Here are examples of **Java Generics**, **Method Overloading**, and **Method Overriding** based on Java 8.

**1. Generics Example**

Generics allow creating classes, interfaces, and methods with type parameters.

**Example: Generic Class and Generic Method**

class Box<T> { // Generic class

private T value;

public void setValue(T value) {

this.value = value;

}

public T getValue() {

return value;

}

}

public class GenericsExample {

public static void main(String[] args) {

Box<Integer> intBox = new Box<>();

intBox.setValue(100);

System.out.println("Integer Value: " + intBox.getValue());

Box<String> strBox = new Box<>();

strBox.setValue("Hello");

System.out.println("String Value: " + strBox.getValue());

}

}

**2. Method Overloading with Generics**

Method Overloading allows multiple methods with the same name but different parameters.

**Example: Overloaded Methods**

class OverloadingExample {

public void display(Integer num) {

System.out.println("Integer: " + num);

}

public void display(String str) {

System.out.println("String: " + str);

}

public <T> void display(T value) { // Generic method

System.out.println("Generic: " + value);

}

public static void main(String[] args) {

OverloadingExample obj = new OverloadingExample();

obj.display(10);

obj.display("Java");

obj.display(10.5); // Uses the generic method

}

}

**3. Method Overriding with Generics**

Method Overriding occurs when a subclass provides a specific implementation of a method already defined in its parent class.

**Example: Overriding a Generic Method**

class Parent<T> {

public void show(T data) {

System.out.println("Parent class: " + data);

}

}

class Child extends Parent<String> {

@Override

public void show(String data) { // Overriding generic method

System.out.println("Child class: " + data);

}

}

public class OverridingExample {

public static void main(String[] args) {

Parent<Integer> p1 = new Parent<>();

p1.show(100); // Parent method with Integer

Child child = new Child();

child.show("Java 8"); // Overridden method in Child class

}

}