Taller 8 de Diseño de Software

Refactorización

Integrantes:

Mario Steeven Cueva

Contenido

[Duplicate Code 3](#_Toc122645722)

[Lazy Class 4](#_Toc122645723)

[Long Method 5](#_Toc122645724)

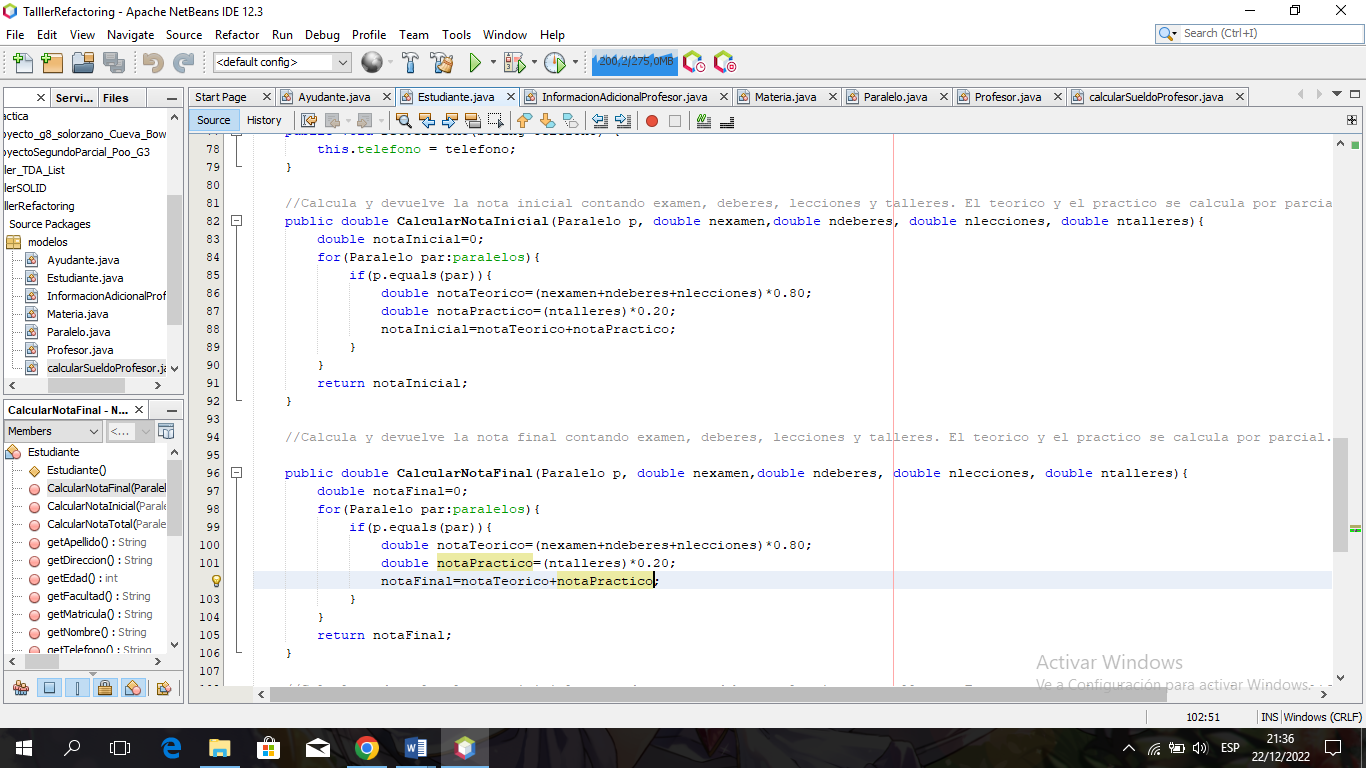
[Inappropriate Intimacy 6](#_Toc122645725)

[Comments 7](#_Toc122645726)

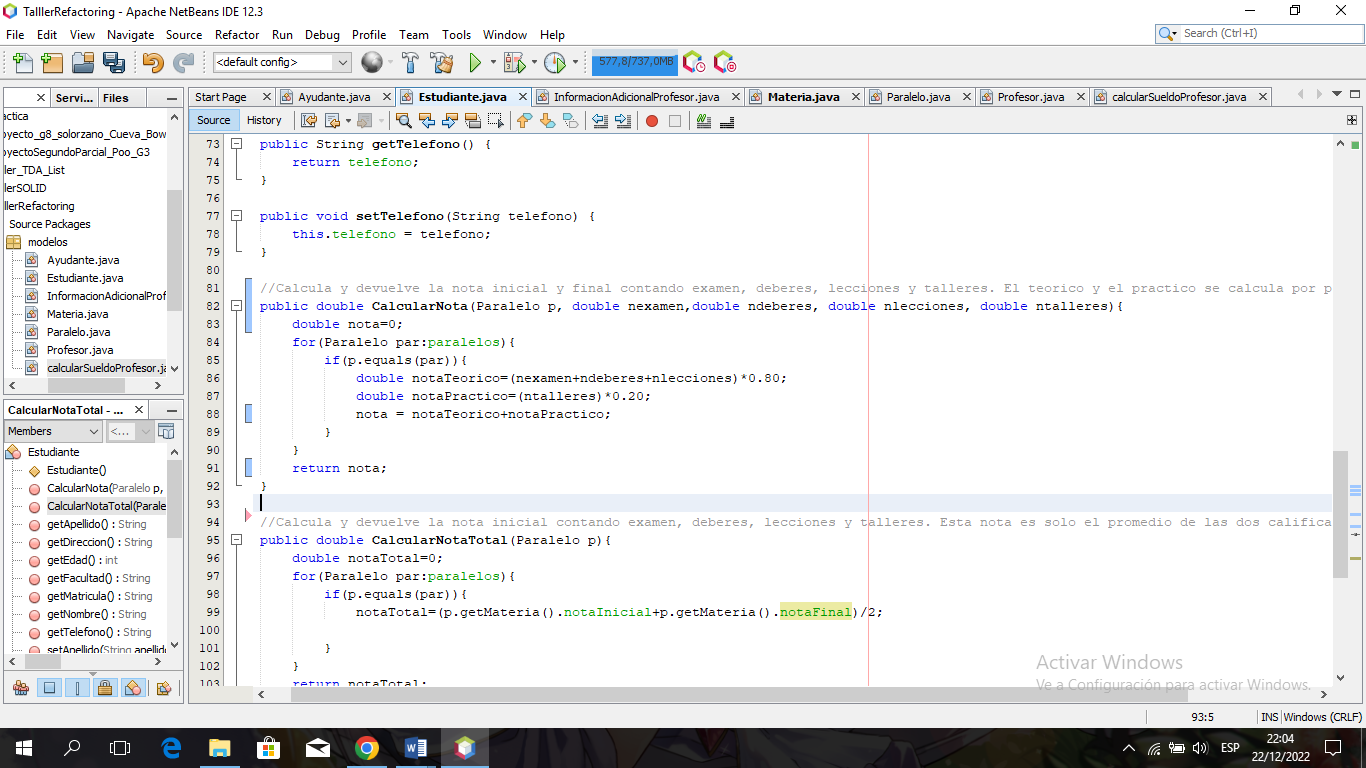
# Duplicate Code

En la parte de calcular notas iniciales y finales se repite el mismo código para ambos método por la tanto para simplificarlo puedo usar Extract Method para agrupar y decir que solo calcula nota y luego asigno ese valor.

Código Antes



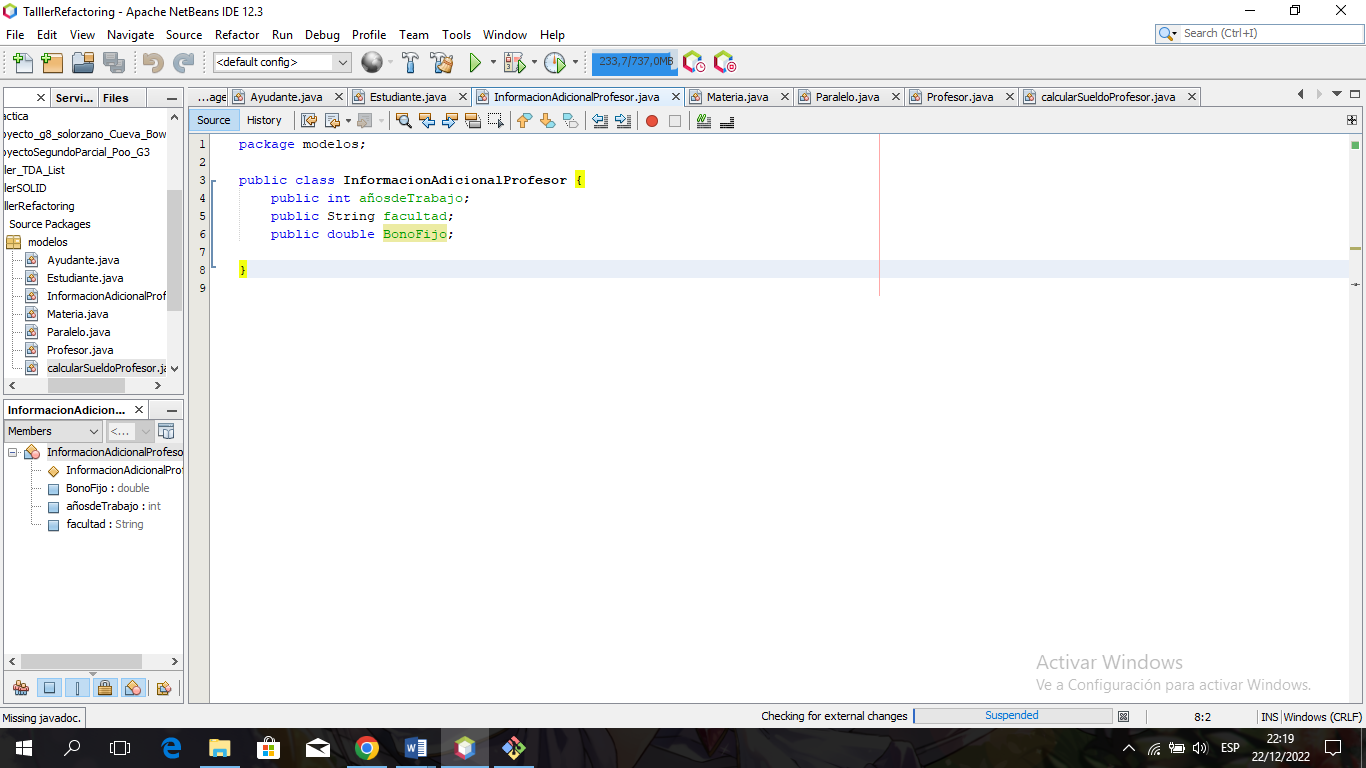
Código después



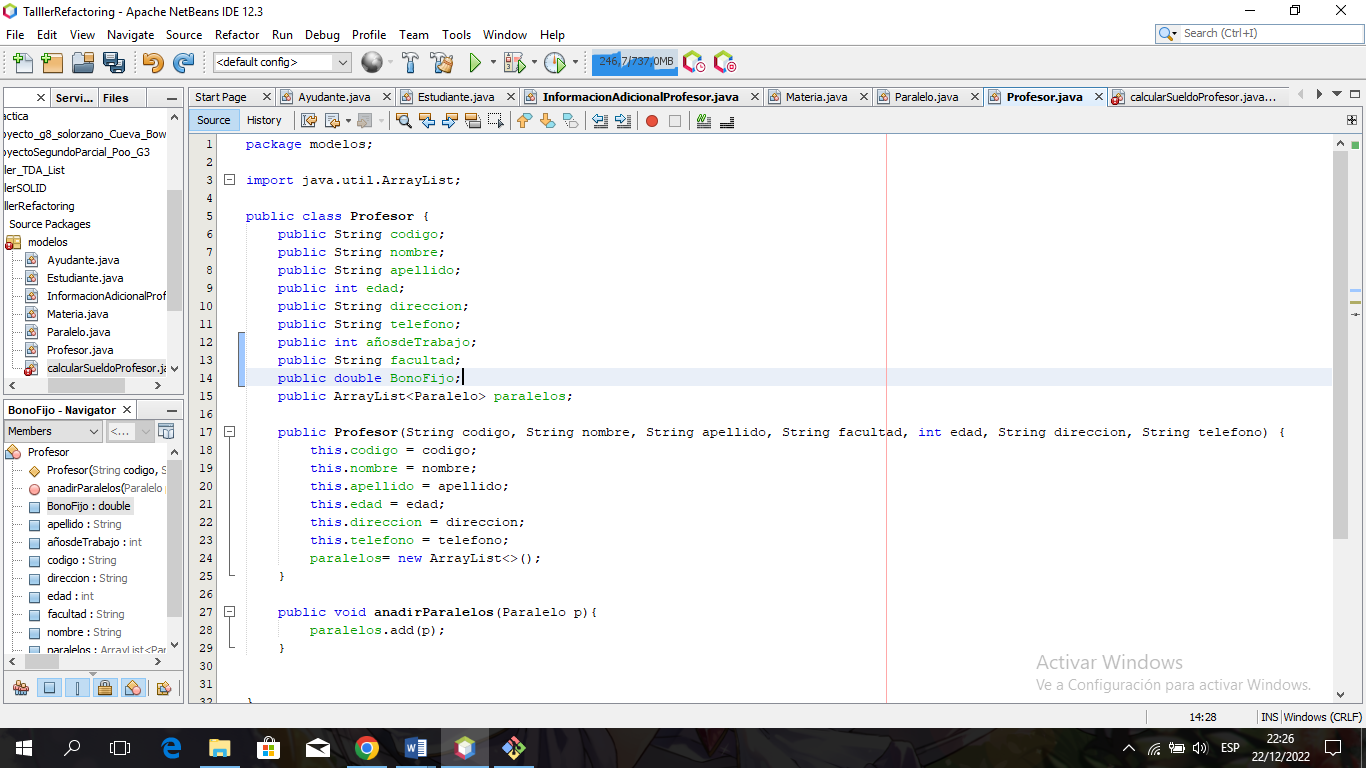
# [Lazy Class](https://refactoring.guru/smells/lazy-class)

La clase Informacion adicinal del profesor seria una lazy class ya que no hace nada y para lo unico que se lo usa es para calcular el sueldo del profesor entonces para refactorizarlo deberia usar Inline Class y mover esos campos a la clase profesor

Codigo antes



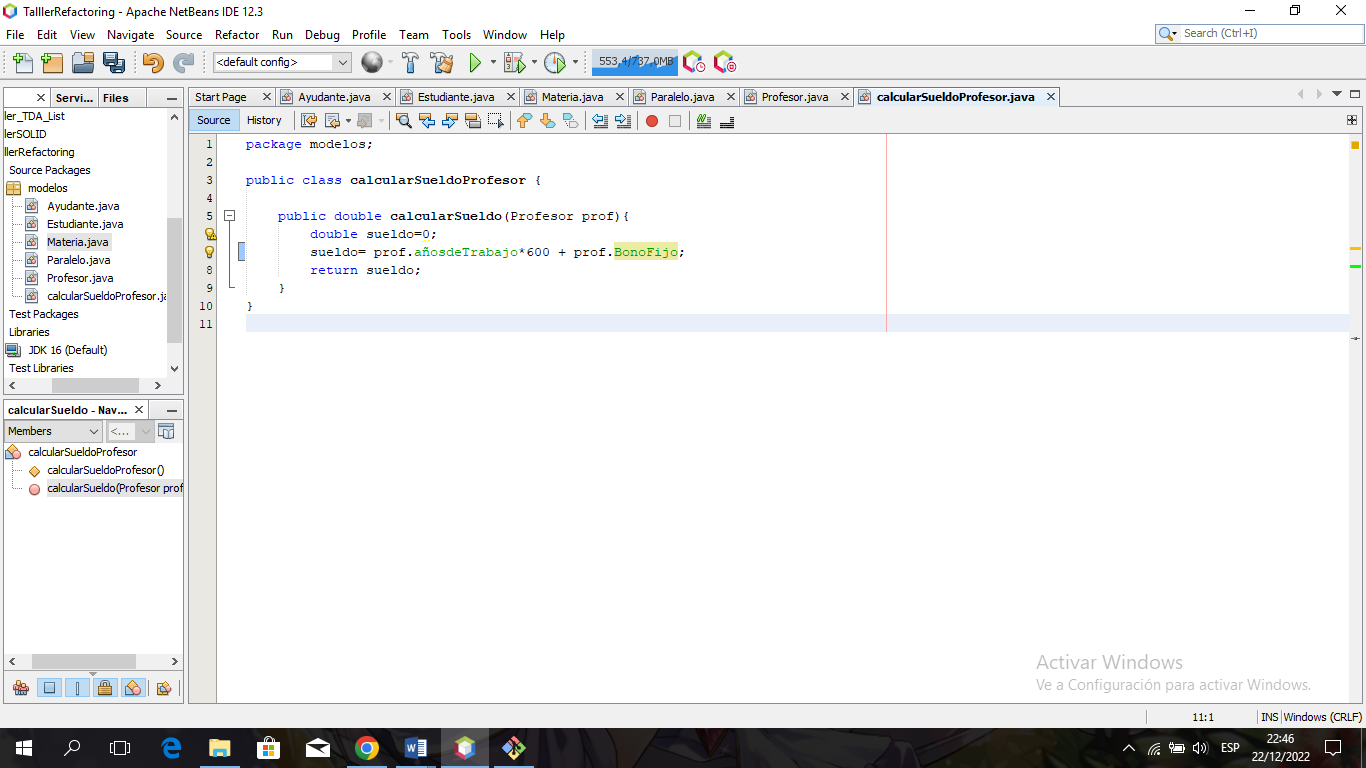
Código Después



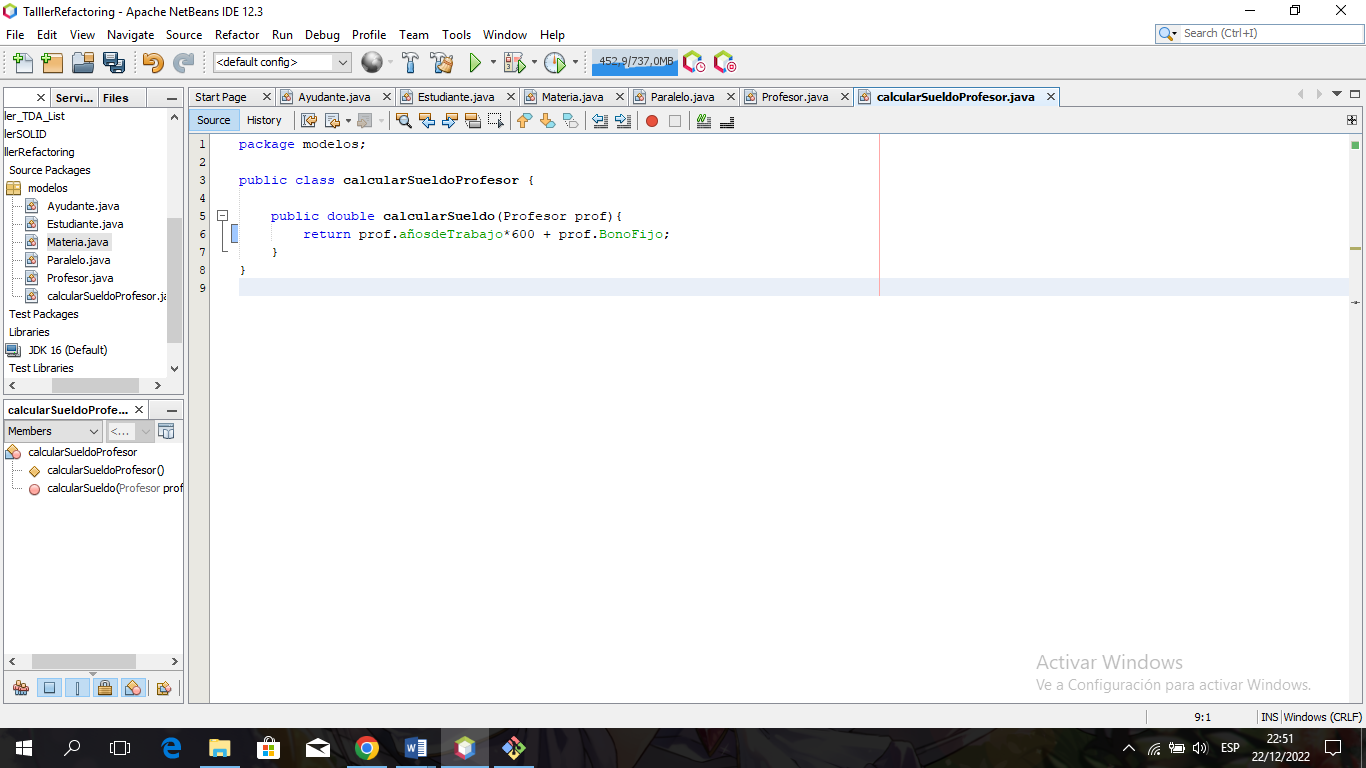
# Long Method

En la clase calcular sueldo Profesor tiene variables temporales que pueden refactorizada utilizando Inline Temp, porque el valor el cual se le está asignando a esa variable temporal puede ser remplazado en el return por ese mismo valor.

Codigo Antes



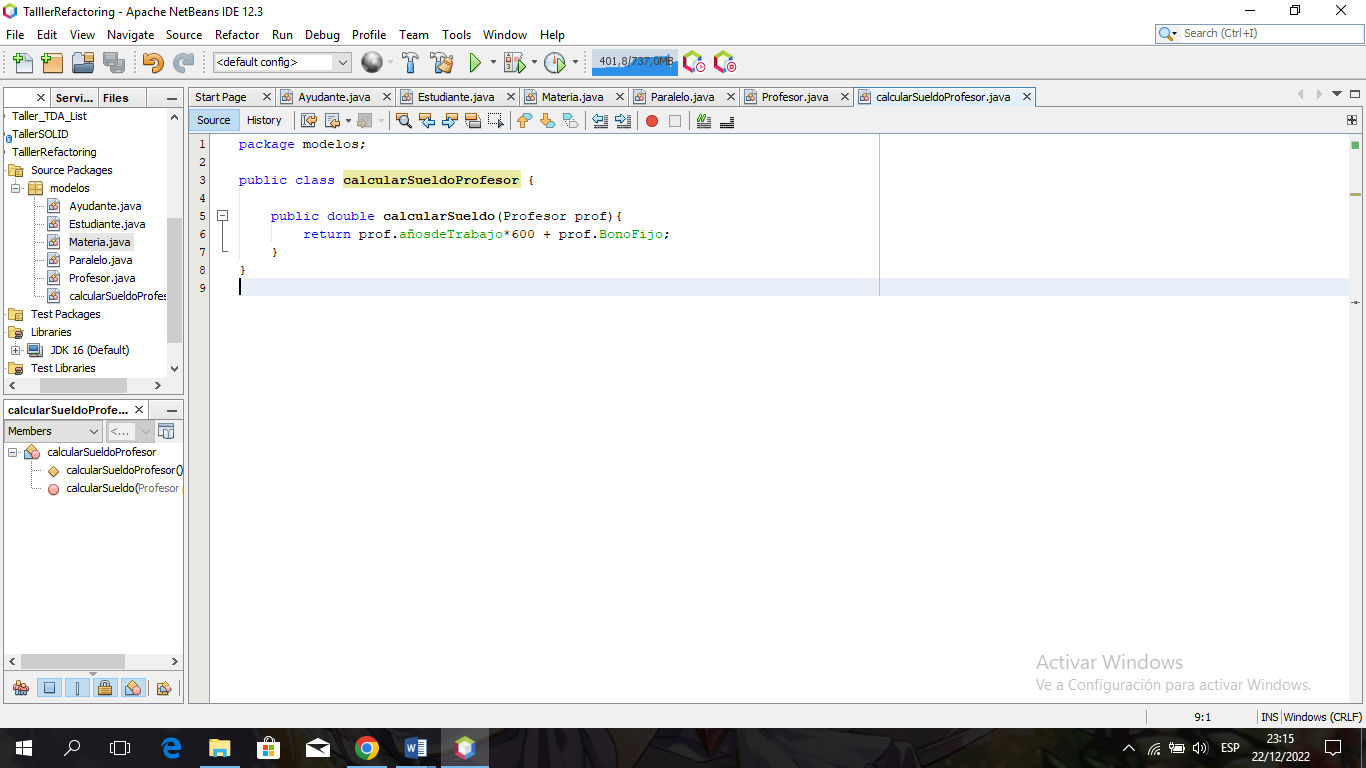
Código Después



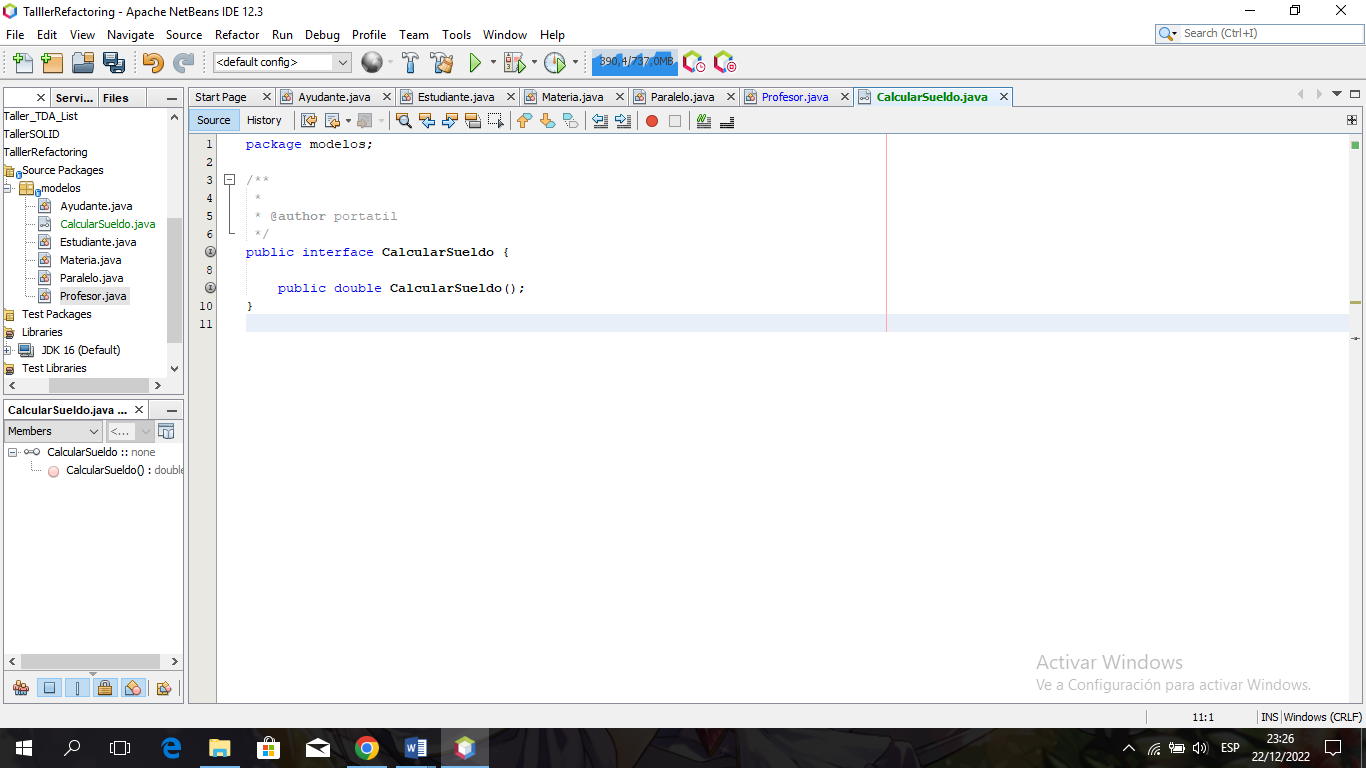
# Inappropriate Intimacy

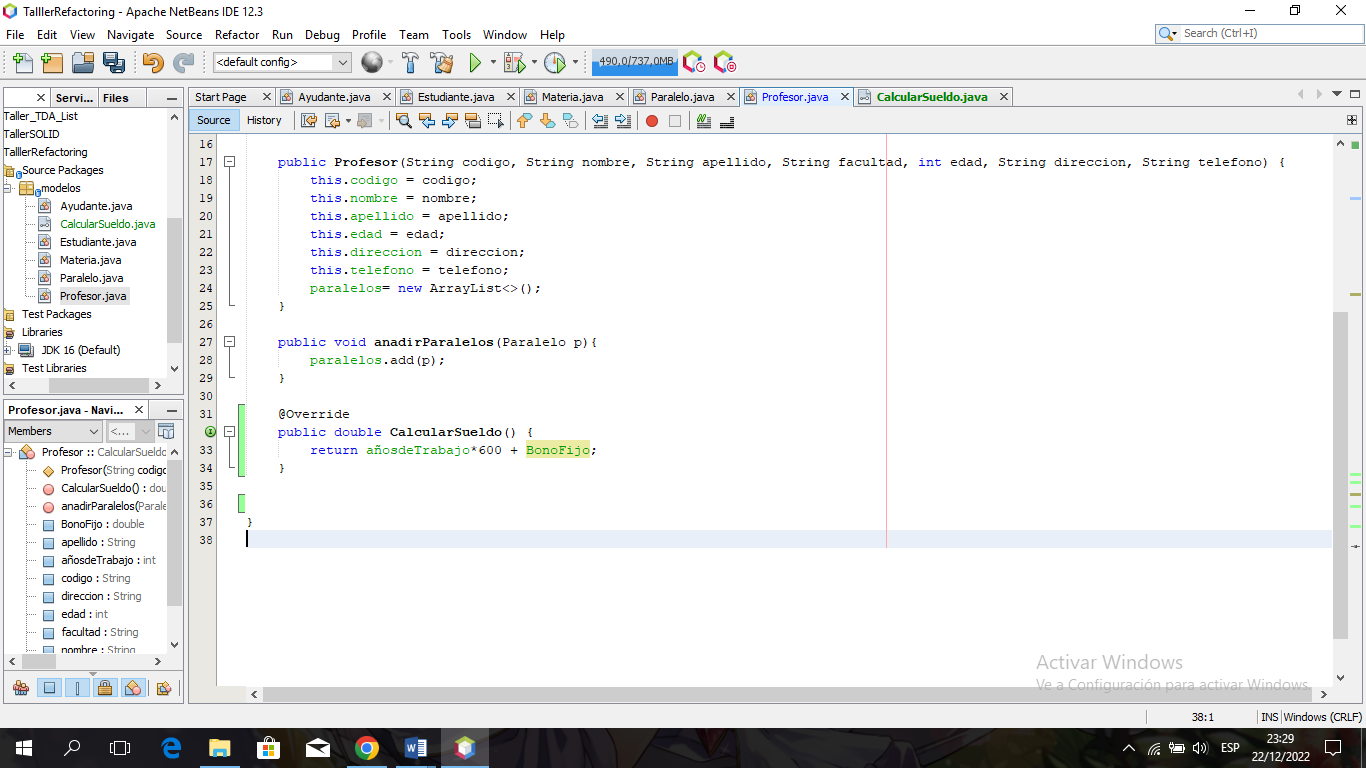
* La clase calcular Sueldo Profesor usa campos que pertenecen a la clase profesor por lo que se podría refactorizar utilizaría [Extract Interface](https://refactoring.guru/extract-interface) que tenga un método para calcular sueldo y que lo implemente la clase profesor. Así implementando esta interfaz también podría usarla en ayudante si se lo requiere.

Código antes



Código después

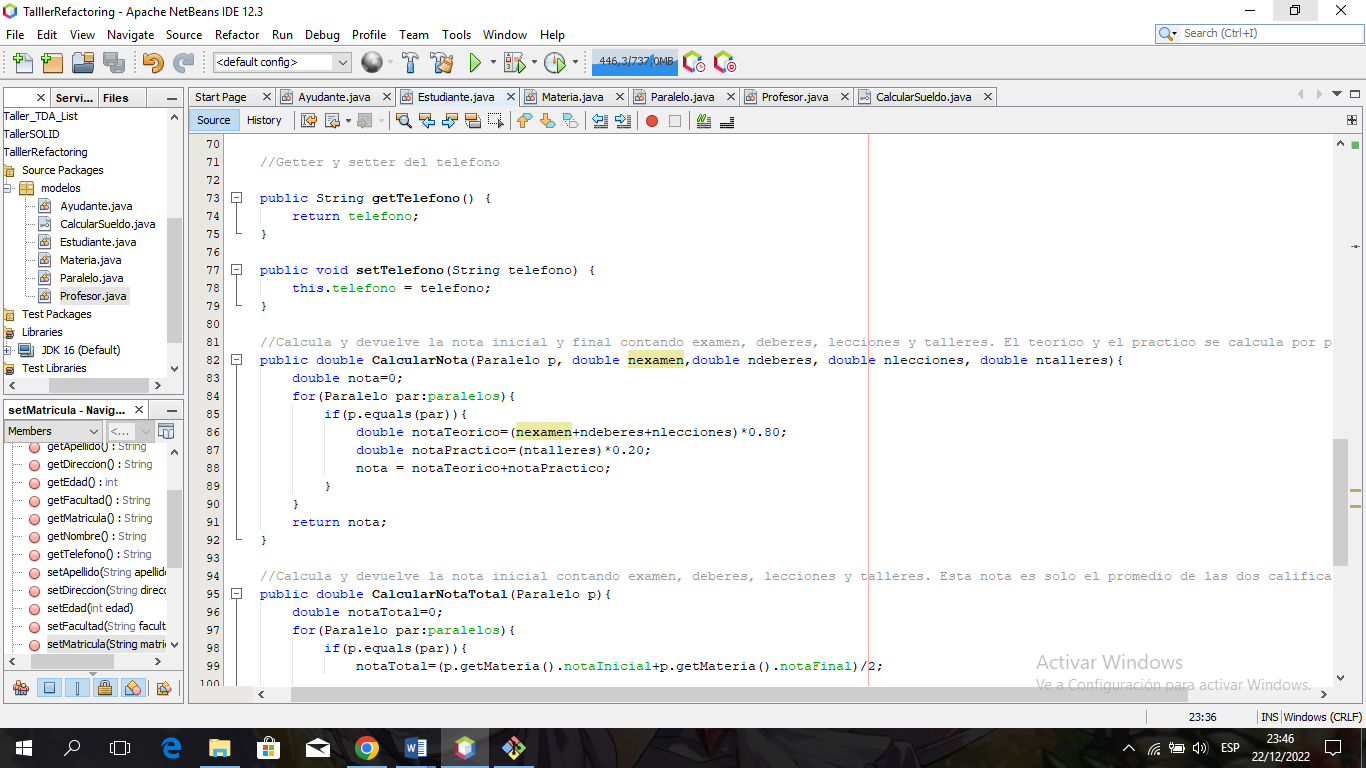




# Comments

Todo el documento tiene demasiados comentarios en la clase estudiante pero ya cuenta con varios nombres bien definido en los métodos y variables por lo tanto sería mejor eliminar los comentarios

Código antes



Codigo después

