**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN**

**DISEÑO DE SOFTWARE**

**TALLER REFACTORING**

**Integrantes:**

* **José Gabriel Jaramillo Arana**
* **Joselyne Isabel Torres Duque**
* **Enrique Steeven Yugcha Chichanda**

Tabla de contenido

[Intimidad inapropiada 3](#_Toc48255034)

[Lazy class 4](#_Toc48255035)

[Duplicate code 5](#_Toc48255036)

[Data Class 6](#_Toc48255037)

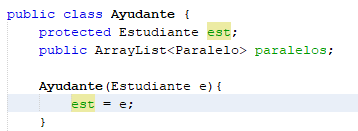
[Data Clump 7](#_Toc48255038)

[Long Parameters Method 8](#_Toc48255039)

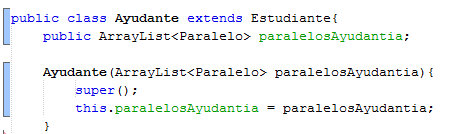
[Shotgun surgery 9](#_Toc48255040)

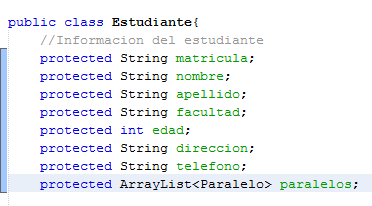
Intimidad inapropiada

La clase Ayudante al tener un constructor para solo instanciar el parámetro de la clase Estudiante, y métodos getters y setters para extraer datos pertenecientes a la misma clase Estudiante cumple con los síntomas de intimidad inapropiada dado que este code smell aparece cuando una clase usa los campos y métodos internos de otra.



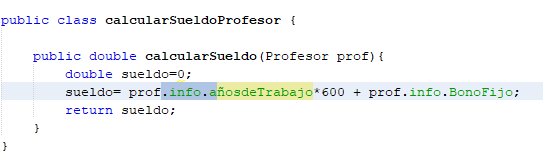
Para solucionar esto se emplea la técnica de refactoring denominada “Replace delegation with inheritance” de esta forma la clase Estudiante pasa a ser el padre de Ayudante, dado que el Ayudante sigue siendo un estudiante la única diferencia es que tiene una lista de paralelos donde el dará las clases. En el constructor del Ayudante ahora se hace uso del super() por la herencia, y ya no se necesitara los getters y setters que estaban antes de la herencia, dado que también hereda estos métodos del Estudiante.



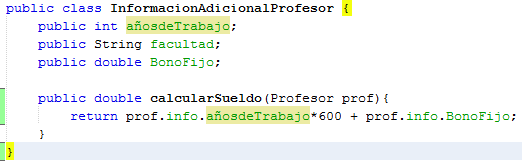


# Lazy class

La clase calcularSueldoProfesor solo contiene un método para ser usado por otras clases y no realiza ninguna otra acción por lo cual se presenta el code smell Lazy Class

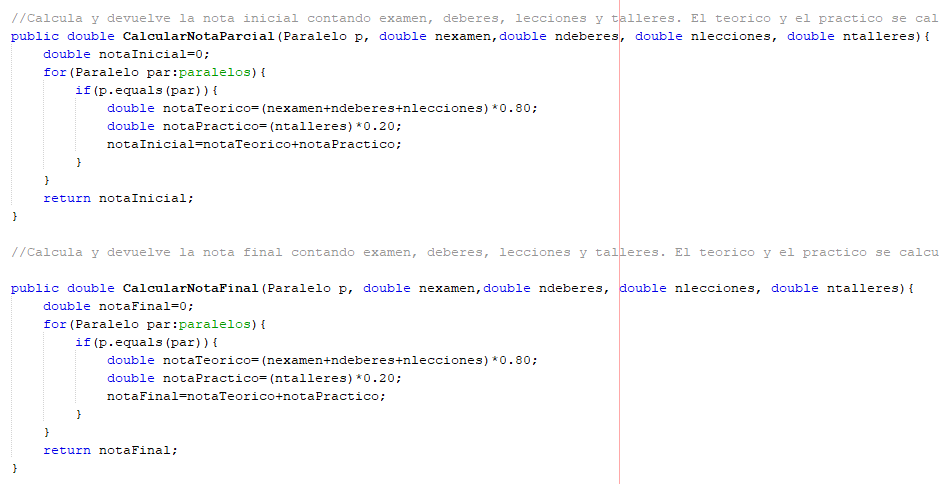


Para solucionar el code smell se aplica la técnica de refactorización Inline class, esto nos permite mover el contenido de la clase calcularSueldoProfesor a la clase InformacionAdicionalProfesor en este caso se mueve

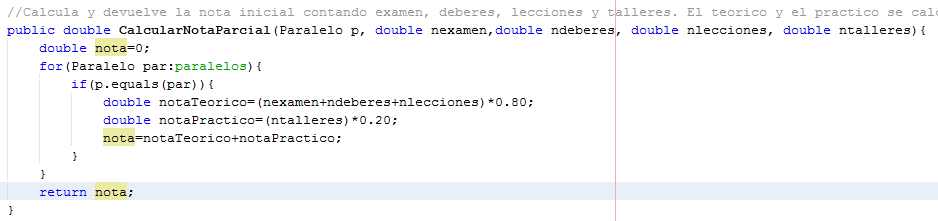


# Duplicate code

El código de los métodos CalcularNotaInicial y CalcularNotaFinal es exactamente igual.

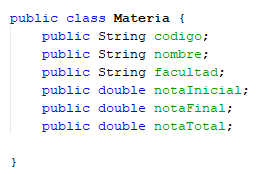


Para corregir este code smell se utiliza la técnica de refactoring Inline method, lo que permite utilizar un solo método para realizar ambos requerimientos, en este caso con el método CalcularNotaParcial se puede hacer el cáculo de la nota inicial y de la nota final, esto permite hacer el código más corto y menos costoso de soportar.

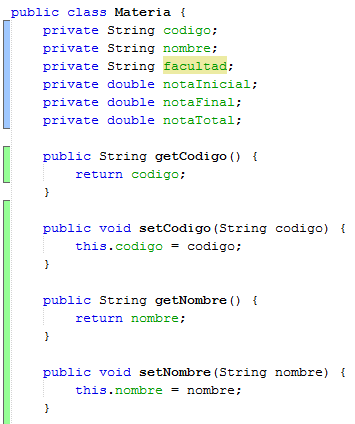


Data Class

La clase Materia solo contiene atributos públicos y ninguna funcionalidad adicional, lo cual provoca el code smell Data class, si se mantiene así esto causa que se peua provocar otro code smell como el de Inappropiate intimacy ya que el uso de estos datos en otras clases se realizaría directamente aumentando el acoplamiento.

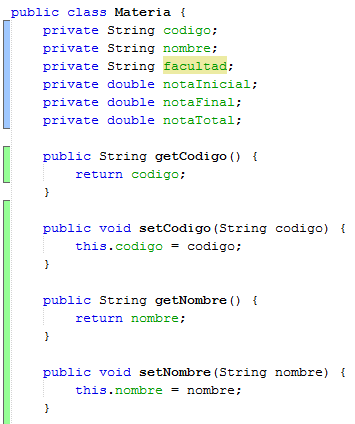


Para resolver este code smell se utiliza la técnica de refactorización Encapsulate Field la cual cambia el acceso a los atributos de la clase para que estos solo puedan ser accedidos a través de getters y setters.

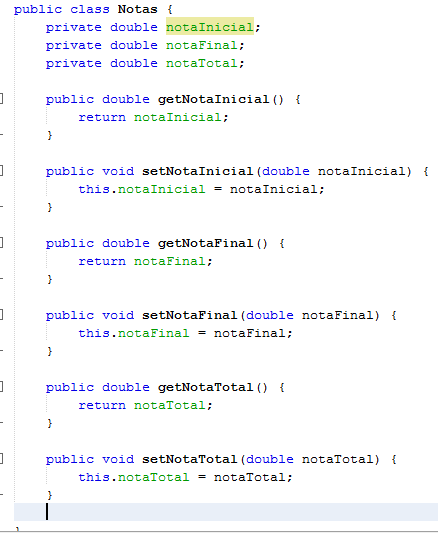


Data Clump

Los parámetros de notas inicial, final y total deben ser manejados en conjunto, ya que si uno deja de ser tomado en cuenta los otros no pueden funcionar correctamente, lo cual provoca el code smell Data clump. Si esto se mantiene como en la imagen la clase Materia además de tener un code smell incumple con el principio SOLID single responsability ya que las notas no deberían ser manejadas dentro de la materia.

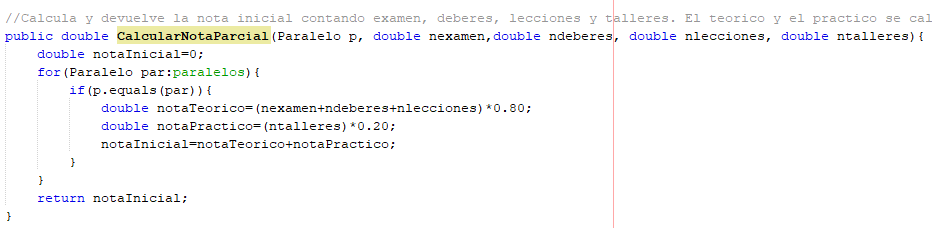


Para corregir este code smell se utiliza Extract class, por lo cual los parámetros correspondientes a las notas se los extrae en una nueva clase llamada Notas los cuales solo pueden ser accedidos mediante sus getters y setters.

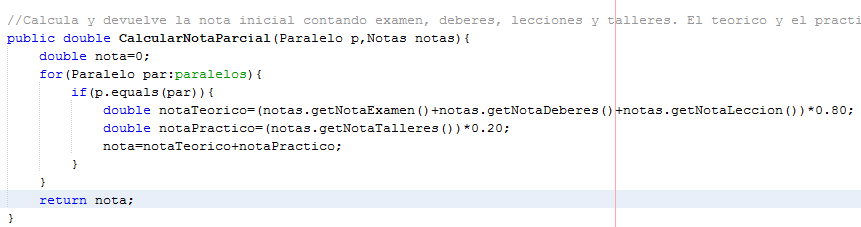


Long Parameters Method

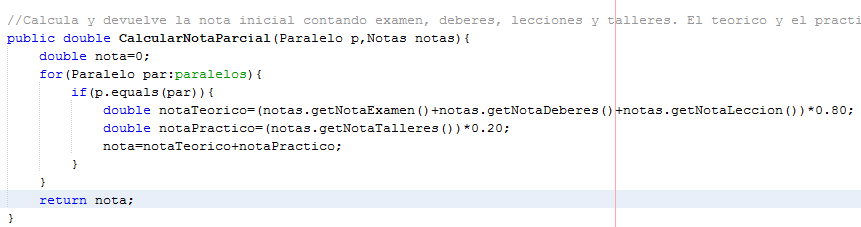
El método CalcularNotaParcial contiene 5 parámetros de entrada y el code smell “long parameters method” indica que a lo mucho debe haber entre 3 o 4 parametros.



Como las variables son de la misma naturaleza (relacionadas con notas), se las movió a la clase notas que fue creada con anterioridad y así se envía un objeto (técnica de refactorización “introduce parameter object”) de tipo notas de la cual se puede obtener las mismas variables.



Shotgun surgery

Si por alguna razón cambia la ponderación de la materia, será necesario cambiar la nueva ponderación en múltiples clases del programa, lo cual causa el code smell Shotgun surgery.

Para eliminar este code smell se puede agregar una variable llamada ponderación a la clase Materia y agregar esta variable en las porciones de código que la necesiten, así funcionará de forma dinámica (técnica de refactoring mover campo).

