Taller Refactoring

Estudiantes:

- Alexander Alzate
- Carlos Moncayo

Link repositorio Git: https://github.com/Aalzate95/Refactory

Tabla de contenido

Middle man

```java
package modelos;

public class calcularSueldoProfesor {

    public double calcularSueldo(Profesor prof){
        double sueldo=0;
        sueldo= prof.info.añosdeTrabajo*600 + prof.info.BonoFijo;
        return sueldo;
    }
}

```

Esta clase solamente tiene un método que sirve para calcular el sueldo de un profesor y recibe a uno como parámetro. Se debe aplicar el refactoring de Inline Class, para trasladar este método a una clase que le saque provecho, como lo es la clase profesor. Además, se eliminaron las variables innecesarias del método (Temporary Field).

```java
    public void setBonoFijo(double BonoFijo) {
        this.BonoFijo = BonoFijo;
    }

    public Profesor(String codigo, String nombre, String apellido, int edad, String direccion, String telefono, ArrayList<Paralelo> paralelos,
        this.codigo = codigo;
        this.nombre = nombre;
        this.apellido = apellido;
        this.edad = edad;
        this.direccion = direccion;
        this.telefono = telefono;
        this.paralelos = new ArrayList();
        this.añosdeTrabajo = añosdeTrabajo;
        this.facultad = facultad;
        this.BonoFijo = BonoFijo;
    }

    public void anadirParalelos(Paralelo p){
        paralelos.add(p);
    }

    public double calcularSueldo(){
        return this.añosdeTrabajo*600 + this.BonoFijo;
    }

}
```

Data Class

```
1   package modelos;
2
3   public class InformacionAdicionalProfesor {
4       public int añosdeTrabajo;
5       public String facultad;
6       public double BonoFijo;
7
8   }
9
```

Esta clase únicamente existe para proveer más información a la clase profesor, no tienen ninguna función adicional, lo que se puede hacer para optimizar este smell, es aplicar el refactoring de Encapsulate Field y mover esta información a la clase profesor, que las utilizara apropiadamente, además de otorgarle sus respectivos getters y setters. Además, se encapsularon las otras variables de clase.

```java
public class Profesor {
    private String codigo;
    private String nombre;
    private String apellido;
    private int edad;
    private String direccion;
    private String telefono;
    private ArrayList<Paralelo> paralelos;
    private int añosdeTrabajo;
    private String facultad;
    private double BonoFijo;

    public int getAñosdeTrabajo() {
        return añosdeTrabajo;
    }

    public void setAñosdeTrabajo(int añosdeTrabajo) {
        this.añosdeTrabajo = añosdeTrabajo;
    }

    public String getFacultad() {
        return facultad;
    }

    public void setFacultad(String facultad) {
        this.facultad = facultad;
    }

    public double getBonoFijo() {
        return BonoFijo;
    }

    public void setBonoFijo(double BonoFijo) {
        this.BonoFijo = BonoFijo;
    }
```

Lazy class+ Message Chains

```java
public class Materia {
    public String codigo;
    public String nombre;
    public String facultad;
    public double notaInicial;
    public double notaFinal;
    public double notaTotal;

}
```

```java
public double CalcularNotaTotal(Paralelo p){
    double notaTotal=0;
    for(Paralelo par:paralelos){
        if(p.equals(par)){
            notaTotal=(p.getMateria().notaInicial+p.getMateria().notaFinal)/2;

        }
    }
    return notaTotal;

}
```

```java
public Materia getMateria() {
    return materia;
}

public void setMateria(Materia materia) {
    this.materia = materia;
}
```

la clase Materia solo es una Lazy class que es instanciada por un Paralelo, y luego un Estudiante accede a Materia a través de Paralelo, lo mejor sería que Paralelo absorba la clase vaga

```java
public double CalcularNotaTotal(Paralelo p){
    double notaTotal=0;
    for(Paralelo par:paralelos){
        if(p.equals(par)){
            notaTotal=(p.getNotaInicial()+p.getNotaFinal())/2;

        }
    }
    return notaTotal;

}
```

```java
public class Paralelo {
    public int numero;
    public Profesor profesor;
    public ArrayList<Estudiante> estudiantes;
    public Ayudante ayudante;
    public String codigoMateria;
    public String nombreMateria;
    public String facultad;
    public double notaInicial;
    public double notaFinal;
    public double notaTotal;

    public int getNumero() {
        return numero;
    }

    public void setNumero(int numero) {
        this.numero = numero;
    }


    public Profesor getProfesor() {
        return profesor;
    }

    public void setProfesor(Profesor profesor) {
        this.profesor = profesor;
    }

    //Imprime el listado de estudiantes registrados
    public void mostrarListado(){
        //No es necesario implementar
    }

```

Temporary Field

```java
public double CalcularNotaInicial(Paralelo p, double nexamen,double ndeberes, double nlecciones, double ntalleres){
    double notaInicial=0;
    for(Paralelo par:paralelos){
        if(p.equals(par)){
            double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
            double notaPractico=(ntalleres)*0.20;
            notaInicial=notaTeorico+notaPractico;
        }
    }
    return notaInicial;
}

//Calcula y devuelve la nota final contando examen, deberes, lecciones y talleres. El teorico y el practico se calcu

public double CalcularNotaFinal(Paralelo p, double nexamen,double ndeberes, double nlecciones, double ntalleres){
    double notaFinal=0;
    for(Paralelo par:paralelos){
        if(p.equals(par)){
            double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
            double notaPractico=(ntalleres)*0.20;
            notaFinal=notaTeorico+notaPractico;
        }
    }
    return notaFinal;
}
```

```java
//Calcula y devuelve la nota inicial contando examen, deberes, lecciones y tallere
public double CalcularNotaTotal(Paralelo p){
    double notaTotal=0;
    for(Paralelo par:paralelos){
        if(p.equals(par)){
            notaTotal=(p.getMateria().notaInicial+p.getMateria().notaFinal)/2;

        }
    }
    return notaTotal;

}
```

Se crean variables temporales innecesarias, se puede retornar directamente el valor calculado, y para el caso de que no exista coincidencia retornar un objeto tipo Null directamente o en caso de querer continuar con el valor de 0 establecido se retorna directamente. Con esto tendríamos mejor código,más claro y organizado.
Nos quedaría de la siguiente manera:

```java
public double CalcularNotaInicial(Paralelo p, double nexamen,double ndeberes, d

    for(Paralelo par:paralelos){
        if(p.equals(par)){
            double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
            double notaPractico=(ntalleres)*0.20;
            return notaTeorico+notaPractico;
        }
    }
    return 0;
}

//Calcula y devuelve la nota final contando examen, deberes, lecciones y taller

public double CalcularNotaFinal(Paralelo p, double nexamen,double ndeberes, dou

    for(Paralelo par:paralelos){
        if(p.equals(par)){
            double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
            double notaPractico=(ntalleres)*0.20;
            return notaTeorico+notaPractico;
        }
    }
    return 0;
}

//Calcula y devuelve la nota inicial contando examen, deberes, lecciones y tall
public double CalcularNotaTotal(Paralelo p){

    for(Paralelo par:paralelos){
        if(p.equals(par)){
            return (p.getMateria().notaInicial+p.getMateria().notaFinal)/2;

        }
    }
    return 0;
```

Inappropriate Intimacy

```
public class Ayudante {
    protected Estudiante est;
    public ArrayList<Paralelo> paralelos;

    Ayudante(Estudiante e){
        est = e;
    }
    public String getMatricula() {
        return est.getMatricula();
    }

    public void setMatricula(String matricula) {
        est.setMatricula(matricula);
    }

    //Getters y setters se delegan en objeto estudiante
    public String getNombre() {
        return est.getNombre();
    }

    public String getApellido() {
        return est.getApellido();
    }

    //Los paralelos se añaden/eliminan directamente del


    //Método para imprimir los paralelos que tiene asig
    public void MostrarParalelos(){
        for(Paralelo par:paralelos){
            //Muestra la info general de cada paralelo
        }
    }
}
```

La clase ayudante contiene métodos que no le pertenecen, por lo que deben ser declarados dentro de sus respectivas clases para así mejorar la calidad del código, facilitando la lectura e interpretación de estos dentro de su misma clase.

```
public class Ayudante {
    protected Estudiante est;
    public ArrayList<Paralelo> paralelos;

    Ayudante(Estudiante e){
        est = e;
    }
    public String getMatricula() {
        return est.getMatricula();
    }

    public void setMatricula(String matricula) {
        est.setMatricula(matricula);
    }

    //Método para imprimir los paralelos que tiene
    public void MostrarParalelos(){
        for(Paralelo par:paralelos){
            //Muestra la info general de cada para
        }
    }
}
```

Alternative Classes with Different Interfaces

```java
public class Estudiante{
    //Informacion del estudiante
    public String matricula;
    public String nombre;
    public String apellido;
    public String facultad;
    public int edad;
    public String direccion;
    public String telefono;
    public ArrayList<Paralelo> paralelos;


    public String getMatricula() {
        return matricula;
    }

    public void setMatricula(String matricula) {
        this.matricula = matricula;
    }

    //Getter y setter del Nombre
    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    //Getter y setter del Apellido
    public String getApellido() {
        return apellido;
    }

    public void setApellido(String apellido) {
        this.apellido = apellido;
    }
```

Las clases ayudante y estudiante, son exactamente iguales, poseen los mismos métodos y atributos, por lo que se pueden combinar hacer una sola clase estudiante que tenga un atributo tipo boolean que indique si es ayudante o no, con esto podemos reducir y hacer el código más entendible.
Quedando de la siguiente forma:

```java
public class Estudiante{
    //Informacion del estudiante
    public String matricula;
    public String nombre;
    public String apellido;
    public String facultad;
    public int edad;
    public String direccion;
    public String telefono;
    public ArrayList<Paralelo> paralelos;
    public boolean isAyudante;
    //Getter y setter de Matricula

    public Estudiante(String matricula, String nombre, String ape
        this.matricula = matricula;
        this.nombre = nombre;
        this.apellido = apellido;
        this.facultad = facultad;
        this.edad = edad;
        this.direccion = direccion;
        this.telefono = telefono;
        this.paralelos = paralelos;
        this.isAyudante = isAyudante;
    }

    public String getMatricula() {
        return matricula;
    }

    public void setMatricula(String matricula) {
        this.matricula = matricula;
    }
```