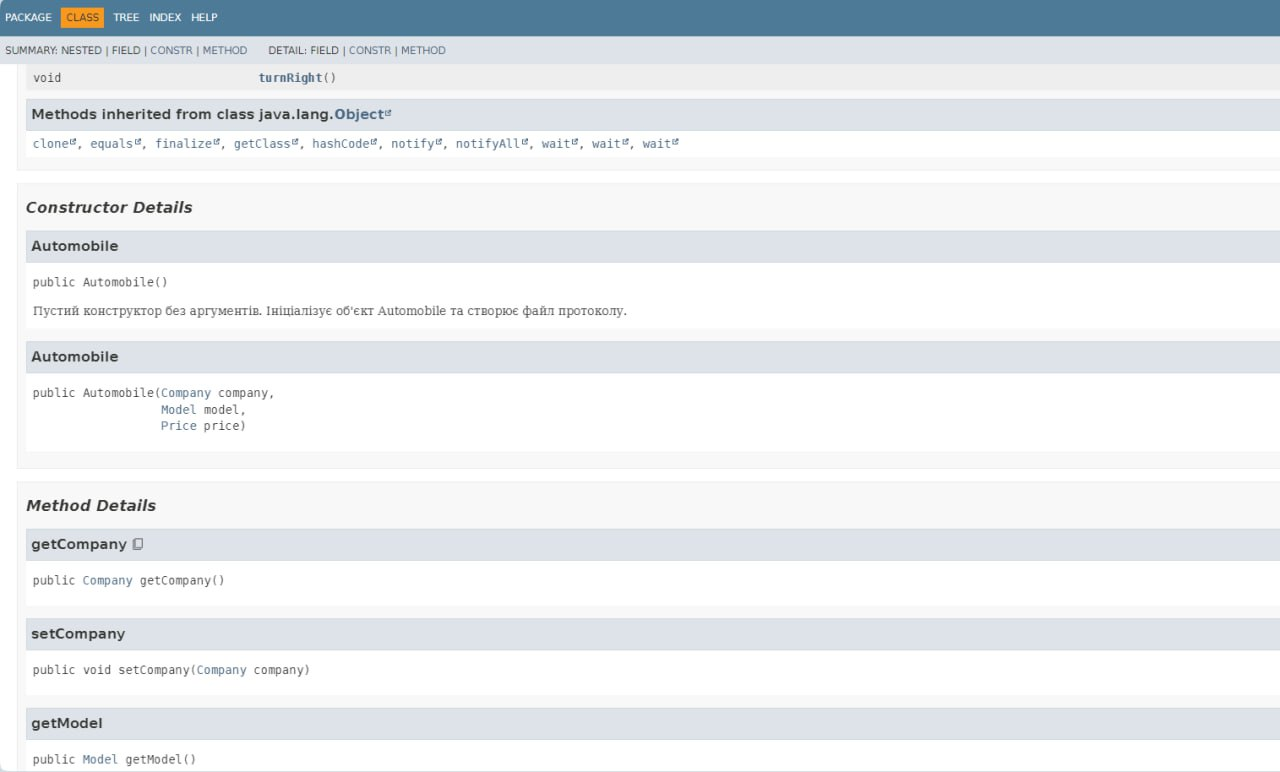
**Результат роботи програми:**

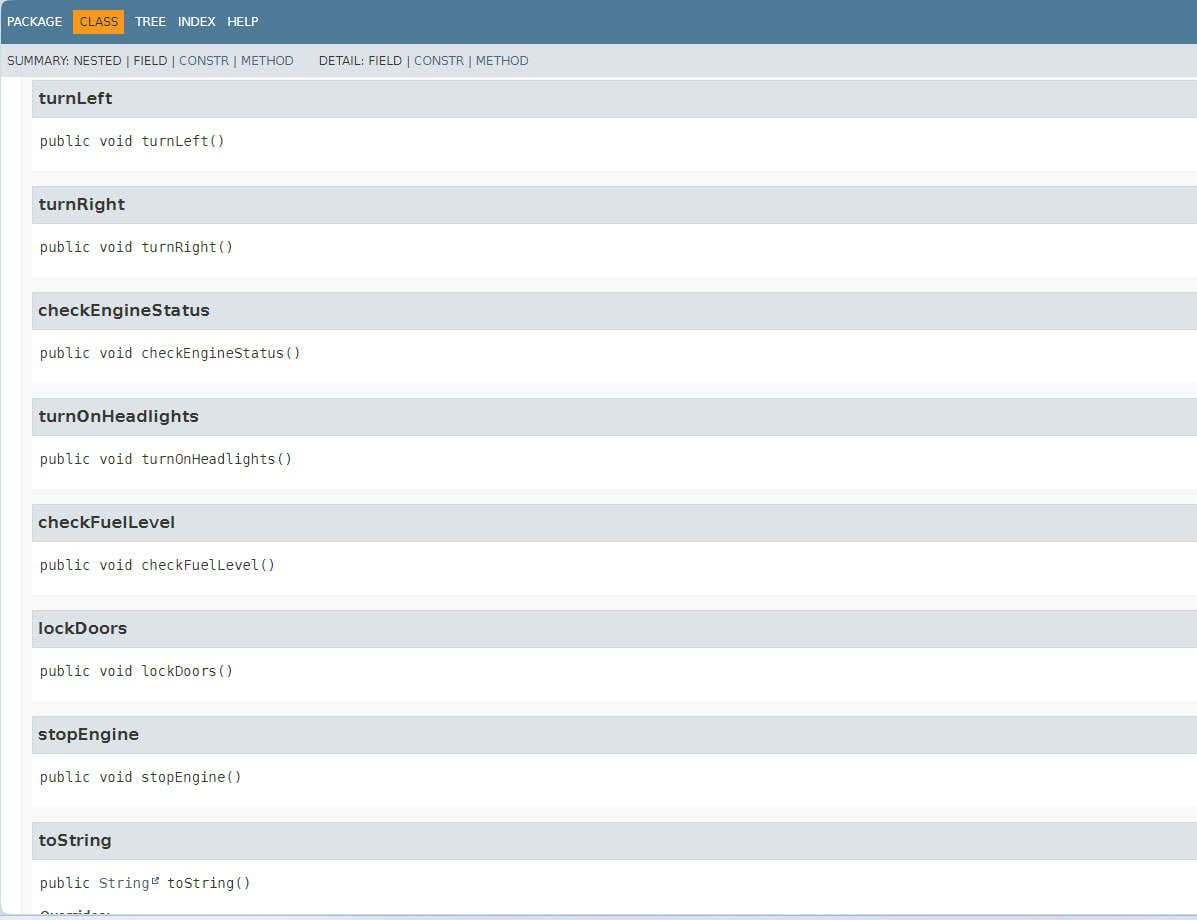


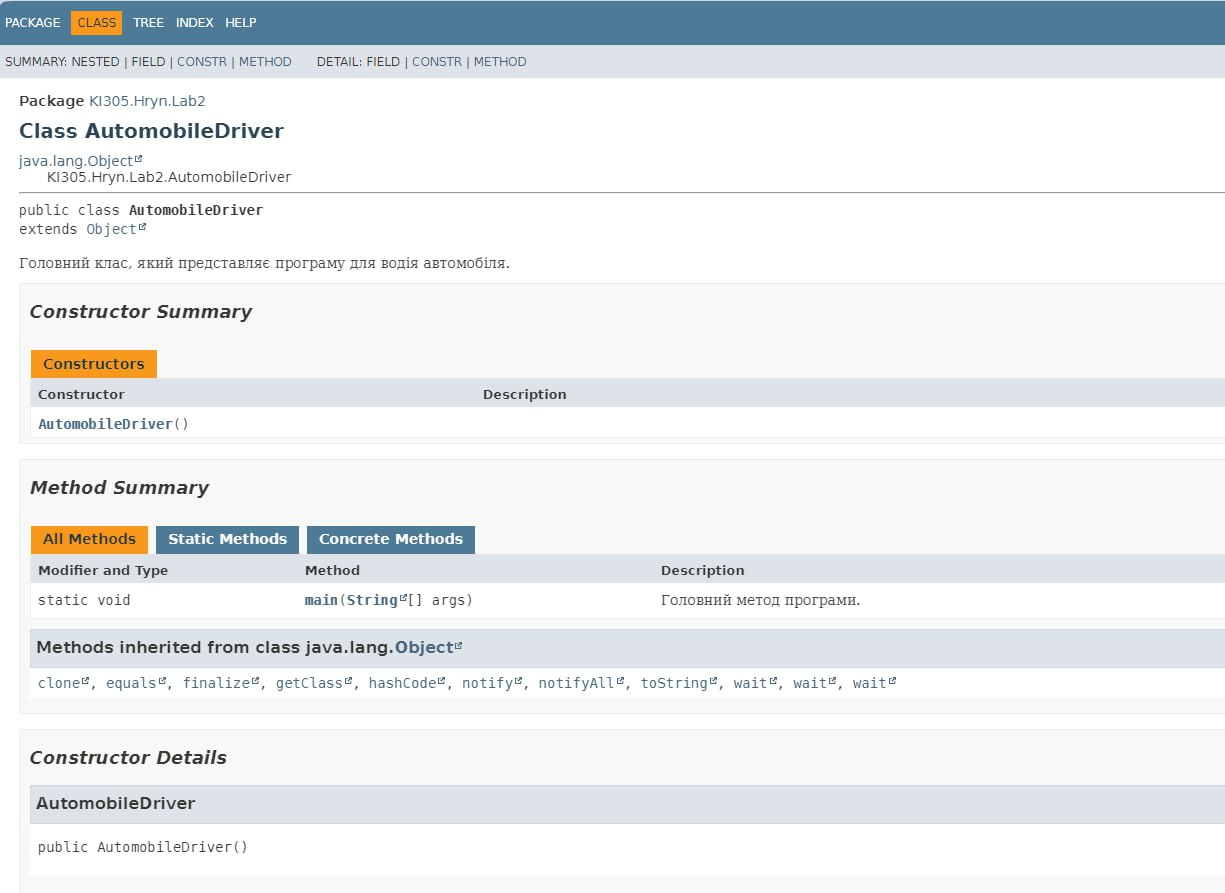


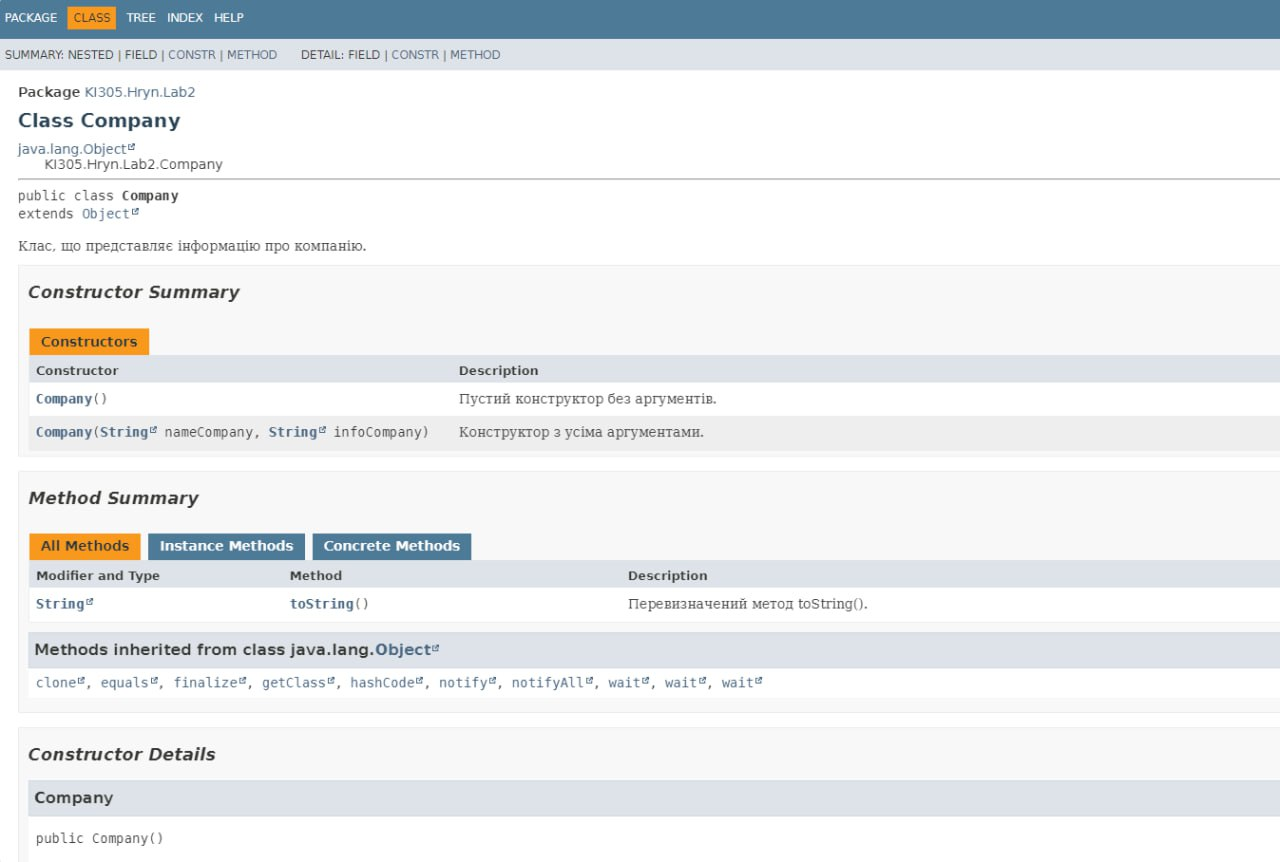


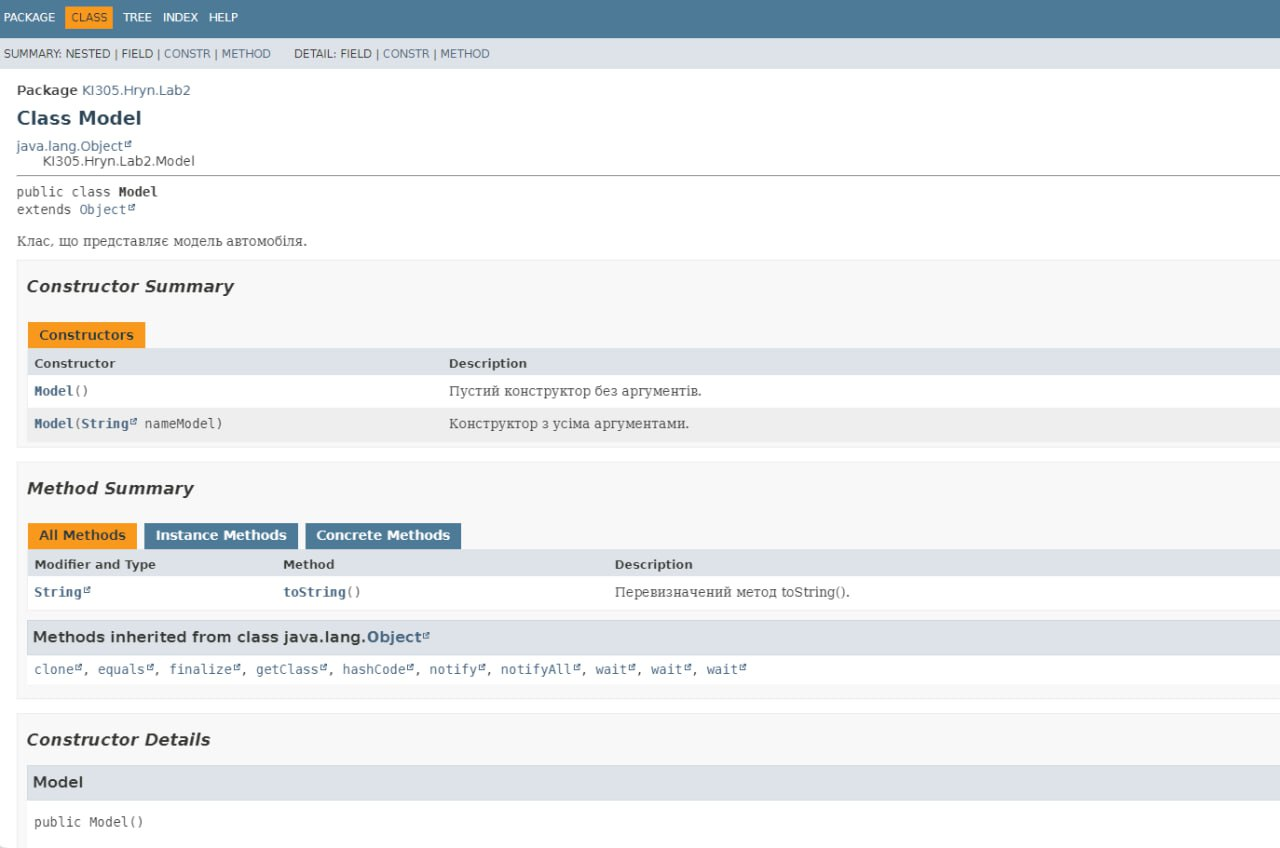


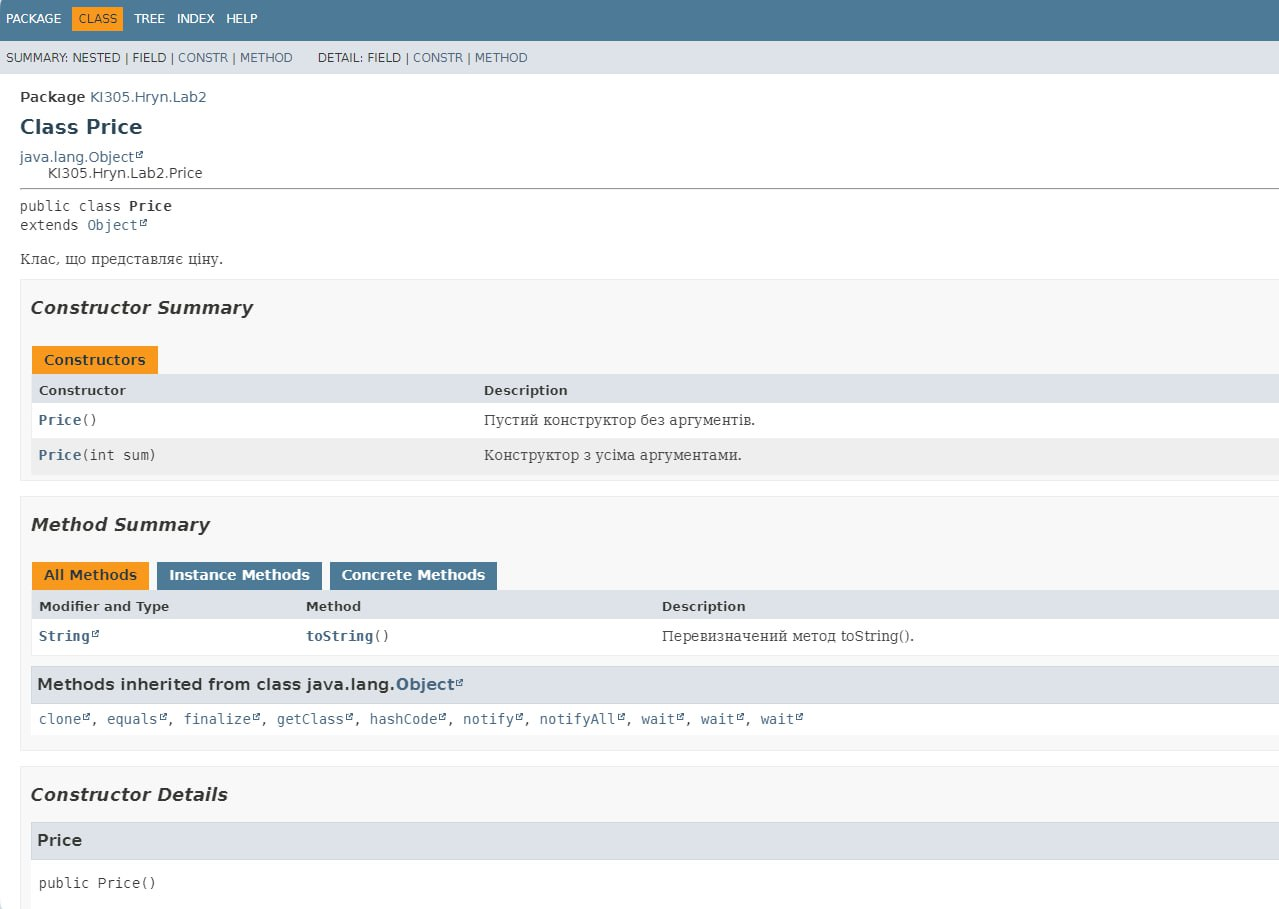






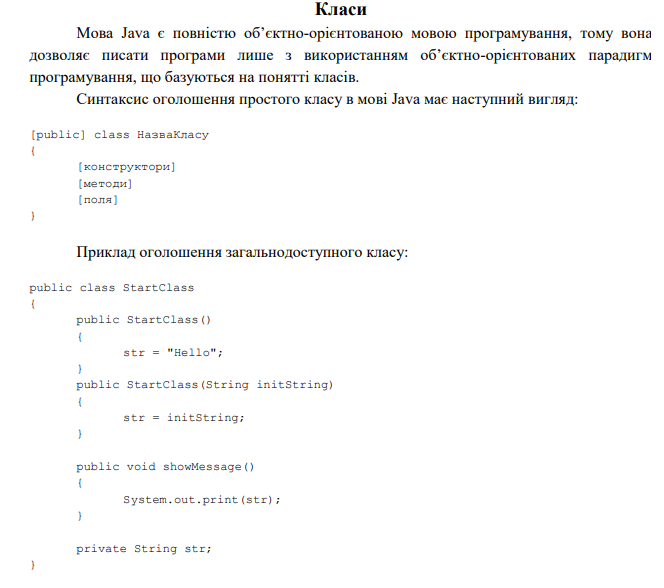




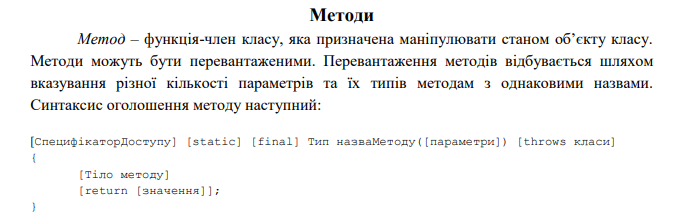


**Контрольні запитання**

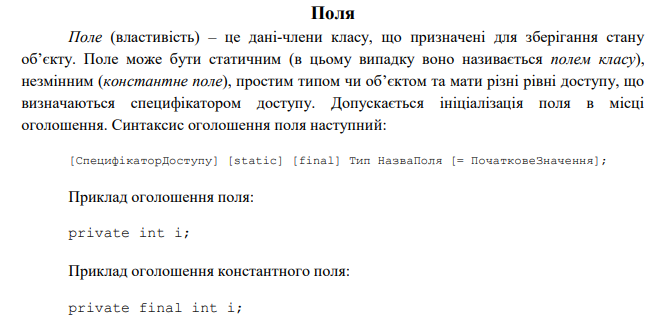
1. ***Синтаксис визначення класу.***

******

***2. Синтаксис визначення методу.***



***3.Синтаксис оголошення поля.***

******

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

public class MyClass {

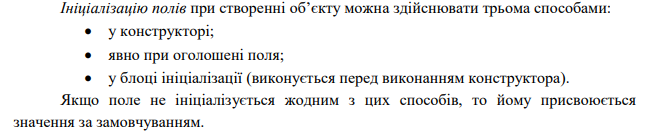
// Константне поле, ініціалізоване одразу

public final int MY\_CONSTANT\_FIELD = 42;

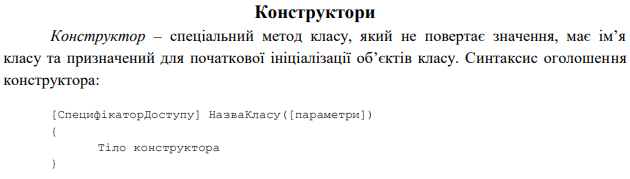
// Решта коду класу

}

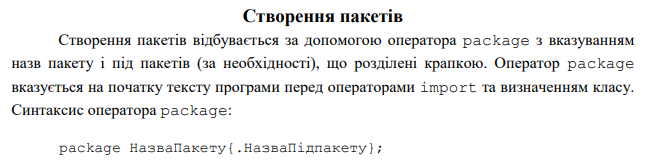
***5. Які є способи ініціалізації полів?***

******

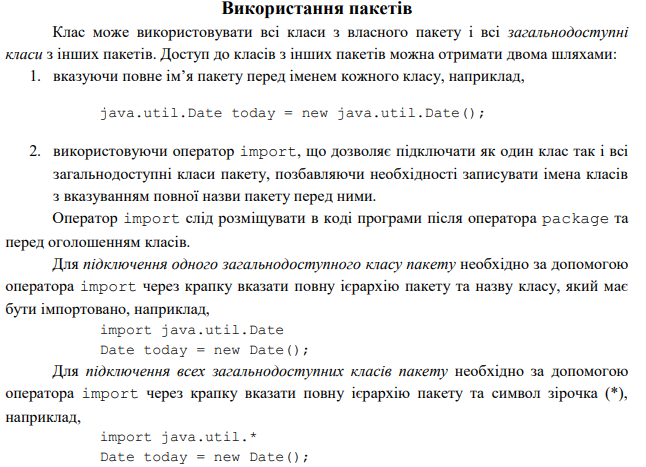
1. ***Синтаксис визначення конструктора.***

******

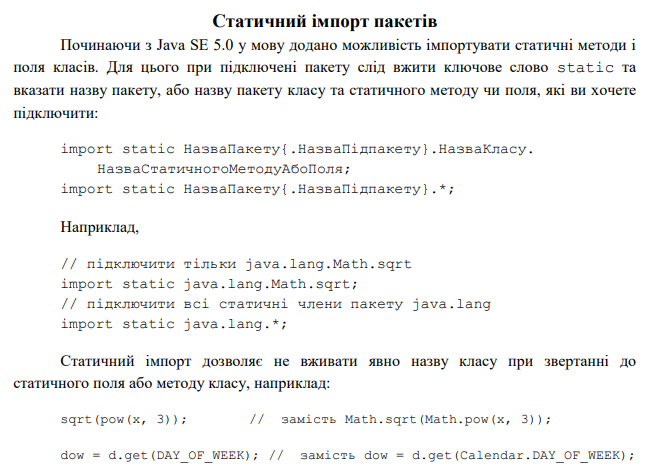
1. ***Синтаксис оголошення пакету.***

******

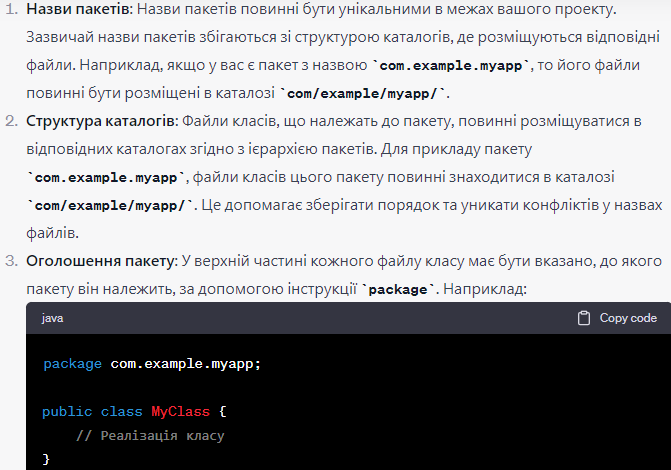
1. ***Як підключити до програми класи, що визначені в зовнішніх пакетах?***

******

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

******

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

****

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