Taller 08

Refactoring

Luis Alejandro Donoso Bravo

Jeremy Joshua Martínez Guamán

Fausto Sebastián Jacome Andreotti

Contenido

[CODE SMELLS 1](#_Toc92373875)

[Large class 1](#_Toc92373876)

[Data clump 3](#_Toc92373877)

[Feature Envy 4](#_Toc92373878)

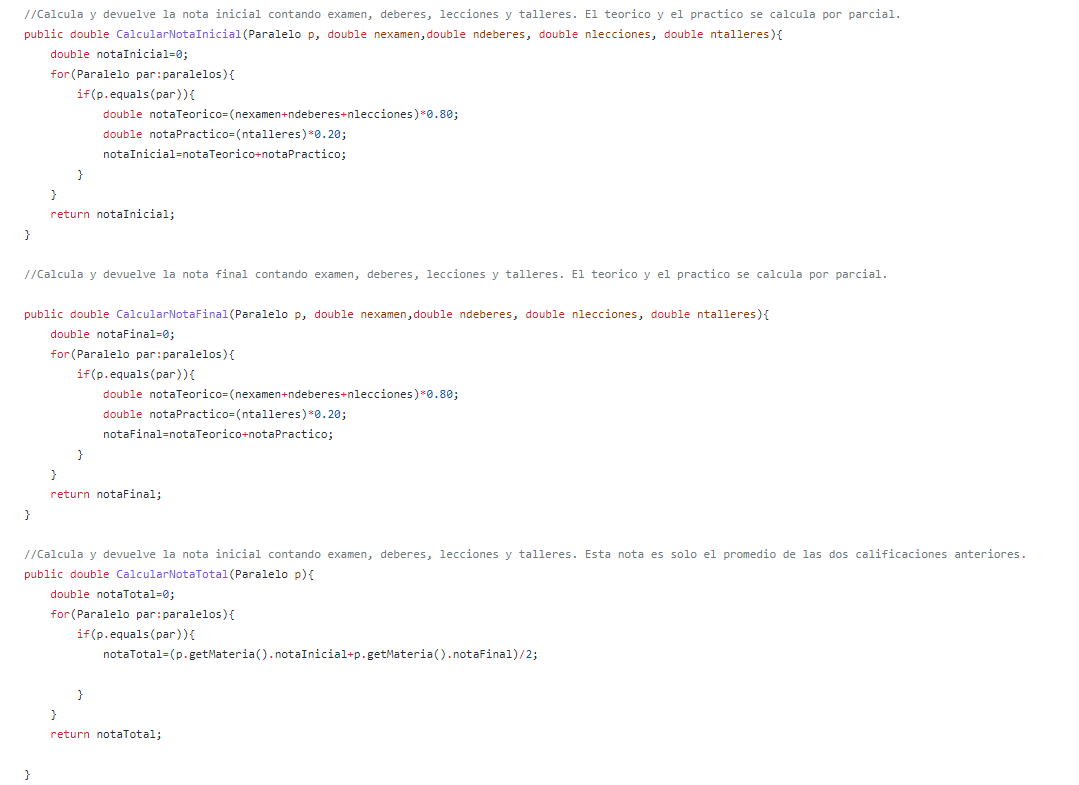
[Lazy Class 5](#_Toc92373879)

[Duplicate Code 6](#_Toc92373880)

[Comments 7](#_Toc92373881)

# CODE SMELLS

## Large class

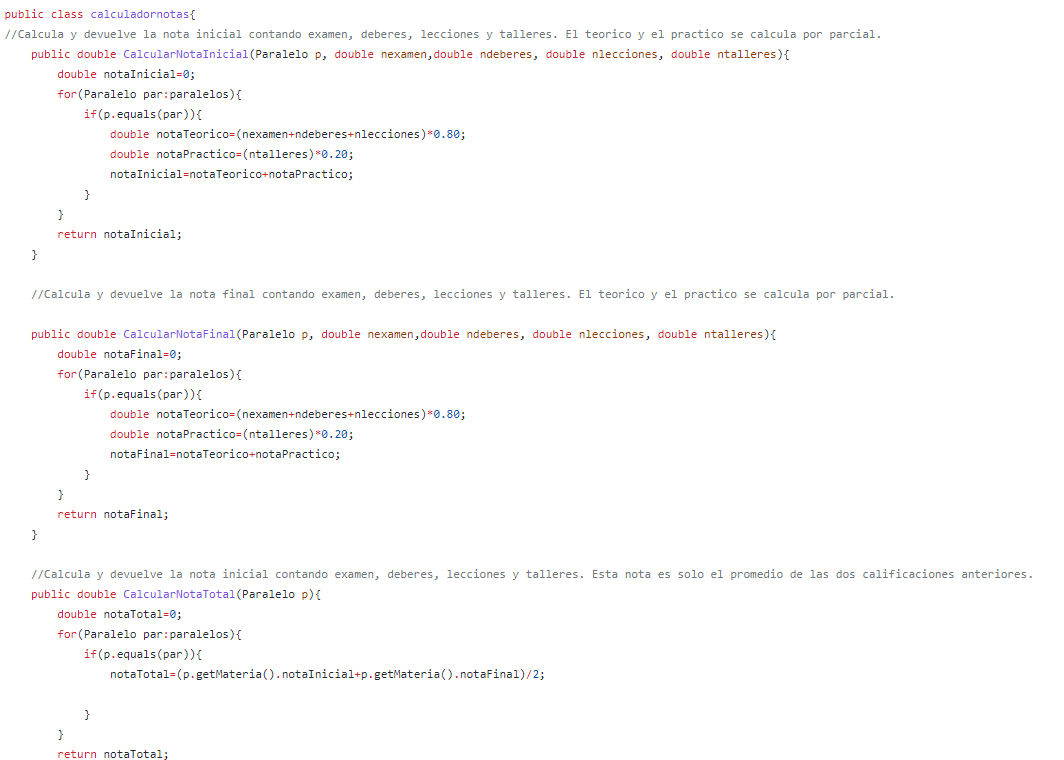


**Problema:** la clase estudiantes cuenta con muchos métodos que pueden ser derivados a otras funcionalidades.

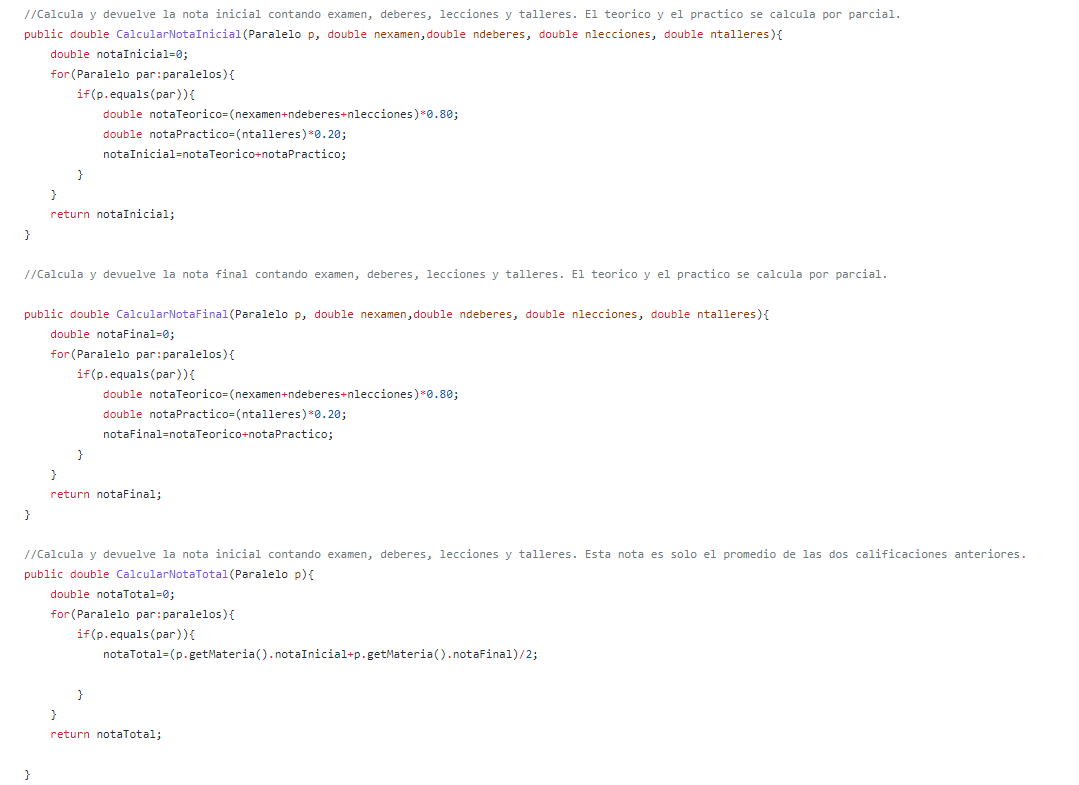
**Consecuencias:** la clase estudiante puede seguir creciendo cuando se busque agregar nuevas funcionalidades haciéndola complicada y difícil de comprender

**Solución:** se puede separar los métodos de cálculo de notas en una clase aparte que se enfoque en esa característica del programa

Usaríamos la técnica de extract class



## Data clump

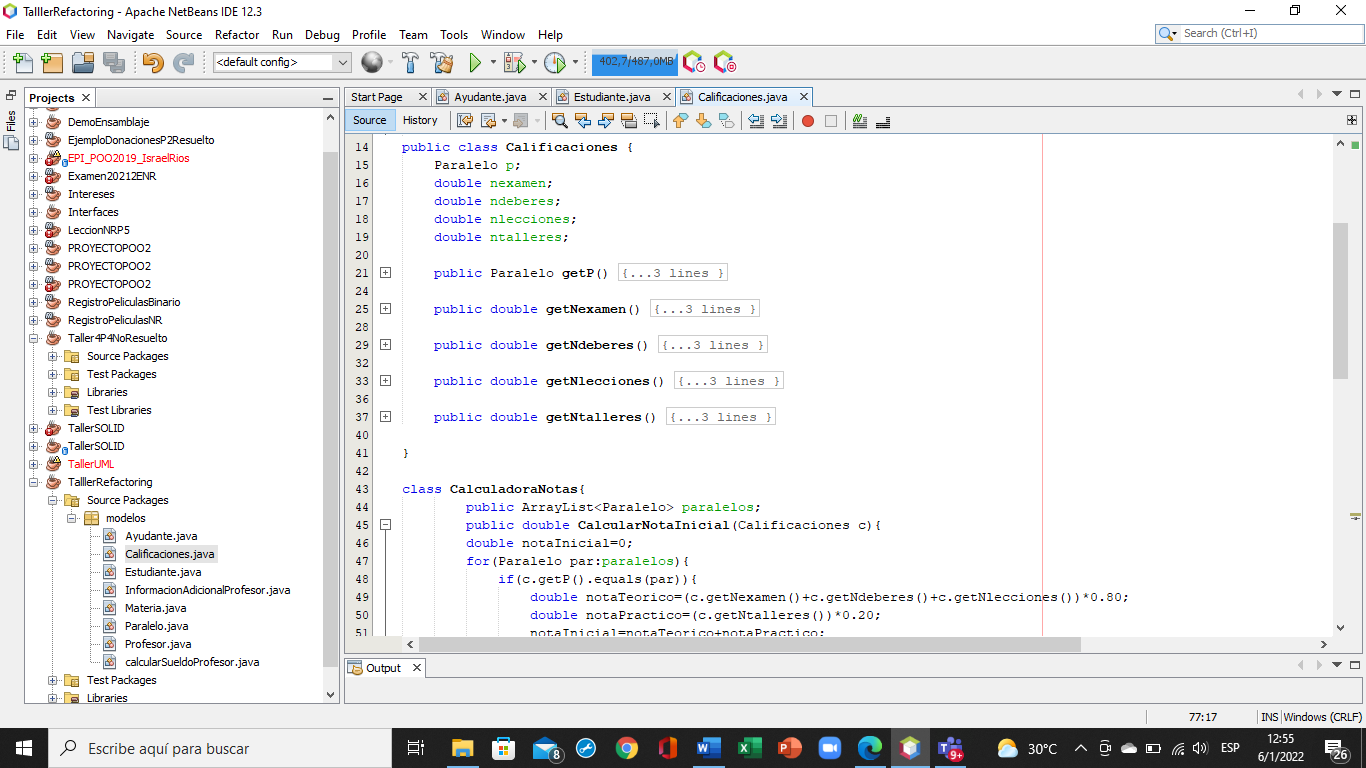
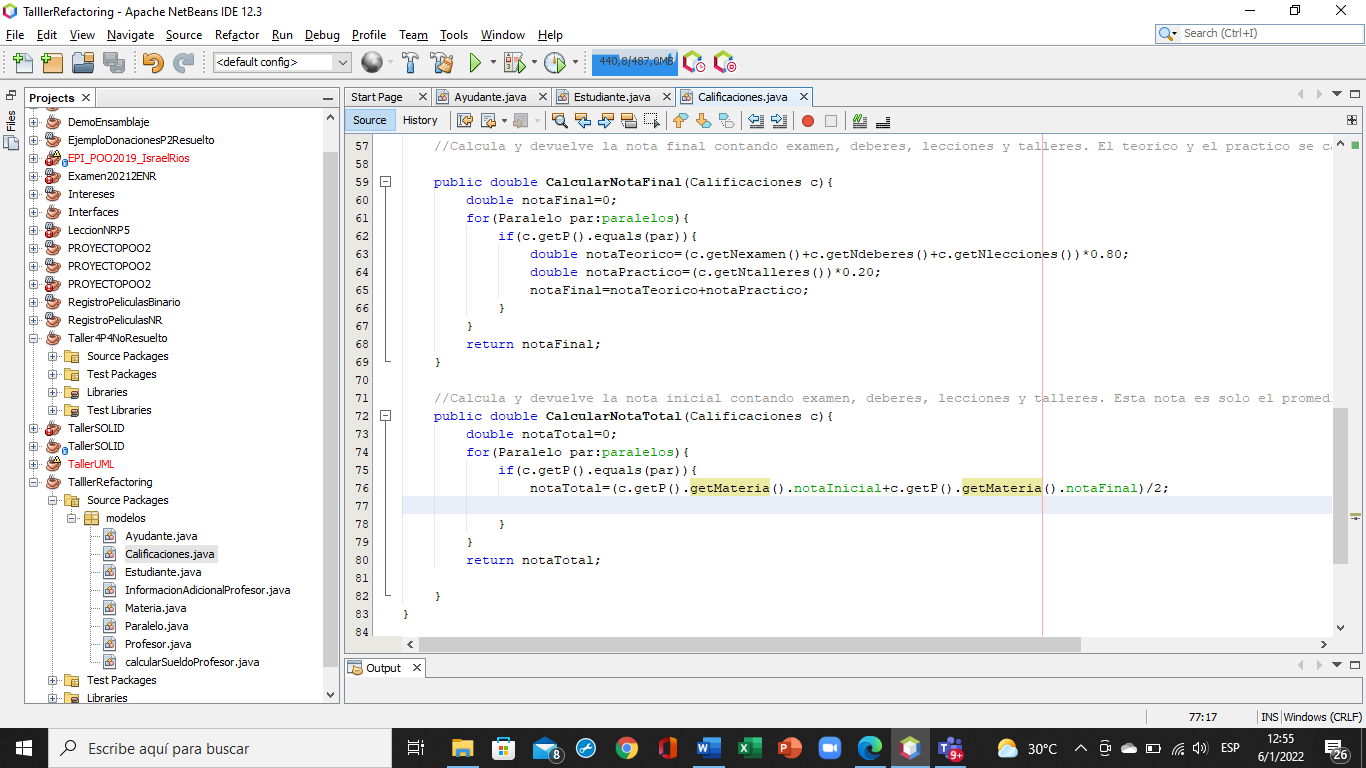


Problema: Luego de haber separado los métodos de estudiante en su propia clase nos damos cuenta de que poseen muchos parámetros

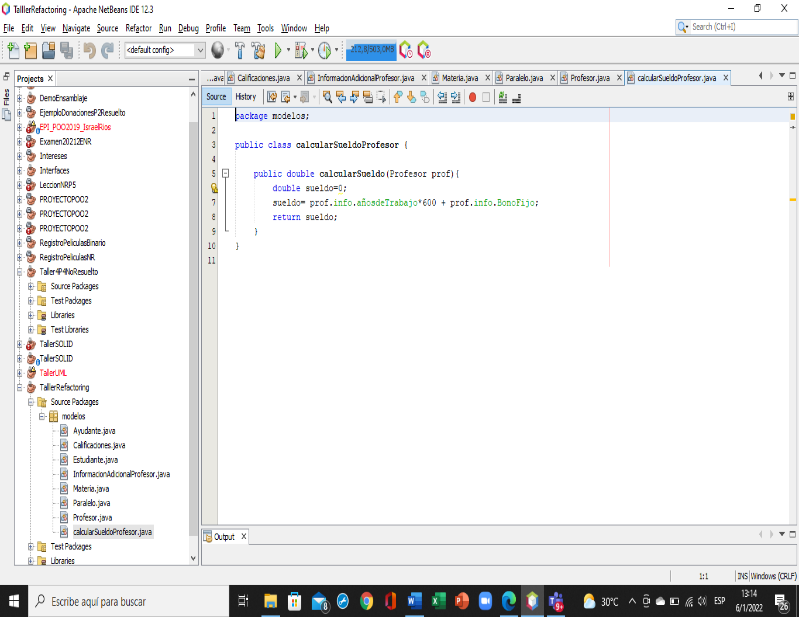
Consecuencias: la clase puede seguir creciendo cuando se busque agregar nuevas funcionalidades haciéndola difícil de entender

Solución: Como ya tenemos la clase calificaciones los métodos de la clase calculadora de notas solo recibirían un objeto calificación

Usamos la técnica de extract class

## Feature Envy

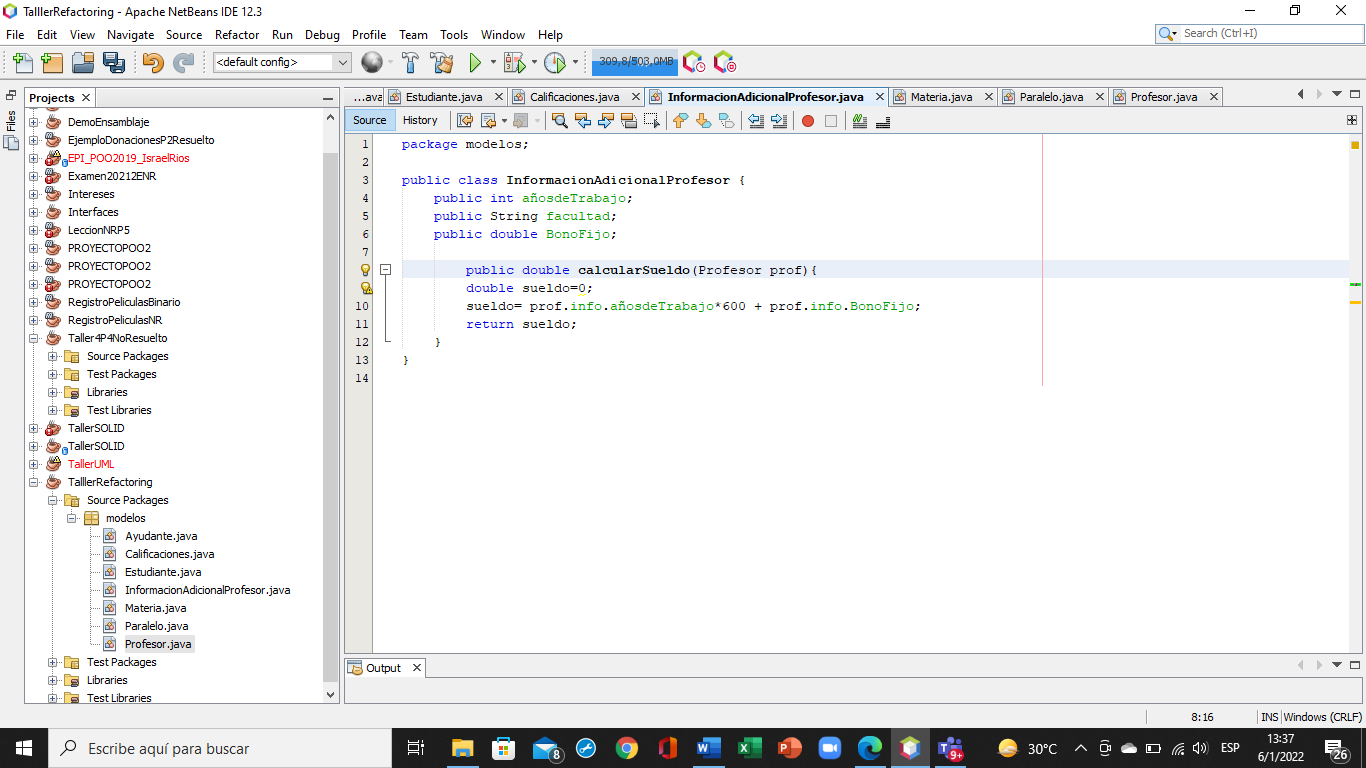


Problema: el método calcular sueldo de la clase calcular sueldo profesor usa la mayoría de los atributos de la clase Información Adicional Profesor

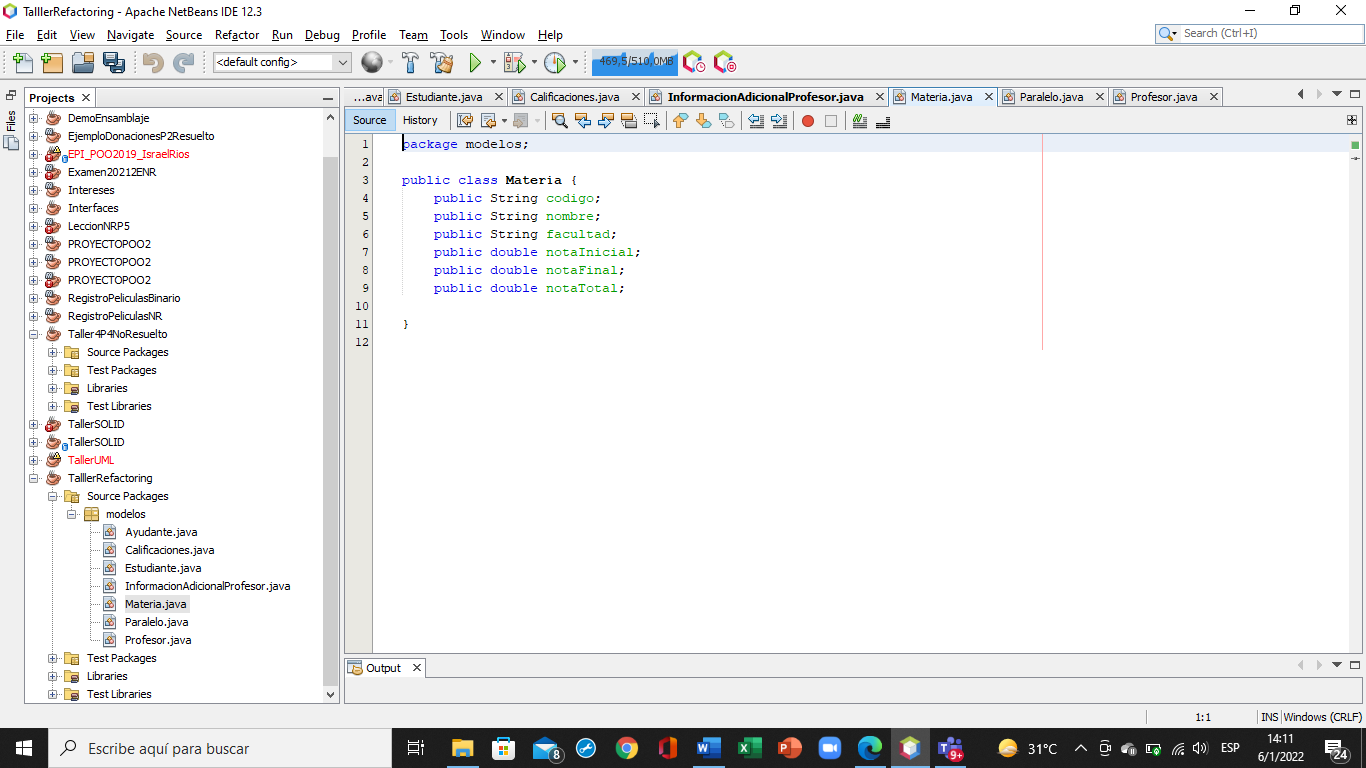
Consecuencias: se puede confundir la funcionalidad principal de la clase por estar accediendo a atributos de otra clase

Solución: mover el método a la clase Información Adicional Profesor y eliminamos la clase de calcular sueldo profesor

Usamos las técnicas de move method e Inline class



## Lazy Class

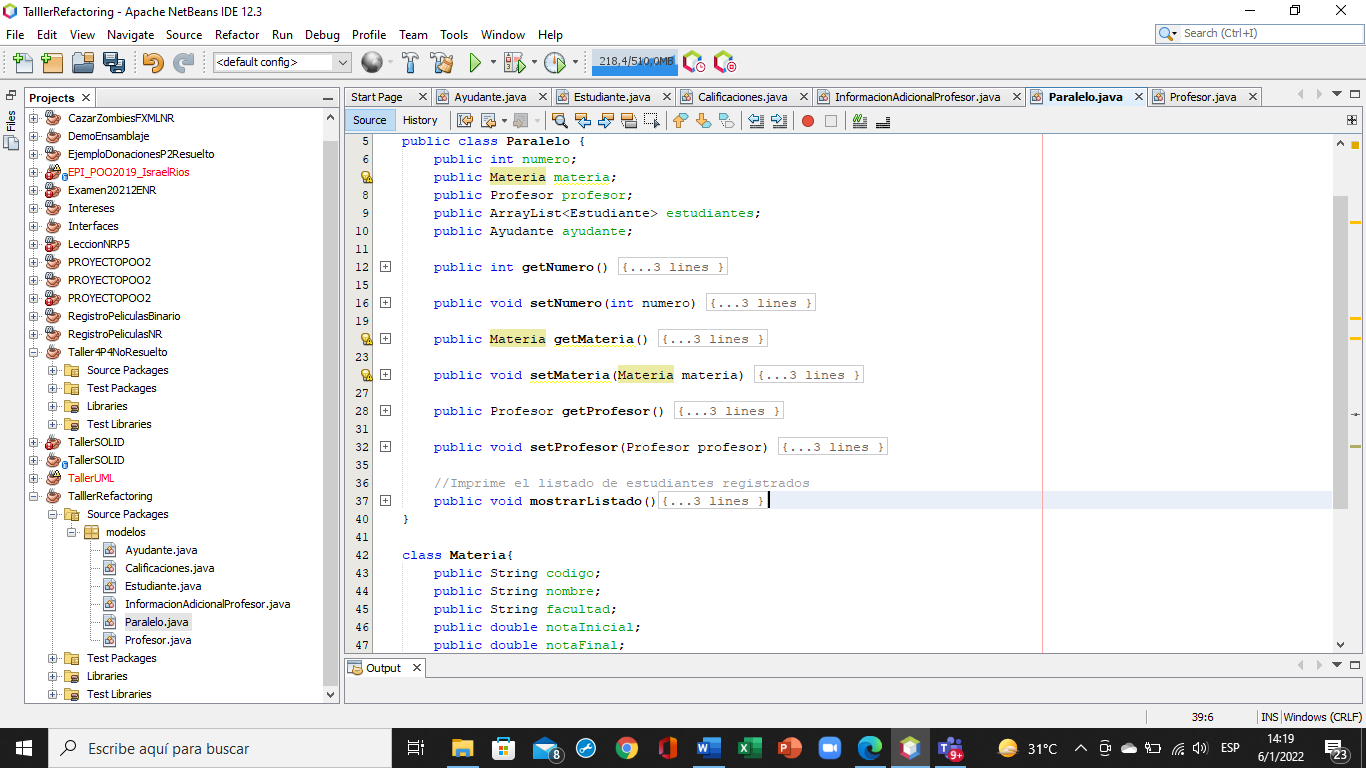


Problema: la clase materia no tiene una función principal si no secundaria en el programa

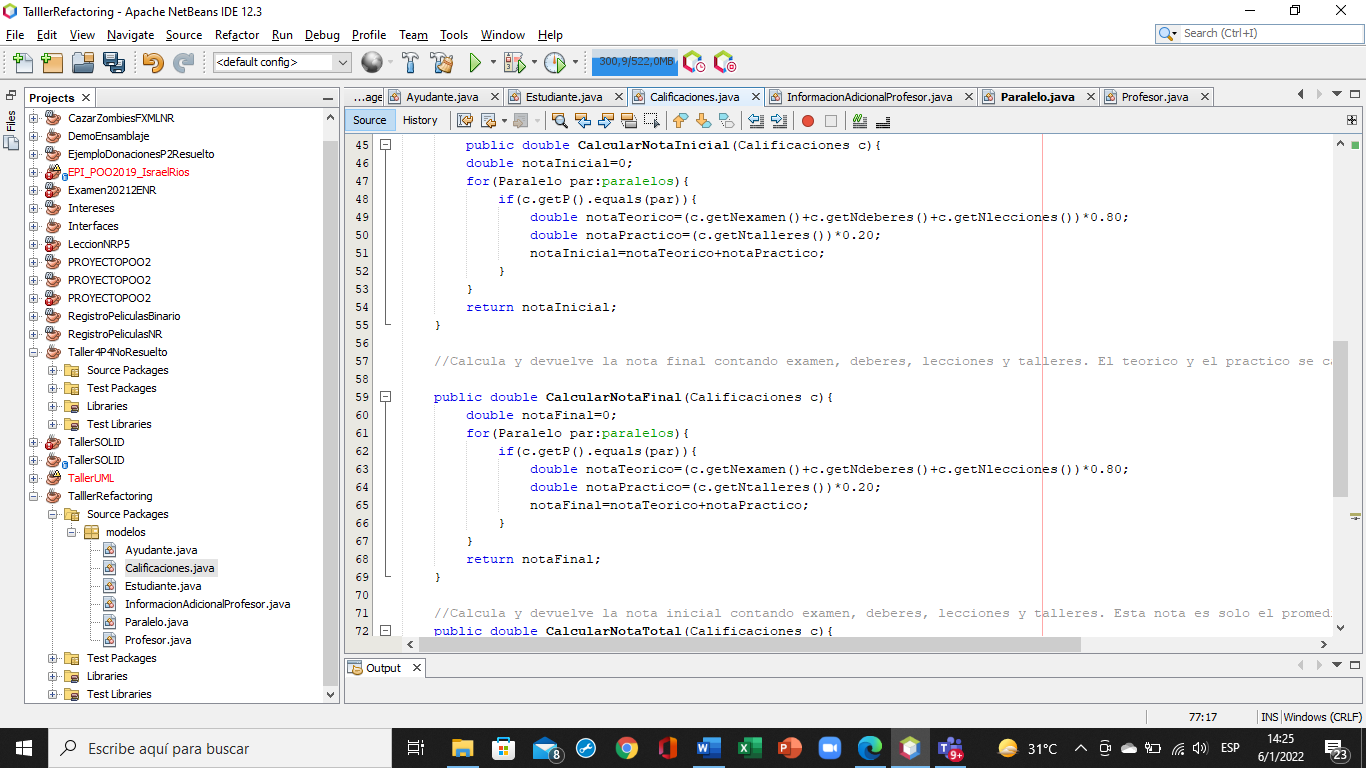
Consecuencias: puede generar confusiones en el proyecto

Solución: mover esa clase como inner class a la clase paralelo

Usamos la técnica de inline class



## Duplicate Code

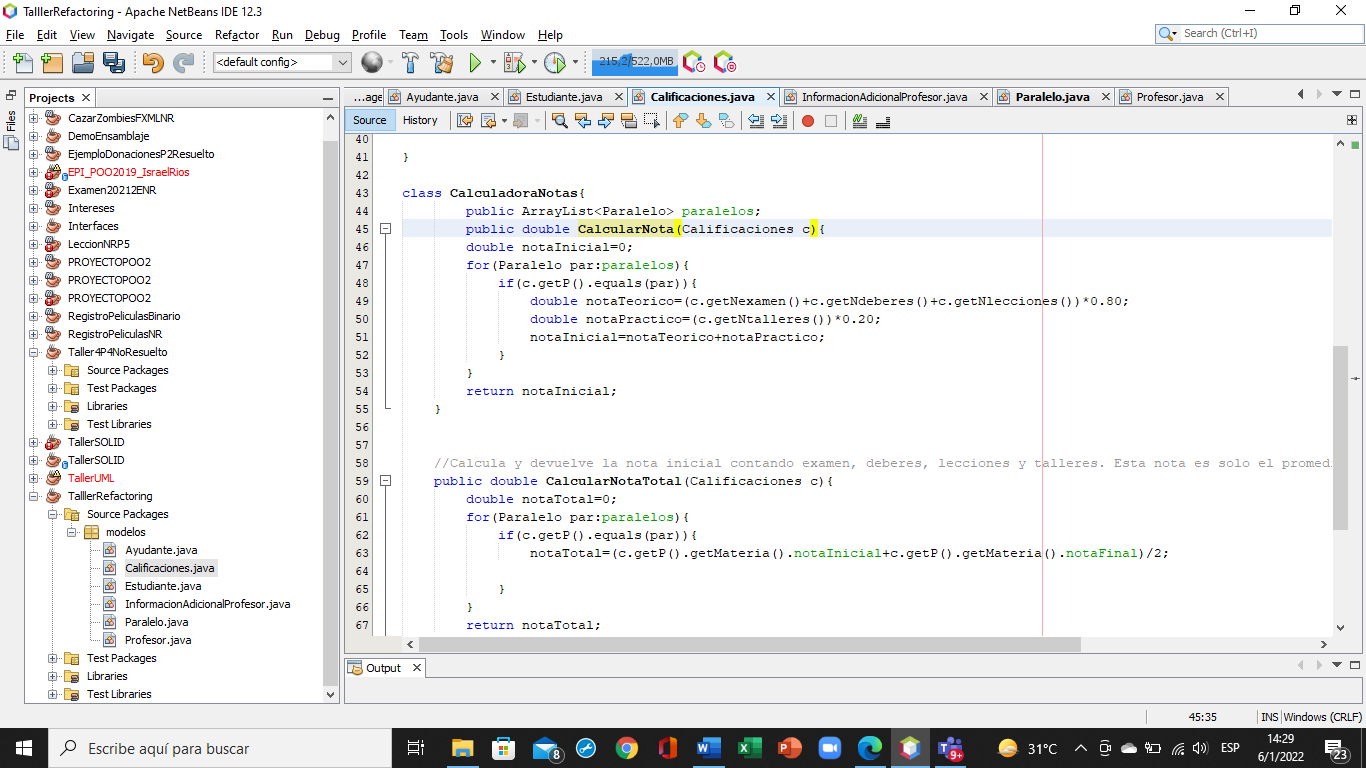


Problema: el cuerpo de los métodos calcular nota inicial y nota final es el mismo

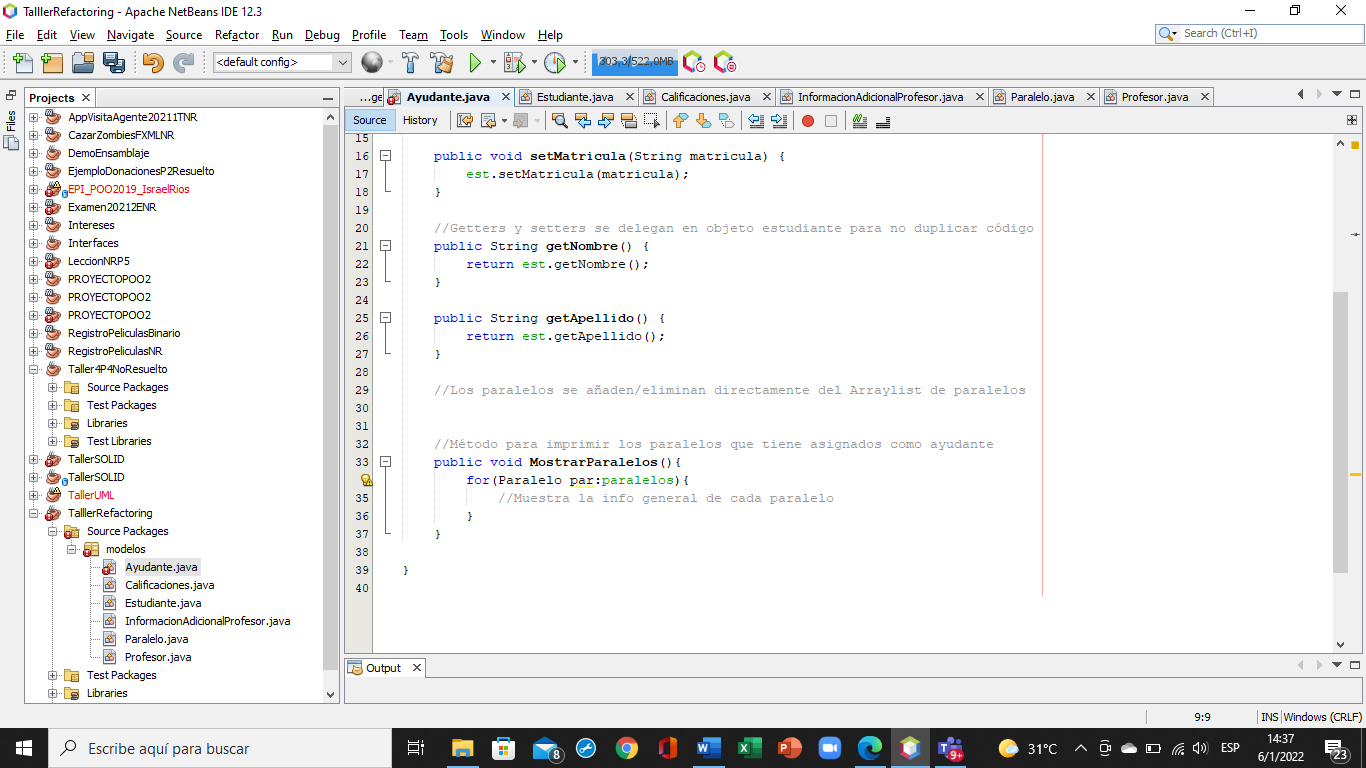
Consecuencias: código repetido que dificulta la comprensión del programa

Solución: unir los 2 metodos y hacerlo 1 solo

Usamos la técnica de extract method



## Comments



Problema: el método Mostrar Paralelos esta comentado 2 veces lo cual no es necesario

Consecuencias: Hace que el código dentro del método se complique al entender puesto que basta con decir lo que realizara el método 1 vez

Solución: des comentar el texto que está dentro del método

Usamos la técnica de introduce assertion para completar el código

