Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



Звіт

з лабораторної роботи № 3

з дисципліни: «Кросплатформенні засоби програмування»

на тему: «Спадкування та інтерфейси»

**Виконав:**

студент групи КІ-307

Трофименко Є.Е.

**Прийняв:**

доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

**Мета роботи:** ознайомитися з спадкуванням та інтерфейсами у мові Java.

**Завдання (варіант № 22)**

1. Написати та налагодити програму на мові Java, що розширює клас, що реалізований у лабораторній роботі No2, для реалізації предметної області заданої варіантом. Суперклас, що реалізований у лабораторній роботі No2, зробити абстрактним. Розроблений підклас має забезпечувати механізми свого коректного функціонування та реалізовувати мінімум один інтерфейс. Програма має розміщуватися в пакеті Група.Прізвище.Lab3 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

2. Автоматично згенерувати документацію до розробленого пакету.

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**

*Lab03\_Class.java*

/\*\*

\* Програма, що реалізує у вигляді класу предметну область автомату.

\*/

package ki307.Trofymenko.Lab3;

/\*\*

\* Клас для створення предметної області автомату...

\* **@version** 1.0

\*/

public class Lab03\_Class{

/\*\*

\* Конструктор за замовченням

\*/

public Lab03\_Class() {}

/\*\*

\* Вхідний метод програми

\* **@param** args визначення методу main

\*/

public static void main(String[] args) {

AssaultRifle rifle = new AssaultRifle(new MagazineCartridge(),new Fuse(),new Trigger());

rifle.Fire();

rifle.Reload();

rifle.TurnOnTheFuse();

rifle.Fire();

rifle.TurnOffTheFuse();

rifle.Fire();

rifle.Fire();

rifle.Fire();

rifle.Fire();

rifle.CheckMagazineCartridge();

rifle.ClosePrintWriter();

}

}

*Rifle.java*

package ki307.Trofymenko.Lab3;

import java.io.FileNotFoundException;

import java.io.PrintWriter;

/\*\*

\* Головний клас програми Автомату

\*/

public abstract class Rifle {

/\*\*

\* new MagazineCartridge

\*/

MagazineCartridge magazineCartridge;

/\*\*

\* new Fuse

\*/

Fuse fuse;

/\*\*

\* new Trigger

\*/

Trigger trigger;

/\*\*

\* new PrintWriter

\*/

PrintWriter printWriter;

/\*\*

\* Constructor

\*/

public Rifle() {

}

/\*\*

\* Constructor

\* **@param** magazineCartridge

\* **@param** trigger

\*/

public Rifle(MagazineCartridge magazineCartridge, Trigger trigger) {

this.magazineCartridge = magazineCartridge;

this.trigger = trigger;

}

/\*\*

\* Constructor

\* **@param** magazineCartridge

\* **@param** fuse

\* **@param** trigger

\*/

public Rifle(MagazineCartridge magazineCartridge, Fuse fuse, Trigger trigger){

this.magazineCartridge = new MagazineCartridge();

this.fuse = new Fuse();

this.trigger = new Trigger();

try {

this.printWriter = new PrintWriter("Log.txt");

} catch (FileNotFoundException e) {

throw new RuntimeException(e);

}

}

/\*\*

\* Method that firing and returns boolean

\* **@return**

\*/

public boolean Fire(){

if(!magazineCartridge.isEmpty() && !fuse.isWorking()) {

UseStock();

TakeAim();

trigger.setOn(true);

System.***out***.println("Firing");

printWriter.println("Firing");

printWriter.flush();

magazineCartridge.setCapacity(magazineCartridge.getCapacity() - 1);

return true;

}

return false;

}

/\*\*

\* Method of reloading that returns void

\*/

public void Reload(){

magazineCartridge.setEmpty(false);

magazineCartridge.setCapacity(30);

System.***out***.println("Reloading");

printWriter.println("Reloading");

printWriter.flush();

}

/\*\*

\* Method that setting fuse true

\*/

public void TurnOnTheFuse(){

fuse.setWorking(true);

System.***out***.println("Putting on the Fuse");

printWriter.println("Putting on the Fuse");

printWriter.flush();

}

/\*\*

\* Method that setting fuse false

\*/

public void TurnOffTheFuse(){

fuse.setWorking(false);

System.***out***.println("Putting off the Fuse");

printWriter.println("Putting off the Fuse");

printWriter.flush();

}

/\*\*

\* Method that check Magazine Capacity

\*/

public void CheckMagazineCartridge(){

magazineCartridge.getCapacity();

System.***out***.println("Checking amount of bullets: " + magazineCartridge.getCapacity());

printWriter.println("Checking amount of bullets: " + magazineCartridge.getCapacity());

printWriter.flush();

}

/\*\*

\* Method that cleaning rifle

\*/

public void CleanRifle(){

System.***out***.println("Cleaning rifle");

printWriter.println("Cleaning rifle");

printWriter.flush();

}

/\*\*

\* Method that taking aim

\*/

public void TakeAim(){

System.***out***.println("Taking aim");

printWriter.println("Taking aim");

printWriter.flush();

}

/\*\*

\* Method that using stock

\*/

public void UseStock(){

System.***out***.println("Using stock");

printWriter.println("Using stock");

printWriter.flush();

}

/\*\*

\* Method that wearing belt

\*/

public void UseBelt(){

System.***out***.println("Wearing belt");

printWriter.println("Wearing belt");

printWriter.flush();

}

/\*\*

\* Method that regulating resistance of trigger

\* **@param** resistance

\*/

public void RegulateResistanceOfTrigger(int resistance){

trigger.setResistance(resistance);

System.***out***.println("Setting Resistance Of Trigger");

printWriter.println("Setting Resistance Of Trigger");

printWriter.flush();

}

/\*\*

\* Method to close file

\*/

public void ClosePrintWriter(){

printWriter.println("Closing file");

printWriter.flush();

printWriter.close();

}

}

*AssaultRifle.java*

package ki307.Trofymenko.Lab3;

public class AssaultRifle extends Rifle implements Bayonet{

/\*\*

\* Constructor

\*/

public AssaultRifle() {

}

/\*\*

\* Constructor

\* **@param** magazineCartridge

\* **@param** trigger

\*/

public AssaultRifle(MagazineCartridge magazineCartridge, Trigger trigger) {

super(magazineCartridge, trigger);

}

/\*\*

\* Constructor

\* **@param** magazineCartridge

\* **@param** fuse

\* **@param** trigger

\*/

public AssaultRifle(MagazineCartridge magazineCartridge, Fuse fuse, Trigger trigger) {

super(magazineCartridge, fuse, trigger);

}

/\*\*

\* Implementing method WearBayonet

\*/

public void WearBayonet() {

System.***out***.println("Wearing Bayonet");

printWriter.print("Wearing Bayonet");

printWriter.flush();

}

}

**Результат виконання програми**

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

**Фрагмент згенерованої документації**

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, белый, чек

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, чек

Автоматически созданное описание

**Відповіді на контрольні запитання**

1. Синтаксис реалізації спадкування.

*Синтаксис реалізації спадкування в мові Java має наступний вигляд:*

*class Підклас extends Суперклас*

*{*

*Додаткові поля і методи*

*}*

1. Що таке суперклас та підклас?

*В термінах мови Java базовий клас найчастіше називається суперкласом, а похідний клас – підкласом.*

1. Як звернутися до членів суперкласу з підкласу?

*У цій ситуації слід використати ключове слово* ***super****, яке вказує, що елемент до якого йде звернення,розташовується у суперкласі, а не у підкласі.*

1. Коли використовується статичне зв’язування при виклику методу?

*Статичне зв'язування відбувається під час компіляції програми.*

1. Як відбувається динамічне зв’язування при виклику методу?

*Поліморфізм реалізується за допомогою механізму динамічного зв’язування, який полягає у тому, що вибір методу, який необхідно викликати, відбувається не на етапі компіляції, а під час виконання програми.*

1. Що таке абстрактний клас та як його реалізувати?

*Абстрактні класи призначені бути основою для розробки ієрархій класів та не дозволяють створювати об’єкти свого класу. Вони реалізуються за допомогою ключового слова abstract.*

1. Для чого використовується ключове слово instanceof?

*Щоби перевірити коректність приведення супертипу до підтипу.*

1. Як перевірити чи клас є підкласом іншого класу?

*Для перевірки того, чи клас є підкласом іншого класу, використовується оператор instanceof*, *який повертає true, якщо посилання посилається на об’єкт фактичний тип якого є не вищим в ієрархії типів, ніж вказаний у операторі instanceof, і false у протилежному випадку.*

1. Що таке інтерфейс?

*Це методи без їхніх реалізацій, яку клас може реалізувати. Клас повинен обо’язково реалізувати все методи, реалізовані в інтерфейсі.*

1. Як оголосити та застосувати інтерфейс?

*Синтаксис*

*оголошення інтерфейсів:*

*[public] interface НазваІнтерфейсу*

*{*

*Прототипи методів та оголошення констант інтерфейсу*

*}*

*Щоб клас реалізував інтерфейс необхідно:*

*1) Оголосити за допомогою ключового слова implements, що клас реалізує інтерфейс. Якщо клас реалізує кілька інтерфейсів, то вони перелічуються через кому після ключового слова implements.*

*2) Визначити у класі усі методи, що вказані у інтерфейсі.*

**Висновок**

На данній лабораторній роботі я ознайомився зі спадкування класів та інтерфейсами у мові Java. Також ознайомився з поліморфізмом та абстрактними класами. Зробив суперклас, що реалізований у лабораторній роботі №2 абстрактним. Також реалізував інтерфейс, згідно індивідуального завдання та автоматично згенерував документацію Javadoc до свого проекту.