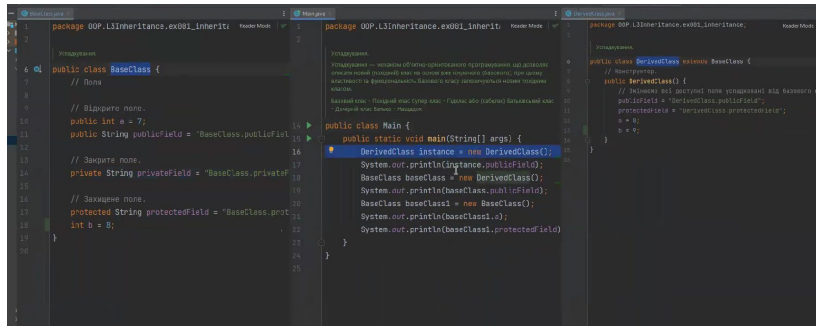


## Успадкування



```
package GJP.L5inheritance.ex001_inherit;

// Підклас
public class BaseClass {
    // Атрибути
    public int a = 7;
    public String publicField = "BaseClass.publicField";
    // Захистені атрибути
    private String privateField = "BaseClass.privateField";
    // Захистені методи
    protected String protectedField = "BaseClass.protectedField";
    int b = 8;
}

package GJP.L5inheritance.ex001_inherit;

// Підклас
public class DerivedClass extends BaseClass {
    // Конструктор
    public DerivedClass() {
        publicField = "DerivedClass.publicField";
        protectedField = "DerivedClass.protectedField";
    }
    // Метод show()
    public void show() {
        System.out.println("Method show from derived class");
    }
}

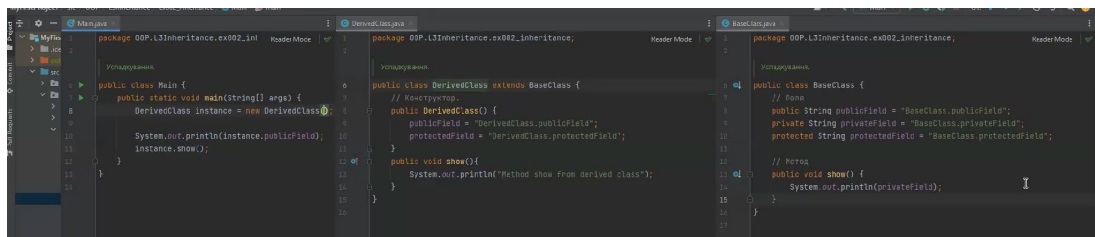
package GJP.L5inheritance.ex001_inherit;

// Підклас
public class Main {
    // Метод main()
    public static void main(String[] args) {
        DerivedClass instance = new DerivedClass();
        System.out.println(instance.publicField);
        System.out.println(instance.privateField);
        System.out.println(instance.protectedField);
    }
}
```

**Посилання на Oracle:** <https://docs.oracle.com/javase/tutorial/java/inheritance/subclasses.html>

**Опис до прикладу:** Даний код використовує успадкування між класами. У головному класі Main створюється об'єкт DerivedClass, який є похідним класом від BaseClass. Це означає, що DerivedClass успадковує всі поля та методи від BaseClass. У конструкторі DerivedClass змінюються значення деяких успадкованих полів, таких як publicField та protectedField. Також в цьому конструкторі встановлюється значення для нового поля c, яке є специфічним для DerivedClass. У головному методі main створюються різні об'єкти: instance типу DerivedClass, baseClass1 типу BaseClass (але з посиланням на об'єкт DerivedClass) та baseClass2 типу BaseClass. При виклику методу System.out.println виводяться значення деяких полів цих об'єктів, таких як publicField, а та protectedField. Успадкування дозволяє класу-спадкоємцю отримати всі поля та методи батьківського класу та, за необхідності, змінити їх значення або додати нові поля та методи.

## Успадкування (#2)



```
package GJP.L5inheritance.ex002_inherit;

// Підклас
public class BaseClass {
    // Атрибути
    public String publicField = "BaseClass.publicField";
    private String privateField = "BaseClass.privateField";
    protected String protectedField = "BaseClass.protectedField";
    // Метод show()
    public void show() {
        System.out.println(privateField);
    }
}

package GJP.L5inheritance.ex002_inherit;

// Підклас
public class DerivedClass extends BaseClass {
    // Конструктор
    public DerivedClass() {
        publicField = "DerivedClass.publicField";
        protectedField = "DerivedClass.protectedField";
    }
    // Метод show()
    public void show() {
        System.out.println("Method show from derived class");
    }
}

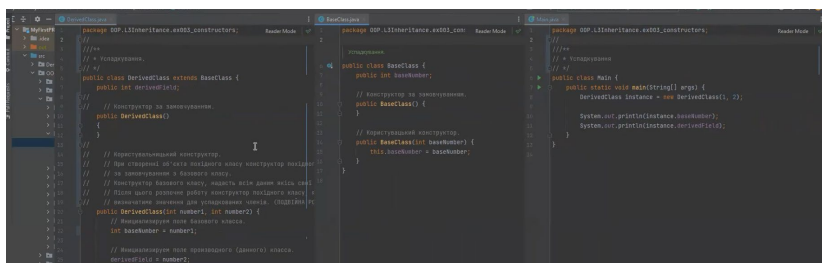
package GJP.L5inheritance.ex002_inherit;

// Підклас
public class Main {
    // Метод main()
    public static void main(String[] args) {
        DerivedClass instance = new DerivedClass();
        System.out.println(instance.publicField);
        System.out.println(instance.privateField);
        System.out.println(instance.protectedField);
    }
}
```

**Посилання на Oracle:** <https://docs.oracle.com/javase/tutorial/java/inheritance/override.html>

**Опис до прикладу:** Успадкування дозволяє класу DerivedClass успадкувати властивості та методи класу BaseClass, тобто DerivedClass стає підкласом BaseClass. Клас BaseClass містить три поля: publicField, privateField та protectedField, а також метод show(). Поле publicField є публічним і доступним для будь-якого коду. Поле privateField є приватним і доступним тільки в межах класу BaseClass. Поле protectedField є захищеним і доступним для підкласів BaseClass. Клас DerivedClass успадковує клас BaseClass і має доступ до його полів та методів. У конструкторі класу DerivedClass змінюються значення полів publicField та protectedField. У методі main класу Main створюється екземпляр класу DerivedClass змінної instance. Далі виводиться значення поля publicField екземпляра instance, яке було змінено у конструкторі DerivedClass. Потім викликається метод show(), який було перевизначено в класі DerivedClass, і він виводить рядок "Method show from derived class".

## Успадкування (#3)



```
package GJP.L5inheritance.ex003_inherit;

// Підклас
public class BaseClass {
    // Атрибути
    public int baseNumber;
    // Метод show()
    public void show() {
        System.out.println(baseNumber);
    }
}

package GJP.L5inheritance.ex003_inherit;

// Підклас
public class DerivedClass extends BaseClass {
    // Конструктор
    public DerivedClass() {
        // Викликає конструктор за замовчуванням
        super();
    }
    // Конструктор за замовчуванням
    public DerivedClass(int number) {
        // Викликає конструктор за замовчуванням
        super();
        baseNumber = number;
    }
    // Конструктор за замовчуванням
    public DerivedClass(int number, int derivedNumber) {
        // Викликає конструктор за замовчуванням
        super(number);
        derivedNumber = derivedNumber;
    }
}

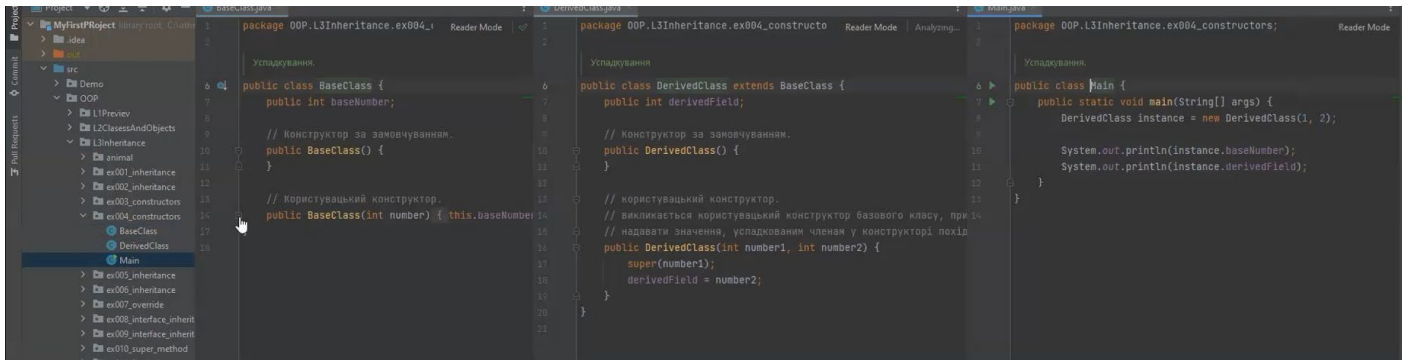
package GJP.L5inheritance.ex003_inherit;

// Підклас
public class Main {
    // Метод main()
    public static void main(String[] args) {
        DerivedClass instance = new DerivedClass();
        System.out.println(instance.baseNumber);
        System.out.println(instance.derivedNumber);
    }
}
```

**Посилання на Oracle:** <https://docs.oracle.com/javase/tutorial/java/inheritance/multipleinheritance.html>

**Опис до прикладу:** У класі BaseClass визначене поле baseNumber, яке використовується як загальне поле для похідних класів. Клас DerivedClass успадковує BaseClass і має додаткове поле derivedField. Він має два конструктори: конструктор за замовчуванням і користувацький конструктор, який приймає два числових параметри. Користувацький конструктор DerivedClass викликає конструктор за замовчуванням BaseClass, щоб ініціалізувати поле baseNumber. Потім він ініціалізує поле derivedField з переданими параметрами. У методі main створюється об'єкт класу DerivedClass з використанням користувацького конструктора і виводяться значення полів baseNumber і derivedField.

## Успадкування (#4)



```
package OOP.L3Inheritance.ex004;

public class BaseClass {
    public int baseNumber;

    // Конструктор за замовчуванням.
    public BaseClass() {}

    // Користувальницький конструктор.
    public BaseClass(int number) { this.baseNumber = number; }
}

package OOP.L3Inheritance.ex004_constructo

public class DerivedClass extends BaseClass {
    public int derivedField;

    // Конструктор за замовчуванням.
    public DerivedClass() {}

    // користувальницький конструктор.
    // викликається користувальницький конструктор базового класу, при чому
    // надавати значення, успадкованим членам у конструкторі похідного класу.
    public DerivedClass(int number1, int number2) {
        super(number1);
        derivedField = number2;
    }
}

package OOP.L3Inheritance.ex004_constructors;

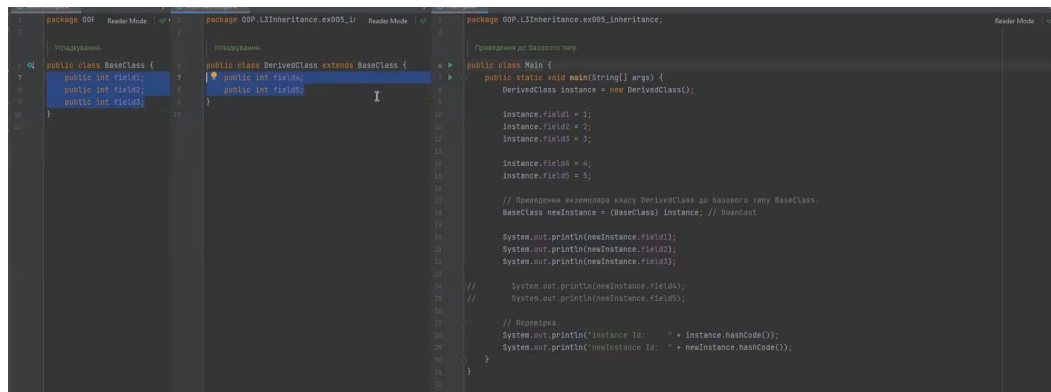
public class Main {
    public static void main(String[] args) {
        DerivedClass instance = new DerivedClass(1, 2);

        System.out.println(instance.baseNumber);
        System.out.println(instance.derivedField);
    }
}
```

Посилання на Oracle: <https://docs.oracle.com/javase/tutorial/java/1andI/super.html>

**Опис до прикладу:** Цей код використовує успадкування для створення похідного класу DerivedClass, який успадковує властивості та методи від базового класу BaseClass. В похідному класі DerivedClass також визначається власне поле derivedField. Конструктор DerivedClass викликає конструктор базового класу BaseClass за допомогою ключового слова super, щоб ініціалізувати успадкований член baseNumber. Код використовує перевизначення методів та полів, поліморфізм та ключове слово super для доступу до членів базового класу з похідного класу.

## Успадкування (#5)



```
package OOP.L3Inheritance.ex005_1;

public class BaseClass {
    public int field1;
    public int field2;
    public int field3;
}

package OOP.L3Inheritance.ex005_1;

public class DerivedClass extends BaseClass {
    public int field4;
    public int field5;
}

package OOP.L3Inheritance.ex005_inheritance;

public class Main {
    public static void main(String[] args) {
        DerivedClass instance = new DerivedClass();

        instance.field1 = 1;
        instance.field2 = 2;
        instance.field3 = 3;

        instance.field4 = 4;
        instance.field5 = 5;

        // Присвоєння екземпляра класу DerivedClass до базового типу BaseClass.
        BaseClass newInstance = (BaseClass) instance; // DownCast

        System.out.println(newInstance.field1);
        System.out.println(newInstance.field2);
        System.out.println(newInstance.field3);

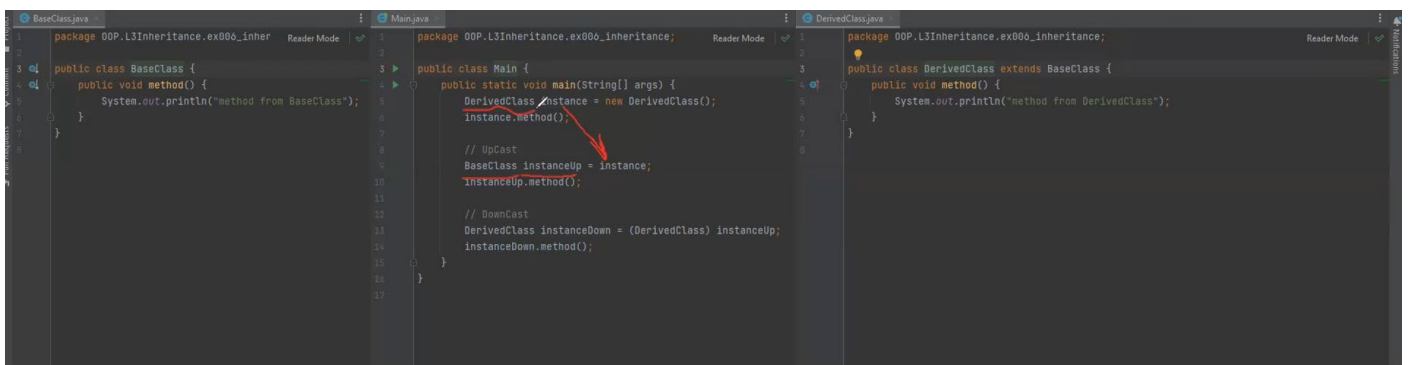
        // System.out.println(newInstance.field4);
        // System.out.println(newInstance.field5);

        // UpCast
        System.out.println("Instance ID: " + instance.hashCode());
        System.out.println("newInstance ID: " + newInstance.hashCode());
    }
}
```

Посилання на Oracle: <https://docs.oracle.com/javase/tutorial/java/1andI/hidevariables.html>

**Опис до прикладу:** Даний код використовує успадкування, де клас DerivedClass успадковує поля (field1, field2, field3) від базового класу BaseClass. В DerivedClass також визначаються додаткові поля (field4, field5). У методі main створюється екземпляр класу DerivedClass з ім'ям instance. Потім значення присвоюються полям цього екземпляра. Далі створюється новий екземпляр класу BaseClass з ім'ям newInstance за допомогою приведення типів (downcasting), де instance приводиться до типу BaseClass. Нарешті, виводяться значення полів newInstance.field1, newInstance.field2 та newInstance.field3. Код закоментований, що стосується полів field4 і field5 класу DerivedClass, оскільки newInstance є екземпляром базового класу і не має доступу до додаткових полів, визначених у похідному класі.

## Успадкування (#6)



```
package OOP.L3Inheritance.ex006_inher

public class BaseClass {
    public void method() {
        System.out.println("method from BaseClass");
    }
}

package OOP.L3Inheritance.ex006_inheritance;

public class Main {
    public static void main(String[] args) {
        DerivedClass instance = new DerivedClass();
        instance.method();

        // UpCast
        BaseClass instanceUp = instance;
        instanceUp.method();

        // DownCast
        DerivedClass instanceDown = (DerivedClass) instanceUp;
        instanceDown.method();
    }
}

package OOP.L3Inheritance.ex006_inheritance;

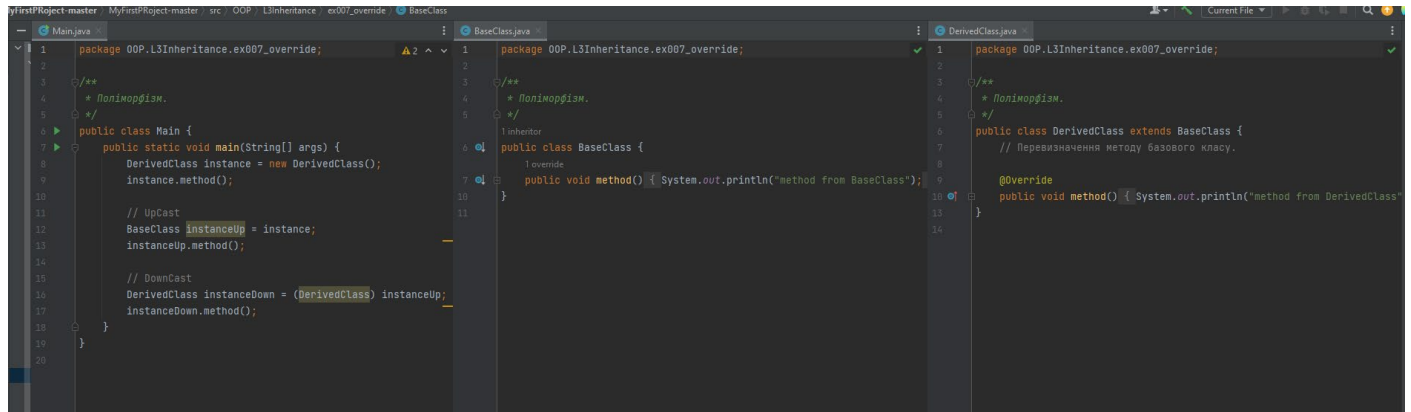
public class DerivedClass extends BaseClass {
    public void method() {
        System.out.println("method from DerivedClass");
    }
}
```

Посилання на Oracle: <https://docs.oracle.com/javase/tutorial/java/1andI/subclasses.html>

**Опис до прикладу:** Код містить два класи: BaseClass і DerivedClass. Клас DerivedClass успадковує клас BaseClass, що означає, що він отримує всі властивості та методи базового класу. У коді визначається метод method() у класі BaseClass, який виводить

рядок "method from BaseClass". Клас DerivedClass перевизначає цей метод і виводить рядок "method from DerivedClass". У методі main() створюється екземпляр класу DerivedClass з іменем instance, і викликається його метод method(), що призводить до виводу рядка "method from DerivedClass". Потім здійснюється використання поліморфізму. Об'єкт instance встановлюється в змінну типу BaseClass з іменем instanceUp за допомогою upcasting. Метод method() викликається з instanceUp, і виводиться рядок "method from DerivedClass". Також здійснюється downcasting, де об'єкт instanceUp приводиться до типу DerivedClass і присвоюється змінній instanceDown. Метод method() викликається з instanceDown, і виводиться рядок "method from DerivedClass".

## Успадкування (#7)



**Посилання на Oracle:** <https://docs.oracle.com/javase/tutorial/java/land/polymorphism.html>

**Опис до прикладу:** Код містить два класи: BaseClass (базовий клас) і DerivedClass (похідний клас), який успадковує властивості та методи від BaseClass. В BaseClass є публічний метод method(), який виводить рядок "method from BaseClass". У DerivedClass перевизначений метод method(), який виводить рядок "method from DerivedClass". Це приклад поліморфізму, де метод з похідного класу перевизначається, щоб мати іншу реалізацію, ніж метод у базовому класі. В Main класі створюється об'єкт DerivedClass, і викликається метод method() для цього об'єкта, що виводить рядок "method from DerivedClass". Далі відбувається використання UpCast та DownCast для створення об'єктів типу BaseClass та DerivedClass відповідно і виклику методу method() для цих об'єктів.