**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**PRIMER TÉRMINO 2020**

**DISEñO DE SOFTWARE**

**TALLER:**

CODE SMELLS

**AUTORES:**

1. Solis Chang Isaac
2. Tuarez Díaz Angie

[**Code Smell 1 : Comments**](#_k8lh81uiscwm) **4**

[Descripción](#_d33seu4inzxd) **4**

[**Imagen**](#_crqkaon44sk2) **4**

[Código Inicial](#_gccnbwbo98gf) 4

[Código Final](#_gccnbwbo98gf) 5

[**Code Smell 2: Data Class**](#_psulum6n6k9w) **6**

[**Descripción**](#_72x8dzxvtlwp) **6**

[**Imagen**](#_7zdlt7pyvx71) **6**

[Código Inicial](#_ms9y0q7fyqfj) 6

[Código Final](#_ndirkim2gwss) 7

[**Code Smell 3: Lazy Class**](#_iez5agphf4re) **8**

[**Descripción**](#_4ld9xengwwkt) **8**

[**Imagen**](#_sz9a43fqty6t) **8**

[Código Inicial](#_8c0hy8ry2632) 8

[Código Final](#_e2r9h77ygqwy) 8

[**Code Smell 4: Temporary Fields**](#_3yshzwoz8z7v) **10**

[**Descripción**](#_sd3pzm7gcdqh) **10**

[**Imagen**](#_alqjyeyndhhm) **10**

[Código Inicial](#_evzh4gwvaxob) 10

[Código Final](#_q6ee4w8poozu) 10

[**Code Smell 5: Inappropriate Intimacy**](#_ajyf8k20mwe5) **11**

[**Descripción**](#_pgzyuhq3jt2s) **11**

[**Imagen**](#_wy5k258kb8et) **11**

[Código Inicial](#_1zv6ogu1uekk) 11

[Código Final](#_og0ggd1d0m0t) 11

[**Code Smell 6: Feature Envy**](#_ta3fdi7j9hs5) **12**

[**Descripción**](#_niqkpeys5kuh) **12**

[**Imagen**](#_nwkzisgluvo) **12**

[Código Inicial](#_9ocuqe30tpv8) 12

[Código Final](#_55kuoued8006) 12

[**Code Smell 7: Duplicated Code**](#_tk7smw2t5ikf) **13**

[**Descripción**](#_w58qa47tsls) **13**

[**Imagen**](#_m9di7gml7ny7) **13**

[Código Inicial](#_7ivz8etk5kmx) 13

[Código Final](#_1tqvg5t1bbig) 13

## **Code Smell 1 : Comments**

## **Descripción**

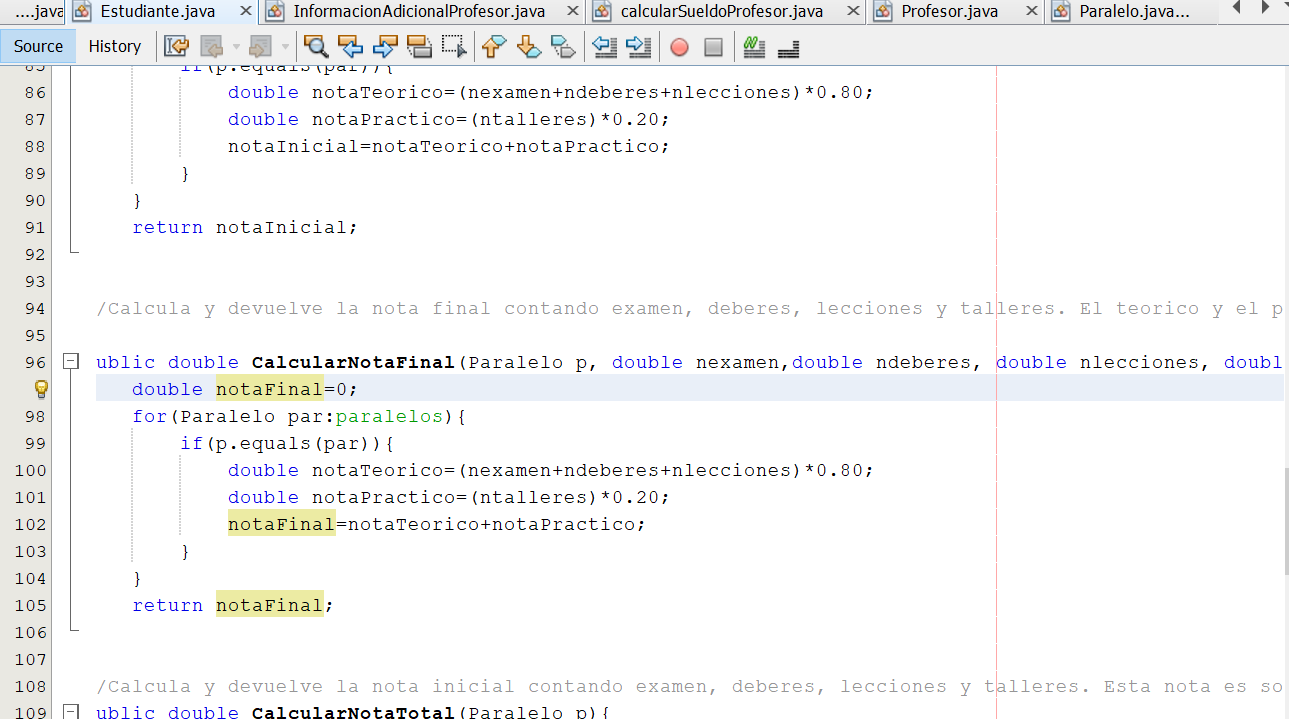
Este Code Smells surge cuando en el código aparecen comentarios innecesarios dentro de los métodos, el programador trata de explicar su código, pero al final tantos comentarios innecesarios no facilita la lectura del código. Este mal olor se presenta en las clases Ayudante, Estudiante y Paralelo.

Este mal olor es fácil de identificar y evitar, así mismo es fácil de limpiar el código, eliminando los comentarios que sean obvios e innecesarios usando la técnica [**Extract Variable**](https://refactoring.guru/es/extract-variable) y [**Extract Method**](https://refactoring.guru/es/extract-method).

## **Imagen**

### Código Inicial

Se muestra una parte del código donde se puede evidenciar



### Código Final

### 

## 

## 

## 

## 

## **Code Smell 2: Data Class**

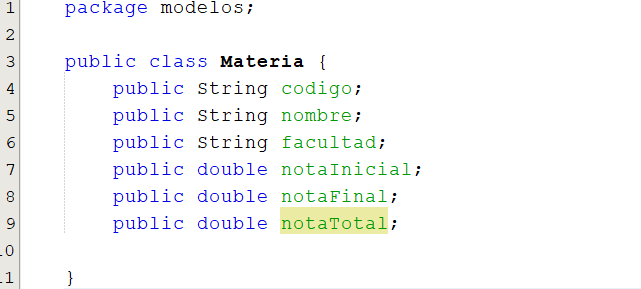
## **Descripción**

Este olor se refiere a una clase que contiene solo campos y métodos getters y setters. Estos son simplemente contenedores de datos utilizados por otras clases. Estas clases no contienen ninguna funcionalidad adicional y no pueden operar de forma independiente con los datos que poseen

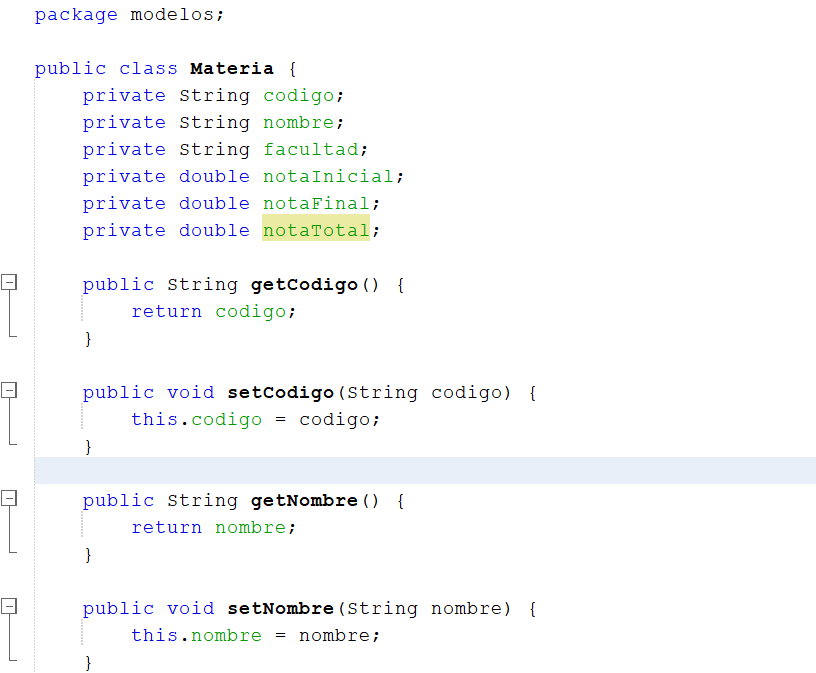
Esto solo lo podemos encontrar en la clase materia ya que solo tiene atributos públicos, por lo que aplicando el método [**Encapsulate Field**](https://refactoring.guru/es/encapsulate-field), se podrá privado los atributos y se crearán métodos públicos para acceder a esos atributos..

## **Imagen**

### Código Inicial



### Código Final



## 

## **Code Smell 3: Lazy Class**

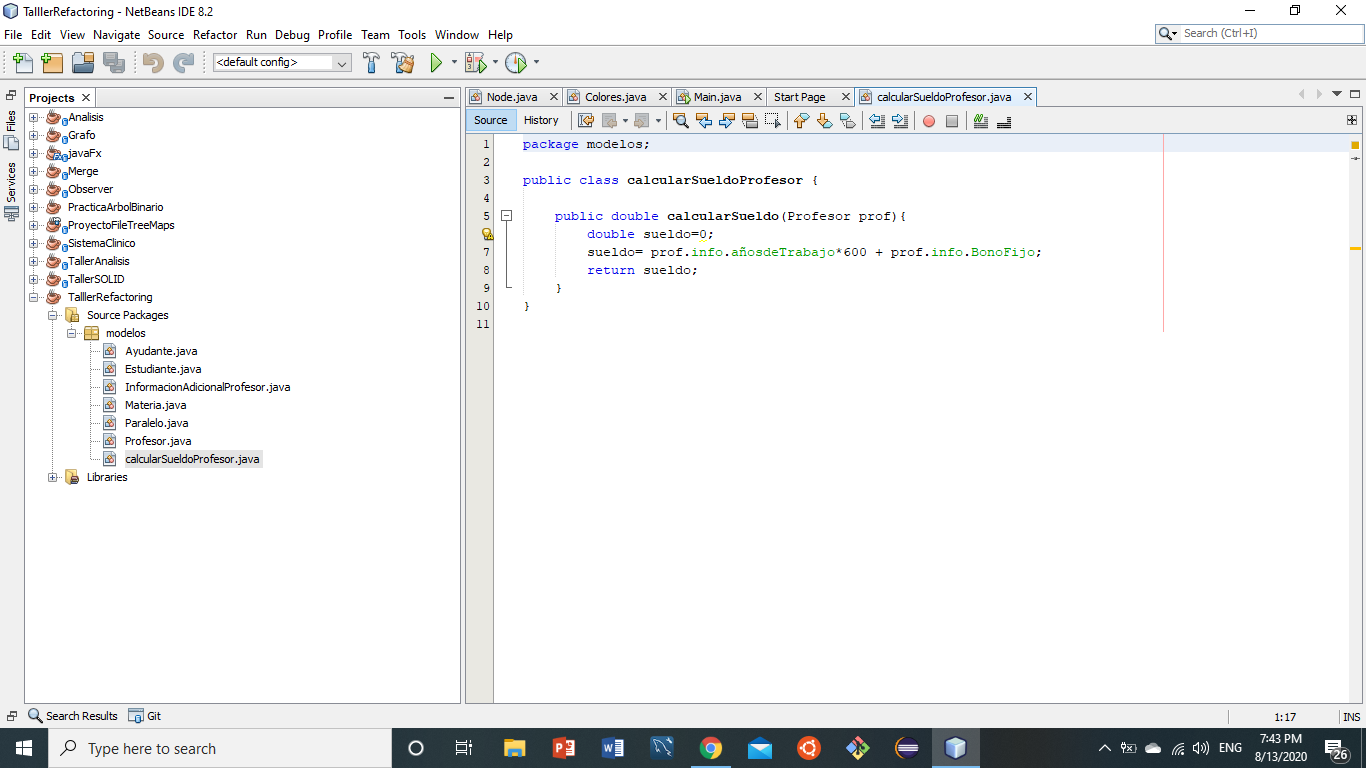
## **Descripción**

Este CodeSmell surge cuando una clase posee muy poca funcionalidad, es decir, muy pocos métodos o métodos que que aportan muy poco al código final, en general. O también suele darse cuando se crea una clase pensando que se la iba a usar o trabajar en un futuro pero quedó obsoleta.

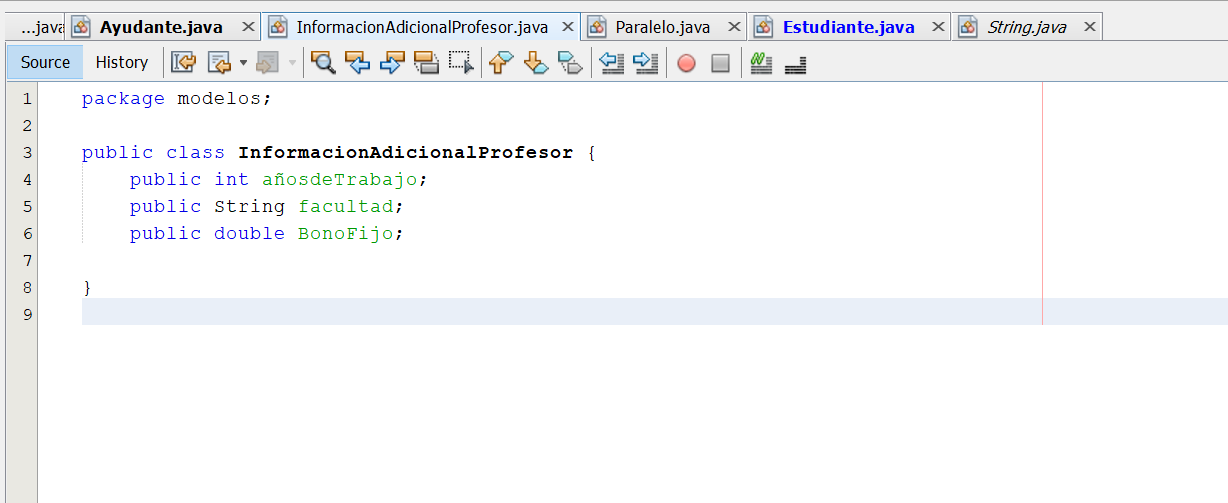
## **Imagen**

### Código Inicial

* En la clase calcularSueldoProfesor

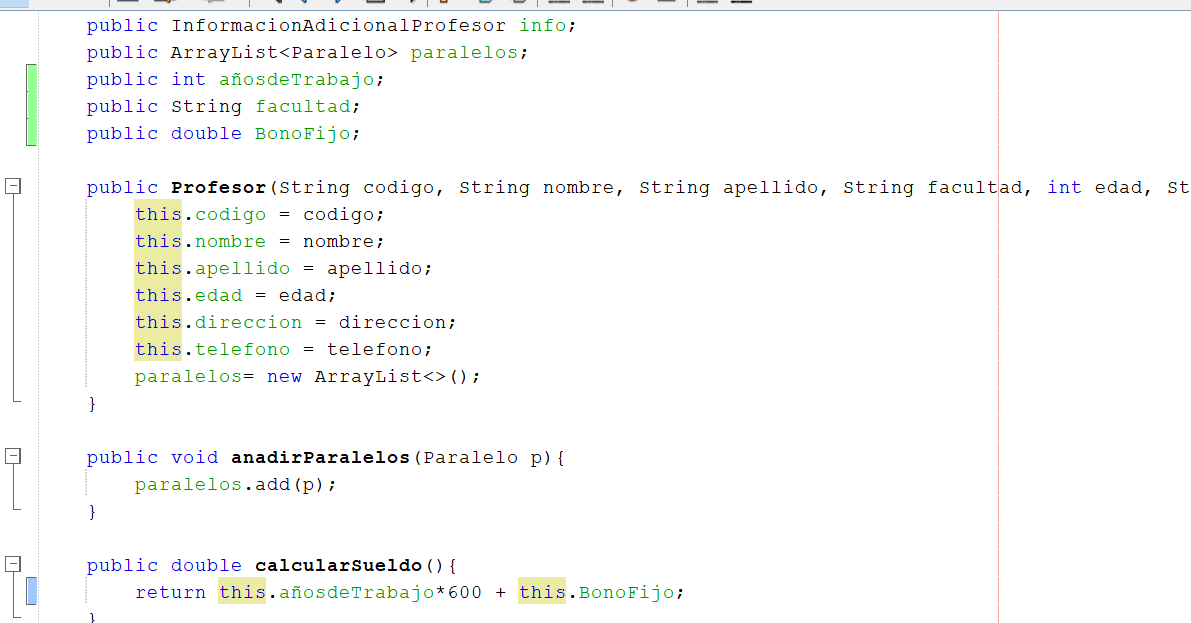


* En cla clase información del profesor



### Código Final

Para los dos casos se usó la técnica [**método**](https://refactoring.guru/es/move-method) [**Move**](https://refactoring.guru/es/move-field) y [**Move Field**](https://refactoring.guru/es/move-field) para mover los atributos y los métodos no usados a la clase profesor

.

## 

## 

## 

## 

## 

## **Code Smell 4: Temporary Fields**

## **Descripción**

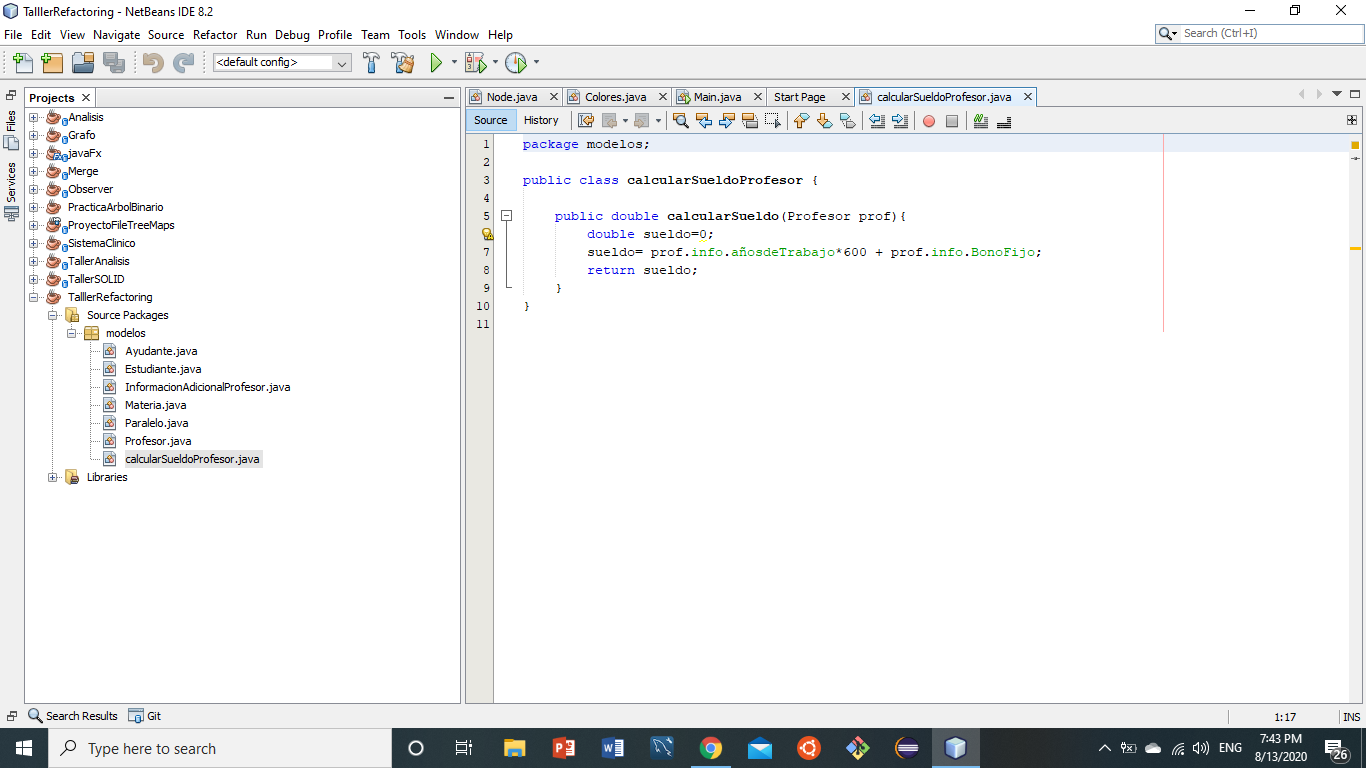
Datos que son usados una sola vez o que solo son llenados bajo ciertas circunstancias, son ejemplo de el code smell **Temp Fields.** Eso ocurre en la clase calcularSueldoProfesor con la variable **sueldo.** Esta variable NO es necesaria dentro del método, puede ser eliminada y simplemente retornar lo que se encuentra dentro de ella.

A pesar de que no produce un problema para el método, la variable sueldo es innecesaria y vuelve difícil de leer el código.

El método de refactorización aplicado para solucionar este smell es **Extraer Variable.**

## **Imagen**

### Código Inicial



### Código Final

## 

## 

## **Code Smell 5: Inappropriate Intimacy**

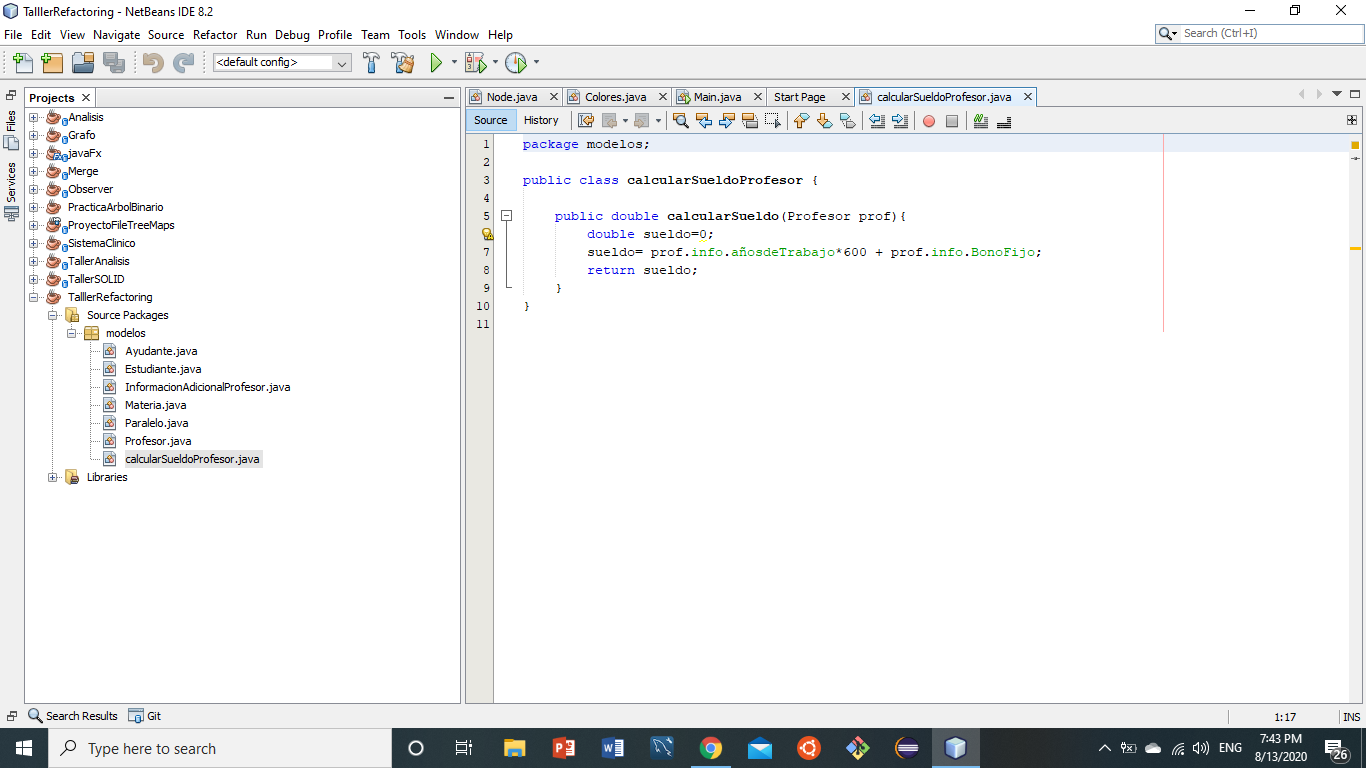
## **Descripción**

Cuando una clase usa los métodos o atributos de otra clase, se produce el code smell Inappropriate Intimacy. Dentro de la clase clase CalcularSueldoProfesor ocurre este smell porque su único método, accede a los atributos presentes en la clase profesor.

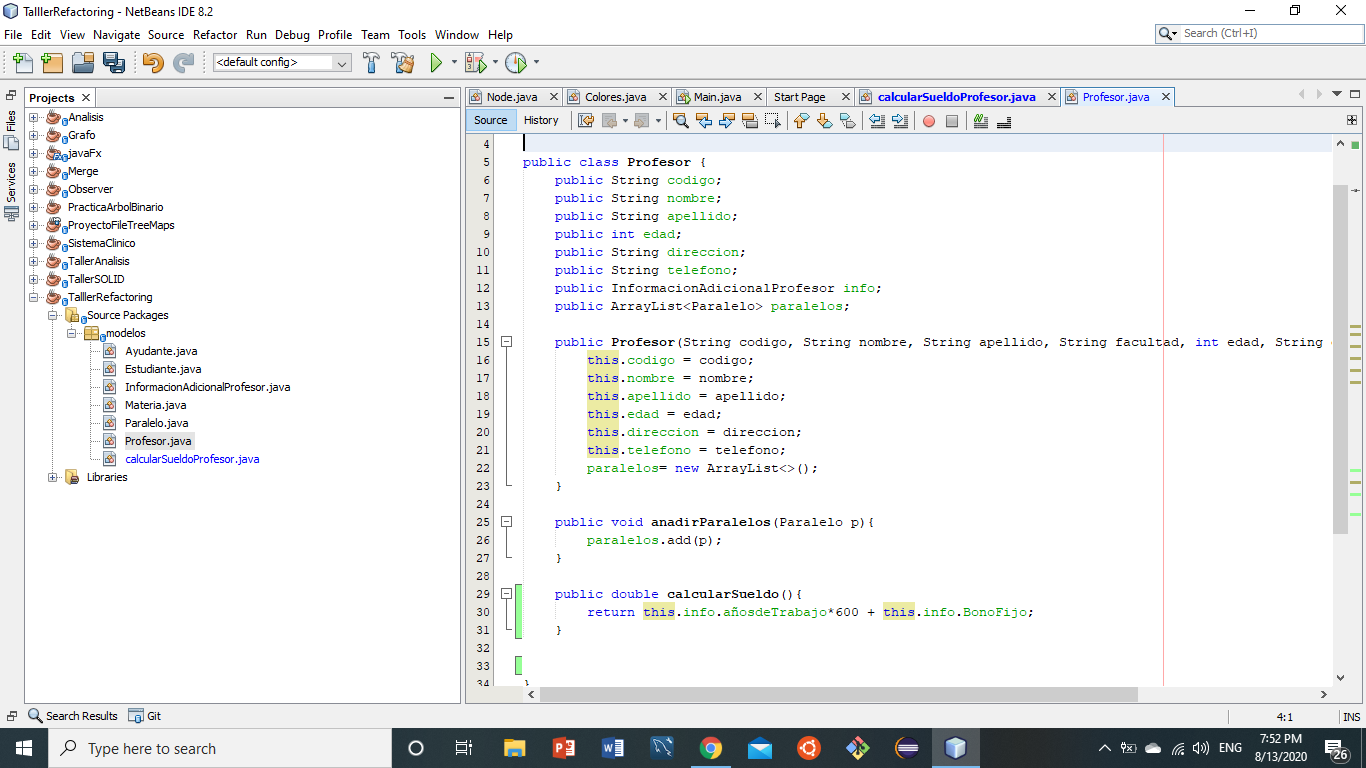
La técnica de refactorización a usar será Mover Método. El método dentro de la clase, será movido a la clase Profesor.

## **Imagen**

### Código Inicial



### Código Final



## **Code Smell 6: Feature Envy**

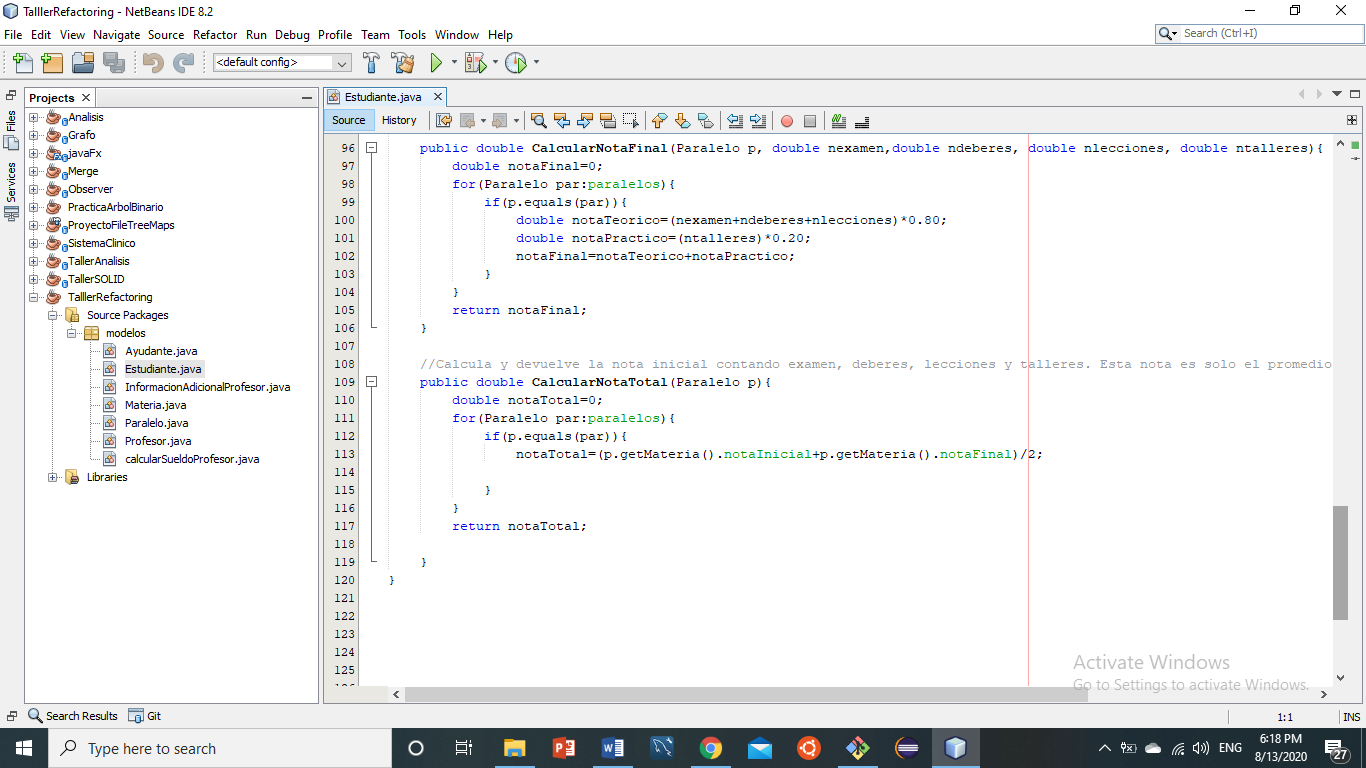
## **Descripción**

Este code smell se presenta cuando un método accede a los datos de otro objeto más que a los suyos.

Podemos ver que el método CalcularNotaTotal, dentro de la clase Estudiante, solo trabaja con datos del paralelo enviado como parámetro por lo que, para solucionar ese code smell, usaremos la técnica de refactorización **Mover Método**. El método será puesto en otra clase.

## **Imagen**

### Código Inicial



### Código Final

### 

## **Code Smell 7: Duplicated Code**

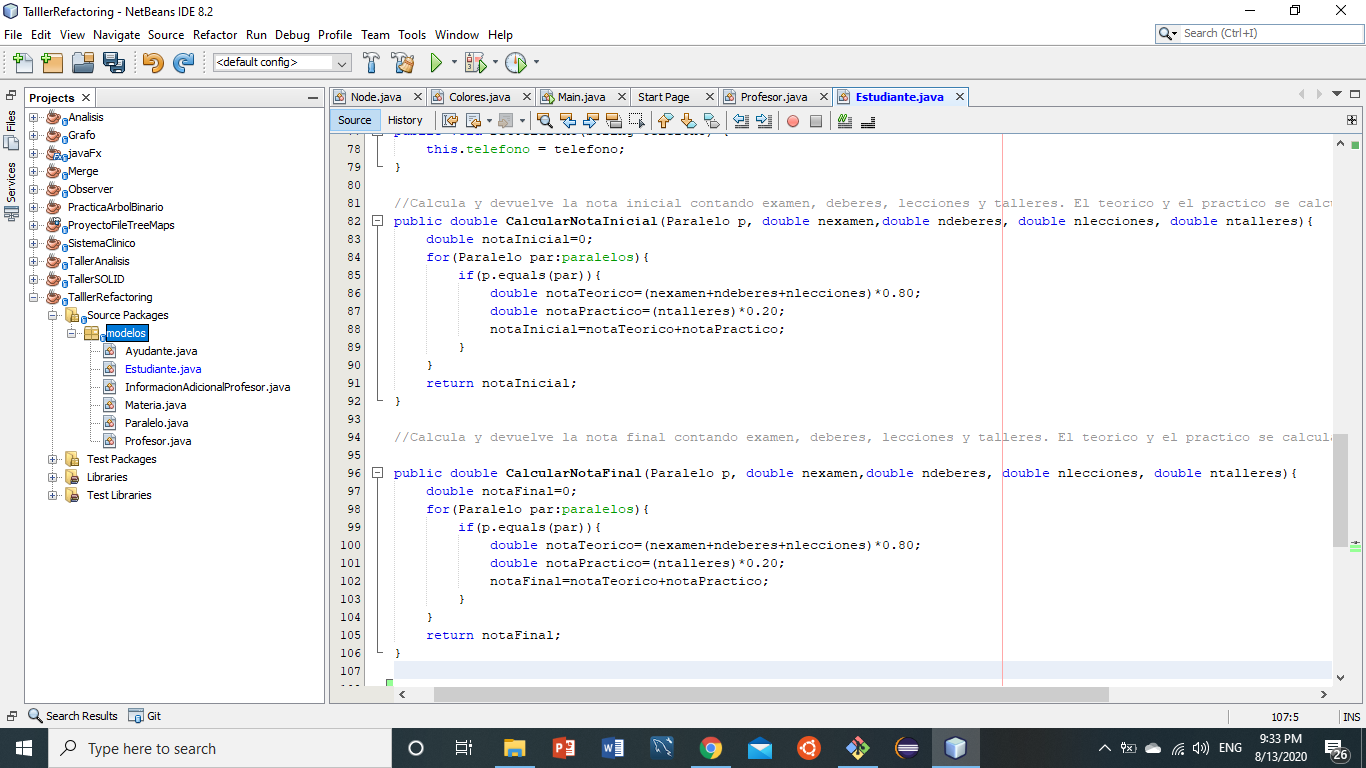
## **Descripción**

En vista del que código dentro de los métodos CalcularNotaFinal y CalcularNotaInicial es el mismo, estamos en presencia del code smell Duplicated Code.

El método de refactorización que se utilizó para manejar este olor es **Extraer Método.** Se colocó ese código repetido dentro un nuevo método, el cual será llamado cuando se lo necesite.

## **Imagen**

### Código Inicial



### Código Final

