Integrantes:

BURNEO ESPIN DANNY JESUS, eras lindao jefferson andrés, moran mendoza nicole madelayne

Taller de refactoring

Grupo 5

**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN**

**DISEÑO DE SOFTWARE**

**Taller de Refactoring - I TÉRMINO 2020**

**Objetivos**

* Identificar malos olores de programación en el código fuente adjunto y las técnicas de refactorización correspondientes.
* Aplicar técnicas de refactorización que pueden aplicarse para eliminar los malos olores previamente identificados.

**Descripción**

En equipos de trabajo, conformados por hasta tres estudiantes, (i) identifique los malos olores de programación encontrados en el código fuente adjunto, (ii) Identifique las técnicas de refactorización adecuadas para eliminar los malos olores encontrados y (iii) Refactorice el código fuente para obtener un código más limpio y fácil de leer. Justifique su respuesta.

**Especificaciones**

Considere un **sistema académico** que permite manejar los estudiantes registrados en ciertos paralelos de distintas materias. Además, cada materia tiene un profesor y podría tener asignado un ayudante.

**Sección A**

Elabore un **reporte** en el que **identifique los code smells** encontrados en el código adjunto. Para cada code smell debe **indicar el nombre**, las **consecuencias** de mantener el mismo en dicho código y la(s) técnicas de refactorización utilizadas para eliminarlo. Para cada mal olor coloque una captura inicial y una del código refactorizado. Indique cualquier asunción que realice. **[60%]**

**Sección B**

Cree un repositorio en GitHub para subir el reporte y realizar las mejoras sobre el código inicial y aplique las técnicas indicadas en el reporte. **[40%]**

**Entregables (enlace a repositorio en Github)**

1. Un **documento** (**disponible en formatos docx o pdf**) que contenga el reporte y las imágenes correspondientes. Además, el documento debe tener una carátula en la que se liste a los integrantes del equipo, una tabla de contenido. Cada code smell debe estar en una página.
2. El repositorio debe contener en el directorio raíz el documento indicado anteriormente y la carpeta del proyecto refactorizado. Subir a Sidweb el enlace a dicho repositorio.
3. Cada estudiante debe entregar una evaluación a los miembros de su equipo de trabajo usando la rúbrica de co-evaluación en la **tarea individual** “Co-evaluación Taller de Refactoring” a disposición en SidWeb.

**Rúbrica de Calificación**

|  |  |
| --- | --- |
| **Sección A** | **Valor** |
| Identificar al menos 6 malos olores | 30 |
| Identificar técnicas para eliminar dichos malos olores. | 30 |
| **Sección B** |  |
| Aplicar correctamente las técnicas de refactorización indicadas en el reporte | 40 |
| **Penalidades** | **Valor** |
| Penalidad por hora o fracción de hora de retraso en la entrega | -30 |
| No subir a SidWeb los entregables requeridos de acuerdo con lo especificado | -50 |
| No subir la rúbrica de coevaluación en tarea individual (afectación individual) | -100 |

Contenido

[CODE SMELL DATA CLASS 3](#_Toc92370361)

[CODE SMELL FEATURE ENVY 7](#_Toc92370362)

[CODE SMELL DUPLICATE CODE 10](#_Toc92370363)

[CODE SMELL FEATURE ENVY 12](#_Toc92370364)

[CODE SMELL DATA CLASSS 14](#_Toc92370365)

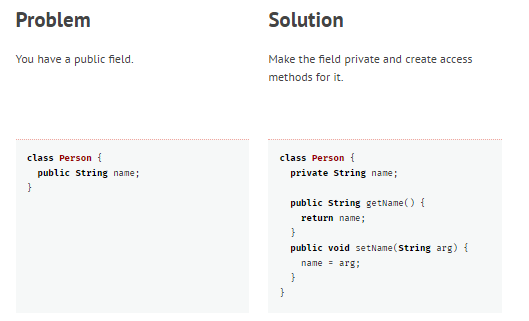
[CODE SMELL CLASE LARGA 16](#_Toc92370366)

**Identificar al menos 6 malos olores y sus técnicas para eliminar dichos olores.**

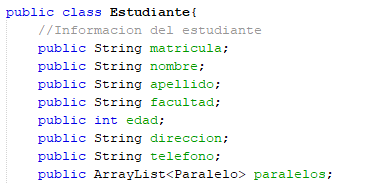
# CODE SMELL DATA CLASS

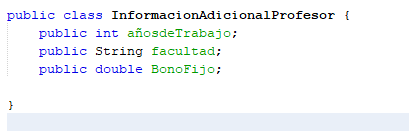
Se está violando el encapsulamiento que es uno de los principios fundamentales de la programación orientada a objetos. Si mantenemos el mismo código todas las clases están vulneradas por otras clases externas. Se utilizará la técnica de refactorización Encapsulate Field que consiste en poner estos atributos en privado y luego hacer getters y setters.

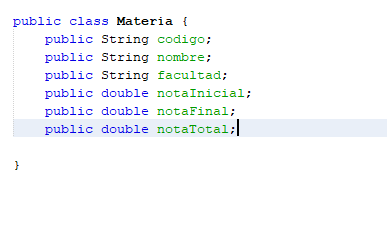
SOLUCION

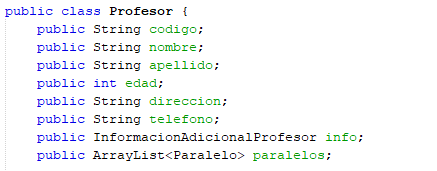


ANTES

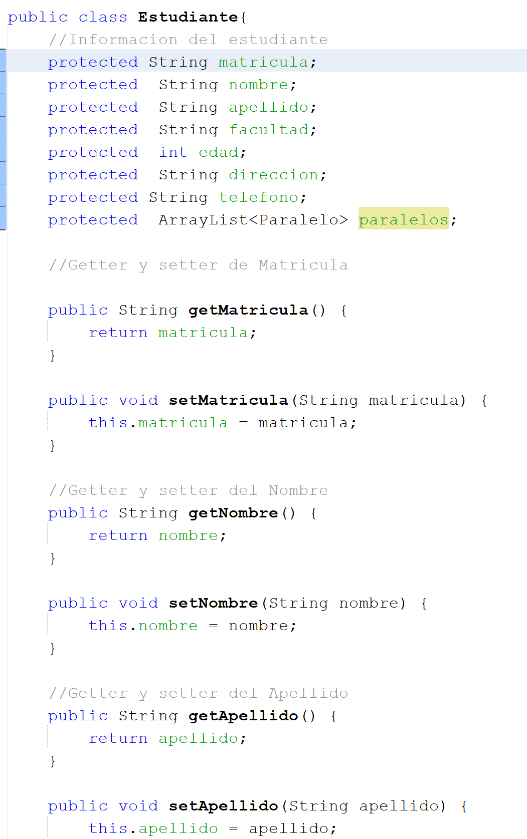


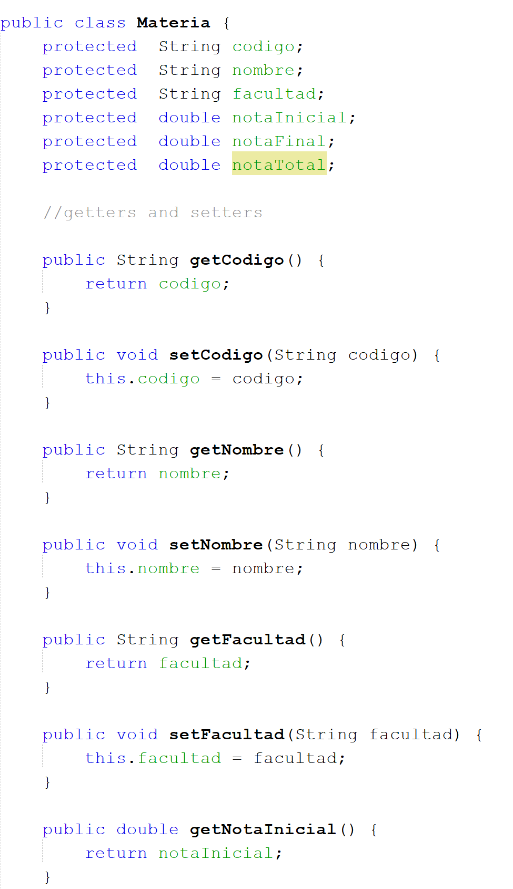


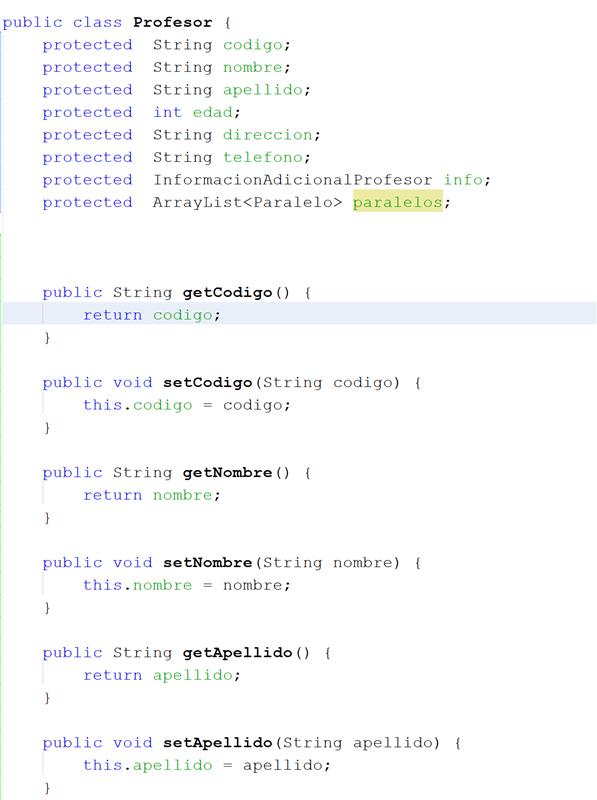




DESPUES



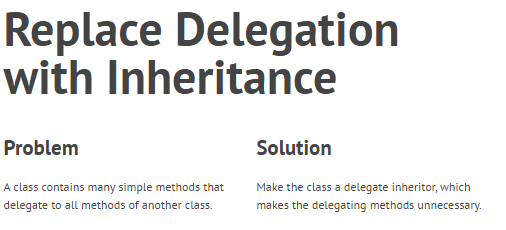


# CODE SMELL FEATURE ENVY

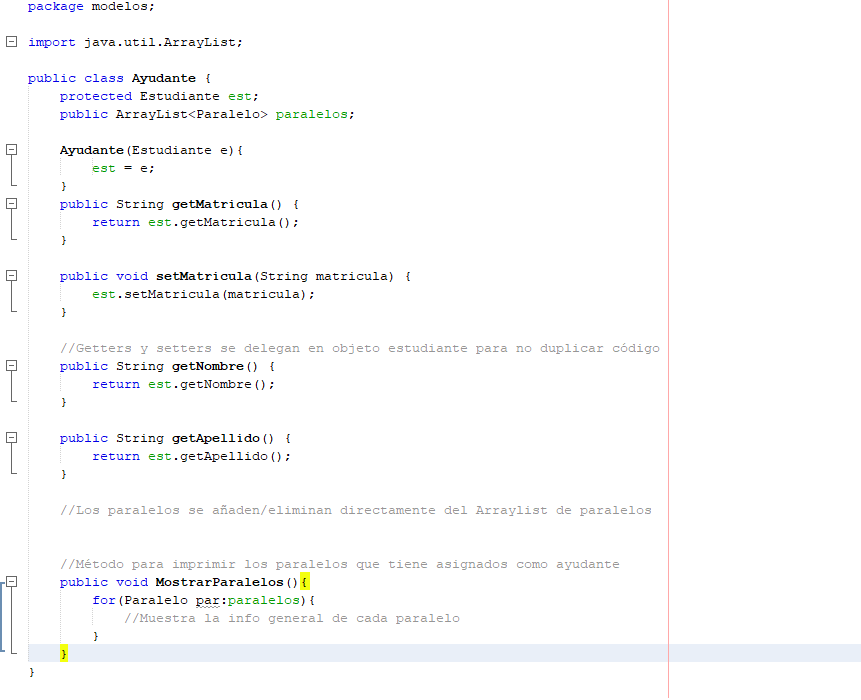
Si mantenemos el mismo código podríamos causar confusión porque no existe alguna heredacion de ayudante y porque no promovemos la reutilización de código y no generaríamos una jerarquía cuando es evidente.

La técnica que vamos a utilizar se llama Reemplazar Delegación con herencia

SOLUCION



ANTES



DESPUES

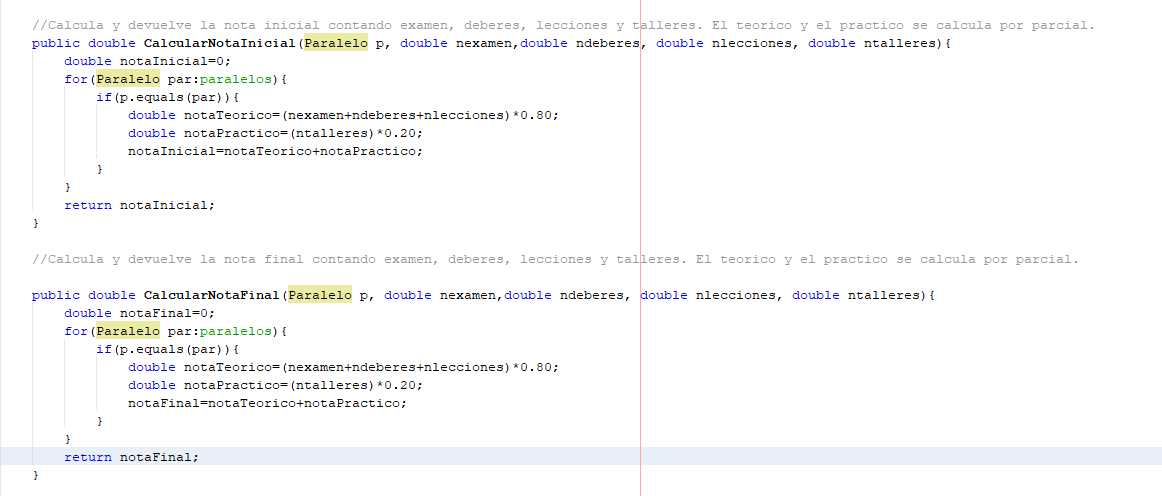


# CODE SMELL DUPLICATE CODE

Si dejamos que existan códigos idénticos nuestro programa seria poco eficiente y muy confuso. Además, aumentamos el tamaño del código dificultando su lectura. Finalmente, algunos de los métodos podrían nunca ser usados y por lo tanto inservibles.

Utilizaremos la técnica de refactorización llamada Extraer Método que nos indica porciones de código que pueden agruparse de una forma para que el método sea más corto.

ANTES



DESPUES

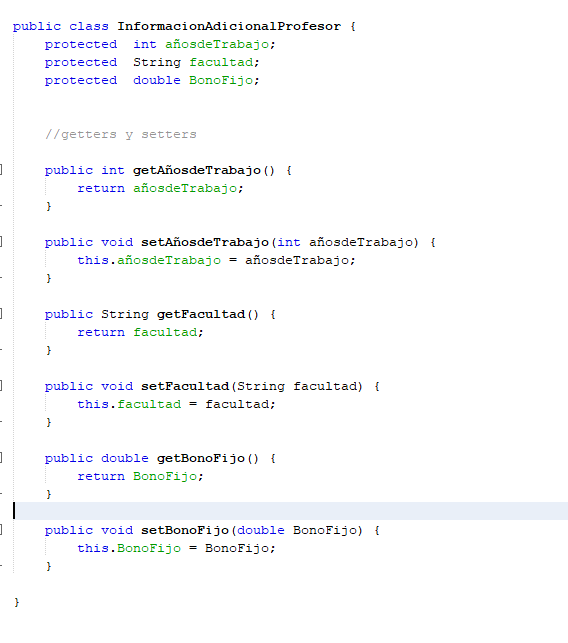


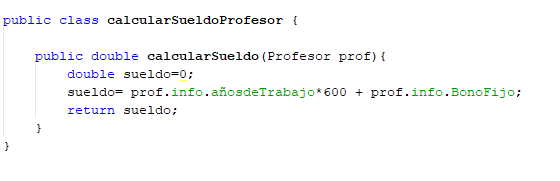
# CODE SMELL FEATURE ENVY

Viola la cohesión de datos y métodos de la clase profesor, no es eficiente separar sus atributos y también sus métodos en otras clases ya que no es necesario y conseguiríamos lo mismo manteniendo todo en una sola clase.

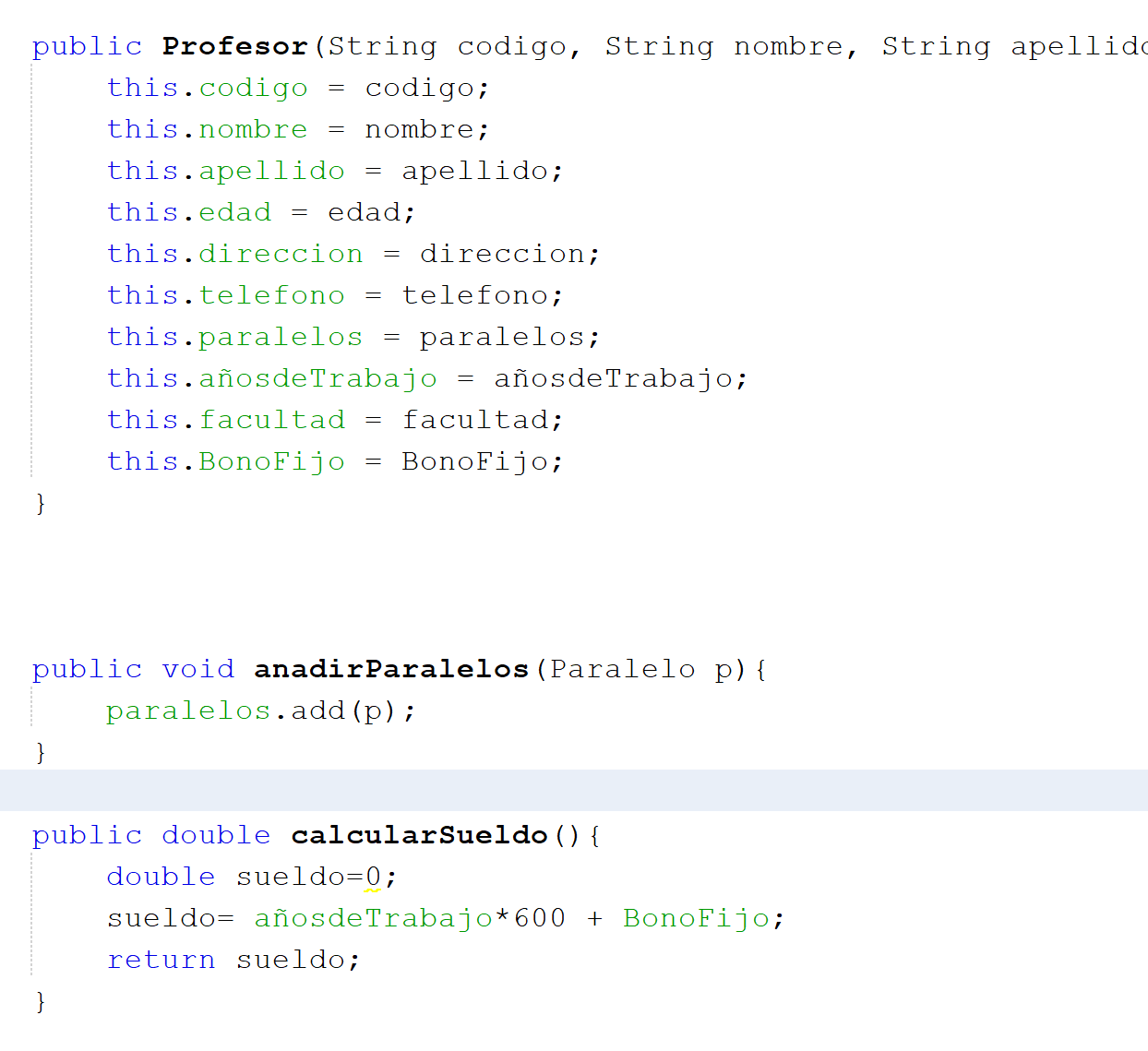
Utilizaremos la técnica de refactorización llamada Mover Metodo.

ANTES





DESPUES



# CODE SMELL DATA CLASSS

Si mantenemos el mismo código no podríamos añadir o eliminar un estudiante en especifico por paralelo, sino que tendríamos que volver a escribir o detallar todos los estudiantes de ese paralelo, haciéndolo tedioso y poco eficiente.

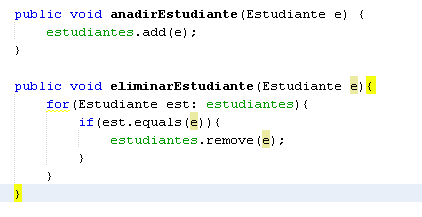
Utilizaremos la técnica de refactorización llamada Encapsular colección que nos indica que está mal tener

Métodos como getColeccion y setColecccion, deberíamos tener métodos como add, remove, get donde se agreguen o se eliminen valores uno a uno.

ANTES



DESPUES

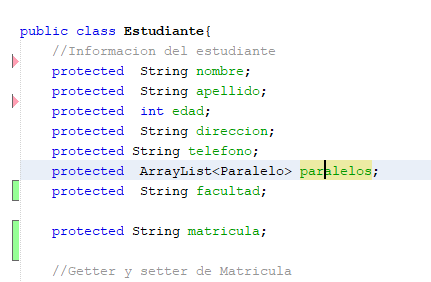


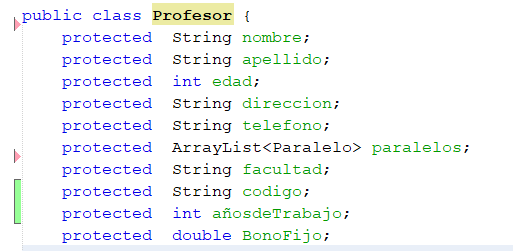
# CODE SMELL CLASE LARGA

Si dejamos que nuestra clase mantenga el mismo código, cuando queramos añadir más información sobre el objeto tendríamos que aumentar el tamaño de la clase. Además, evitamos la duplicación de código y promovemos una fácil lectura.

Utilizaremos la técnica de refactorización llamada Extraer superclase.

Antes





Después



