Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ

Звіт

до лабораторної роботи № 3

з дисципліни: «Кросплатформні засоби програмування»

на тему: «Спадкування та інтерфейси»

Варіант №13

Виконав:
ст. гр. КІ-203
Панченко Д. В.

Прийняв:
доцент кафедри ЕОМ
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: 13 – Мобільний телефон.

Вихідний код програми:

Файл PhoneMusic.java:

```
/**
 * lab 3 package
 */
package KI_306.Panchenko.Lab3;
import java.io.*;

/**
 * interface for playing music
 */
interface MusicPlayer {
    void playMusic(String song);
}

/**
 * Class <code>PhoneMusic</code> A subclass that implements an interface
 */
public class PhoneMusic extends Phone implements MusicPlayer{

      private String NameList;
      private String MusicList;

      /**
      * Constructor
      * @throws FileNotFoundException is exception
      */
      public PhoneMusic() throws FileNotFoundException
    {
      NameList = "My list";
```

```
        MusicList = "";

    }

        /**
        * Constructor
        * @param NameList The name of music list
        * @throws FileNotFoundException is exception
        */
        public PhoneMusic(String NameList) throws FileNotFoundException
        {
        this.NameList = NameList;
        MusicList = "";

        }

        /**
         * Implementation of the interface method for playing music
         * @param song The song they want to play
         */
        @Override
    public void playMusic(String song) {
             System.out.println(" \n");
             fout.print(" \n");
         if (super.isOn) {
             if (MusicList.contains(song)) {
                   System.out.println("Відтворення пісні: " + song + " \n");
                   fout.print("Відтворення пісні: " + song + " \n");
             }
             else {
                   System.out.println("Пісні під назвою: " + song + " не знайдено \n");
                   fout.print("Пісні під назвою: " + song + " не знайдено \n");
             }
         } else {
            System.out.println("Телефон вимкнено, щоб відтворити музику ввімкніть його
\n");
            fout.print("Телефон вимкнено, щоб відтворити музику ввімкніть його \n");
         }
    }

        /**
     * Method for downloading music
     * @param song The song they want to downloading
     */
        public void downloading(String song) {
              System.out.println(" \n");
              fout.print(" \n");
              if (super.isOn) {
                   if(MusicList == "") {
                         MusicList += song;
                   }
                   else {
                         MusicList += ", " + song;
                   }
                   System.out.println("Пісню " + song + " завантажено \n");
            fout.print("Пісню " + song + " завантажено \n");

             } else {
            System.out.println("Телефон вимкнено, щоб завантажити музику ввімкніть його
\n");
            fout.print("Телефон вимкнено, щоб завантажити музику ввімкніть його \n");
         }
        }
```

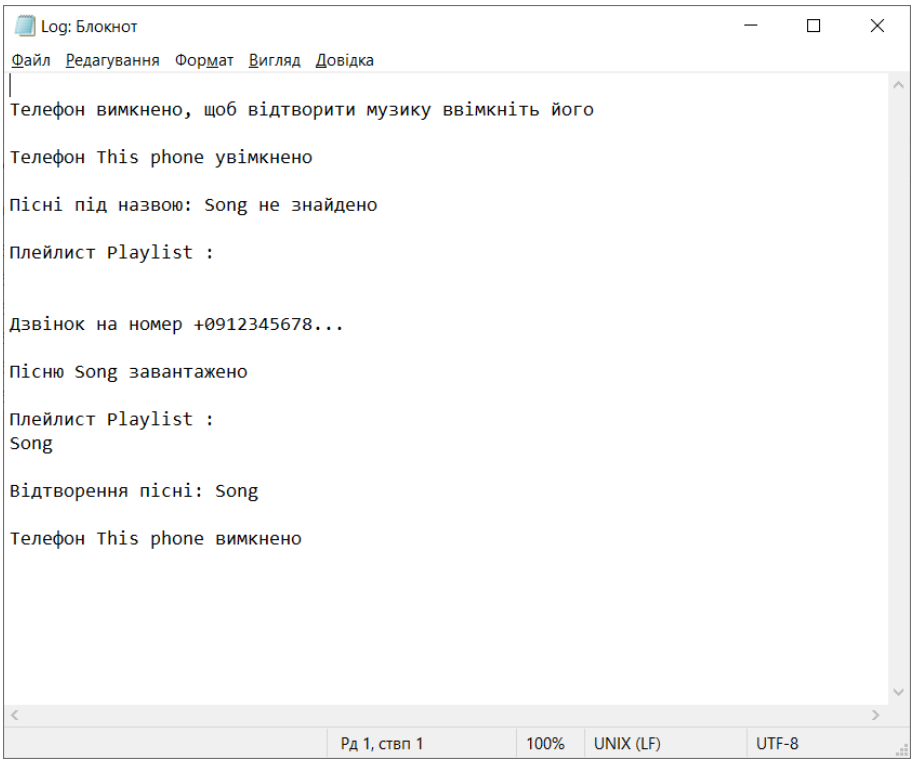
```
	/**
	* Method for list display
	*/
	public void listDisplay() {
		System.out.println(" \n");
		fout.print(" \n");
		if (super.isOn) {
			System.out.println("Плейлист " + NameList + " : \n");
		fout.print("Плейлист " + NameList + " : \n");
			System.out.println(MusicList + " \n");
		fout.print(MusicList + " \n");

		} else {
		System.out.println("Телефон вимкнено, щоб подивитись список пісень ввімкніть його \n");
		fout.print("Телефон вимкнено, щоб подивитись список пісень ввімкніть його \n");
	}
	}

}
```

Файл PhoneMusicApp.java:

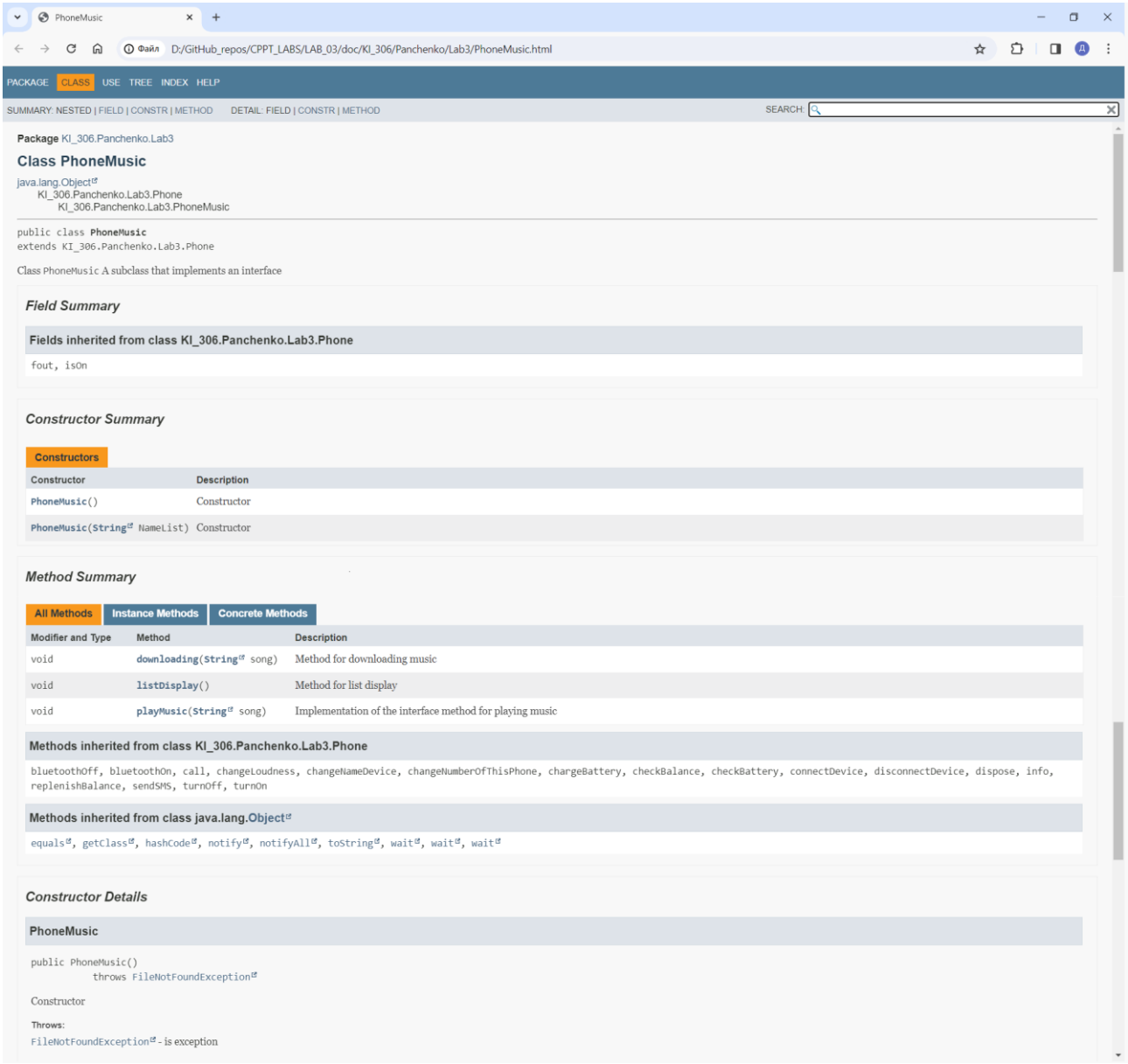
```
/**
 * lab 3 package
 */
package KI_306.Panchenko.Lab3;

import java.io.FileNotFoundException;

/**
 * Phone Music Application class implements main method for PhoneMusic class abilities demonstration
 */
public class PhoneMusicApp {

	/**
	 * @param args is arguments
	 * @throws FileNotFoundException is exception
	 */
	public static void main(String[] args) throws FileNotFoundException {

		PhoneMusic myPhoneMusic = new PhoneMusic("Playlist");

		myPhoneMusic.playMusic("Song");

		myPhoneMusic.turnOn();

		myPhoneMusic.playMusic("Song");

		myPhoneMusic.listDisplay();

		myPhoneMusic.call("+0912345678");

		myPhoneMusic.downloading("Song");

		myPhoneMusic.listDisplay();

		myPhoneMusic.playMusic("Song");

		myPhoneMusic.turnOff();
```

```
            myPhoneMusic.dispose();

        }

}
```

Результат виконання програми:



Фрагмент згенерованої документації:



PhoneMusic

Файл D:/GitHub_repos/CPPT_LABS/LAB_03/doc/KI_306/Panchenko/Lab3/PhoneMusic.html

PACKAGE CLASS USE TREE INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SEARCH:

Package KI_306.Panchenko.Lab3

Class PhoneMusic

java.lang.Object
KI_306.Panchenko.Lab3.Phone
KI_306.Panchenko.Lab3.PhoneMusic

```
public class PhoneMusic
extends KI_306.Panchenko.Lab3.Phone
```

Class PhoneMusic A subclass that implements an interface

Field Summary

Fields inherited from class KI_306.Panchenko.Lab3.Phone

fout, isOn

Constructor Summary

Constructors

Constructor	Description
PhoneMusic()	Constructor
PhoneMusic(String NameList)	Constructor

Method Summary

All Methods **Instance Methods** **Concrete Methods**

Modifier and Type	Method	Description
void	downloading(String song)	Method for downloading music
void	listDisplay()	Method for list display
void	playMusic(String song)	Implementation of the interface method for playing music

Methods inherited from class KI_306.Panchenko.Lab3.Phone

bluetoothOff, bluetoothOn, call, changeLoudness, changeNameDevice, changeNumberOfThisPhone, chargeBattery, checkBalance, checkBattery, connectDevice, disconnectDevice, dispose, info, replenishBalance, sendSMS, turnOff, turnOn

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

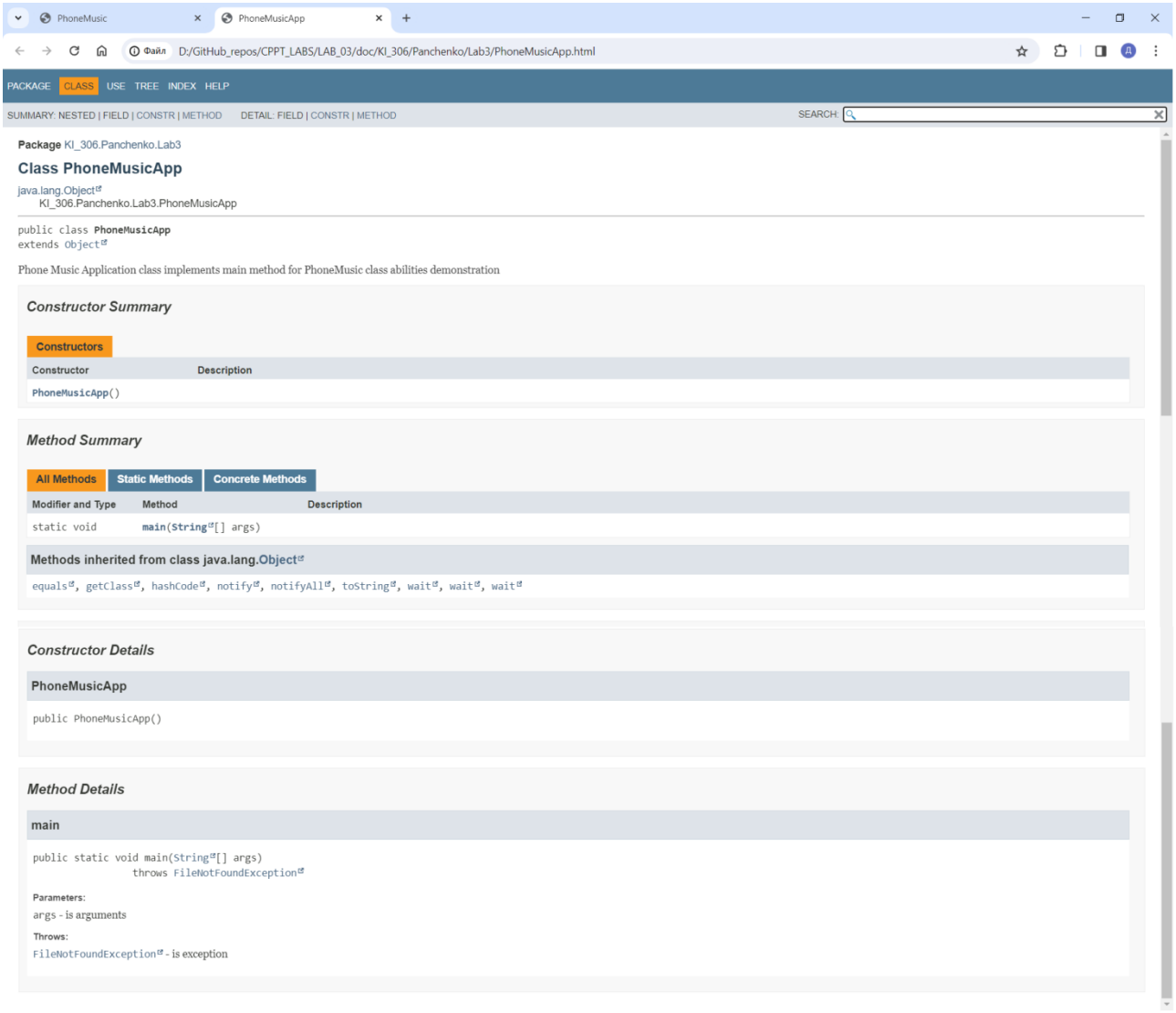
PhoneMusic

```
public PhoneMusic()
            throws FileNotFoundException
```

Constructor

Throws:

FileNotFoundException - is exception



Відповіді на контрольні запитання:

1. Синтаксис реалізації спадкування.

Синтаксис реалізації спадкування:

class Підклас extends Суперклас
{
Додаткові поля і методи
}

2. Що таке суперклас та підклас?

В термінах мови Java базовий клас найчастіше називається суперкласом, а похідний клас – підкласом. Дана термінологія запозичена з теорії множин, де підмножина міститься у супермножині.

3. Як звернутися до членів суперкласу з підкласу?

Синтаксис звертання до елементів суперкласу:

super.назваМетоду([параметри]); // виклик методу суперкласу
super.назваПоля // звертання до поля суперкласу

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

Якщо метод є приватним, статичним, фінальним або конструктором, то для нього застосовується механізм статичного зв'язування.

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

Механізм динамічного (пізнього) зв'язування полягає у тому, що вибір методу, який необхідно викликати, відбувається не на етапі компіляції, а під час виконання програми.

6. Що таке абстрактний клас та як його реалізувати?

Абстрактні класи призначені бути основою для розробки ієрархій класів та не дозволяють створювати об'єкти свого класу. Вони реалізуються за допомогою ключового слова abstract.

7. Для чого використовується ключове слово instanceof?

Наявність бодай одної виключної ситуації призводить до аварійного завершення програми. Щоб уникнути цього слід перед приведенням типів використати оператор instanceof, який повертає true, якщо посилання посилається на об'єкт фактичний тип якого є не вищим в ієрархії типів, ніж вказаний у операторі instanceof, і false у протилежному випадку. Синтаксис оператора instanceof:

посилання instanceof Ім'яТипу

8. Як перевірити чи клас є підкласом іншого класу?

В Java можна використовувати ключове слово instanceof для перевірки того, чи об'єкт належить до певного класу або його підкласу. Відповідний вираз має вигляд:

```
if (об'єкт instanceof Клас) {
 // код, який виконується, якщо об'єкт є екземпляром Класу або його підкласу
} else {
   // код, який виконується, якщо об'єкт не є екземпляром Класу або його підкласу
}
```

9. Що таке інтерфейс?

Інтерфейси вказують що повинен робити клас не вказуючи як саме він це повинен робити. Інтерфейси покликані компенсувати відсутність множинного спадкування у мові Java та гарантують визначення у класах оголошених у собі прототипів методів.

10. Як оголосити та застосувати інтерфейс?

Синтаксис оголошення інтерфейсів:

```
[public] interface НазваІнтерфейсу
{
Прототипи методів та оголошення констант інтерфейсу
}
```

Щоб клас реалізував інтерфейс необхідно:

1. Оголосити за допомогою ключового слова implements, що клас реалізує інтерфейс. Якщо клас реалізує кілька інтерфейсів, то вони перелічуються через кому після ключового слова implements.
2. Визначити у класі усі методи, що вказані у інтерфейсі.

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