

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-306
Бокало П.М.

Прийняв:
доцент кафедри ЕОМ
Іванов Ю. С.

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

Завдання(варіант №2):

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.

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.

Індивідуальне завдання:

2. Космічний корабель

Вихідний код програми:

Файл KI306.Bokalo.Lab2.java:

```
package KI306.Bokalo.Lab2;

import java.io.FileNotFoundException;

/**
 * Ця програма реалізовує симуляцію написаного класу згідно до 2 варіанту - Космічний корабель
 * @author Petro Bokalo KI-203
 * @version 1.0
 * @since version 1.0
 * @throws FileNotFoundException - call when it is not a file available
 */

public class Lab2BokaloKI306
{
```

```

    public static void main(String[] args) throws FileNotFoundException
    {

        Starship ship = new Starship("King", "Sir Rodock");

        ship.stats();
        ship.move();
        ship.fight();
        ship.retreat();
        ship.call_captain();
        ship.stats();

        for(int i = 0; i < 10; i++)
        {
            ship.fight();
        }
        ship.stats();
        ship.repair();
        ship.call_captain();

        ship.move();
        ship.move();
        ship.stats();
        ship.refuel();
        ship.stats();

        ship.closer();

    }

}

```

Файл Starship.java

```

package KI306.Bokalo.Lab2;

import java.io.FileNotFoundException;
import java.io.*;

/**
 * Тут реалізовано класи Starship і Captain
 * @author Petro Bokalo KI-203
 * @version 1.0
 * @since version 1.0
 */

public class Starship
{
    private int fuel;

    private String s_name;

    private int s_health;

```

```

private int staff;

private Captain captain;

private boolean is_fighting = false;

private PrintWriter fout;


public int getFuel() {
    fout.write("\n public int getFuel() : was used\n");
    return fuel;
}


public void setFuel(int fuel) {
    this.fuel = fuel;
    fout.write("\n public void setFuel(int fuel) : was used\n");
}


public String getS_name() {
    fout.write("\n public String getS_name() : was used\n");
    return s_name;
}


public void setS_name(String s_name) {
    fout.write("\n public void setS_name() : was used\n");
    this.s_name = s_name;
}


public int getS_health() {
    fout.write("\n public int getS_health() : was used\n");
    return s_health;
}


public void setS_health(int s_health) {
    fout.write("\n public void setS_health(int s_health) : was used\n");
    this.s_health = s_health;
}


public int getStaff() {
    fout.write("\n public int getStaff() : was used\n");

```

```

        return staff;
    }

    public void setStaff(int staff) {
        fout.write("\n public void setStaff(int staff)() : was used\n");
        this.staff = staff;
    }

    public Starship()
{
    fuel = 100;
    s_name = "void";
    s_health = 100;
    staff = 10;
    captain = new Captain();
    try
    {
        fout = new PrintWriter(new BufferedWriter(new FileWriter("Starship_log.txt", true)));
    } catch (IOException e)
    {
        System.err.println("Помилка: файл Captain_log.txt не може бути створений або
записаний.");
    }

    fout.write("\n public Starship() : was used\n");
}

    public Starship(String name, String c_name)
{
    fuel = 100;
    this.s_name = name;
    s_health = 100;
    staff = 10;
    captain = new Captain(c_name);
    try
    {
        fout = new PrintWriter(new BufferedWriter(new FileWriter("Starship_log.txt", true)));
    } catch (IOException e)
    {

```

```
        System.err.println("Помилка: файл Captain_log.txt не може бути створений або записаний.");
```

```
    }
```

```
        fout.write("\n public Starship(String name, String c_name) : was used\n");
    }
```

```
public void closer()
{
    fout.write("public void closer() : was used\n");
    if (fout != null)
    {
        fout.close();
        captain.close();
    }
}
```

```
public void call_captain()
{
    captain.Captain_stats();
}
```

```
public void move()
{
    if(fuel > 10)
    {
        fuel -=10;
    }
    else
    {
        System.out.print("\n Can't move!!! \n");
    }
    fout.write("\n public move() : was used\n");
}
```

```
public void damage()
{
    if(s_health > 0 && staff > 0)
    {
```

```

        s_health -= 10;
        staff -= 1;

    }else if (s_health == 0)
    {
        System.out.print("\n Starship is critically damaged !!! \n");
    }
    fout.write("\n  public damage()  : was used\n");
}

public void repair()
{
    if(s_health < 100)
    {
        staff += 5;
        int coins = captain.getCoins();
        coins -= 50;
        captain.setCoins(coins);
    }
    else
    {
        System.out.print("Repaired\n");
    }
    fout.write("\n  public repair()  : was used\n");
}

public void fight ()
{
    is_fighting = true;
    damage();
    int coins = captain.getCoins();
    captain.setCoins(coins + 25);

    fout.write("\n  public fight()  : was used\n");
}

public void retreat ()
{
    is_fighting = false;

```

```

        fout.write("\n  public retreat()  : was used\n");
    }

    public void stats ()
    {
        System.out.print("Stats : \n");
        System.out.print("Name : " + s_name + "\n");
        System.out.print("Fuel : " + fuel + "\n");
        System.out.print("Health : " + s_health + "\n");
        System.out.print("Staff : " + staff + "\n");
        System.out.print("Fighting : " + is_fighting + "\n\n");
        fout.write("\n  public stats()  : was used\n");

    }

    public void refuel()
    {
        fuel = 100;
        int coins = captain.getCoins();
        captain.setCoins(coins - 25);
        fout.write("\n  public refuel()  : was used\n");
    }

    public void clearShip_logs()

    {
        fout.write("public void clearShip_logs() : was used\n");
        try
        {
            PrintWriter clearWriter = new PrintWriter("Starship_log.txt");
            clearWriter.close();
        } catch (FileNotFoundException e) {
            System.err.println("Помилка: файл 'Starship_log.txt' не може бути очищений.");
        }
    }

}

```



```
class Captain {  
    private String name;  
    private int coins;  
    private int health;  
    private static int id = 0;  
    private PrintWriter fout;  
  
    public String getName() {  
        fout.write("\n public String getName() : was used\n");  
        return name;  
    }  
  
    public void setName(String name) {  
        fout.write("\n public void setName(String name) : was used\n");  
        this.name = name;  
    }  
  
    public int getCoins() {  
        fout.write("\n public int getCoins() : was used\n");  
        return coins;  
    }  
  
    public void setCoins(int coins) {  
        fout.write("\n public void setCoins(int coins) : was used\n");  
        this.coins = coins;  
    }  
  
    public int getHealth() {  
        fout.write("\n public int getHealth() : was used\n");  
        return health;  
    }  
  
    public void setHealth(int health) {  
        fout.write("\n public void setHealth(int health) : was used\n");  
        this.health = health;  
    }  
  
    public Captain() {
```

```

        this.name = "Noname";

        this.coins = 100;

        this.health = 100;

        id++;

        try {
            fout = new PrintWriter(new BufferedWriter(new FileWriter("Captain_log.txt", true)));
        } catch (IOException e) {
            System.err.println("Помилка: файл Captain_log.txt не може бути створений або записаний.");
        }

        fout.write("\n public Captain() : was used\n");
        Captain_stats_file();
    }

    public Captain(String name) {

        this.name = name;

        coins = 100;

        health = 100;

        id++;

        try {
            fout = new PrintWriter(new BufferedWriter(new FileWriter("Captain_log.txt", true)));
        } catch (IOException e) {
            System.err.println("Помилка: файл Captain_log.txt не може бути створений або записаний.");
        }

        fout.write("\n public Captain (String name) : was used\n");
        Captain_stats_file();
    }

    public void Captain_stats_file() {
        fout.write("\n public void Captain_stats_file() : was used \n\n");

        fout.write("New Captain created :\n");

        fout.write("ID : " + id + "\n");

        fout.write("Name : " + name + "\n");
    }

```

```

        fout.write("Coins : " + coins + "\n");
        fout.write("Health : " + health + "\n\n");
        fout.flush(); // Очистити буфер і записати дані в файл
    }

    public void Captain_stats() {
        fout.write("public void Captain_stats : was used \n");
        System.out.print("Your captain has : \n");
        System.out.print("Name : " + name + "\n");
        System.out.print("Coins : " + coins + "\n");
        System.out.print("Health : " + health + "\n\n");
    }

    public void close()
    {
        fout.write("public void close() : was used\n");
        if (fout != null)
        {
            fout.close();
        }
    }

    public void clearCaptain_logs()
    {
        fout.write("public void clearCaptain_logs() : was used\n");
        try
        {
            PrintWriter clearWriter = new PrintWriter("Captain_log.txt");
            clearWriter.close();
        } catch (FileNotFoundException e) {
            System.err.println("Помилка: файл 'Captain_log.txt' не може бути очищений.");
        }
    }
}

```

Результат роботи програми:

```
Stats :  
Name : King  
Fuel : 100  
Health : 100  
Staff : 10  
Fighting : false  
  
Your captain has :  
Name : Sir Rodock  
Coins : 125  
Health : 100  
  
Stats :  
Name : King  
Fuel : 90  
Health : 90  
Staff : 9  
Fighting : false  
  
Starship is critically damaged !!!  
Stats :  
Name : King  
Fuel : 90  
Health : 0  
Staff : 0  
Fighting : true  
  
Your captain has :  
Name : Sir Rodock  
Coins : 325  
Health : 100  
  
Stats :  
Name : King  
Fuel : 70  
Health : 0  
Staff : 5  
Fighting : true  
  
Stats :  
Name : King  
Fuel : 100  
Health : 0  
Staff : 5  
Fighting : true
```

Вміст файлів :

Файл Captain.txt:

public Captain (String name) : was used

public void Captain_stats_file() : was used

New Captain created :

ID : 1

Name : Sir Rodock

Coins : 100

Health : 100

public int getCoins() : was used

public void setCoins(int coins) : was used

public void Captain_stats : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public void Captain_stats : was used

public int getCoins() : was used

public void setCoins(int coins) : was used

public void close() : was used

Файл Starship.txt

public Starship(String name, String c_name) : was used

public stats() : was used

public move() : was used

public damage() : was used

public fight() : was used

public retreat() : was used

public stats() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public damage() : was used

public fight() : was used

public stats() : was used

public repair() : was used

public move() : was used

public move() : was used

public stats() : was used

public refuel() : was used

public stats() : was used

public void closer() : was used

Фрагмент згенерованої документації:

PACKAGE

CLASS

USE

TREE

INDEX

HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD

DETAIL: FIELD | CONSTR | METHOD

Package

KI306.Bokalo.Lab2

Class

Starship

java.lang.Object

KI306.Bokalo.Lab2.Starship

public class Starship

extends Object

Тут реалізовано класи Starship і Captain

Since:
version 1.0

Version:
1.0

Author:
Petro Bokalo KI-203

Constructor Summary

Constructors

Constructor	Description
Starship()	
Starship(String name, String c_name)	

Method Summary

All Methods

Instance Methods

Concrete Methods

Modifier and Type	Method	Description
void	call_captain()	
void	clearShip_logs()	
void	closer()	

Відповідь на контрольні питання

- Синтаксис визначення класу.
[public] class НазваКласу
- Синтаксис визначення методу.
[СпецифікаторДоступу] [static] [final] Тип назваМетоду([параметри]) [throws класи]
{
[Тіло методу]
[return [значення]];
}

3. Синтаксис оголошення поля.

[СпецифікаторДоступу] [static] [final] Тип НазваПоля [= ПочатковеЗначення];

4. Як оголосити та ініціалізувати константне поле?

```
public class MyClass {  
    final int myConstant = 42;  
}
```

5. Які є способи ініціалізації полів?

- У конструкторі
- Явно при оголошенні
- У блоці ініціалізації

6. Синтаксис визначення конструктора.

```
public MyClass([parameters])  
{  
}
```

7. Синтаксис оголошення пакету.

package [Назва пакету].[Назва під пакетів]

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

За допомогою ключового слова import

9. В чому суть статичного імпорту пакетів?

Це дає можливість імпортувати статичні члени (методи і поля) з класів без необхідності використовувати префікс імені класу.

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

- *Клас повинен мати відповідну директорію відносно кореневого каталогу проекту.*
- *Ім'я файлу повинно точно відповідати імені класу і має мати розширення .java для файлів з вихідним кодом і .class для скомпільованих файлів.*
- *Шлях до кореневої директорії проекту повинен бути встановлений в системі, а інтерпретатор Java повинен мати доступ до цього шляху для пошуку класів і пакетів.*

Висновок: на цій лабораторній роботі, я ознайомився з класами та пакетами в мові програмування java. Написав програму згідно до свого варіанту. Навчився оголошувати і використовувати класи, поля, методи та пакети.