Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних технологій, автоматики та метрології

Кафедра ЕОМ



**Звіт**

**До лабораторної роботи №3**

# З дисципліни: «Кросплатформні засоби програмування»

На тему «Спадкування та інтерфейси»

# Варіант №17

Виконав: ст. гр. КІ-305

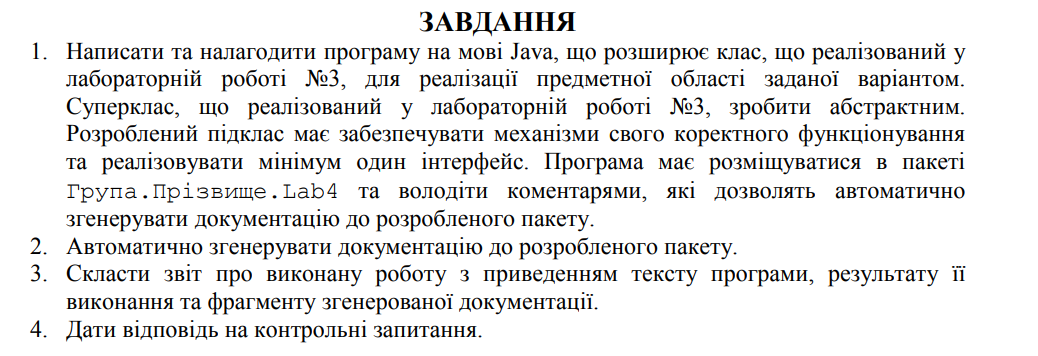
Легкобит Н.В

Прийняв:

Олексів М.В

Львів – 2023

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





**Виконання:**

**Лістинг програми:**

*/\*\*  
 \* Class Button  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public class Button  
{  
 private String action;  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *action  
 \*/* public Button(String action) {  
 this.action = action;  
 }  
  
 */\*\*  
 \* Getter for action  
 \** ***@return*** *action  
 \*/* public String getAction() {  
 return action;  
 }  
  
 */\*\*  
 \* Setter for action  
 \** ***@param*** *action  
 \*/* public void setAction(String action) {  
 this.action = action;  
 }  
  
 @Override  
 public String toString() {  
 return "Button{ " +  
 "action = '" + action + '\'' +  
 '}';  
 }  
}

*/\*\*  
 \* Class Hard Disk  
 \** ***@author*** *\** ***@version*** *\*/*public class HardDisk  
{  
 private double capacity;  
 private String producer;  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *capacity  
 \** ***@param*** *producer  
 \*/* public HardDisk(double capacity, String producer)  
 {  
 this.capacity = capacity;  
 this.producer = producer;  
 }  
  
 */\*\*  
 \* Getter for capacity  
 \** ***@return*** *capacity  
 \*/* public double getCapacity() {  
 return capacity;  
 }  
  
 */\*\*  
 \* Setter for capacity  
 \** ***@param*** *capacity  
 \*/* public void setCapacity(double capacity) {  
 this.capacity = capacity;  
 }  
  
 */\*\*  
 \* Getter for producer  
 \** ***@return*** *producer  
 \*/* public String getProducer() {  
 return producer;  
 }  
  
 */\*\*  
 \* Setter for producer  
 \** ***@param*** *producer  
 \*/* public void setProducer(String producer) {  
 this.producer = producer;  
 }  
  
 @Override  
 public String toString() {  
 return "HardDisk{ " +  
 "capacity = " + capacity + " mb." +  
 ", producer = '" + producer + '\'' +  
 '}';  
 }  
}

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*** *\** ***@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*;  
 }  
  
}

public class Main {  
 public static void main(String[] args) {  
 VideoMagnitophone videoMagnitophone = new VideoMagnitophone(new Screen(7.8, "720x1980"), new HardDisk(1000, "Harman"));  
  
 videoMagnitophone.AddVideo("My birthday video");  
 videoMagnitophone.AddVideo("First time in France");  
 videoMagnitophone.AddVideo("Lviv");  
 videoMagnitophone.AddVideo("Party");  
 videoMagnitophone.AddVideo("Karate competition");  
  
 videoMagnitophone.TurnOnNextVideo();  
 videoMagnitophone.TurnOnNextVideo();  
 videoMagnitophone.TurnOnNextVideo();  
  
 videoMagnitophone.TurnOnPrevVideo();  
 videoMagnitophone.TurnOnPrevVideo();  
 videoMagnitophone.TurnOnPrevVideo();  
 videoMagnitophone.TurnOnPrevVideo();  
  
 System.*out*.println(videoMagnitophone);  
 }  
}

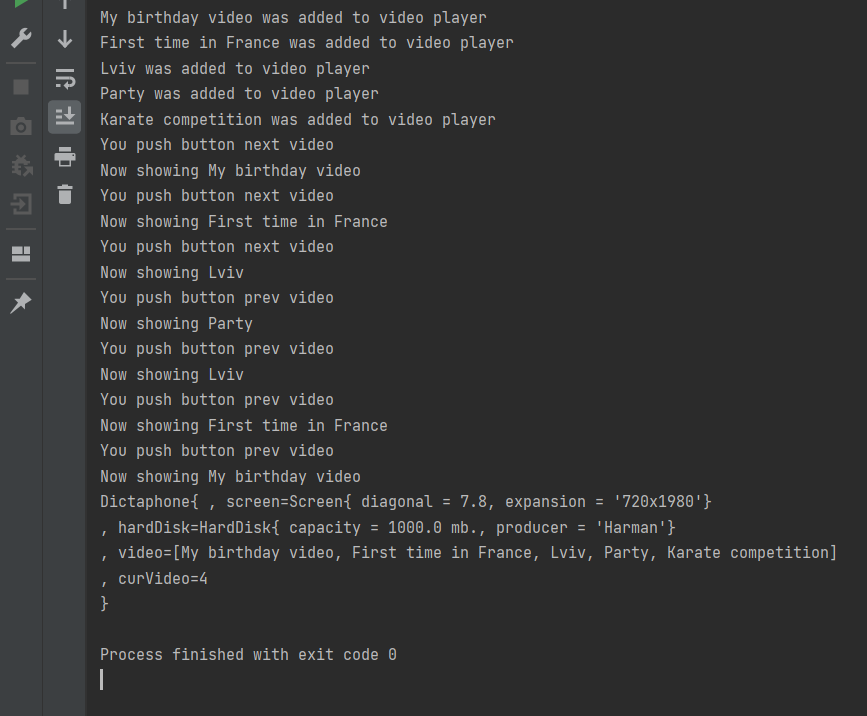
*/\*\*  
 \* Class Screen  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public class Screen  
{  
 private double diagonal;  
 private String expansion;  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *diagonal  
 \** ***@param*** *expansion  
 \*/* public Screen(double diagonal, String expansion) {  
 this.diagonal = diagonal;  
 this.expansion = expansion;  
 }  
  
 */\*\*  
 \* Getter for Diagonal  
 \** ***@return*** *diagonal  
 \*/* public double getDiagonal() {  
 return diagonal;  
 }  
  
 */\*\*  
 \* Setter for diagonal  
 \** ***@param*** *diagonal  
 \*/* public void setDiagonal(double diagonal) {  
 this.diagonal = diagonal;  
 }  
  
 */\*\*  
 \* Getter for expansion  
 \** ***@return*** *\*/* public String getExpansion() {  
 return expansion;  
 }  
  
 */\*\*  
 \* Setter for expansion  
 \** ***@param*** *expansion  
 \*/* public void setExpansion(String expansion) {  
 this.expansion = expansion;  
 }  
  
 @Override  
 public String toString() {  
 return "Screen{ " +  
 "diagonal = " + diagonal +  
 ", expansion = '" + expansion + '\'' +  
 '}';  
 }  
}

*/\*\*  
 \* Class VideoMagnitophone  
 \** ***@version*** *1.0  
 \*/*public class VideoMagnitophone extends VideoPlayer implements VideoRecord  
{  
 */\*\*  
 \* Constructor  
 \*  
 \** ***@param*** *screen  
 \** ***@param*** *hardDisk  
 \*/* public VideoMagnitophone(Screen screen, HardDisk hardDisk) {  
 super(screen, hardDisk);  
 }  
  
 */\*\*  
 \* Overrided method to turn on next video  
 \*/* @Override  
 public void TurnOnNextVideo() {  
 logger.log(logger.infoFlag + "TurnOnNextVideo VideoPlayer method was called");  
 if(curVideo == videos.size() - 1)  
 {  
 System.*out*.println("You push button " + nextVideo.getAction());  
 System.*out*.println("Now showing " + videos.get(curVideo));  
 curVideo = 0;  
 } else if (curVideo < videos.size() - 1) {  
 System.*out*.println("You push button " + nextVideo.getAction());  
 System.*out*.println("Now showing " + videos.get(curVideo));  
 curVideo++;  
 }  
 }  
 */\*\*  
 \* Overrided method to turn on prev video  
 \*/* @Override  
 public void TurnOnPrevVideo() {  
 logger.log(logger.infoFlag + "TurnPrevNextVideo VideoPlayer method was called");  
 if(curVideo == 0)  
 {  
 System.*out*.println("You push button " + prevVideo.getAction());  
 System.*out*.println("Now showing " + videos.get(curVideo));  
 curVideo = videos.size() - 1;  
 } else if (curVideo > 0) {  
 System.*out*.println("You push button " + prevVideo.getAction());  
 System.*out*.println("Now showing " + videos.get(curVideo));  
 curVideo--;  
 }  
 }  
  
 @Override  
 public void RecordVideo(String voice) {  
 videos.add("Voice Record - " + voice);  
 }  
  
 @Override  
 public String toString() {  
 return "Dictaphone{ " +  
 ", screen=" + screen + "\n" +  
 ", hardDisk=" + hardDisk + "\n" +  
 ", video=" + videos + "\n" +  
 ", curVideo=" + curVideo + "\n" +  
 '}';  
 }  
}

import java.util.ArrayList;  
  
*/\*\*  
 \* Class VideoPlayer  
 \** ***@author*** *\** ***@version*** *1.0  
 \*/*public abstract class VideoPlayer  
{  
 protected final Button nextVideo = new Button("next video");  
 protected final Button prevVideo = new Button("prev video");  
 protected final Button pause = new Button("pause");  
 protected Logger logger = Logger.*getLogger*("logs.txt");  
 protected Screen screen;  
 protected HardDisk hardDisk;  
 protected ArrayList<String> videos = new ArrayList<>();  
 protected int curVideo = 0;  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *screen  
 \** ***@param*** *hardDisk  
 \*/* public VideoPlayer(Screen screen, HardDisk hardDisk) {  
 logger.log(logger.infoFlag + "VideoPlayer constructor called");  
 this.screen = screen;  
 this.hardDisk = hardDisk;  
 }  
  
 */\*\*  
 \* Method to add new video to player  
 \** ***@param*** *video  
 \*/* public void AddVideo(String video)  
 {  
 videos.add(video);  
 System.*out*.println(video + " was added to video player");  
 logger.log(logger.infoFlag + "VideoPlayer AddVideo method was called");  
 }  
  
  
 */\*\*  
 \* Method to turn on next video  
 \*/* public abstract void TurnOnNextVideo();  
  
 */\*\*  
 \* Method to turn on prev video  
 \*/* public abstract void TurnOnPrevVideo();  
  
  
 public Button getNextVideo() {  
 return nextVideo;  
 }  
  
 public Button getPrevVideo() {  
 return prevVideo;  
 }  
  
 public Button getPause() {  
 return pause;  
 }  
  
 public Logger getLogger() {  
 return logger;  
 }  
  
 public void setLogger(Logger logger) {  
 this.logger = logger;  
 }  
  
 public Screen getScreen() {  
 return screen;  
 }  
  
 public void setScreen(Screen screen) {  
 this.screen = screen;  
 }  
  
 public HardDisk getHardDisk() {  
 return hardDisk;  
 }  
  
 public void setHardDisk(HardDisk hardDisk) {  
 this.hardDisk = hardDisk;  
 }  
  
 public ArrayList<String> getVideos() {  
 return videos;  
 }  
  
 public void setVideos(ArrayList<String> songs) {  
 this.videos = songs;  
 }  
  
 public int getCurVideo() {  
 return curVideo;  
 }  
  
 public void setCurVideo(int curVideo) {  
 this.curVideo = curVideo;  
 }  
  
 @Override  
 public String toString() {  
 return "AudioPlayer{ " +  
 " screen=" + screen + "\n" +  
 ", hardDisk=" + hardDisk + "\n" +  
 ", videos=" + videos + "\n" +  
 ", curVideo=" + curVideo + "\n" +  
 '}';  
 }  
}

public interface VideoRecord  
{  
 void RecordVideo(String video);  
}

**Результати:**

****

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