Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



Звіт

з лабораторної роботи № 2

з дисципліни: «Кросплатформенні засоби програмування»

на тему: «КЛАСИ ТА ПАКЕТИ»

**Виконав:**

студент групи КІ-301

*Солтисяк О.Р.***Прийняв:**

доцент кафедри ЕОМ

*Іванов Ю. С.*

Львів – 2024

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

**Завдання (варіант № 26)**

1. Написати та налагодити програму на мові Java, що реалізує у вигляді класу предметну область згідно варіанту **(Шлюпка на веслах)**. 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.
2. Автоматично згенерувати документацію до розробленої програми.
3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
5. Дати відповідь на контрольні запитання.

**Вихідний код програми:**

***Файл App(main)***

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Copyright (c) 2013-2023 Lviv Polytechnic National University. All  
 \* Rights Reserved.  
 \*  
 \* This program and the accompanying materials are made available under the  
 \* terms of the Academic Free License v. 3.0 which accompanies this  
 \* distribution, and is available at https://opensource.org/license/afl-3-0-php/  
 \*  
 \* SPDX-License-Identifier: AFL-3.0  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
  
/\*\*  
 \* lab 2 package  
 \*/*package org.example;  
  
import static java.lang.System.*out*;  
import java.io.\*;  
  
*/\*\*  
 \* Клас `App` реалізує метод `main` для демонстрації можливостей класу `Shlyubka`.  
 \*/*public class App  
{  
 */\*\*  
 \** ***@param*** *args аргументи командного рядка  
 \** ***@throws*** *FileNotFoundException якщо файл не знайдено  
 \*/* public static void main(String[] args) throws FileNotFoundException  
 {  
 // Створення екземплярів компонентів  
 Vesla vesla = new Vesla(4);  
 Korpus korpus = new Korpus("Алюміній");  
 Sidinnya sidinnya = new Sidinnya(4);  
  
 // Створення об'єкта шлюпки  
 Shlyubka shlyubka = new Shlyubka(vesla, korpus, sidinnya);  
  
 // Демонстрація методів  
 shlyubka.row();  
 shlyubka.accelerate();  
 *out*.println("Поточна швидкість: " + shlyubka.getSpeed() + " км/год");  
 shlyubka.turnLeft();  
 shlyubka.decelerate();  
 *out*.println("Поточна швидкість: " + shlyubka.getSpeed() + " км/год");  
 shlyubka.stopRowing();  
 shlyubka.anchor();  
 shlyubka.liftAnchor();  
  
 // Отримання властивостей шлюпки  
 *out*.println("Кількість весел: " + shlyubka.getNumberOfOars());  
 *out*.println("Матеріал корпусу: " + shlyubka.getHullMaterial());  
 *out*.println("Кількість сидінь: " + shlyubka.getNumberOfSeats());  
  
 // Закриття ресурсів  
 shlyubka.dispose();  
 }  
}

***Файл Korpus.java***

package org.example;  
  
*/\*\*  
 \* Клас <code>Korpus</code> представляє корпус шлюпки.  
 \*/*class Korpus {  
 private String material;  
  
 */\*\*  
 \* Конструктор за замовчуванням.  
 \*/* public Korpus()  
 {  
 material = "Дерево";  
 }  
  
 */\*\*  
 \* Конструктор з параметром.  
 \** ***@param*** *mat матеріал корпусу  
 \*/* public Korpus(String mat)  
 {  
 material = mat;  
 }  
  
 */\*\*  
 \* Метод повертає матеріал корпусу.  
 \** ***@return*** *матеріал корпусу  
 \*/* public String getMaterial()  
 {  
 return material;  
 }  
  
 */\*\*  
 \* Метод встановлює матеріал корпусу.  
 \** ***@param*** *mat матеріал корпусу  
 \*/* public void setMaterial(String mat)  
 {  
 material = mat;  
 }  
}

**Shlyubka.java**

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Copyright (c) 2013-2023 Lviv Polytechnic National University.  
 \* All Rights Reserved.  
 \*  
 \* This program and the accompanying materials are made available under the  
 \* terms of the Academic Free License v. 3.0 which accompanies this  
 \* distribution, and is available at https://opensource.org/license/afl-3-0-php/  
 \*  
 \* SPDX-License-Identifier: AFL-3.0  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
  
/\*\*  
 \* lab 2 package  
 \*/*package org.example;  
  
import java.io.\*;  
  
*/\*\*  
 \* Клас <code>Shlyubka</code> реалізує модель шлюпки на веслах.  
 \*/*public class Shlyubka {  
 private Vesla vesla;  
 private Korpus korpus;  
 private Sidinnya sidinnya;  
 private PrintWriter fout;  
 private double speed;  
  
 */\*\*  
 \* Конструктор за замовчуванням.  
 \** ***@throws*** *FileNotFoundException якщо файл не знайдено  
 \*/* public Shlyubka() throws FileNotFoundException  
 {  
 vesla = new Vesla();  
 korpus = new Korpus();  
 sidinnya = new Sidinnya();  
 speed = 0.0;  
 fout = new PrintWriter(new File("Log.txt"));  
 }  
  
 */\*\*  
 \* Конструктор з параметрами.  
 \** ***@param*** *v об'єкт класу Vesla  
 \** ***@param*** *k об'єкт класу Korpus  
 \** ***@param*** *s об'єкт класу Sidinnya  
 \** ***@throws*** *FileNotFoundException якщо файл не знайдено  
 \*/* public Shlyubka(Vesla v, Korpus k, Sidinnya s) throws FileNotFoundException  
 {  
 vesla = v;  
 korpus = k;  
 sidinnya = s;  
 speed = 0.0;  
 fout = new PrintWriter(new File("Log.txt"));  
 }  
  
 */\*\*  
 \* Метод імітує початок веслування.  
 \*/* public void row()  
 {  
 fout.print("Почали веслувати\n");  
 fout.flush();  
 speed = 5.0;  
 }  
  
 */\*\*  
 \* Метод імітує зупинку веслування.  
 \*/* public void stopRowing()  
 {  
 fout.print("Зупинили веслування\n");  
 fout.flush();  
 speed = 0.0;  
 }  
  
 */\*\*  
 \* Метод імітує поворот ліворуч.  
 \*/* public void turnLeft()  
 {  
 fout.print("Поворот ліворуч\n");  
 fout.flush();  
 }  
  
 */\*\*  
 \* Метод імітує поворот праворуч.  
 \*/* public void turnRight()  
 {  
 fout.print("Поворот праворуч\n");  
 fout.flush();  
 }  
  
 */\*\*  
 \* Метод імітує прискорення.  
 \*/* public void accelerate()  
 {  
 fout.print("Прискорилися\n");  
 fout.flush();  
 speed += 2.0;  
 }  
  
 */\*\*  
 \* Метод імітує уповільнення.  
 \*/* public void decelerate()  
 {  
 fout.print("Уповільнилися\n");  
 fout.flush();  
 speed -= 2.0;  
 if (speed < 0) speed = 0;  
 }  
  
 */\*\*  
 \* Метод імітує опускання якоря.  
 \*/* public void anchor()  
 {  
 fout.print("Опустили якір\n");  
 fout.flush();  
 speed = 0.0;  
 }  
  
 */\*\*  
 \* Метод імітує підняття якоря.  
 \*/* public void liftAnchor()  
 {  
 fout.print("Підняли якір\n");  
 fout.flush();  
 }  
  
 */\*\*  
 \* Метод повертає поточну швидкість шлюпки.  
 \** ***@return*** *поточна швидкість  
 \*/* public double getSpeed()  
 {  
 fout.print("Отримали поточну швидкість\n");  
 fout.flush();  
 return speed;  
 }  
  
 */\*\*  
 \* Метод повертає кількість весел.  
 \** ***@return*** *кількість весел  
 \*/* public int getNumberOfOars()  
 {  
 return vesla.getNumberOfOars();  
 }  
  
 */\*\*  
 \* Метод повертає матеріал корпусу.  
 \** ***@return*** *матеріал корпусу  
 \*/* public String getHullMaterial()  
 {  
 return korpus.getMaterial();  
 }  
  
 */\*\*  
 \* Метод повертає кількість сидінь.  
 \** ***@return*** *кількість сидінь  
 \*/* public int getNumberOfSeats()  
 {  
 return sidinnya.getNumberOfSeats();  
 }  
  
 */\*\*  
 \* Метод звільняє використані ресурси.  
 \*/* public void dispose()  
 {  
 fout.close();  
 }  
}

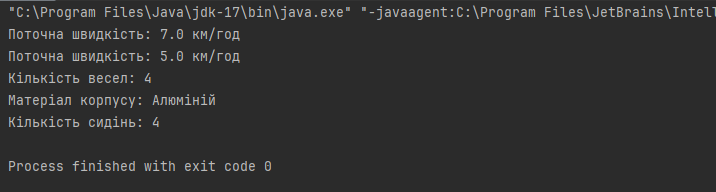
**SIDINNYA**

package org.example;  
  
*/\*\*  
 \* Клас <code>Sidinnya</code> представляє сидіння в шлюпці.  
 \*/*class Sidinnya {  
 private int numberOfSeats;  
  
 */\*\*  
 \* Конструктор за замовчуванням.  
 \*/* public Sidinnya()  
 {  
 numberOfSeats = 2;  
 }  
  
 */\*\*  
 \* Конструктор з параметром.  
 \** ***@param*** *num кількість сидінь  
 \*/* public Sidinnya(int num)  
 {  
 numberOfSeats = num;  
 }  
  
 */\*\*  
 \* Метод повертає кількість сидінь.  
 \** ***@return*** *кількість сидінь  
 \*/* public int getNumberOfSeats()  
 {  
 return numberOfSeats;  
 }  
  
 */\*\*  
 \* Метод встановлює кількість сидінь.  
 \** ***@param*** *num кількість сидінь  
 \*/* public void setNumberOfSeats(int num)  
 {  
 numberOfSeats = num;  
 }  
}

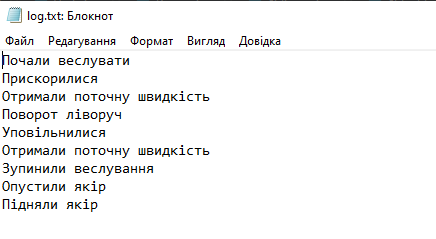
**VESLA**

package org.example;  
  
*/\*\*  
 \* Клас <code>Vesla</code> представляє весла.  
 \*/*class Vesla {  
 private int numberOfOars;  
  
 */\*\*  
 \* Конструктор за замовчуванням.  
 \*/* public Vesla()  
 {  
 numberOfOars = 2;  
 }  
  
 */\*\*  
 \* Конструктор з параметром.  
 \** ***@param*** *num кількість весел  
 \*/* public Vesla(int num)  
 {  
 numberOfOars = num;  
 }  
  
 */\*\*  
 \* Метод повертає кількість весел.  
 \** ***@return*** *кількість весел  
 \*/* public int getNumberOfOars()  
 {  
 return numberOfOars;  
 }  
  
 */\*\*  
 \* Метод встановлює кількість весел.  
 \** ***@param*** *num кількість весел  
 \*/* public void setNumberOfOars(int num)  
 {  
 numberOfOars = num;  
 }  
}

**РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ**



**Вміст файлу log.txt:**



**Відповіді на контрольні запитання:**

1. Синтаксис визначення класу.

public class ClassName {

// Вміст класу

}

1. Синтаксис визначення методу.

[модифікатори] [тип повернення] ім'яМетоду([параметри]) {

// Тіло методу

}

1. Синтаксис оголошення поля.

[модифікатори] [тип] ім'яПоля;

1. Як оголосити та ініціалізувати константне поле?

[модифікатори] static final тип ІМЯ\_КОНСТАНТИ = значення;

1. Які є способи ініціалізації полів?

* через конструктор
* в блоку ініціалізації
* при оголошенні або в методі.

1. Синтаксис визначення конструктора.

[модифікатори] Ім'яКласу([параметри]) {

// Тіло конструктора

}

1. Синтаксис оголошення пакету.

package ім'яПакету;

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

Використовуючи імпорт: import ім'яПакету.ім'яКласу;

1. В чому суть статичного імпорту пакетів?

Суть статичного імпорту пакетів полягає в тому, що можна імпортувати статичні члени класу (методи або поля) і використовувати їх без префіксу класу.

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

Вимоги до файлів та каталогів при використанні пакетів включають в себе правильну структуру каталогів, іменування файлів та розташування пакетів у відповідних папках, що відповідають іменам пакетів.

**Висновок:**

На даній лабораторній роботі я отримав навички розробки класів та пакетів у мові програмування Java. Також на цій лабораторній роботі я ознайомився з базовими конструкціями Java, такими як оголошення класів, методів та полів. Було здобуто навички структурувати свій код, визначати доступ до класів та їх членів, а також використовувати модифікатори доступу для керування видимістю.