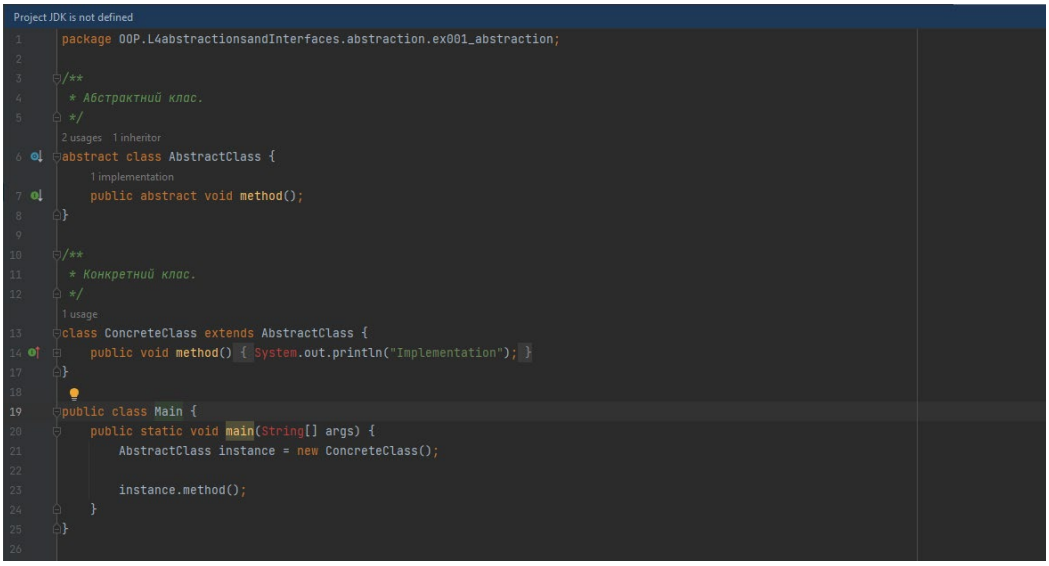
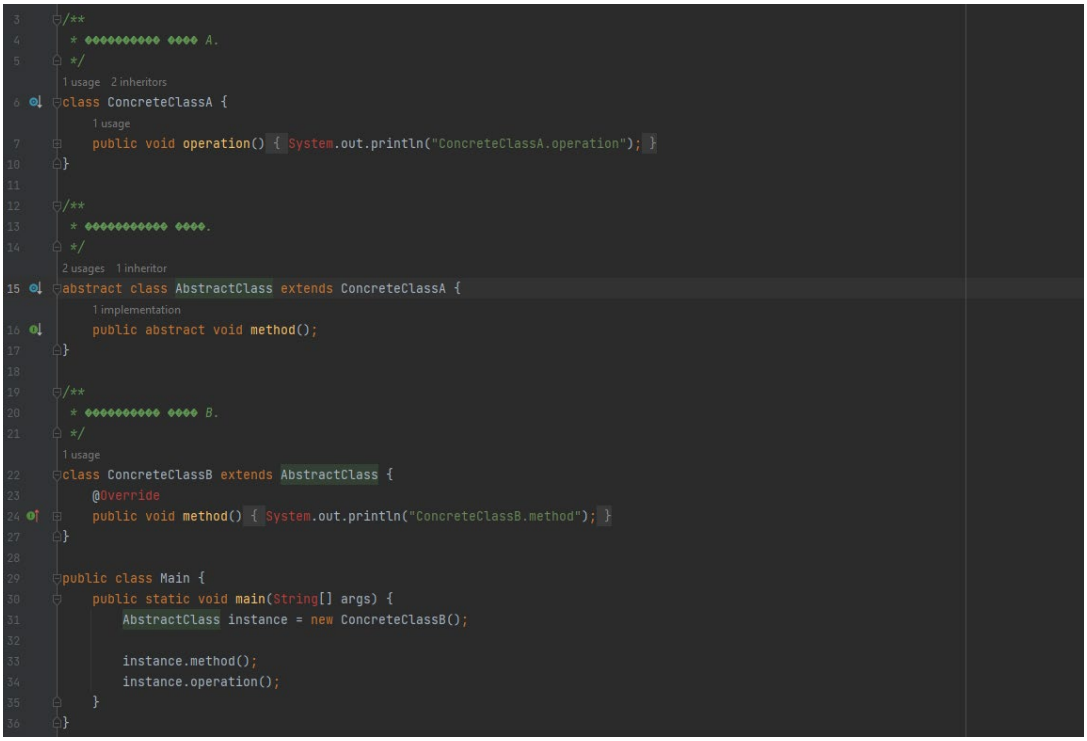


Розділ на Oracle: https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

Опис роботи коду: Клас AbstractClass є абстрактним і містить один абстрактний метод method(), який не має реалізації у самому класі. Клас ConcreteClass є конкретною реалізацією абстрактного класу AbstractClass. Він перевизначає метод method(), додаючи конкретну реалізацію, у разі виведення рядка " Implementation " на консоль. У методі main класу Main створюється екземпляр ConcreteClass, який присвоюється змінній типу AbstractClass. Потім викликається метод method() на цій змінній. Під час виконання програми на консоль буде виведено рядок "Implementation", що є результатом виклику методу method() у класі ConcreteClass.



Розділ на Oracle: https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

https://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html

https://docs.oracle.com/javase/tutorial/java/IandI/polymorphism.html

Опис роботи коду: Клас ConcreteClassA - конкретний клас, який містить метод operation(), що виводить повідомлення "ConcreteClassA.operation". Абстрактний клас AbstractClass, який є

спадкоємцем класу ConcreteClassA. Він містить абстрактний метод method(), який має бути реалізований у його підкласах. Клас ConcreteClassB - конкретний клас, який успадковується від абстрактного класу AbstractClass і реалізує його абстрактний метод method(). Метод method() виводить повідомлення "ConcreteClassB.method". У методі main створюється об'єкт instance типу AbstractClass, який посилається на екземпляр класу ConcreteClassB. Потім викликаються методи method() і operation() на об'єкті instance. Таким чином, код демонструє успадкування, абстрактні класи та поліморфізм у Java.

```
package OOP.L4abstractionsandInterfaces.abstraction.ex003_abstraction;

/**
 * ◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆ ◆◆◆◆ ◆◆◆◆ ◆◆◆◆◆◆◆◆◆◆◆◆ ◆◆ ◆◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆◆.
 * ◆◆◆◆◆◆◆◆◆ ◆◆◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆◆◆ ◆ ◆◆◆◆◆◆◆◆ ◆◆◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆◆, ◆ ◆◆◆◆◆◆◆◆◆ ◆◆◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆ ◆ ◆◆ ◆ ◆◆◆◆'◆◆◆◆◆◆◆◆.
 */

/**
 * ◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆ A.
 */
2 usages  2 inheritors
abstract class AbstractClassA {
    2 usages  1 implementation
    public abstract void operationA();
}

/**
 * ◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆ B.
 */
2 usages  1 inheritor
abstract class AbstractClassB extends AbstractClassA {
    1 usage  1 implementation
    public abstract void operationB();
}

/**
 * ◆◆◆◆◆◆◆◆◆◆ ◆◆◆◆.
 */
2 usages
class ConcreteClass extends AbstractClassB {
    2 usages
    @Override
    public void operationA() { System.out.println("ConcreteClass.operationA"); }

    1 usage
    @Override
    public void operationB() { System.out.println("ConcreteClass.operationB"); }
}
public class Main {
    public static void main(String[] args) {
        AbstractClassA instance = new ConcreteClass();

        instance.operationA();

//        instance.operationB();  // ◆◆◆◆◆◆◆: ◆◆◆◆ ◆◆◆◆◆◆◆◆◆◆◆ ◆◆◆ ◆◆◆◆◆?
        AbstractClassB instance1 = new ConcreteClass();

        instance1.operationB();
        instance1.operationA();

    }
}
```

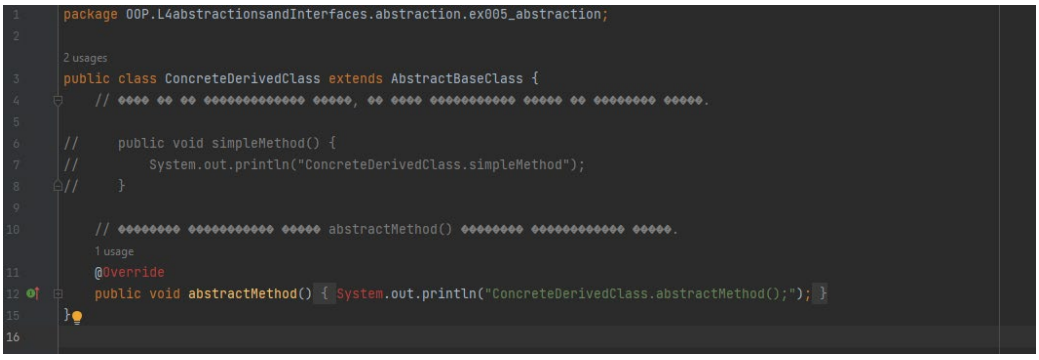
Розділ на Oracle: https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

Опис роботи коду: Абстрактний клас AbstractClassA, який має абстрактний метод operationA(). Абстрактний клас AbstractClassB, який розширює клас AbstractClassA і має абстрактний метод operationB(). Клас ConcreteClass, який успадковує клас AbstractClassB і реалізує методи operationA() та operationB(). У методі main створюється екземпляр класу ConcreteClass і присвоюється змінній типу AbstractClassA. Це робиться, оскільки ConcreteClass є підкласом AbstractClassA. Далі викликається

метод operationA() на цьому екземплярі, що виводить рядок "ConcreteClass.operationA". У коментарі нижче закоментований виклик методу operationB(). Це тому, що хоча змінна instance фактично містить екземпляр класу ConcreteClass, яка реалізує метод operationB(), тип змінної вказує на AbstractClassA, який не має такого методу. Тому, виклик цього методу є недоступним. Також використовується інша змінна instance1 типу AbstractClassB, якій також присвоєний екземпляр класу ConcreteClass. На цьому екземплярі викликаються методи operationB() та operationA(), які виводять рядки "ConcreteClass.operationB" і "ConcreteClass.operationA" відповідно.

Розділ на Oracle: https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

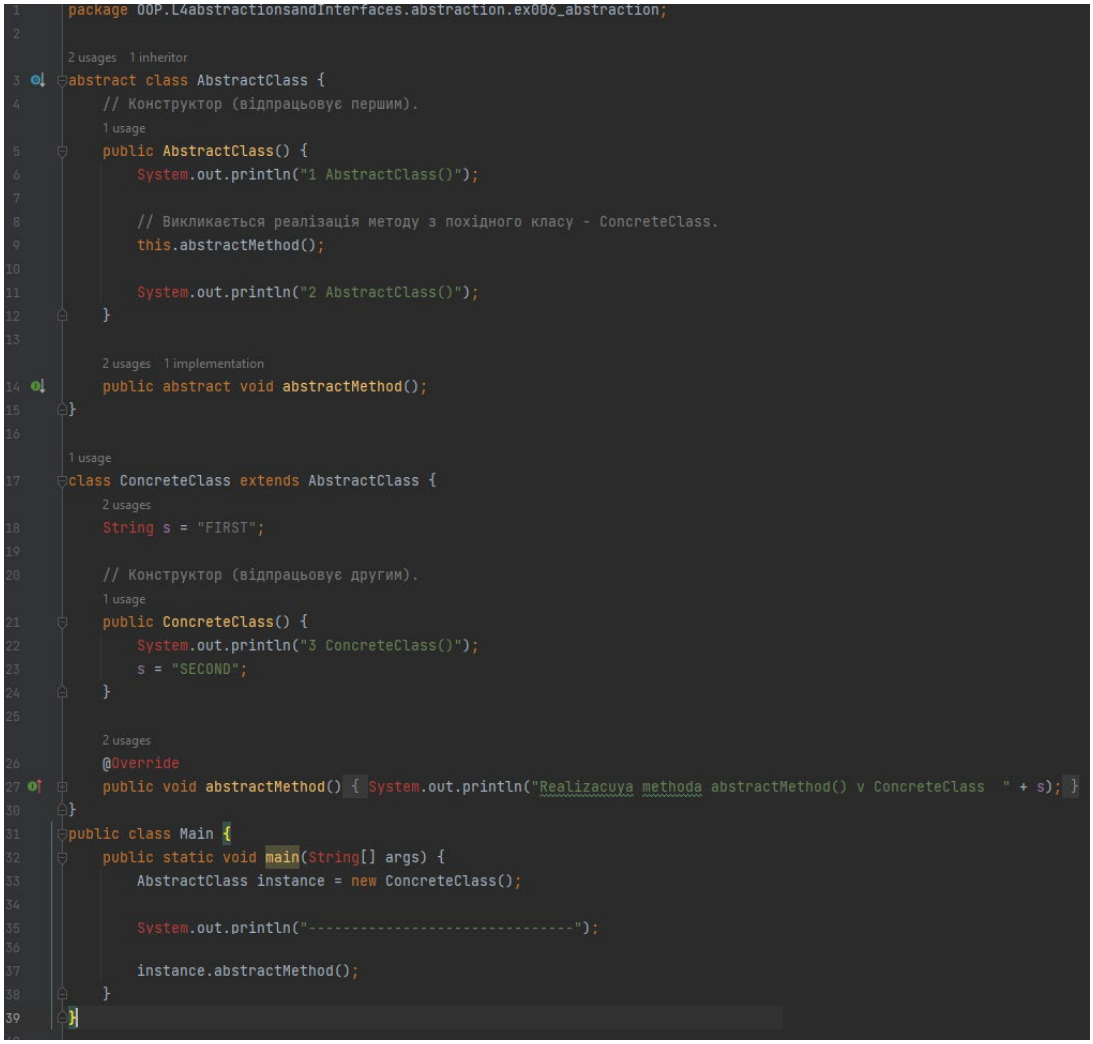
Опис роботи коду: Метод "simpleMethod()" - це публічний метод без аргументів, який надає просту реалізацію. У даному випадку, він просто виводить повідомлення "AbstractBaseClass.simpleMethod" на консоль. Метод "overriddenMethod()" - також публічний метод без аргументів, але з покажчиком "public". Цей метод може бути перевизначений (overridden) у похідних класах. У даному випадку, він також виводить повідомлення "AbstractBaseClass.overriddenMethod" на консоль. Метод "abstractMethod()" - це абстрактний метод, який не має реалізації у базовому класі. Абстрактні методи призначені для перевизначення у похідних класах. У даному випадку, цей метод просто оголошується без будь-якого коду. Клас "AbstractBaseClass" є абстрактним класом, що означає, що його не можна створювати окремо, а лише використовувати як базовий для похідних класів. Інші класи можуть успадковувати від нього і перевизначати його методи за необхідності.



Розділ на Oracle: https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

Опис роботи коду: Код представляє похідний клас "ConcreteDerivedClass", який успадковує від абстрактного базового класу "AbstractBaseClass" на Java. Давайте розглянемо його роботу:

Клас "ConcreteDerivedClass" успадковує всі поля і методи з класу "AbstractBaseClass" і може додавати власні поля та методи. У даному випадку, у похідному класі "ConcreteDerivedClass" немає перевизначеного методу "simpleMethod()". Код методу з коментарем "//" є закоментованим, що означає, що цей метод не використовується в похідному класі і використовується реалізація з базового класу. Метод "abstractMethod()" перевизначений у похідному класі "ConcreteDerivedClass" з використанням анотації "@Override". Він має реалізацію, де виводиться повідомлення "ConcreteDerivedClass.abstractMethod();" на консоль. Таким чином, похідний клас "ConcreteDerivedClass" успадковує методи базового класу і може перевизначати абстрактні методи за необхідності.



```
package OOP.L4abstractionsandInterfaces.abstraction.ex006_abstraction;

2 usages  1 inheritor
abstract class AbstractClass {
    // Конструктор (відпрацьовує першим).
    1 usage
    public AbstractClass() {
        System.out.println("1 AbstractClass()");

        // Викликається реалізація методу з похідного класу - ConcreteClass.
        this.abstractMethod();

        System.out.println("2 AbstractClass()");
    }

    2 usages  1 implementation
    public abstract void abstractMethod();
}

1 usage
class ConcreteClass extends AbstractClass {
    2 usages
    String s = "FIRST";

    // Конструктор (відпрацьовує другим).
    1 usage
    public ConcreteClass() {
        System.out.println("3 ConcreteClass()");
        s = "SECOND";
    }

    2 usages
    @Override
    public void abstractMethod() { System.out.println("Realizacuya methoda abstractMethod() v ConcreteClass  " + s); }
}
public class Main {
    public static void main(String[] args) {
        AbstractClass instance = new ConcreteClass();

        System.out.println("-----------------------------");

        instance.abstractMethod();
    }
}
```

Розділ на Oracle: https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

https://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html

https://docs.oracle.com/javase/tutorial/java/IandI/override.html

Опис роботи коду: Клас "AbstractClass" є абстрактним і містить абстрактний метод "abstractMethod()". Він також має конструктор, який викликається при створенні об'єкта. У конструкторі виводиться повідомлення "1 AbstractClass()", потім викликається абстрактний метод "abstractMethod()" (який буде реалізований у похідних класах) і потім виводиться повідомлення "2 AbstractClass()". Клас "ConcreteClass" успадковує від "AbstractClass" і реалізує абстрактний метод "abstractMethod()". Він також має своє поле "s" зі значенням "FIRST". У конструкторі "ConcreteClass" виводиться повідомлення "3 ConcreteClass()" і значенню поля "s" присвоюється "SECOND". У класі "Main" в методі "main" створюється об'єкт типу "AbstractClass" з посиланням на об'єкт "ConcreteClass". Це можливо,

оскільки "ConcreteClass" є похідним класом від "AbstractClass". Виводиться повідомлення "1 AbstractClass()" з конструктора базового класу, потім викликається реалізація методу "abstractMethod()" у похідному класі "ConcreteClass", виводиться повідомлення "Realizacuya methoda abstractMethod() v ConcreteClass SECOND", і нарешті виводиться повідомлення "2 AbstractClass()". Після цього виводиться розділювальна лінія "------------------------------". Викликається метод "abstractMethod()" на об'єкті "instance" типу "AbstractClass". В цьому випадку викликається реалізація методу "abstractMethod()" у похідному класі "ConcreteClass". Виводиться повідомлення "Realizacuya methoda abstractMethod() v ConcreteClass SECOND". Таким чином, код демонструє роботу з абстрактними класами, успадкуванням і перевизначенням методів у похідних класах.