Task 3: Reflection API
Use reflection to inspect a class's methods, fields, and constructors, and modify the access level of a private field, setting its value during runtime

```
package com.wipro.model;

import java.lang.reflect.*;

class MyClass {

private int privateField;

public MyClass(int privateField) {

this.privateField = privateField;

}

private void privateMethod() {

System.out.println("Inside privateMethod");

}

public void publicMethod() {

System.out.println("Inside publicMethod");

}

}

public class ReflectionExample {

public static void main(String[] args)

throws NoSuchFieldException,IllegalAccessException,

NoSuchMethodException,InvocationTargetException,InstantiationException {

Class<?> clazz=MyClass.class;

System.out.println("Fields:");

Field[] fields=clazz.getDeclaredFields();

for (Field f:fields) {

System.out.println(f.getName()+"(Type: "+f.getType()+", Modifier: "+

Modifier.toString(f.getModifiers()) + ")");

}

Field privateField = clazz.getDeclaredField("privateField");

privateField.setAccessible(true);

MyClass i = (MyClass) clazz.getDeclaredConstructor(int.class).newInstance(10);
privateField.setInt(i, 20);
System.out.println("Modified privateField value: " + privateField.getInt(i));
System.out.println("\nMethods:");
Method[] methods=clazz.getDeclaredMethods();
for (Method m:methods) {
System.out.println(m.getName()+" (Return type: "+
m.getReturnType()+", Modifier: "+Modifier.toString(m.getModifiers())+")");
}
Method privateMethod=clazz.getDeclaredMethod("privateMethod");
privateMethod.setAccessible(true);
privateMethod.invoke(i);
Method publicMethod=clazz.getDeclaredMethod("publicMethod");
publicMethod.invoke(i);
}
}
```