Міністерство освіти та науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



**ЗВІТ**

**з лабораторної роботи № 4**

**з дисципліни:** «Кросплатформенні засоби програмування»

**на тему:** «СПАДКУВАННЯ ТА ІНТЕРФЕЙСИ»

Виконав: ст. гр. КІ-35

Флик Є.В.

Прийняв:

доцент кафедри ЕОМ

Іванов Ю.С

Львів-2022

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

**ЗАВДАННЯ**

1. Написати та налагодити програму на мові Java, що розширює клас, що реалізований у лабораторній роботі №3, для реалізації предметної області заданої варіантом. Суперклас, що реалізований у лабораторній роботі №3, зробити абстрактним. Розроблений підклас має забезпечувати механізми свого коректного функціонування та реалізовувати мінімум один інтерфейс. Програма має розміщуватися в пакеті Група.Прізвище.Lab4 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.
2. Автоматично згенерувати документацію до розробленого пакету.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.
4. Дати відповідь на контрольні запитання.

**Варіант 22:**

****

**Текст програми:**

|  |
| --- |
| ***Лістинг AutomaticGun***  */\*\*  \* lab 3 package  \*/* package KI35.Flyk.Lab4;  import java.io.\*;  */\*\*  \* Class <code>AutomaticGun</code> implements automatic gun  \** ***@author*** *Flyk Yevhenii  \** ***@version*** *1.0  \*/* public abstract class AutomaticGun {  public static int *count*;  private Magazine m\_magazine;  private Fuse m\_fuse;  private Barrel m\_barrel;   */\*\*  \* Constructor  \*/* public AutomaticGun(){  m\_magazine =new Magazine();  m\_fuse = new Fuse();  m\_barrel = new Barrel();  }  */\*\*  \* Constructor  \** ***@param*** *\_length Length of barrel  \** ***@param*** *\_caliber Caliber of barrel  \*/* public AutomaticGun(int \_length,float \_caliber){  m\_magazine =new Magazine();  m\_fuse= new Fuse();  m\_barrel =new Barrel(\_length,\_caliber);   *count*++;  }  */\*\*  \* Method implements get of count  \*/* public int getCount(){  return *count*;  }  */\*\*  \* Method implements reload ofmagazine  \*/* public void ReloadAutomaticGun(){  m\_magazine.load();  m\_magazine.setAmmunition(30);  System.*out*.println("RELOAD!!!");  }  */\*\*  \* Method implements single firing  \*/* public void SingleFiring(){  if(m\_magazine.getAmmunition()>0 && !m\_fuse.isState()){  m\_magazine.reduction();   System.*out*.println("SHOOT!!!");  }else  {  System.*out*.println("Shooting is impossible!");  if(m\_magazine.getAmmunition()==0){  System.*out*.println("Reload the AutomaticGun");  }else  {  System.*out*.println("Turn off the fuse");  }  }  }  */\*\*  \* Method implements burst firing  \*/* public void BurstFiring(){  if(m\_magazine.getAmmunition()>0 && !m\_fuse.isState()){  for(int i=0;i<3;++i) {  m\_magazine.reduction();    System.*out*.println("SHOOT!!!");  }  }else  {  System.*out*.println("Shooting is impossible!");  if(m\_magazine.getAmmunition()==0){  System.*out*.println("Reload the AutomaticGun");  }else  {  System.*out*.println("Turn off the fuse");  }  }  }  */\*\*  \* Method implements switch on fuse  \*/* public void SwitchOnFuse(){  if(!m\_fuse.isState()){  m\_fuse.setState(true);    System.*out*.println("Fuse switching!!!");  }  else{  System.*out*.println("The fuse is already on!");  }  }  */\*\*  \* Method implements switch off fuse  \*/* public void SwitchOffFuse(){   if(m\_fuse.isState()){  m\_fuse.setState(false);    System.*out*.println("Fuse switching!!!");  }  else{  System.*out*.println("The fuse is already off!");  }  }   */\*\*  \* Method implements set information about barrel  \** ***@param*** *\_length Length of barrel  \** ***@param*** *\_caliber Caliber of barrel  \*/* public void SetBarrel(int \_length,float \_caliber){  m\_barrel.setLength(\_length);  m\_barrel.setCaliber(\_caliber);   }  */\*\*  \* Method implements printing information about gun  \*/* public void PrintGun(){  System.*out*.println("Amount of ammunition: " + m\_magazine.getAmmunition());  System.*out*.println("Current state of magazine: "+m\_magazine.isInserted());  System.*out*.println("Current state of fuze: "+m\_fuse.isState());  System.*out*.println("Length of barrel: "+m\_barrel.getLength());  System.*out*.println("Caliber of barrel: "+m\_barrel.getCaliber());  }  */\*\*  \* Method implements set of magazine  \** ***@param*** *\_ammunition  \** ***@param*** *\_state  \*/* public void SetMagazine(int \_ammunition,boolean \_state){  m\_magazine.setAmmunition(\_ammunition);  m\_magazine.setInserted(\_state);   }  */\*\*  \* Method implements set of fuse  \** ***@param*** *\_state  \*/* public void SetFuse(boolean \_state){  m\_fuse.setState(\_state);   }  } */\*\*  \* Class <code>Magazine</code> implements magazine  \** ***@author*** *Flyk Yevhenii  \** ***@version*** *1.0  \*/* class Magazine{  //number of ammunition  private int ammunition;  //current state of magazine  private boolean inserted;   */\*\*  \* Constructor  \*/* public Magazine() {  this.ammunition = 30;  this.inserted =true;  }  */\*\*  \* Method returns ammunition  \** ***@return*** *Number of ammunition  \*/* public int getAmmunition() {  return ammunition;  }  */\*\*  \* Method returns current state magazine  \** ***@return*** *Current state magazine  \*/* public boolean isInserted() {  return inserted;  }  */\*\*  \* Method set current state magazine  \** ***@param*** *inserted New current state magazine  \*/* public void setInserted(boolean inserted) {  this.inserted = inserted;  }  */\*\*  \* Method implements a reduction in ammunition  \*/* public void reduction(){  --(this.ammunition);  }  */\*\*  \* Method implements store removal  \*/* public void unLoad(){  this.inserted=false;  }  */\*\*  \* Method implements magazine insertion  \*/* public void load(){  this.inserted=true;  }  */\*\*  \* Method set current state magazine  \** ***@param*** *ammunition New current state magazine  \*/* public void setAmmunition(int ammunition) {  this.ammunition = ammunition;  } } */\*\*  \* Class <code>Fuse</code> implements fuse  \** ***@author*** *Flyk Yevhenii  \** ***@version*** *1.0  \*/* class Fuse {  //State fuse  private boolean state;  */\*\*  \* Constructor  \*/* public Fuse(){  this.state = false;  }  */\*\*  \* Method returns current state  \** ***@return*** *Returns current state of Fuse  \*/* public boolean isState() {  return state;  }  */\*\*  \* Method set state of fuze  \** ***@param*** *state Set state of fuze  \*/* public void setState(boolean state) {  this.state = state;  } } */\*\*  \* Class <code>Barrel</code> implements barrel  \** ***@author*** *Flyk Yevhenii  \** ***@version*** *1.0  \*/* class Barrel{  //length of barrel(mm)  int length;  //caliber of barrel  float caliber;   */\*\*  \* Constructor  \*/* public Barrel(){  this.length = 415;//standard in Ukraine  this.caliber = (float) 5.45;//standard in Ukraine  }  */\*\*  \* Constructor with parameters  \*/* public Barrel(int \_length , float \_caliber ){  this.length = \_length;  this.caliber = \_caliber;  }  */\*\*  \* Method length of barrel  \** ***@return*** *Returns length  \*/* public int getLength() {  return length;  }  */\*\*  \* Method set length of barrel  \** ***@param*** *length of barrel  \*/* public void setLength(int length) {  this.length = length;  }  */\*\*  \* Method caliber of barrel  \** ***@return*** *Returns caliber  \*/* public float getCaliber() {  return caliber;  }  */\*\*  \* Method set caliber of barrel  \** ***@param*** *caliber of barrel  \*/* public void setCaliber(float caliber) {  this.caliber = caliber;  } }  ***Лістинг RifleGun***  */\*\*  \* lab 4 package  \*/* package KI35.Flyk.Lab4;  import java.io.\*; import java.io.FileNotFoundException;  */\*\*  \* Interface which is implemented in the class  \*/* interface ScoreChoice{  void scoreChoice(ScoreSight \_scoreSight);  }  */\*\*  \*  \*/* interface ReductionScore{  void reductionScore (); } */\*\*  \*  \*  \*/  /\*\*  \* Class <code>RifleGun</code> implements rifle gun  \** ***@author*** *Flyk Yevhenii  \** ***@version*** *1.0  \*/* public class RifleGun extends AutomaticGun implements ScoreChoice , ReductionScore {  //object ScoreSight  private ScoreSight scoreSight;   */\*\*  \* Constructor  \*/* public RifleGun() {  super();  scoreSight=ScoreSight.*x1*;  }   */\*\*  \* Constructor  \** ***@param*** *\_length  \** ***@param*** *\_caliber  \** ***@param*** *\_scoreSight  \*/* public RifleGun(int \_length, float \_caliber, ScoreSight \_scoreSight) {  super(\_length, \_caliber);  scoreSight= \_scoreSight;  }   */\*\*  \* Method which return scoreSight  \** ***@return*** *scoreSight  \*/* public ScoreSight getScoreSight() {  return scoreSight;  }   */\*\*  \* Method which set scoreSight  \** ***@param*** *scoreSight  \*/* public void setScoreSight(ScoreSight scoreSight) {  this.scoreSight = scoreSight;  }   */\*\*  \* Method which implements an increase in sight mode  \*/* public void magnification() {  ScoreSight[] score\_arr = ScoreSight.*values*();  for (int i = 0; i < 4; ++i) {  if (scoreSight.ordinal() == 3) {  scoreSight = score\_arr[0];  } else if (scoreSight.ordinal() == i) {  scoreSight = score\_arr[i + 1];  }   }  System.*out*.println("Options of sight was increased");  }   */\*\*  \* Method which implements sight mode reduction  \*/* public void reductionScore () {  ScoreSight[] score\_arr = ScoreSight.*values*();  for (int i = 3; i >= 0; --i) {  if (scoreSight.ordinal() == i) {  scoreSight = score\_arr[i - 1];  } else if (scoreSight.ordinal() == 0) {  scoreSight = score\_arr[3];  }  }  System.*out*.println("Options of sight was reduced");  }   */\*\*  \* Method that implements the prototype of the method declared in the interface  \** ***@param*** *\_scoreSight  \*/* public void scoreChoice(ScoreSight \_scoreSight){  scoreSight=\_scoreSight;  }   }  ***Лістинг ScoreSight***  */\*\*  \* lab 4 package  \*/* package KI35.Flyk.Lab4;  */\*\*  \* Enum <code>ScoreSight</code> implements enum ScoreSight  \** ***@author*** *Flyk Yevhenii  \** ***@version*** *1.0  \*/* public enum ScoreSight {  *x1*,  *x2*,  *x4*,  *x8* }  ***Лістинг RifleGunApp***  */\*\*  \* lab 4 package  \*/* package KI35.Flyk.Lab4;  import java.io.FileNotFoundException;  */\*\*  \* Rifle Gun Application class implements main method for RifleGun  class possibilities demonstration  \** ***@author*** *Flyk Yevhenii  \** ***@version*** *1.0  \*/* public class RifleGunApp {  */\*\*  \** ***@param*** *args  \** ***@throws*** *FileNotFoundException  \*/* public static void main(String[] args){  RifleGun rifle = new RifleGun();  rifle.SwitchOffFuse();  rifle.BurstFiring();  rifle.ReloadAutomaticGun();  rifle.SwitchOnFuse();  rifle.getScoreSight();  rifle.magnification();   } } |

**Результат роботи:**

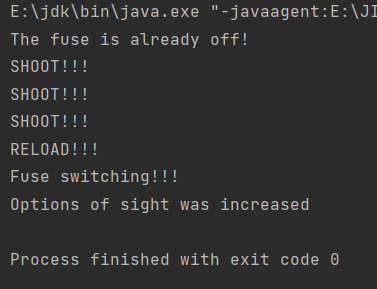
****

Рис.1 Результат роботи програми

**Фрагмент створеної документації:**

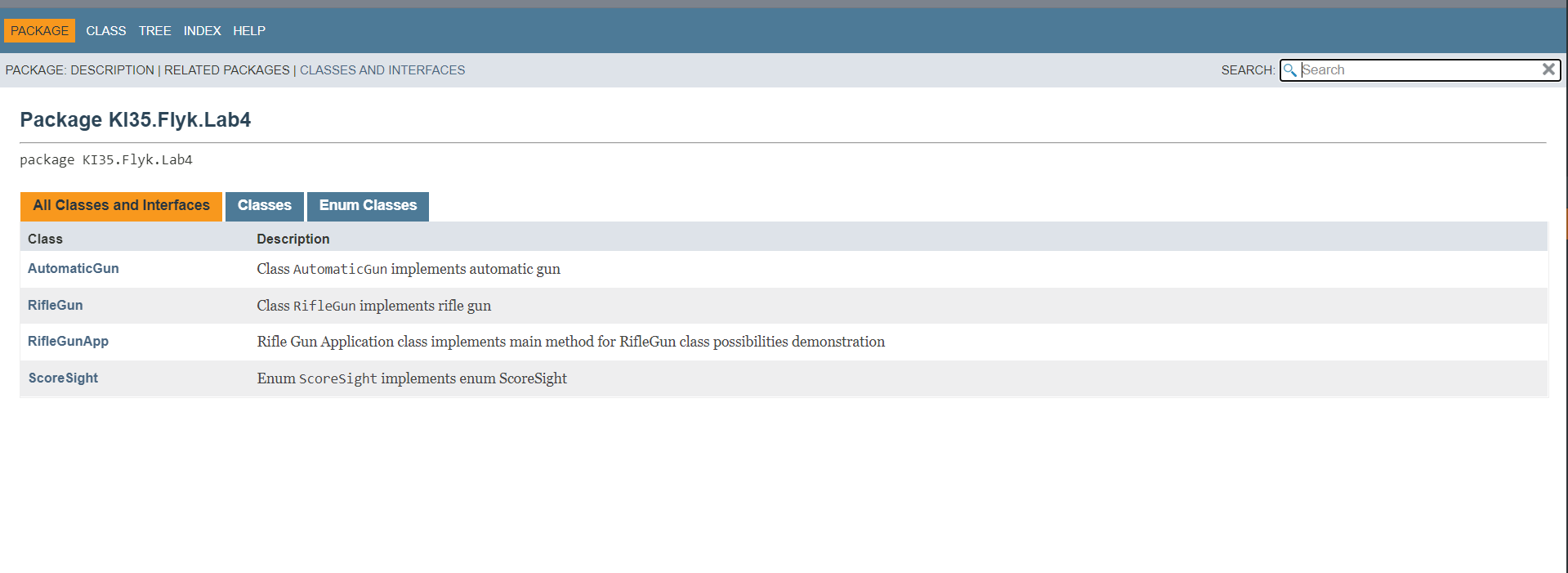
****

Рис.2.1 Фрагмент створеної документації

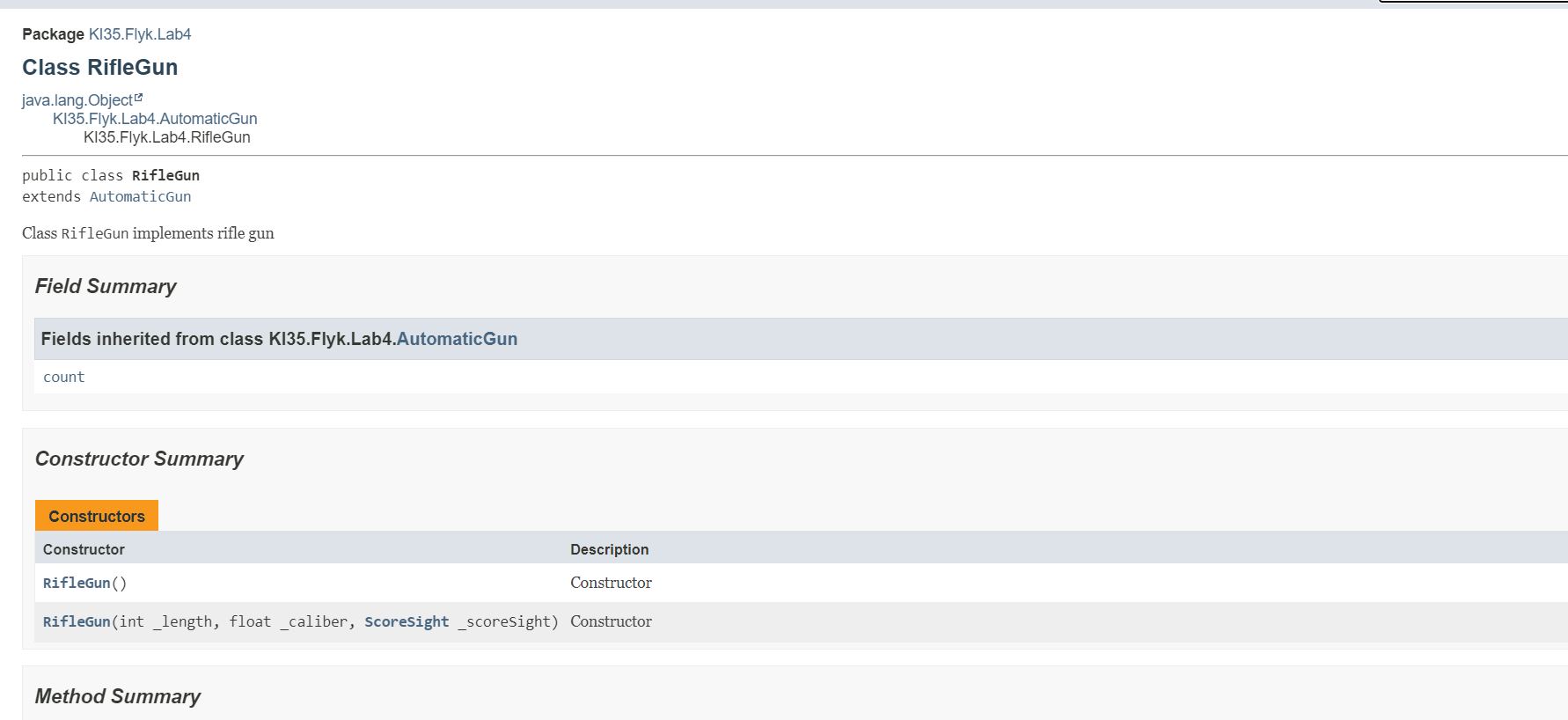
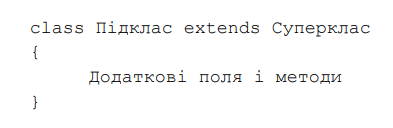


Рис.2.2 Фрагмент створеної документації класу RifleGun

**Відповіді на контрольні запитання:**

1. Синтаксис реалізації спадкування.

Спадкування в ООП призначене для розширення функціональності існуючих класів шляхом утворення нових класів на базі вже існуючих. У Java реалізована однокоренева архітектура класів згідно якої всі класи мають єдиного спільного предка (кореневий клас в ієрархії класів) – клас Object. Решта класів мови Java утворюються шляхом успадковування даного класу. Будь-яке спадкування у мові Java є відкритим, при цьому аналогів захищеному і приватному спадкуванню мови С++ не існує. На відміну від С++ у Java можливе спадкування лише одного базового класу (множинне спадкування відсутнє). Спадкування реалізується шляхом вказування ключового слова class піcля якого вказується назва підкласу, ключове слово extends та назва суперкласу, що розширюється у новому підкласі. Синтаксис реалізації спадкування:



1. Що таке суперклас та підклас?

В термінах мови Java базовий клас найчастіше називається суперкласом, а похідний клас – підкласом. Дана термінологія запозичена з теорії множин, де підмножина міститься у супермножині.

**Висновок:**

Під час виконання роботи я ознайомився з спадкуванням та інтерфейсами у мові Java.