**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Чорноморський національний університет   
імені Петра Могили**

**Факультет комп’ютерних наук**

**Кафедра інженерії програмного забезпечення**

**ЗВІТ**

*з лабораторної роботи № 4*

**«Абстрактні класи. Інтерфейси»**

**Варіант № 15**

Дисципліна «Об'єктно-орієнтоване програмування»

Спеціальність «Інженерія програмного забезпечення»

121–ЛР.04–209.22210929

***Здобувачка*** *\_\_\_\_\_\_\_\_\_\_****О. К.******Шумакова***

*(підпис)*

*\_\_17.01.2024 \_\_*

*(дата)*

***Викладач*** *\_\_\_\_\_\_\_\_\_\_\_\_\_****Г. В.******Горбань***

*(підпис)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(дата)*

**Миколаїв – 2024**

**Лабораторна робота №4**

**Тема роботи:**Абстрактні класи. Інтерфейси.

**Завдання:**

1. Створити абстрактний клас та інтерфейс за варіантом. Оголосити один метод в абстрактному класі. Оголосити один метод за замовчуванням в інтерфейсі.
2. Створити два класи, які реалізують інтерфейс та наслідують абстрактний клас. Для створених класів перевизначити методи *toString*() та *equals*(), використати оператор *instanceof*.
3. Класи, створені в завданні 2, мають реалізовувати інтерфейси *Comparable*та *Cloneable*. Продемонструвати в методі *main*() сортування та клонування об'єктів.
4. Створити метод, який приймає в якості параметру об'єкт типу абстрактного класу за варіантом та викликає його метод. Створити метод, який приймає в якості параметру об'єкт типу інтерфейсу за варіантом та викликає його метод.

**Лістинг коду:**

**import** java**.**util**.**Arrays**;**

abstract class FlyingMachine **implements** Comparable**<**FlyingMachine**>** **{**

private String model**;**

private int passengerCapacity**;**

private double maxSpeed**;**

FlyingMachine**(**String model**,** int passengerCapacity**,** double maxSpeed**)** **{**

**this.**model **=** model**;**

**this.**passengerCapacity **=** passengerCapacity**;**

**this.**maxSpeed **=** maxSpeed**;**

**}**

abstract void fly**();**

String getModel**()** **{**

**return** model**;**

**}**

int getPassengerCapacity**()** **{**

**return** passengerCapacity**;**

**}**

double getMaxSpeed**()** **{**

**return** maxSpeed**;**

**}**

@Override

public int compareTo**(**FlyingMachine o**)** **{**

**return** Double**.**compare**(this.**maxSpeed**,** o**.**maxSpeed**);**

**}**

@Override

public boolean equals**(**Object obj**)** **{**

**if** **(this** **==** obj**)** **return** **true;**

**if** **(**obj **==** **null** **||** getClass**()** **!=** obj**.**getClass**())** **return** **false;**

FlyingMachine that **=** **(**FlyingMachine**)** obj**;**

**return** passengerCapacity **==** that**.**passengerCapacity **&&**

Double**.**compare**(**that**.**maxSpeed**,** maxSpeed**)** **==** 0 **&&**

model**.**equals**(**that**.**model**);**

**}**

**}**

interface Flyable **{**

**default** void takeOff**()** **{**

System**.**out**.**println**(**"Taking off..."**);**

**}**

**}**

class Airplane **extends** FlyingMachine **implements** Flyable**,** Cloneable **{**

Airplane**(**String model**,** int passengerCapacity**,** double maxSpeed**)** **{**

**super(**model**,** passengerCapacity**,** maxSpeed**);**

**}**

@Override

void fly**()** **{**

System**.**out**.**println**(**"Airplane is flying..."**);**

**}**

@Override

protected Object clone**()** **throws** CloneNotSupportedException **{**

**return** **super.**clone**();**

**}**

@Override

public String toString**()** **{**

**return** "Airplane{" **+**

"model='" **+** getModel**()** **+** '\'' **+**

", passengerCapacity=" **+** getPassengerCapacity**()** **+**

", maxSpeed=" **+** getMaxSpeed**()** **+**

'}'**;**

**}**

**}**

class Helicopter **extends** FlyingMachine **implements** Flyable**,** Cloneable **{**

Helicopter**(**String model**,** int passengerCapacity**,** double maxSpeed**)** **{**

**super(**model**,** passengerCapacity**,** maxSpeed**);**

**}**

@Override

void fly**()** **{**

System**.**out**.**println**(**"Helicopter is flying..."**);**

**}**

@Override

protected Object clone**()** **throws** CloneNotSupportedException **{**

**return** **super.**clone**();**

**}**

@Override

public String toString**()** **{**

**return** "Helicopter{" **+**

"model='" **+** getModel**()** **+** '\'' **+**

", passengerCapacity=" **+** getPassengerCapacity**()** **+**

", maxSpeed=" **+** getMaxSpeed**()** **+**

'}'**;**

**}**

**}**

public class Main **{**

public static void main**(**String**[]** args**)** **throws** CloneNotSupportedException **{**

Airplane airplane1 **=** **new** Airplane**(**"Boeing 747"**,** 366**,** 988**);**

Airplane airplane2 **=** **(**Airplane**)** airplane1**.**clone**();**

Helicopter helicopter1 **=** **new** Helicopter**(**"Bell 206"**,** 5**,** 217**);**

Helicopter helicopter2 **=** **(**Helicopter**)** helicopter1**.**clone**();**

callFly**(**airplane1**);**

callTakeOff**(**airplane1**);**

printDetails**(**airplane1**);**

callFly**(**helicopter1**);**

callTakeOff**(**helicopter1**);**

printDetails**(**helicopter1**);**

callFly**(**airplane2**);**

callTakeOff**(**airplane2**);**

printDetails**(**airplane2**);**

callFly**(**helicopter2**);**

callTakeOff**(**helicopter2**);**

printDetails**(**helicopter2**);**

System**.**out**.**println**(**"Airplane1 equals Airplane2: " **+** airplane1**.**equals**(**airplane2**));**

System**.**out**.**println**(**"Helicopter1 equals Helicopter2: " **+** helicopter1**.**equals**(**helicopter2**));**

FlyingMachine**[]** flyingMachines **=** **{**airplane1**,** airplane2**,** helicopter1**,** helicopter2**};**

Arrays**.**sort**(**flyingMachines**);**

System**.**out**.**println**(**"After sorting by max speed:"**);**

**for** **(**FlyingMachine flyingMachine **:** flyingMachines**)** **{**

printDetails**(**flyingMachine**);**

**}**

**}**

static void callFly**(**FlyingMachine flyingMachine**)** **{**

flyingMachine**.**fly**();**

**}**

static void callTakeOff**(**Flyable flyable**)** **{**

flyable**.**takeOff**();**

**}**

static void printDetails**(**FlyingMachine flyingMachine**)** **{**

System**.**out**.**println**(**flyingMachine**);**

**}**

**}**

**Результати роботи програми:**

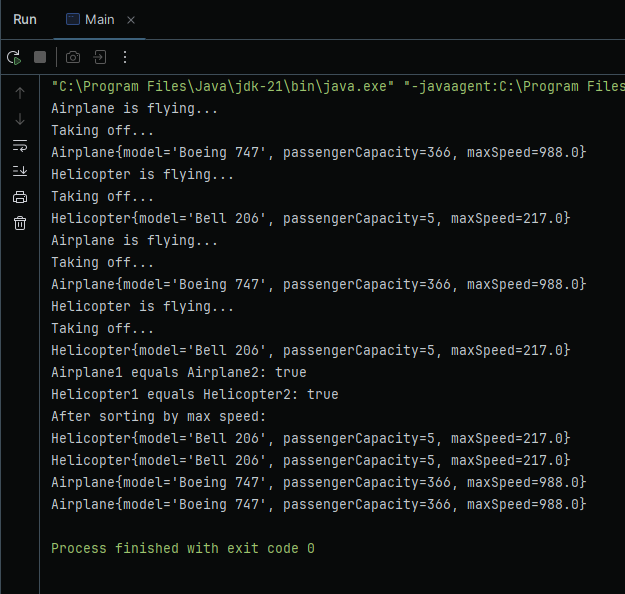


Рисунок 1