Міністерство освіти і науки України

Національний університет “Львівська політехніка”



**Звіт**

**до лабораторної роботи № 3**

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

**на тему: «Спадкування та інтерфейси»**

Варіант № 20

Підготував:

студент групи КІ-307

Скрипник Т.О.

Перевірив:

Іванов Ю.С.

Львів 2023

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

**Завдання:**

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. Автоматично згенерувати документацію до розробленого пакету. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації. Дати відповідь на контрольні запитання.

**Лістинг ShoesApp.java:**

package KI307.Skrypnyk.lab3;

import static java.lang.System.out;

import java.io.\*;

import KI307.Skrypnyk.lab3.Shoes.Tyer;

/\*\*

\* Shoes Application class implements main method for Shoes

class possibilities demonstration

\* @author Pashko Andrii

\* @version 1.0

\*/

public class ShoesApp

{

/\*\*

\* @param args

\* @throws FileNotFoundException

\*/

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

{

Tyer.States dir = null;

Boots shoes = new Boots();

shoes.TieShoes();

out.print(shoes.getLeftShoeTieCount() + "\n");

out.print(shoes.getRightShoeTieCount() + "\n");

shoes.setShoeSize(38, 40);

out.print(shoes.getLeftShoeSize() + "\n");

out.print(shoes.getRightShoeSize() + "\n");

shoes.move(150.23);

out.print(shoes.durability() + "\n");

dir = shoes.getTieState();

if (dir == Tyer.States.Tied)

{

out.print("Tied\n");

}else if (dir == Tyer.States.Untied)

{

out.print("Untied\n");

}else if (dir == Tyer.States.Without)

{

out.print("No laces\n");

}

shoes.dispose();

}

}

**Лістинг Shoes.java:**

package KI307.Skrypnyk.lab3;

import java.io.\*;

/\*\*

\* Class <code>Shoes</code> implements shoes

\* **@version** 1.0

\*/

abstract public class Shoes {

private Tyer TieDevice;

private ShoeSize size;

private Shoe RShoe;

private Shoe LShoe;

private PrintWriter fout;

/\*\*

\* Constructor

\* **@throws** FileNotFoundException

\*/

public Shoes() throws FileNotFoundException

{

TieDevice = new Tyer();

size = new ShoeSize();

RShoe = new Shoe();

LShoe = new Shoe();

fout = new PrintWriter (new File("Log.txt"));

}

public Shoes(int TieCountR,int TieCountL) throws FileNotFoundException

{

TieDevice = new Tyer();

size = new ShoeSize();

RShoe = new Shoe(TieCountR);

LShoe = new Shoe(TieCountL);

fout = new PrintWriter (new File("Log.txt"));

}

public void increseShoeSize(int rInc, int lInc)

{

size.setrSize(size.getrSize() + rInc);

size.setlSize(size.getlSize() + lInc);

fout.print("New Right Shoe Size: " + size.getrSize() + "\n");

fout.print("New Left Shoe Size: " + size.getlSize() + "\n");

}

public void setShoeSize(int r, int l)

{

size.setrSize(r);

size.setlSize(l);

fout.print("New Right Shoe Size: " + size.getrSize() + "\n");

fout.print("New Left Shoe Size: " + size.getlSize() + "\n");

}

public int getRightShoeSize()

{

return size.getrSize();

}

public int getLeftShoeSize()

{

return size.getlSize();

}

public void TieRightShoe()

{

RShoe.TieShoe();

fout.print("New ties count for Right Shoe: " + RShoe.getTieCount() + "\n");

}

public void TieLefttShoe()

{

LShoe.TieShoe();

fout.print("New ties count for Left Shoe: " + LShoe.getTieCount() + "\n");

}

public int getLeftShoeTieCount()

{

return LShoe.getTieCount();

}

public int getRightShoeTieCount()

{

return RShoe.getTieCount();

}

public void TieShoes()

{

TieDevice.SetStateTied();

RShoe.TieShoe();

LShoe.TieShoe();

fout.print("Shoes were Tied\n");

}

public void UntieShoes()

{

TieDevice.SetStateUntied();

fout.print("Shoes were Untied\n");

}

public void RemoveLaces()

{

TieDevice.SetStateWithout();

fout.print("Laces were removed from shoes\n");

}

public Tyer.*States* getTieState()

{

return TieDevice.getState();

}

/\*\*

\* Method releases used recourses

\*/

public void dispose()

{

fout.close();

}

class Tyer

{

enum *States* {***Tied***, ***Untied***, ***Without***};

private *States* state;

/\*\*

\* Constructor

\*/

public Tyer()

{

state = *States*.***Untied***;

}

/\*\*

\* Method sets state to Tied

\*/

public void SetStateTied()

{

state = *States*.***Tied***;

}

/\*\*

\* Method sets state to Untied

\*/

public void SetStateUntied()

{

state = *States*.***Untied***;

}

/\*\*

\* Method sets state to Without

\*/

public void SetStateWithout()

{

state = *States*.***Without***;

}

/\*\*

\* Method resets state to Untied

\*/

public void resetState()

{

SetStateUntied();

}

/\*\*

\* Method returns stae

\* **@return** State of <code>Tyer.States</code> type

\*/

public *States* getState()

{

return state;

}

}

class ShoeSize

{

private int rSize, lSize;

/\*\*

\* Constructor

\*/

public ShoeSize()

{

rSize = 38;

lSize = 38;

}

/\*\*

\* Constructor

\* **@param** <code>R</code> The rSize value

\* **@param** <code>L</code> The lSize value

\*/

public ShoeSize(int R, int L)

{

rSize = R;

lSize = L;

}

/\*\*

\* Method returns the rSize value

\* **@return** The rSize value

\*/

public int getrSize()

{

return rSize;

}

/\*\*

\* Method returns the lSize value

\* **@return** The lSize value

\*/

public int getlSize()

{

return lSize;

}

/\*\*

\* Method returns sizes in the <code>obj</code>,

that is passed into method through method parameter

\* **@param** <code>obj</code> The object, where sizes are set

\*/

public void getPosition(ShoeSize obj)

{

obj.rSize = rSize;

obj.lSize = lSize;

}

/\*\*

\* Method sets the rSize value

\* **@param** <code>R</code> The rSize value

\*/

public void setrSize(int R)

{

rSize = R;

}

/\*\*

\* Method sets the lSize value

\* **@param** <code>L</code> The lSize value

\*/

public void setlSize(int L)

{

lSize = L;

}

}

class Shoe

{

private int TieCount;

/\*\*

\* Constructor

\*/

public Shoe()

{

TieCount = 0;

}

/\*\*

\* Constructor

\* **@param** <code>count</code> Shoe ties count

\*/

public Shoe(int count)

{

TieCount = count;

}

/\*\*

\* Method simulates Shoe tying

\*/

public void TieShoe()

{

TieCount++;

}

/\*\*

\* Method returns Tie count

\* **@return** TieCount

\*/

public int getTieCount()

{

return TieCount;

}

}

}

interface Move

{

void move (double x);

}

interface Usage extends Move

{

double durability();

}

class Boots extends Shoes implements Usage

{

private double distance;

public Boots() throws FileNotFoundException {

super();

distance = 0;

}

public void move (double x)

{

distance = distance + x;

}

public double durability()

{

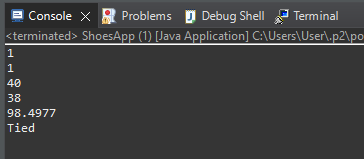
return 100 - distance \* 0.01;

}

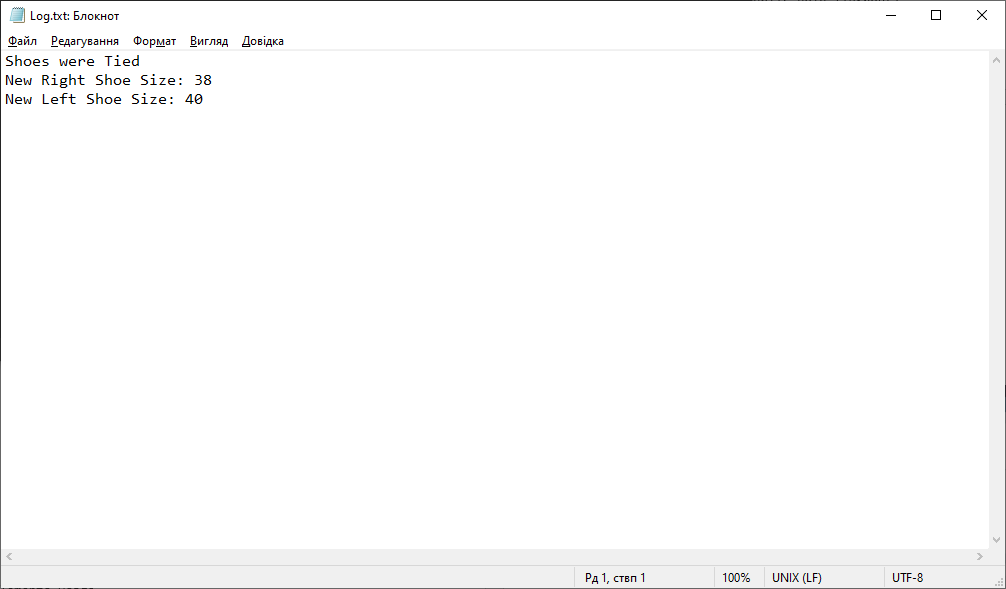
}

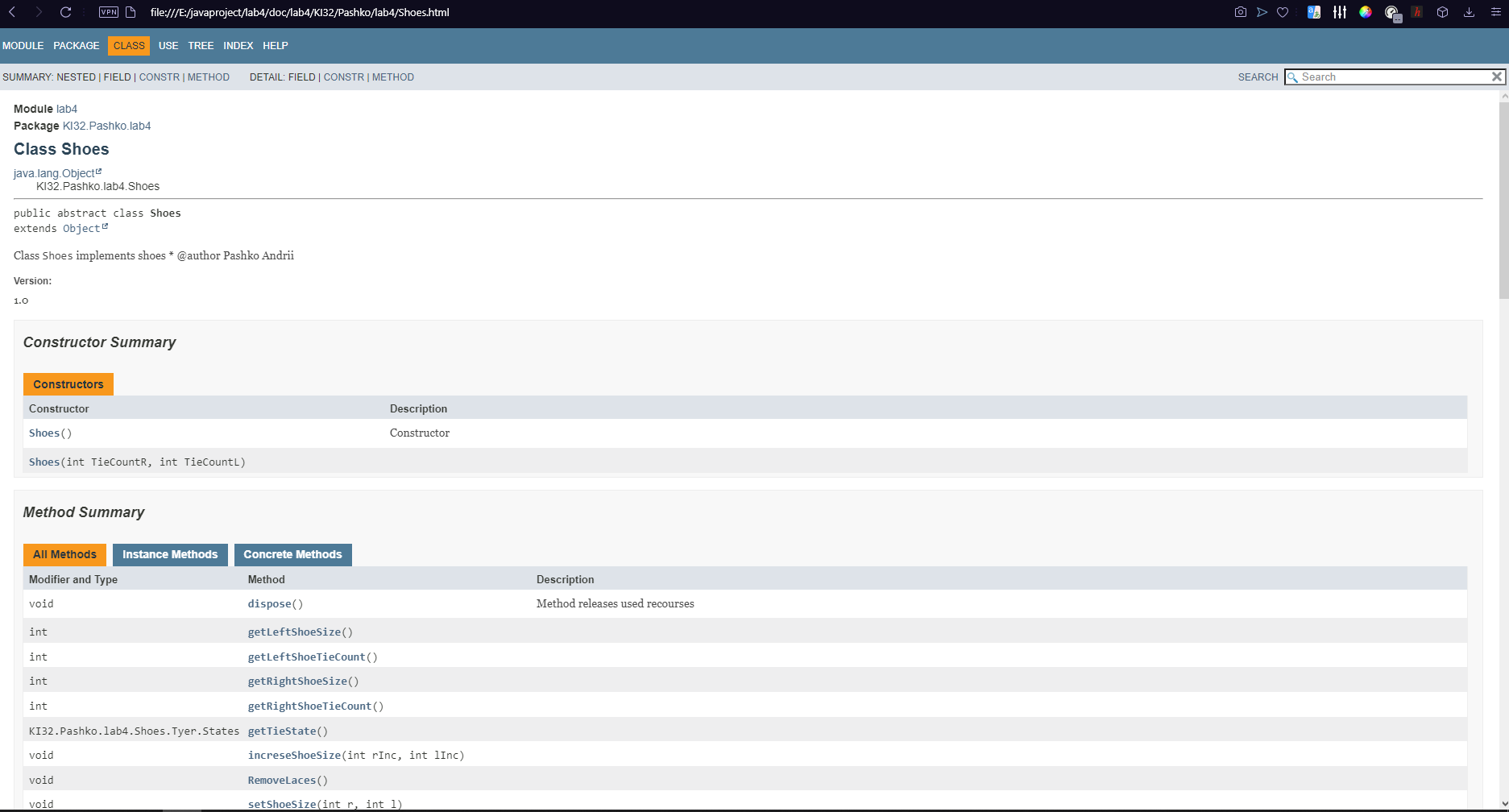
}

}

} **Результати виконання програми:** 

**Файл Log.txt**





Сформував документацію за допомогою javadoc

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