Міністерство освіти і науки України Національний

«Львівська політехніка»

Кафедра ЕОМ



Звіт

до лабораторної роботи № 3

з дисципліни: «Кросплатформні засоби програмування» «Спадкування та інтерфейси» Варіант - 1

Виконав:

Студент групи КІ-305

Вознюк О. М.

Прийняв:

Іванов Ю. С.

Львів 2023

**Мета:** 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 №2, для реалізації предметної області заданої варіантом. Суперклас, що реалізований у лабораторній роботі №2, 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.

Варіант завдання:



Код програми:

**File HumanApp.java**

package KI305.Vozniuk.Lab3;  
import java.io.FileNotFoundException;  
*/\*\*  
 \* Class App  
 \* @version 1.0  
 \*/*public class HumanApp {  
 */\*\*  
 \* @param args  
 \*/* public static void main(String[] args) throws FileNotFoundException  
 {  
 Sportsman person = new Sportsman("Сristiano", 25, 185,7 , "healthy","lab4.txt");  
 System.*out*.println(person.getName() + " have: " + person.getAge() + " years old");  
 person.ShowIndex();  
 person.running(100);  
 person.dispose();  
 }  
}

**File Sportsman.java**

package KI305.Vozniuk.Lab3;  
import java.io.FileNotFoundException;  
  
  
interface ISportsman{  
 void running(int metres);  
}  
  
*/\*\*  
 \* Class Sportsman  
 \* @version 1.0  
 \*/*public class Sportsman extends Human implements ISportsman{  
 */\*\*  
 \* Constructor  
 \* @param name person name  
 \* @param height person height  
 \* @param weight person weight  
 \* @param HealthResults person weight  
 \* @param outPutFile output file  
 \* @throws FileNotFoundException  
 \*/* public Sportsman(String name , int age, double height, double weight, String HealthResults,String outPutFile) throws FileNotFoundException{  
 super(name, age, height, weight, HealthResults,outPutFile);  
 }  
 */\*\*  
 \* Constructor to calculating how many matres Name have to run in seconds  
 \*/* public void running(int metres)  
 {  
 System.*out*.println("Olympic running standarts for sportmen to run: " + metres + " metres");  
 int sec = metres/7;  
 System.*out*.println(super.getName() + " have to finish running in " + sec + " seconds");  
 }  
}

**File Person.java**

*/\*\*  
 \* lab 2 package  
 \*/*package KI305.Vozniuk.Lab3;  
import java.io.\*;  
import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
*/\*\*  
 \* Class Human  
 \* @version 1.0  
 \*/*public class Human {  
 private String name;  
 private AthleteDetails athletics;  
 private HealthRecords health;  
 private FinancialDetails finances;  
 private PrintWriter fout;  
  
 */\*\*  
 \* Constructor  
 \*  
 \* @param name person name  
 \* @param height person height  
 \* @param weight person weight  
 \* @param weight person HealthResults  
 \* @param outPutFile output file  
 \* @throws FileNotFoundException  
 \*/* public Human(String name , int age, double height, double weight, String HealthResults, String outPutFile) throws FileNotFoundException{  
 this.name = name;  
 athletics = new AthleteDetails(age, height, weight);  
 health = new HealthRecords(HealthResults);  
 finances = new FinancialDetails();  
 fout = new PrintWriter(new File(outPutFile));  
 }  
 */\*\*  
 \* this.name = name; athletics = new AthleteDetails(age, height, weight);  
 \* health = new HealthRecords(HealthResults); finances = new FinancialDetails();  
 \* fout = new PrintWriter(new File(outPutFile));  
 \* <p>  
 \* Method to return name  
 \*/* public String getName(){  
 return name;  
 }  
 */\*\*  
 \* Method returns age  
 \*/* public int getAge(){  
 return athletics.getAge();  
 }  
 */\*\*  
 \* Method returns height  
 \*/* public double getHeight(){  
 return athletics.getHeight();  
 }  
 */\*\*  
 \* Method returns weight  
 \*/* public double getWeight(){  
 return athletics.getWeight();  
 }  
 */\*\*  
 \* Method for calculation weight index  
 \*/* public void ShowIndex(){  
 athletics.idx();  
 }  
 */\*\*  
 \* Method for calculation is person need to get vaccinated  
 \*/* public void health(){  
 if (health.getHealthResults().equals("healthy")) {  
 System.*out*.println("You don't need get vaccinated");  
 fout.println("You don't need get vaccinated");  
 fout.flush();  
 }else if(health.getHealthResults().equals("unhealthy")){  
 System.*out*.println("You should get vaccinated");  
 fout.println("You should get vaccinated");  
 fout.flush();  
 }else{  
 System.*out*.println("You entered incorrect values");  
 fout.println("You entered incorrect values");  
 fout.flush();  
 }  
 }  
  
  
 */\*\*  
 \* Method returns random salary from 4000 to 5000  
 \*/* public int getSalary(){  
 return finances.CalculateSalary();  
 }  
 */\*\*  
 \* Method returns salary with 19.5% taxes  
 \*/* public int getSalaryWithTaxes(){  
 return finances.getSalaryWithTaxes();  
 }  
 */\*\*  
 \* Method returns BankInfo  
 \*/* public String getBankInfo(){  
 return finances.getBankInfo();  
 }  
 */\*\*  
 \* Method exits  
 \*/* public void dispose(){  
 fout.flush();  
 fout.close();  
 }  
}

**AthleteDetails.java**

package KI305.Vozniuk.Lab3;  
  
public class AthleteDetails {  
 private int age;  
 private double height;  
 private double weight;  
 */\*\*  
 \* Constructor  
 \* @param age person name  
 \* @param height person height  
 \* @param weight person weight  
 \*/* public AthleteDetails(int age, double height, double weight)  
 {  
 this.age = age;  
 this.height = height;  
 this.weight = weight;  
 }  
 */\*\*  
 \* Method returns age  
 \*/* public int getAge(){  
 return age;  
 }  
 */\*\*  
 \* Method returns the height  
 \*/* public double getHeight() {  
 return height;  
 }  
 */\*\*  
 \* Method returns the weight  
 \*/* public double getWeight() {  
 return weight;  
 }  
 */\*\*  
 \* Method calculations weight index  
 \*/* public void idx(){  
 double y = height - weight + 10;  
 if(y <= weight + 10){  
 System.*out*.println("Your weight index is not normal");  
 }else {  
 System.*out*.println("Your weight index is normal");  
 }  
 }  
}

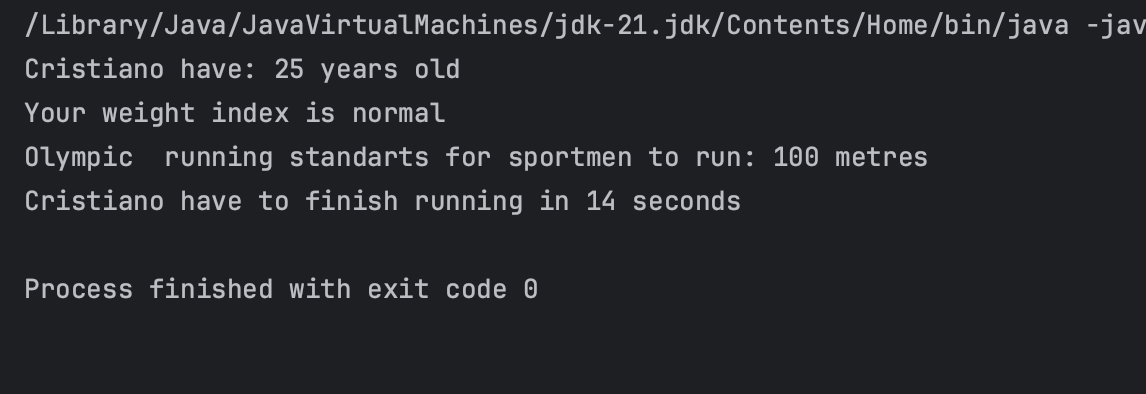
**File HealthRecords.java**

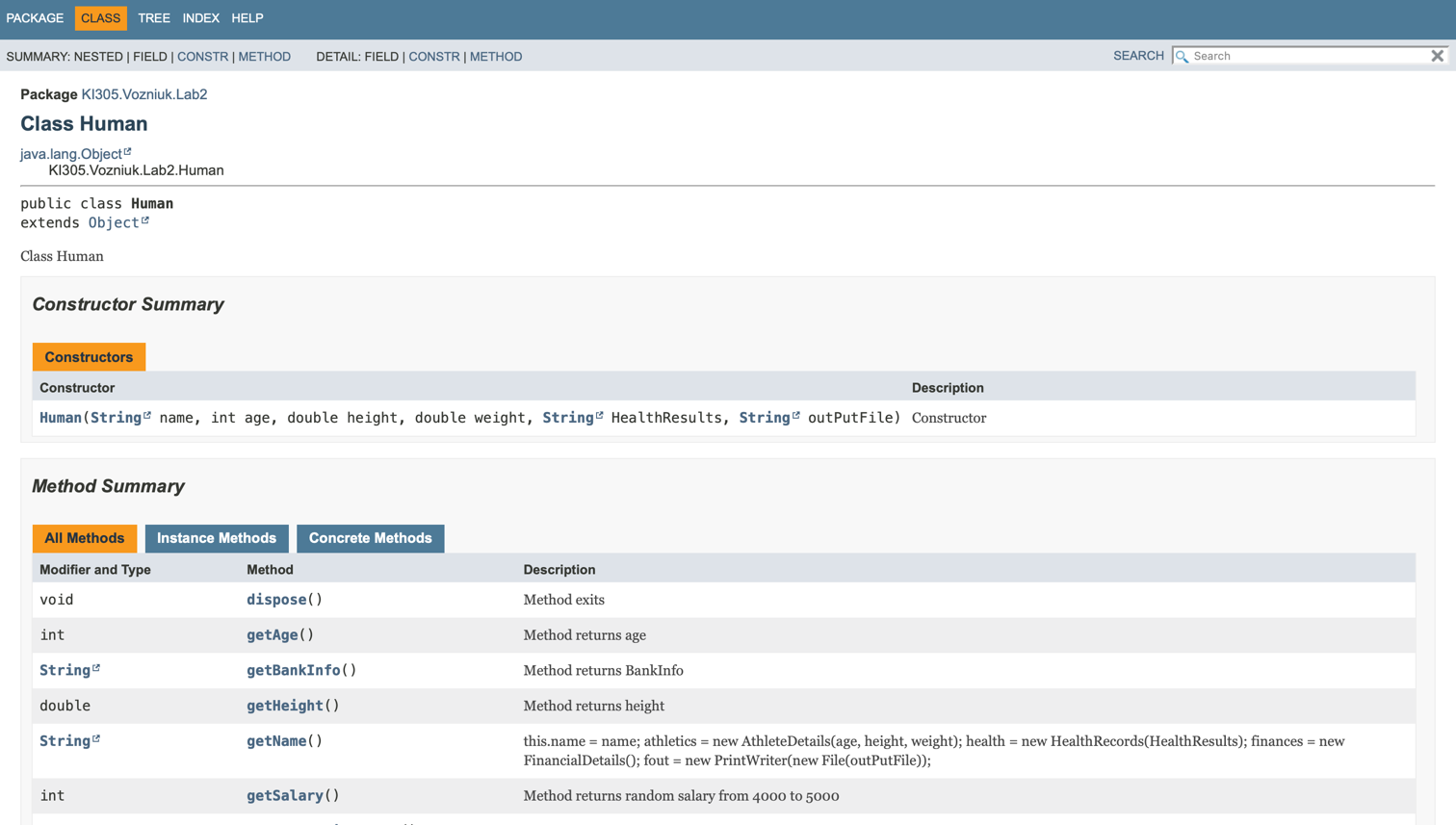
package KI305.Vozniuk.Lab3;  
  
public class HealthRecords {  
 private String testResults;  
 */\*\*  
 \* Constructor  
 \* @param testResults person testResults  
 \*/* public HealthRecords(String testResults) {  
 this.testResults = testResults;  
 }  
 */\*\*  
 \* Method returns HealthResults  
 \*/* public String getHealthResults() {  
 return testResults;  
 }  
}

**File HealthRecords.java**

package KI305.Vozniuk.Lab3;  
  
public class FinancialDetails{  
 private int salary;  
 private String bankInfo;  
 */\*\*  
 \* Constructor without parametres  
 \*/* public FinancialDetails() {  
 salary = 0;  
 bankInfo = "4444 5555 6666 7777";  
 }  
 */\*\*  
 \* Method calculate and return salary  
 \*/* public int CalculateSalary(){  
 salary = (int)(Math.*random*()\*1000 + 4000);  
 return salary;  
 }  
 */\*\*  
 \* Method returns salary with taxes  
 \*/* public int getSalaryWithTaxes(){  
 return (int) (0.805 \* salary);  
 }  
 */\*\*  
 \* Method returns bankInfo  
 \*/* public String getBankInfo(){  
 return bankInfo;  
 }  
}

Результат виконання програми:



****

Відповіді на контрольні запитання

1. Синтаксис реалізації спадкування.

- class МійКлас implements Інтерфейс {

// тіло класу }

2. Що таке суперклас та підклас?

- суперклас - це клас, від якого інший клас успадковує властивості та методи.

Підклас - це клас, який успадковує властивості та методи від суперкласу.

3. Як звернутися до членів суперкласу з підкласу?

- super.назваМетоду([параметри]); // виклик методу суперкласу

super.назваПоля; // звернення до поля суперкласу

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

- Статичне зв’язування використовується, коли метод є приватним, статичним,

фінальним або конструктором. В таких випадках вибір методу відбувається на

етапі компіляції.

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

- вибір методу для виклику відбувається під час виконання програми на основі

фактичного типу об’єкта.

6. Що таке абстрактний клас та як його реалізувати?

- це клас, який має один або більше абстрактних методів (методів без

реалізації).

Щоб створити абстрактний клас, використовується ключове слово abstract.

Приклад:

abstract class АбстрактнийКлас {

abstract void абстрактнийМетод(); }

7. Для чого використовується ключове слово instanceof?

- для перевірки, чи об'єкт належить до певного класу або інтерфейсу.

Синтаксис:

if (об'єкт instanceof Клас) {

// код, який виконується, якщо об'єкт належить до класу }

8. Як перевірити чи клас є підкласом іншого класу?

- В Java використовується ключове слово extends, щоб вказати, що клас є

підкласом іншого класу. Перевірити, чи один клас є підкласом іншого класу

можна шляхом аналізу ієрархії успадкування.

9. Що таке інтерфейс?

- це абстрактний тип даних, який визначає набір методів, але не надає їх

реалізацію. Всі методи інтерфейсу є загальнодоступними та автоматично є

public. Інтерфейси використовуються для створення контрактів, які класи

повинні реалізувати.

10. Як оголосити та застосувати інтерфейс?

• - Для оголошення інтерфейсу використовується ключове слово interface.

Синтаксис:

interface Інтерфейс {

// оголошення методів та констант }

• - Для застосування інтерфейсу в класі використовується ключове слово

implements.

Синтаксис:

class МійКлас implements Інтерфейс {

// реалізація методів інтерфейсу }

Висновок: У ході виконання даної лабораторної роботи, я отримала навички роботи з концепціями спадкування та інтерфейсами в мові програмування Java. Ознайомившись з цими важливими аспектами об'єктно-орієнтованого програмування, я зрозуміла їх роль у створенні більш структурованих і гнучких програм.