Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних технологій, автоматики та метрології

Кафедра ЕОМ



**Звіт**

**До лабораторної роботи №4**

# З дисципліни: «Кросплатформні засоби програмування»

На тему «Спадкування та інтерфейси»

# Варіант №27

Виконав: ст. гр. КІ-36

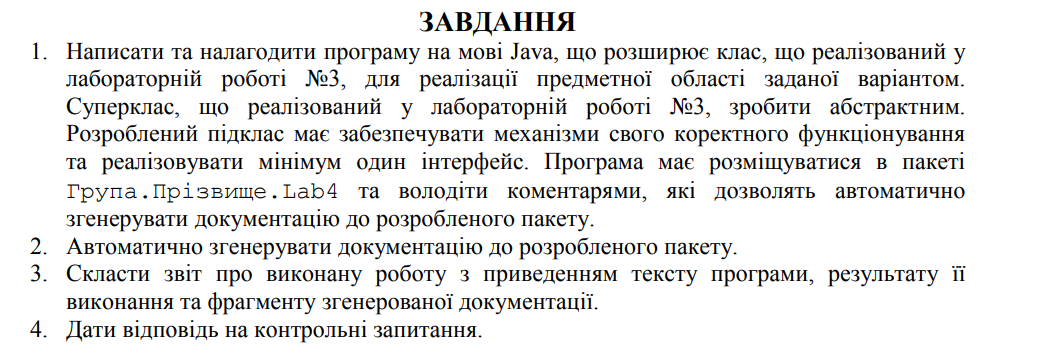
Галаджун Т. В.

Прийняв:

Іванов Ю. С.

Львів – 2022

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





**Виконання:**

**Лістинг програми:**

import java.util.ArrayList;  
  
public class Main  
{  
 public static void main(String[] args)  
 {  
 int size = 15;  
 ArrayList<IntellectualPatron> patrons = new ArrayList<>(size);  
 for (int i = 0; i < size - 1; i++)  
 {  
 patrons.add(new IntellectualPatron(new Projectile("steal", "traser")  
 , new Sleeve("conus", 0.5)  
 , new Capsule("capsule", 1.9)  
 , 7.65, 90));  
 }  
  
 for(var patron : patrons)  
 {  
 patron.ToShoot();  
 }  
 }  
}

*/\*\*  
 \* Class Sleeve  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public class Sleeve  
{  
 private String form;  
 private double weight;  
 private Logger logger = Logger.*getLogger*("logs.txt");  
  
 */\*\*  
 \* Sleeve constructor  
 \** ***@param*** *form  
 \** ***@param*** *weight  
 \*/* public Sleeve(String form, double weight)  
 {  
 logger.log(logger.infoFlag + " Sleeve constructor was called");  
 this.form = form;  
 this.weight = weight;  
 }  
  
 */\*\*  
 \* Getter for form  
 \** ***@return*** *form  
 \*/* public String getForm()  
 {  
 logger.log(logger.infoFlag + " Sleeve getForm was called");  
 return form;  
 }  
  
 */\*\*  
 \* Setter for form  
 \** ***@param*** *form  
 \*/* public void setForm(String form)  
 {  
 logger.log(logger.infoFlag + " Sleeve setForm was called");  
 this.form = form;  
 }  
  
 */\*\*  
 \* Getter for weight  
 \** ***@return*** *weight  
 \*/* public double getWeight()  
 {  
 logger.log(logger.infoFlag + " Sleeve getWeight was called");  
 return weight;  
 }  
  
 */\*\*  
 \* Setter for weight  
 \** ***@param*** *weight  
 \*/* public void setWeight(double weight)  
 {  
 logger.log(logger.infoFlag + " Sleeve setWeight was called");  
 this.weight = weight;  
 }  
  
 */\*\*  
 \* PrintInfo method  
 \*/* public void PrintInfo()  
 {  
 logger.log(logger.infoFlag + " Sleeve PrintInfo method was called");  
 System.*out*.println("Sleeve { form : " + "\"" + form + "\"; weight : " + "\"" + weight + "\"; }");  
 }  
}

import java.util.Random;  
  
*/\*\*  
 \* Class IntellectualPatron  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public class IntellectualPatron extends Patron implements PrintData  
{  
  
 */\*\*  
 \* Patron constructor  
 \*  
 \** ***@param*** *projectile  
 \** ***@param*** *sleeve  
 \** ***@param*** *capsule  
 \** ***@param*** *caliber  
 \** ***@param*** *probability  
 \*/* public IntellectualPatron(Projectile projectile, Sleeve sleeve, Capsule capsule, double caliber, double probability) {  
 super(projectile, sleeve, capsule, caliber, probability);  
 }  
  
 */\*\*  
 \* Overrided method ToShoot  
 \*/* @Override  
 public void ToShoot()  
 {  
 logger.log(logger.infoFlag + " Patron ToShoot method was called");  
 if (new Random().nextInt(100) < probability)  
 {  
 System.*out*.println("The patron hit the target");  
 }  
 else  
 {  
 System.*out*.println("The patron do not hit the target");  
 }  
 }  
  
 */\*\*  
 \* Overrided interface PrintInfo  
 \*/* @Override  
 public void PrintInfo()  
 {  
 logger.log(logger.infoFlag + " Patron printInfo method was called");  
 System.*out*.println("Patron { caliber : " + "\"" + caliber + "\"; probability : " + "\"" + probability + "\"; }");  
 projectile.PrintInfo();  
 sleeve.PrintInfo();  
 capsule.PrintInfo();  
 }  
}

import java.io.File;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
*/\*\*  
 \* Class Logger. Was created to log information, errors and warnings. Also there was implemented Singelton  
 \** ***@author*** *Serhii  
 \** ***@version*** *1.0  
 \*/*public class Logger  
{  
 private static Logger *logger*;  
 private final String fileName;  
  
 protected final String infoFlag = new String("[INFO] ");  
 protected final String errorFlag = new String("[ERROR] ");  
 protected final String warningFlag = new String("[WARNING] ");  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *fileName  
 \*/* private Logger(String fileName)  
 {  
 this.fileName = fileName;  
 File loggerFile = null;  
 FileWriter fout = null;  
 try  
 {  
 loggerFile = new File(fileName);  
 fout = new FileWriter(loggerFile, true);  
 SimpleDateFormat formatter= new SimpleDateFormat("yyyy-MM-dd 'at' HH:mm:ss z");  
 Date date = new Date(System.*currentTimeMillis*());  
 fout.write("[" + formatter.format(date) + "] " + "Logger start to work\n");  
 }  
 catch (IOException e)  
 {  
 System.*err*.println("Something wrong with log file" + e.getMessage());  
 System.*exit*(1);  
 }  
 finally  
 {  
 try  
 {  
 fout.flush();  
 fout.close();  
 }  
 catch (IOException e)  
 {  
 System.*out*.println(e.getMessage());  
 }  
 }  
 }  
  
 */\*\*  
 \* Method to do logging  
 \** ***@param*** *massege  
 \*/* public void log(String massege)  
 {  
 File loggerFile = null;  
 FileWriter fout = null;  
 try  
 {  
 loggerFile = new File(this.fileName);  
 fout = new FileWriter(loggerFile, true);  
 SimpleDateFormat formatter= new SimpleDateFormat("yyyy-MM-dd 'at' HH:mm:ss z");  
 Date date = new Date(System.*currentTimeMillis*());  
 fout.write("[" + formatter.format(date) + "] " + massege + "\n");  
 }  
 catch (IOException e)  
 {  
 System.*err*.println("Something wrong with log file" + e.getMessage());  
 System.*exit*(1);  
 }  
 finally  
 {  
 try  
 {  
 fout.flush();  
 fout.close();  
 }  
 catch (IOException | NullPointerException e)  
 {  
 System.*out*.println(e.getMessage());  
 }  
 }  
 }  
  
 */\*\*  
 \* Singleton implementation  
 \** ***@param*** *fileName  
 \** ***@return*** *\*/* public static Logger getLogger(String fileName)  
 {  
 if (*logger* == null)  
 {  
 *logger* = new Logger(fileName);  
 }  
 return *logger*;  
 }  
  
 */\*\*  
 \* Getter for logger  
 \** ***@return*** *logger  
 \*/* public static Logger getLogger()  
 {  
 return *logger*;  
 }  
  
}

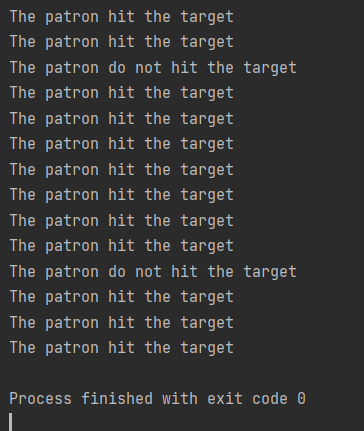
*/\*\*  
 \* Class Capsule  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public class Capsule  
{  
 private String type;  
 private double powder\_weight;  
 private Logger logger = Logger.*getLogger*("logs.txt");  
  
 */\*\*  
 \* Capsule constructor  
 \** ***@param*** *type  
 \** ***@param*** *powder\_weight  
 \*/* public Capsule(String type, double powder\_weight)  
 {  
 logger.log(logger.infoFlag + " Capsule constructor was called");  
 this.type = type;  
 this.powder\_weight = powder\_weight;  
 }  
  
 */\*\*  
 \* Getter for type  
 \** ***@return*** *type  
 \*/* public String getType()  
 {  
 logger.log(logger.infoFlag + " Capsule getType was called");  
 return type;  
 }  
  
 */\*\*  
 \* Setter for type  
 \** ***@param*** *type  
 \*/* public void setType(String type)  
 {  
 logger.log(logger.infoFlag + " Capsule setType was called");  
 this.type = type;  
 }  
  
 */\*\*  
 \* Getter for powder weight  
 \** ***@return*** *powder\_weight  
 \*/* public double getPowder\_weight()  
 {  
 logger.log(logger.infoFlag + " Capsule getPowder\_weight was called");  
 return powder\_weight;  
 }  
  
 */\*\*  
 \* Setter for powder weight  
 \** ***@param*** *powder\_weight  
 \*/* public void setPowder\_weight(double powder\_weight)  
 {  
 logger.log(logger.infoFlag + " Capsule setPowder\_weight was called");  
 this.powder\_weight = powder\_weight;  
 }  
  
 */\*\*  
 \* PrintInfo method  
 \*/* public void PrintInfo()  
 {  
 logger.log(logger.infoFlag + " Capsule PrintInfo method was called");  
 System.*out*.println("Projectile { powder weight : " + "\"" + powder\_weight + "\"; type : " + "\"" + type + "\"; }");  
 }  
}

public interface PrintData  
{  
 void PrintInfo();  
}

*/\*\*  
 \* Class Projectile  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public class Projectile  
{  
 private String material;  
 private String type;  
 private Logger logger = Logger.*getLogger*("logs.txt");  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *material  
 \** ***@param*** *type  
 \*/* public Projectile(String material, String type)  
 {  
 this.material = material;  
 this.type = type;  
 logger.log(logger.infoFlag + " Projectile constructor was called");  
 }  
  
 */\*\*  
 \* Getter for material  
 \** ***@return*** *material  
 \*/* public String getMaterial()  
 {  
 logger.log(logger.infoFlag + " Projectile getMaterial was called");  
 return material;  
 }  
  
 */\*\*  
 \* Setter for material  
 \** ***@param*** *material  
 \*/* public void setMaterial(String material)  
 {  
 logger.log(logger.infoFlag + " Projectile setMaterial was called");  
 this.material = material;  
 }  
  
 */\*\*  
 \* Getter for type  
 \** ***@return*** *type  
 \*/* public String getType()  
 {  
 logger.log(logger.infoFlag + " Projectile getType was called");  
 return type;  
 }  
  
 */\*\*  
 \* Setter for type  
 \** ***@param*** *type  
 \*/* public void setType(String type)  
 {  
 logger.log(logger.infoFlag + " Projectile setType was called");  
 this.type = type;  
 }  
  
 */\*\*  
 \* PrintInfo method  
 \*/* public void PrintInfo()  
 {  
 logger.log(logger.infoFlag + " Projectile PrintInfo method was called");  
 System.*out*.println("Projectile { material : " + "\"" + material + "\"; type : " + "\"" + type + "\"; }");  
 }  
}

import java.util.Random;  
  
*/\*\*  
 \* Class Patron  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public abstract class Patron  
{  
 protected Projectile projectile;  
 protected Sleeve sleeve;  
 protected Capsule capsule;  
 protected double caliber;  
 protected double probability;  
 protected Logger logger = Logger.*getLogger*("logs.txt");  
  
 */\*\*  
 \* Patron constructor  
 \** ***@param*** *projectile  
 \** ***@param*** *sleeve  
 \** ***@param*** *capsule  
 \** ***@param*** *caliber  
 \** ***@param*** *probability  
 \*/* public Patron(Projectile projectile, Sleeve sleeve, Capsule capsule, double caliber, double probability)  
 {  
 logger.log(logger.infoFlag + " Patron constructor was called");  
 this.projectile = projectile;  
 this.sleeve = sleeve;  
 this.capsule = capsule;  
 this.caliber = caliber;  
 this.probability = probability;  
 }  
  
 */\*\*  
 \* Getter for probability  
 \** ***@return*** *probability  
 \*/* public double getProbability()  
 {  
 logger.log(logger.infoFlag + " Patron getProbability was called");  
 return probability;  
 }  
  
 */\*\*  
 \* Setter for probability  
 \** ***@param*** *probability  
 \*/* public void setProbability(double probability)  
 {  
 logger.log(logger.infoFlag + " Patron setProbability");  
 this.probability = probability;  
 }  
  
 */\*\*  
 \* Getter for projectile  
 \** ***@return*** *projectile  
 \*/* public Projectile getProjectile()  
 {  
 logger.log(logger.infoFlag + " Patron getProjectile was called");  
 return projectile;  
 }  
  
 */\*\*  
 \* Setter for projectile  
 \** ***@param*** *projectile  
 \*/* public void setProjectile(Projectile projectile)  
 {  
 logger.log(logger.infoFlag + " Patron setProjectile was called");  
 this.projectile = projectile;  
 }  
  
 */\*\*  
 \* Getter for sleeve  
 \** ***@return*** *sleeve  
 \*/* public Sleeve getSleeve()  
 {  
 logger.log(logger.infoFlag + " Patron getSleeve was called");  
 return sleeve;  
 }  
  
 */\*\*  
 \* Setter for sleeve  
 \** ***@param*** *sleeve  
 \*/* public void setSleeve(Sleeve sleeve)  
 {  
 logger.log(logger.infoFlag + " Patron setSleeve was called");  
 this.sleeve = sleeve;  
 }  
  
 */\*\*  
 \* Getter for capsule  
 \** ***@return*** *capsule  
 \*/* public Capsule getCapsule()  
 {  
 logger.log(logger.infoFlag + " Patron getCapsule was called");  
 return capsule;  
 }  
  
 */\*\*  
 \* Setter for capsule  
 \** ***@param*** *capsule  
 \*/* public void setCapsule(Capsule capsule)  
 {  
 logger.log(logger.infoFlag + " Patron setCapsule was called");  
 this.capsule = capsule;  
 }  
  
 */\*\*  
 \* Getter for caliber was called  
 \** ***@return*** *caliber  
 \*/* public double getCaliber()  
 {  
 logger.log(logger.infoFlag + " Patron getCaliber was called");  
 return caliber;  
 }  
  
 */\*\*  
 \* Setter for caliber  
 \** ***@param*** *caliber  
 \*/* public void setCaliber(double caliber)  
 {  
 logger.log(logger.infoFlag + " Patron setCaliber was called");  
 this.caliber = caliber;  
 }  
  
 */\*\*  
 \* ToShoot method  
 \*/* public abstract void ToShoot();  
}

**Результати:**

****

**Висновок:** ознайомився з спадкуванням та інтерфейсами у мові Java.