**Міністерство освіти і науки України**

**Національний університет «Львівська політехніка»**

**Кафедра ЕОМ**



Звіт

до лабораторної роботи № 4

з дисципліни: «Кросплатформні засоби програмування»

«СПАДКУВАННЯ ТА ІНТЕРФЕЙСИ»

Варіант №16

Виконав:

ст.гр. КІ-34

Палій Н.С­­.

Прийняв:

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 : 16

**Диктофон**

**Код програми :**

**DictaphoneApp.java**

*/\*\*  
 \* lab 4 package  
 \*/*package KI\_34.Palii.lab4;  
import java.io.\*;  
*/\*\*  
 \* Computer Mouse Application class implements main method for ComputerMouse  
 class possibilities demonstration  
 \** ***@author*** *EOM Stuff  
 \** ***@version*** *1.0  
 \*/*public class DictaphoneApp {  
 */\*\*  
 \** ***@param*** *args  
 \** ***@throws*** *FileNotFoundException  
 \*/* public static void main(String[] args) throws FileNotFoundException {  
  
 Dictaphone micro = new Dictaphone();  
  
  
 micro.Discharge\_player(15);  
 micro.ShowAllINFO();  
 System.*out*.println("\n\n");  
  
 micro.Charge(10);  
 micro.ShowAllINFO();  
  
 micro.show\_memory();  
  
 //interface realisation  
 System.*out*.println("Power of dictophone: " + micro.power());  
 System.*out*.println("Color of dictophone: " + micro.*color* );  
  
 }  
}

**Dictaphone.java**

package KI\_34.Palii.lab4;  
  
import java.io.File;  
import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
  
public class Dictaphone extends Player implements Interface{  
  
 */\*\*  
 \* Constructor  
 \** ***@throws*** *FileNotFoundException  
 \*/* public Dictaphone() throws FileNotFoundException {  
 Battery battery;  
 Memory memory ;  
 Stop\_Play stopPlay;  
 Volume volume;  
 }  
 */\*\*  
 \* Method show memory  
 \*/* public void show\_memory(){  
 System.*out*.println("Hours left to record: " + (memory.ret\_memory()/60));  
 }  
  
 */\*\*  
 \* Interface method  
 \*/* @Override  
 public float power() {  
 return (memory.ret\_memory()\* battery.ret\_battery());  
 }  
}

**Player.java**

*/\*\*  
 \* lab 3 package  
 \*/*package KI\_34.Palii.lab4;  
import java.io.File;  
import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
  
  
*/\*\*  
 \* Class <code>Player</code> implements audio player  
 \** ***@author*** *Nazarii Palii  
 \** ***@version*** *1.0  
 \*/*public abstract class Player {  
  
 public Battery battery;  
 public Memory memory ;  
 public Stop\_Play stopPlay;  
 public Volume volume;  
 public final PrintWriter fout;  
  
  
 */\*\*  
 \* Constructor  
 \** ***@throws*** *FileNotFoundException  
 \*/* public Player()throws FileNotFoundException {  
 battery = new Battery();  
 memory = new Memory();  
 stopPlay = new Stop\_Play();  
 volume = new Volume();  
  
 fout = new PrintWriter(new File("Log.txt"));  
 }  
  
  
  
 */\*\*  
 \* Constructor  
 \** ***@throws*** *FileNotFoundException  
 \*/* public Player(int percentage,int memory\_has, int songs\_dwnl, int vol)throws FileNotFoundException {  
 battery = new Battery(percentage);  
 memory = new Memory(memory\_has,songs\_dwnl);  
 stopPlay = new Stop\_Play();  
 volume = new Volume(vol);  
  
 fout = new PrintWriter(new File("Log.txt"));  
 }  
  
  
  
 */\*\*  
 \* Method show info about player  
 \*/* void ShowAllINFO(){  
 battery.Show\_percentage();  
 System.*out*.print("Memory: " + memory.ret\_memory() + "\n");  
 System.*out*.print("Status of playing: " + stopPlay.status + "\n");  
 System.*out*.print("Volume level: " + volume.volume + "\n");  
  
 fout.print("info showed\n\n");  
  
 }  
  
  
 */\*\*  
 \* Method show memory  
 \*/* public void show\_memory(){  
 System.*out*.print("\nMemory: " + memory.ret\_memory() + "\nStored memory: " + memory.ret\_stored\_memory() + "\n");  
 }  
  
  
  
  
 */\*\*  
 \* Method simulates battery save mode  
 \*/* void Energy\_save\_mode(){  
 if(battery.*percentage*<25)  
 {  
 battery.save\_mode = true;  
 System.*out*.print("Battery in save mode\n");  
 volume.SetVolume(20);  
 }else{  
 battery.save\_mode = false;  
 System.*out*.print("Battery is well charged\n");  
 }  
 fout.println("battery save mode is called\n");  
 }  
  
  
 */\*\*  
 \* Method simulates playing music  
 \*/* void PlayMusic(){  
 stopPlay.Play();  
 fout.println("music playing\n");  
 }  
  
  
 */\*\*  
 \* Method show status of playing  
 \*/* void Status\_of\_play(){  
 if(stopPlay.status){  
 System.*out*.print("Playing\n");  
 }else{  
 System.*out*.print("Stopped\n");  
 }  
 fout.println("status of music showed\n");  
 }  
  
  
 */\*\*  
 \* Method simulates charging player  
 \*/* void Charge(int perc){  
 battery.Charge(perc);  
 fout.println("battery was charged\n");  
 }  
  
 */\*\*  
 \* Method simulates discharging player  
 \*/* void Discharge\_player(int perc){  
 Battery.*DecreasePercentage*(perc);  
 fout.println("battery was discharged\n");  
 }  
  
  
 */\*\*  
 \* Method simulates loading songs  
 \*/* private void load\_song(int amout)  
 {  
 memory.DownloadSongs(amout);  
 }  
  
 */\*\*  
 \* Method releases used recourses  
 \*/* public void CloseFile (){  
 fout.close();  
 }  
  
  
  
  
  
  
  
  
  
  
  
  
}

**Interface.java**

package KI\_34.Palii.lab4;  
  
public interface Interface {  
 String *color* = "Blue";  
 float power();  
}

**Other classes**

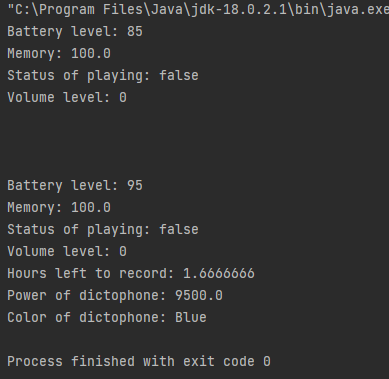
package KI\_34.Palii.lab4;  
  
public class Memory {  
 public int memory;  
 public int stored\_memory;  
 public int memory\_left;  
 public int songs\_downloaded;  
 public int songs\_to\_download\_left;  
  
 public Memory(){  
 memory = 100;  
 stored\_memory = 20;  
 memory\_left = memory - stored\_memory;  
 songs\_downloaded = stored\_memory/5;  
 songs\_to\_download\_left = memory\_left/5;  
 }  
  
 public Memory(int MegaBytes\_has, int Songs\_downloaded){  
 memory = MegaBytes\_has;  
 songs\_downloaded = Songs\_downloaded;  
 stored\_memory = songs\_downloaded\*5;  
 memory\_left = memory - stored\_memory;  
 songs\_to\_download\_left = memory\_left/5;  
 }  
  
 void DownloadSongs(int Number\_of\_songs){  
 songs\_downloaded = songs\_downloaded + Number\_of\_songs;  
 if(songs\_downloaded > songs\_to\_download\_left){  
 System.*out*.print("Memory is full\n Downloaded only " + (Number\_of\_songs-songs\_to\_download\_left) + "songs\n");  
 }  
 }  
  
  
 float ret\_memory(){  
 return memory;  
 }  
  
 float ret\_stored\_memory(){  
 return stored\_memory;  
 }  
  
 void ShowSongs(){  
 System.*out*.print("\nSongs: " + songs\_downloaded + "\nSongs left to download: " + songs\_to\_download\_left + "\n");  
 }  
  
}

package KI\_34.Palii.lab4;  
  
public class Battery {  
 public static int *percentage*;  
 public int left\_to\_play;  
 public boolean save\_mode;  
  
 public Battery(){  
 *percentage* = 100;  
 left\_to\_play = 15\**percentage*;  
 save\_mode = false;  
 }  
  
 public Battery (int percentage\_of\_battery){  
 *percentage* = percentage\_of\_battery;  
 left\_to\_play = 15\**percentage*;  
 save\_mode = false;  
 }  
  
 float ret\_battery(){  
 return (*percentage*);  
 }  
  
 void Show\_percentage (){  
 System.*out*.print("Battery level: " + *percentage*);  
 }  
  
 void Charge(int perc){  
 *percentage* = *percentage* + perc;  
 if(*percentage*>100){  
 *percentage* = 100;  
 }  
 }  
  
 static void DecreasePercentage(int decrease\_perc)  
 {  
 *percentage* = *percentage* - decrease\_perc;  
 }  
}

package KI\_34.Palii.lab4;  
  
public class Stop\_Play {  
 public boolean status;  
  
 public Stop\_Play(){  
 status = false;  
 }  
  
 void Play(){  
 if(Battery.*percentage*>6){  
 status = true;  
 Battery.*DecreasePercentage*(5);  
 System.*out*.print("...playing\n");  
 }else {  
 System.*out*.print("\nCharge your battery");  
 }  
 }  
  
 void Stop(){  
  
 status = false;  
 }  
  
}

package KI\_34.Palii.lab4;  
  
public class Volume {  
 public int volume;  
  
 public Volume(){  
 volume = 0;  
 }  
  
 public Volume(int vol){  
 volume = vol;  
 }  
  
 void SetVolume(int volumeRate){  
 volume = volumeRate;  
 if(volumeRate>100)  
 {  
 volume = 100;  
 }  
  
 if(volume < 20){  
 Battery.*DecreasePercentage*(1);  
 } else if (volume>20 && volume<50) {  
 Battery.*DecreasePercentage*(3);  
 } else if (volume>50 && volume<80) {  
 Battery.*DecreasePercentage*(4);  
 } else if (volume>80) {  
 Battery.*DecreasePercentage*(5);  
 }  
 }  
}

**Вивід в консоль:**

****

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