

**Міністерство освіти і науки України**

**Національний університет “Львівська політехніка”**

**Кафедра ЕОМ**



**Звіт**

**З лабораторної роботи №4**

**Варіант – 21**

**З дисципліни: «Кросплатформенні засоби програмування»**

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

**Виконав:** ст. гр. КІ-35

Сухан Д. В.

**Перевірив:** доцент кафедри ЕОМ

Іванов Ю.С.

**Львів 2022**

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

## Завдання (варіант № 21)

### Варіант завдання : “Водяний Пістолет”

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.

### Хід роботи:

#### Текст програми

##### Pistol.java

```
package KI35.Sukhan.Lab4;

import java.io.*;

/**
 * Клас Pistol реалізує пістолет
 *
 * @author Denys Sukhan
 * @version 1.0
 * @since version 1.0
 */
public abstract class Pistol extends Object{
    public Trigger trigger;
    private Equipment equipment;
    private Cartridge cartridge;
```

```

public PrintWriter fout;
private boolean fuse;
private String model,company;
private double caliber,shoot_range;

/* Constructor
 * @throws FileNotFoundException
 */

public Pistol() throws FileNotFoundException {
    model ="Desert Eagle";
    company="American weapons.inc";
    caliber=7.62;
    shoot_range=2.2;
    fuse=true;
    trigger = new Trigger();
    equipment = new Equipment();
    cartridge = new Cartridge();
    fout = new PrintWriter(new File("KI35/Sukhan/Lab4/MyFile.txt"));
}

/* Constructor
 * @param model - модель пістолета
 * @param caliber - калібр
пістолета
 * @param sight - приціл
 * @param company - коспанія виробник
 * @param shoot_range - дальність пострілу
 * @param flashlight - ліхтарик
 * @throws FileNotFoundException
 * @param laser - лазер
 */

public Pistol(String model,String company,double caliber,double shoot_range,boolean
laser,boolean sight, boolean flashlight) throws FileNotFoundException {
    this.model =model;
    this.company=company;
    this.caliber=caliber;
    this.shoot_range=shoot_range;
    fuse=true;
    trigger = new Trigger();
    equipment = new Equipment(laser,sight,flashlight);

```

```

        cartridge = new Cartridge();

        fout = new PrintWriter(new File("KI35/Sukhan/Lab4/MyFile.txt"));
    }

    /**
     * Method змінює положення запобіжника
     */
    public void change_fuse_state(){
        fuse = fuse==false? true:false;
        if(fuse){
            System.out.println("Fuse is changed to [turn on state]\nPlease [turn off]
fuse state to make shot.");
            fout.println("Fuse is changed to [turn on state]\nPlease [turn off] fuse
state to make shot.");
            fout.flush();
        }
        else{
            System.out.println("Fuse is changed to [turn off state]");
            fout.println("Fuse is changed to [turn off state]");
            fout.flush();
        }
    }

    /**
     * Method змінює положення курка
     */
    public void change_push_state(){
        boolean push = trigger.get_push();
        trigger.set_push(!push);
        if(!push){
            System.out.println("Push is changed to [turn on state]");
            fout.println("Push is changed to [turn on state]");
            fout.flush();
        }
        else{
            System.out.println("Push is changed to [turn off state]\nPlease [turn
on] push state to make shot.");

```

```

        fout.println("Push is changed to [turn off state]\nPlease [turn on] push
state to make shot.");
        fout.flush();
    }

}

/**
 * Method виконує постріл
 */
public abstract void shoot();

/**
 * Method перезаряджає обойму
 */
public void reload(){
    int cur_bullets = cartridge.get_cur_bullets();
    int max_bullets = cartridge.get_max_bullets();
    if (cur_bullets == max_bullets){
        System.out.println("Pistol doesn't require reloading!");
        fout.println("Pistol doesn't require reloading!");
        fout.flush();
    }
    else{
        cartridge.set_cur_bullets(max_bullets);
        System.out.println("The gun is reloaded !");
        fout.println("The gun is reloaded !");
        fout.flush();
    }
}

/**
 * Method змінює комплектування лазером
 */
public void switch_laser(){
    boolean laser = equipment.get_laser();
    System.out.println(laser==false? "Laser is [turned on] ": "Laser is [turned
on]");

```

```
fout.println(laser==false? "Laser is [turned on] ": "Laser is [turned on]");
fout.flush();
equipment.set_laser(!laser);
}

/**
 * Method змінює комплектування прицілом
 */
public void switch_sight(){
    boolean sight = equipment.get_sight();
    System.out.println(sight==false? "Sight is [turned on] ": "Sight is [turned
on]");
    fout.println(sight==false? "Sight is [turned on] ": "Sight is [turned on]");
    fout.flush();
    equipment.set_sight(!sight);
}

/**
 * Method змінює комплектування ліхтариком
 */
public void switch_flashlight(){
    boolean flashlight = equipment.get_flashlight();
    System.out.println(flashlight==false? "Flashlight is [turned on] ": "Flashlight
is [turned on]");
    fout.println(flashlight==false? "Flashlight is [turned on] ": "Flashlight is
[turned on]");
    fout.flush();
    equipment.set_flashlight(!flashlight);
}

/**
 * Method об'єм магазину патронів
 */
public void change_cartridge_capacity(int capacity){
    cartridge.set_max_bullets(capacity);
    cartridge.set_cur_bullets(capacity);
    System.out.println("Cartridge capacity was changed to :" + capacity);
```

```

        fout.println("Cartridge capacity was changed to :" + capacity);
        fout.flush();
    }

    /**
     * Method виводить інформацію про пістолет
     */
    public void get_info(){
        System.out.println("Pistol model (" + model+")\nMade in company (" +
company+)\nCaliber(" + caliber+)\nShoot range(" + shoot_range+)\nCurrent bullets(" +
cartridge.get_cur_bullets()+")\nCatrige capacity(" +
cartridge.get_max_bullets()+")\n");
        fout.println("Pistol model (" + model+)\nMade in company (" +
company+)\nCaliber(" + caliber+)\nShoot range(" + shoot_range+)\nCurrent bullets(" +
cartridge.get_cur_bullets()+")\nCatrige capacity(" +
cartridge.get_max_bullets()+")\n");
        fout.flush();
    }
}

class Trigger{
    private boolean push;

    /**
     * Constructor
     */
    public Trigger(){
        push = false;
    }

    public void set_push(boolean push){
        this.push = push;
    }

    public boolean get_push(){
        return push;
    }
}

```

```
}
```

```
class Equipment{  
    private boolean laser,sight,flashlight;  
    /*  
     * Constructor  
     */  
    public Equipment(){  
        laser=false;  
        sight=false;  
        flashlight=false;  
    }  
    /* Constructor  
     * @param shoot_range - дальність пострілу  
     * @param flashlight - ліхтарик  
     * @param laser - лазер  
     */  
    public Equipment(boolean laser,boolean sight, boolean flashlight){  
        this.laser=laser;  
        this.sight=sight;  
        this.flashlight=flashlight;  
    }  
  
    public void set_laser(boolean laser){  
        this.laser = laser;  
    }  
    public void set_sight(boolean sight){  
        this.sight = sight;  
    }  
    public void set_flashlight(boolean flashlight){  
        this.flashlight = flashlight;  
    }  
    public boolean get_laser(){  
        return laser;  
    }  
    public boolean get_sight(){  
        return sight;  
    }  
}
```



```

    }

    public boolean get_flashlight(){
        return flashlight;
    }
}

class Cartridge {
    private int max_bullets;
    private int cur_bullets;
    /*
     * Constructor
     */
    public Cartridge(){
        max_bullets = 15;
        cur_bullets= max_bullets;
    }
    public void set_max_bullets(int max_bullets){
        this.max_bullets = max_bullets;
    }

    public void set_cur_bullets(int cur_bullets){
        this.cur_bullets = cur_bullets;
    }

    public int get_max_bullets(){
        return max_bullets;
    }

    public int get_cur_bullets(){
        return cur_bullets;
    }
}

```

## WaterPistol.java та клас Main

```
package KI35.Sukhan.Lab4;
import java.io.*;

interface ShootMode {
    void setShootMode(int mode);
    int getShootMode();
}

/**
 * Клас WaterPistol реалізує Водяний пістолет
 * @author Denys Sukhan
 * @version 1.0
 * @since version 1.0
 */

public class WaterPistol extends Pistol implements
ShootMode {
    private int mode;
    int curWaterCapacity; // Capacity value of water
pistol cartridge
    int maxWaterCapacity; // in range ( 0 to 100) percent

    /* Constructor
     * @throws FileNotFoundException
     */

    public WaterPistol() throws FileNotFoundException{
        super();
        mode = 0;
    }

    /**
     * Method Виконує вистріл водою різними способами
     */
    @Override
    public void shoot(){
```

```

boolean push = trigger.get_push();
int curWaterCapacity = get_curWaterCapacity();
if (push == true && curWaterCapacity > 0){
    if(mode==1){
        set_curWaterCapacity(curWaterCapacity=0);
    }else if(mode==0){
        set_curWaterCapacity(curWaterCapacity-20);
    }
    else{
        System.out.println("Wrong input mode of type
Water Shoot!");
    }
    trigger.set_push(false);
    System.out.println("Water pistol fired, cartridge
capacity (" +curWaterCapacity+"%)");
    fout.println("Water pistol fired, cartridge
capacity (" +curWaterCapacity+"%)");
    fout.flush();
}
else{
    System.out.println("Water pistol can't shoot!");
    fout.println("Water pistol can't shoot!");
    fout.flush();
}
}

```

```

/**
 * @return shootMode
 */
public int getShootMode() {
    System.out.println(mode);
    return mode;
}

```

```

/**
 * @param mode - спосіб

```

```

    */
    public void setShootMode(int mode) {
        this.mode = mode;
        System.out.println("Set shooting mode a water
pistol to: " + mode);
        fout.println("Set shooting mode a water pistol to:
" + mode);
        fout.flush();
    }

    /**
     * Method встановлює максимальний рівень води
     * */
    public void set_maxWaterCapacity(int
maxWaterCapacity){
        this.maxWaterCapacity = maxWaterCapacity;
    }

    /**
     * Method встановлює наявний рівень води
     * */
    public void set_curWaterCapacity(int
curWaterCapacity){
        this.curWaterCapacity =curWaterCapacity;
    }

    /**
     * Method повертає максимальний рівень води
     * */
    public int get_maxWaterCapacity(){
        return maxWaterCapacity;
    }

    /**
     * Method повертає наявний рівень води
     * */
    public int get_curWaterCapacity(){

```

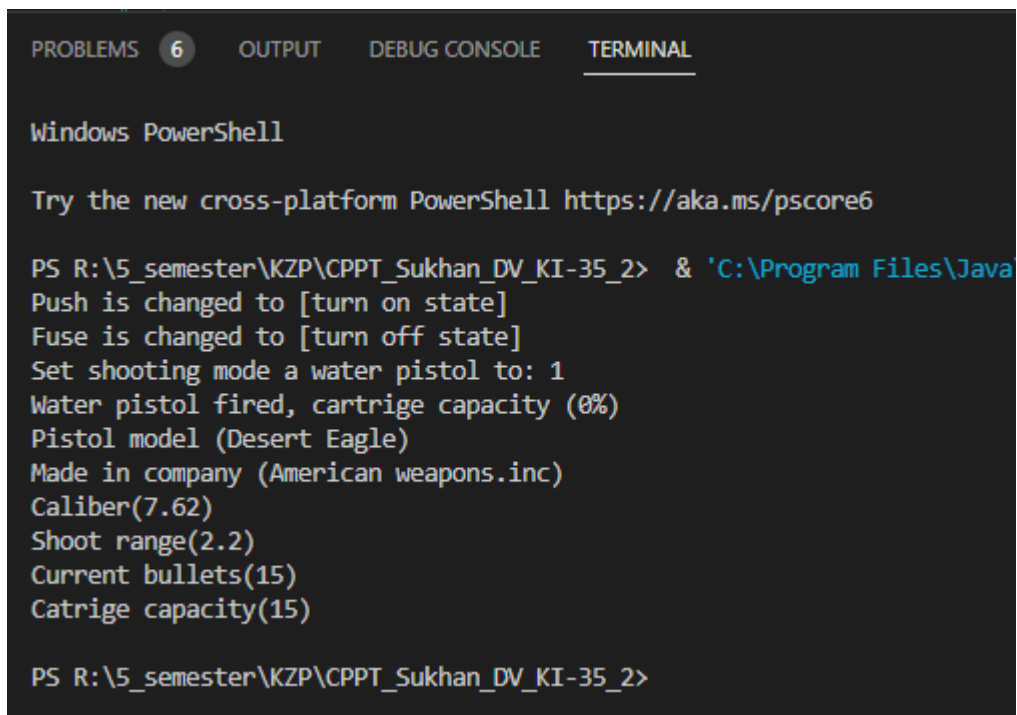
```

        return curWaterCapacity;
    }
}

class Main {
    public static void main(String[] arg)throws
FileNotFoundException{
        WaterPistol wp = new WaterPistol();
        wp.set_maxWaterCapacity(5);
        wp.set_curWaterCapacity(2);
        wp.change_push_state();
        wp.change_fuse_state();
        wp.setShootMode(1);
        wp.shoot();
        wp.get_info();
    }
}

```

### Результат виконання програми



```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell

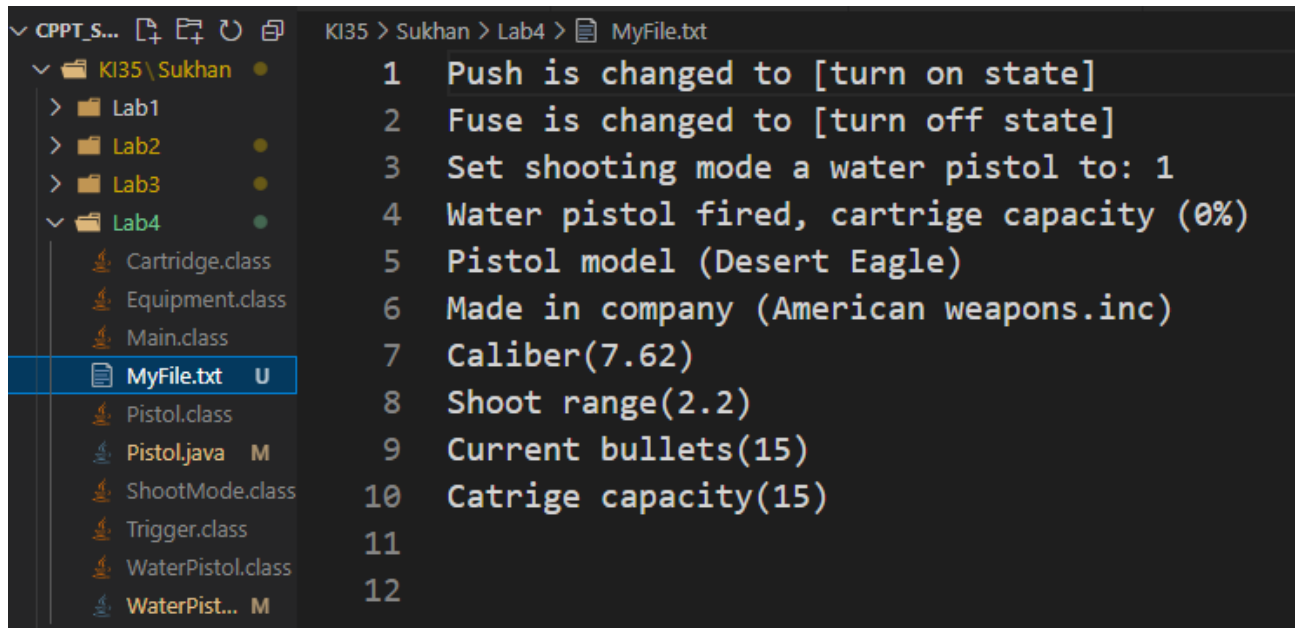
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS R:\5_semester\KZP\CPPT_Sukhan_DV_KI-35_2> & 'C:\Program Files\Java
Push is changed to [turn on state]
Fuse is changed to [turn off state]
Set shooting mode a water pistol to: 1
Water pistol fired, cartridge capacity (0%)
Pistol model (Desert Eagle)
Made in company (American weapons.inc)
Caliber(7.62)
Shoot range(2.2)
Current bullets(15)
Cartridge capacity(15)

PS R:\5_semester\KZP\CPPT_Sukhan_DV_KI-35_2>

```

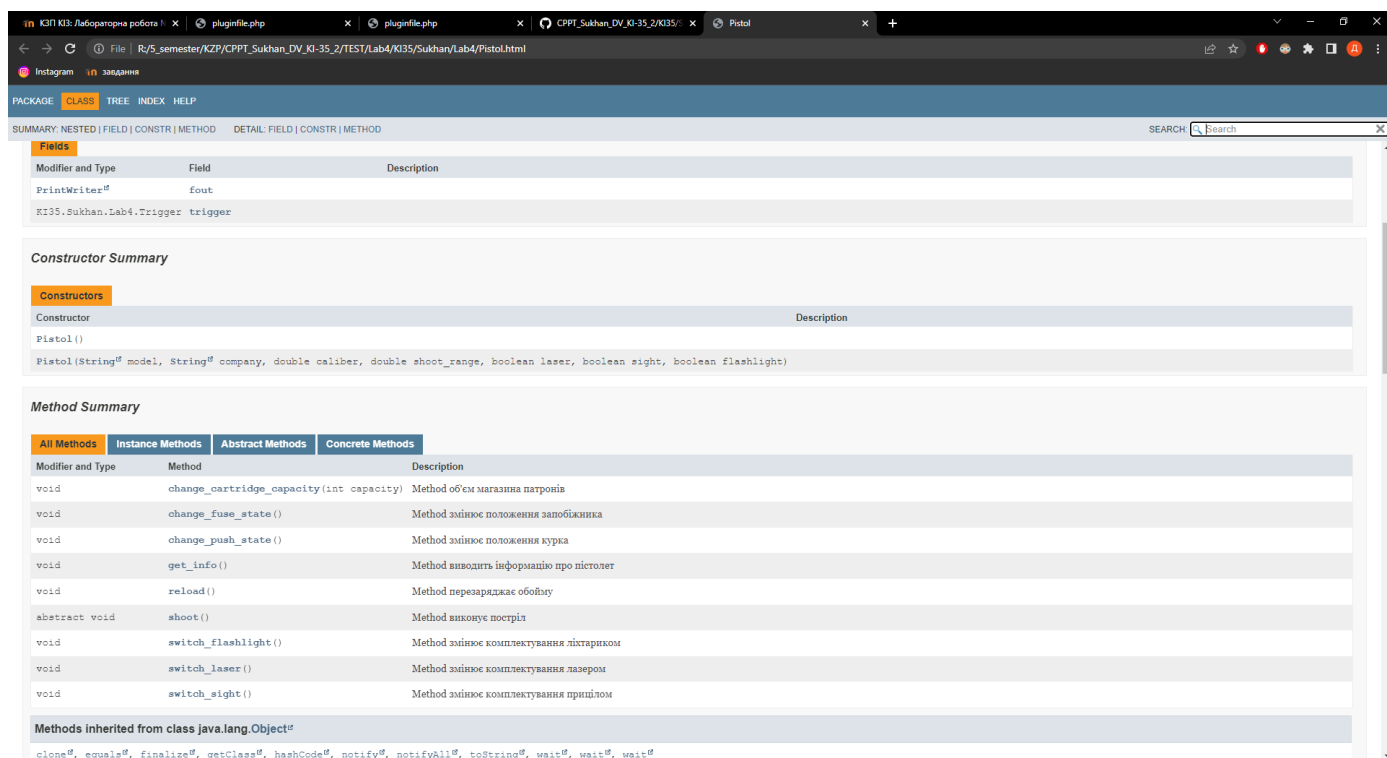
## Вміст файлу MyFile.txt



```
1 Push is changed to [turn on state]
2 Fuse is changed to [turn off state]
3 Set shooting mode a water pistol to: 1
4 Water pistol fired, cartrige capacity (0%)
5 Pistol model (Desert Eagle)
6 Made in company (American weapons.inc)
7 Caliber(7.62)
8 Shoot range(2.2)
9 Current bullets(15)
10 Catrige capacity(15)
11
12
```

## Фрагмент згенерованої документації

### Pistol.html



PACKAGE CLASS TREE INDEX HELP

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

#### Fields

Modifier and Type	Field	Description
PrintWriter	fout	
K135.Sukhan.Lab4.Trigger	trigger	

#### Constructor Summary

Constructor	Description
Pistol()	
Pistol(String model, String company, double caliber, double shoot_range, boolean laser, boolean sight, boolean flashlight)	

#### Method Summary

Modifier and Type	Method	Description
void	change_cartridge_capacity(int capacity)	Method об'єм магазину патронів
void	change_fuse_state()	Method змінює положення запобіжника
void	change_push_state()	Method змінює положення курка
void	get_info()	Method виводить інформацію про пістолет
void	reload()	Method перезаряджає обойму
abstract void	shoot()	Method виконує постріл
void	switch_flashlight()	Method змінює комплектування ліхтариком
void	switch_laser()	Method змінює комплектування лазером
void	switch_sight()	Method змінює комплектування прицілом

Methods inherited from class java.lang.Object

clone(), equals(), finalize(), getClass(), hashCode(), notify(), notifyAll(), toString(), wait(), wait(), wait()

# WaterPistol.html

**Constructors**

Constructor	Description
WaterPistol()	

**Method Summary**

Modifier and Type	Method	Description
int	get_curWaterCapacity()	Method повертає наявний рівень води
int	get_maxWaterCapacity()	Method повертає максимальний рівень води
int	getShootMode()	
void	set_curWaterCapacity(int curWaterCapacity)	Method встановлює наявний рівень води
void	set_maxWaterCapacity(int maxWaterCapacity)	Method встановлює максимальний рівень води
void	setShootMode(int mode)	
void	shoot()	Method Виконує вистріл водою різними способами

**Methods inherited from class K135.Sukhan.Lab4.Pistol**

change\_cartridge\_capacity, change\_fuse\_state, change\_push\_state, get\_info, reload, switch\_flashlight, switch\_laser, switch\_sight

**Methods inherited from class java.lang.Object**

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

**Constructor Details**

**WaterPistol**

```
public WaterPistol()
    throws FileNotFoundException
```

Throws:

## Відповідь на контрольні запитання:

### 1. Синтаксис реалізації спадкування.

**Відповідь:**

class Підклас extends Суперклас

{

Додаткові поля і методи

}

### 2. Що таке суперклас та підклас?

**Відповідь:** Суперклас – батьківський клас. Підклас – дочірній.

### 3. Як звернутися до членів суперкласу з підкласу?

**Відповідь:** super.назваМетоду([параметри]); super.назваПоля;

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

**Відповідь:** метод є приватним, статичним, фінальним або конструктором. Механізм статичного зв'язування передбачає визначення методу, який необхідно викликати, на етапі компіляції.

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

**Відповідь:** метод, що необхідно викликати, визначається по фактичному типу неявного параметру.

## 6. Що таке абстрактний клас та як його реалізувати?

**Відповідь:** Це клас який оголошений з ключовим словом `abstract`. Об'єкт такого класу не може бути створеним, може вміщати абстрактні методи.

## 7. Для чого використовується ключове слово `instanceof`?

**Відповідь:** Для встановлення чи є певний клас спадкоємцем другого.

## 8. Як перевірити чи клас є підкласом іншого класу?

**Відповідь:** використати ключове слово `instanceof`.

## 9. Що таке інтерфейс?

**Відповідь:** 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?

**Відповідь:**

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

Застосувати можна імплементуючи його, або створюючи посилання на дочірній об'єкт класу.

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