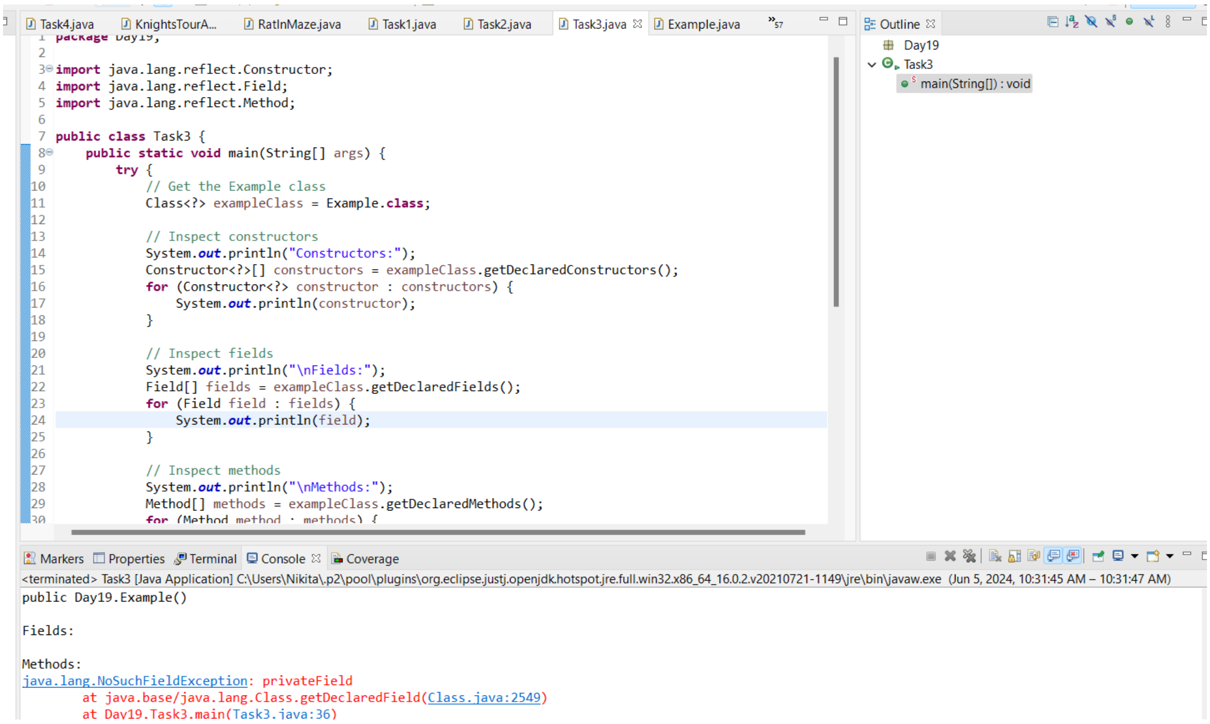
Task 3: Reflection API

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 Day19;

import java.lang.reflect.Constructor;
import java.lang.reflect.Field;
import java.lang.reflect.Method;

public class Task3 {
    public static void main(String[] args) {
        try {
            // Get the Example class
            Class<?> exampleClass = Example.class;

            // Inspect constructors
            System.out.println("Constructors:");
            Constructor<?>[] constructors = exampleClass.getDeclaredConstructors();
            for (Constructor<?> constructor : constructors) {
                System.out.println(constructor);
            }

            // Inspect fields
            System.out.println("\nFields:");
            Field[] fields = exampleClass.getDeclaredFields();
            for (Field field : fields) {
                System.out.println(field);
            }

            // Inspect methods
            System.out.println("\nMethods:");
            Method[] methods = exampleClass.getDeclaredMethods();
            for (Method method : methods) {
                System.out.println(method);
            }

            // Modify the access level of the private field and set its value
            Example exampleInstance = new Example();
            Field privateField = exampleClass.getDeclaredField("privateField");
            privateField.setAccessible(true);
            privateField.set(exampleInstance, "Modified Value");

            // Verify the private field value
            System.out.println("\nModified private field value: " +
privateField.get(exampleInstance));

            // Invoke the private method
            Method privateMethod = exampleClass.getDeclaredMethod("privateMethod");
            privateMethod.setAccessible(true);
            privateMethod.invoke(exampleInstance);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```



```java
package Day19;

import java.lang.reflect.Constructor;
import java.lang.reflect.Field;
import java.lang.reflect.Method;

public class Task3 {
    public static void main(String[] args) {
        try {
            // Get the Example class
            Class<?> exampleClass = Example.class;

            // Inspect constructors
            System.out.println("Constructors:");
            Constructor<?>[] constructors = exampleClass.getDeclaredConstructors();
            for (Constructor<?> constructor : constructors) {
                System.out.println(constructor);
            }

            // Inspect fields
            System.out.println("\nFields:");
            Field[] fields = exampleClass.getDeclaredFields();
            for (Field field : fields) {
                System.out.println(field);
            }

            // Inspect methods
            System.out.println("\nMethods:");
            Method[] methods = exampleClass.getDeclaredMethods();
            for (Method method : methods) {
```

<terminated> Task3 [Java Application] C:\Users\Nikita\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe (Jun 5, 2024, 10:31:45 AM – 10:31:47 AM)

```
public Day19.Example()

Fields:

Methods:
java.lang.NoSuchFieldException: privateField
	at java.base/java.lang.Class.getDeclaredField(Class.java:2549)
	at Day19.Task3.main(Task3.java:36)
```