Programación Orientada a Objetos. Práctica 4.1

Tema 4. Clases Básicas de Java

Ejercicio 1. (proyecto prNotas)

Se va a crear una aplicación para anotar las calificaciones obtenidas por alumnos en una asignatura. Para ello se crearán las clases `Alumno`, `Asignatura` y `AlumnoException`.

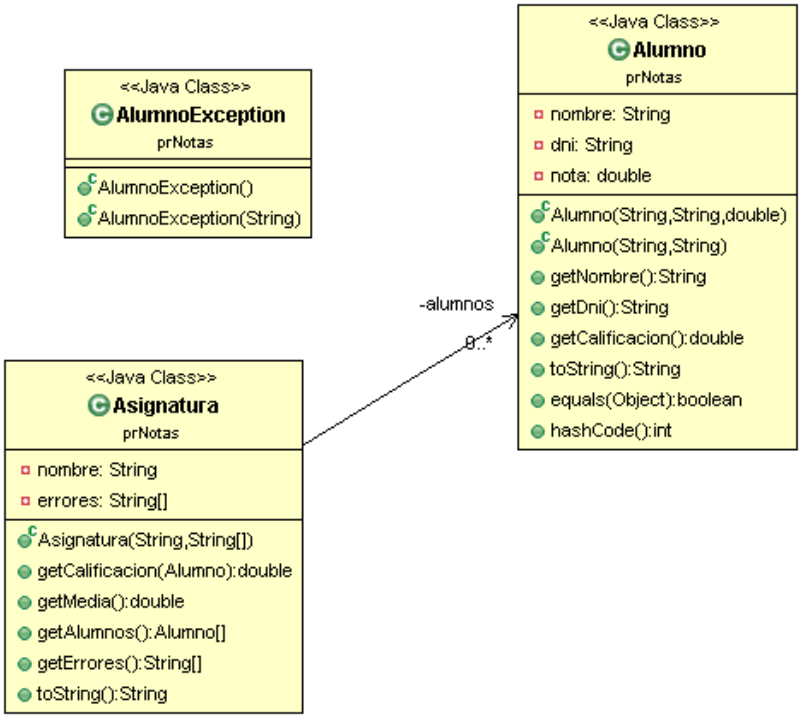

Figura 1: Diagrama de clases UML

Clase `AlumnoException`

Crea la excepción **comprobada** `AlumnoException` para manejar situaciones excepcionales que podrán producirse en las siguientes clases.

Clase `Alumno`

Crea la clase `Alumno` que mantiene información de un alumno del cual se conocen el *dni* (`String`), el *nombre* (`String`) y la *calificación* obtenida en una asignatura (`double`). La clase tendrá dos constructores, uno en el que se proporcionan el dni, el nombre y la calificación y otro con solo el dni y el nombre, siendo en este caso la calificación igual a cero. Si la calificación dada es negativa se deberá lanzar una excepción `AlumnoException` con el mensaje `"Calificación negativa"`.

Los siguientes métodos permiten acceder al nombre, al dni y a la calificación:

- `String getNombre();`
- `String getDni();`
- `double getCalificacion();`

Dos alumnos son iguales si coinciden sus nombres y sus dni. La letra del dni podrá estar indistintamente en mayúsculas o minúsculas.

La representación textual de un alumno contiene el nombre y el dni, en ese orden, pero no la calificación, según el formato del siguiente ejemplo (nótese que la coma forma parte del nombre, no del formato):

```
Lopez Turo, Manuel 23322443K
```

Aplicación `PruebaAlumno`

Crea una aplicación (clase distinguida `PruebaAlumno`) para probar la clase anterior. En esta clase se crean dos alumnos con los datos siguientes:

```
DNI: 22456784F Nombre: Gonzalez Perez, Juan Nota: 5.5
DNI: 33456777S Nombre: Gonzalez Perez, Juan Nota: 3.4
```

Y se muestra por pantalla el nombre de cada alumno, así como sus calificaciones. Además, se comprueba si ambos alumnos son iguales, indicándolo por pantalla. Ten en cuenta que la excepción `AlumnoException` es de obligado tratamiento a la hora de implementar `PruebaAlumno`. Ejecuta el programa.

A continuación modifica los datos del segundo alumno tal y como se indica a continuación, ejecuta de nuevo el programa y observa lo que sucede.

```
DNI: 33456777S Nombre: Gonzalez Perez, Juan Nota: -3.4
```

Clase `Asignatura`

Crea la clase `Asignatura`. Una asignatura se crea a partir del nombre de la misma y de un *array* de `String` en la que cada elemento del array contendrá toda la información para crear un alumno con el siguiente formato (deben aparecer siempre los tres tokens separados por `;`)[1] [2]

```
"<Dni>;<Apellidos, nombre>;<Calificación>"
```

El constructor recibe el nombre de la asignatura y el array de `String` descrito anteriormente y para cada elemento en el array deberá crear, si es posible, el alumno con el dni, nombre y calificación extraídas del `String`, y almacenarlos en el array de alumnos. Por ejemplo, para la siguiente entrada:

```
"55343442L;Godoy Molina, Marina;6.31"
```

creará un alumno con DNI `"55343442L"`, nombre `"Godoy Molina, Marina"`, y nota `6.31`, que será añadido al array de alumnos.

Si no fuera posible crear un determinado alumno, entonces deberá almacenar esta entrada errónea en otro array de `String` (denominado `errores`) precedido de un comentario que indique cual ha sido el problema por el que no se ha podido crear el alumno. Por ejemplo, ante las entradas:

```
"53553421D;Santana Medina, Petra;-7.1"
"55343442L,Godoy Molina, Marina;6.3"
"342424f2J;Fernandez Vara, Pedro;xxx"
```

se incluirá en el array de `errores` los siguientes `String`:

```
"ERROR. Calificación negativa: 53553421D;Santana Medina, Petra;-7.1"
"ERROR. Faltan datos: 55343442L,Godoy Molina, Marina;6.3"
"ERROR. Nota no numérica: 342424f2J;Fernandez Vara, Pedro;xxx"
```

El siguiente método de la clase `Asignatura` devuelve la calificación del alumno `al` dado, si existe.

- `double getCalificacion(Alumno al) throws AlumnoException;`

Si el alumno no existe lanzará una excepción `AlumnoException` con un mensaje como en el siguiente ejemplo, donde `"Fernandez Vara, Pedro 34242432J"` es la representación textual del alumno que no se encuentra.

```
"El alumno Fernandez Vara, Pedro 34242432J no se encuentra"
```

Los siguientes métodos devuelven el array de alumnos y el array de entradas erróneas respectivamente:

[1] Tanto en el método `split` de la clase `String`, como en el método `useDelimiter` de la clase `Scanner`, se puede utilizar la expresión regular `"\\s*[;]\\s*"` como delimitador de tokens.

[2] En la clase `Scanner`, para poder leer números decimales con el separador punto (ej. `7.1`), se debe usar un objeto `Scanner sc` al que se le envía el mensaje `sc.useLocale(Locale.ENGLISH)`.

- `Alumno[] getAlumnos();`
- `String[] getErrores();`

Además, dispondrá de una representación textual de los objetos de la clase como la que se muestra en el siguiente ejemplo (donde los saltos de línea se han introducido para aumentar la legibilidad), se debe utilizar `StringBuilder` o `StringJoiner` para crear la representación.

```
Algebra: { [Garcia Gomez, Juan 25653443S, Lopez Turo, Manuel 23322443K,
Merlo Martinez, Juana 24433522M, Lopez Gama, Luisa 42424312G],
[ERROR. Calificación negativa: 53553421D;Santana Medina, Petra;-7.1,
ERROR. Faltan datos: 55343442L,Godoy Molina, Marina;6.3,
ERROR. Calificación no numérica: 34242432J;Fernandez Vara, Pedro;2.k] }
```

Por último, el siguiente método devuelve la media de las calificaciones de los alumnos de la asignatura, considerando que si no hay alumnos registrados, entonces este método lanzará la excepción `AlumnoException` con el mensaje `"No hay alumnos"`.

- `double getMedia() throws AlumnoException;`

Aplicación `PruebaAsignatura`

Crea una aplicación (clase distinguida `PruebaAsignatura`) para probar la clase `Asignatura`. En esta clase se crea la asignatura `POO` con tres alumnos con los siguientes datos:

```
DNI: 12455666F Nombre: Lopez Perez, Pedro Nota: 6.7
DNI: 33678999D Nombre: Merlo Gomez, Isabel Nota: 5.8
DNI: 23555875G Nombre: Martinez Herrera, Lucia Nota: 9.1
```

A continuación muestra la *media* de las calificaciones de la asignatura y accede a los alumnos de la asignatura, mostrando por pantalla el DNI de cada uno de ellos. Por último, imprime la calificación del alumno `Lopez Perez, Pedro`. De nuevo ten en cuenta que la excepción `AlumnoException` es de obligado tratamiento a la hora de implementar `PruebaAsignatura`.

Después cambia el nombre del alumno cuya calificación se ha de imprimir por `Lopez Lopez, Pedro`. Ejecuta de nuevo el programa y observa lo que sucede.

Aplicación `Main`

Para finalizar se presenta un ejemplo de uso más completo de las clases `Alumno`, `Asignatura` y `AlumnoException` y la salida correspondiente.

```
import prNotas.AlumnoException;
import prNotas.Alumno;
import prNotas.Asignatura;

public class Main {
    static final String[] als = {
        "25653443S;Garcia Gomez, Juan;8.1",
        "23322443K;Lopez Turo, Manuel;4.3",
        "24433522M;Merlo Martinez, Juana;5.3",
        "53553421D;Santana Medina, Petra;-7.1",
        "55343442L,Godoy Molina, Marina;6.3",
        "34242432J;Fernandez Vara, Pedro;2.k",
        "42424312G;Lopez Gama, Luisa;7.1" };
    public static void main(String[] args) {
        try {
            Asignatura algebra = new Asignatura("Algebra", als);
            try {
                Alumno al1 = new Alumno("23322443k", "Lopez Turo, Manuel");
                Alumno al2 = new Alumno("34242432J", "Fernandez Vara, Pedro");
                System.out.println("Calificacion de " + al1 + ": "
                                   + algebra.getCalificacion(al1));
```

```
                System.out.println("Calificacion de " + al2 + ": "
                                   + algebra.getCalificacion(al2));
            } catch (AlumnoException e) {
                System.err.println(e.getMessage());
            }
            try {
                System.out.printf("Media %4.2f\n", algebra.getMedia());
            } catch (AlumnoException e) {
                System.err.println(e.getMessage());
            }
            System.out.println("Alumnos...");
            for (Alumno alumno : algebra.getAlumnos()) {
                System.out.println(alumno + ": " + alumno.getCalificacion());
            }
            System.out.println("Errores...");
            for (String error : algebra.getErrores()) {
                System.out.println(error);
            }
            System.out.println(algebra);
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
    }
}
```

La salida al ejecutar el programa anterior es:

```
Calificacion de Lopez Turo, Manuel 23322443k: 4.3
El alumno Fernandez Vara, Pedro 34242432J no se encuentra
Media: 6.20
Alumnos...
Garcia Gomez, Juan 25653443S: 8.1
Lopez Turo, Manuel 23322443K: 4.3
Merlo Martinez, Juana 24433522M: 5.3
Lopez Gama, Luisa 42424312G: 7.1
Errores...
ERROR. Calificación negativa: 53553421D;Santana Medina, Petra;-7.1
ERROR. Faltan datos: 55343442L,Godoy Molina, Marina;6.3
ERROR. Calificación no numérica: 34242432J;Fernandez Vara, Pedro;2.k
Algebra: { [Garcia Gomez, Juan 25653443S, Lopez Turo, Manuel 23322443K,
Merlo Martinez, Juana 24433522M, Lopez Gama, Luisa 42424312G],
[ERROR. Calificación negativa: 53553421D;Santana Medina, Petra;-7.1,
ERROR. Faltan datos: 55343442L,Godoy Molina, Marina;6.3,
ERROR. Calificación no numérica: 34242432J;Fernandez Vara, Pedro;2.k] }
```

Ejercicio 2. (proyecto prNotasInterfazMedia)

Se desea modificar la clase `Asignatura` del proyecto `prNotas` para que sea posible indicar la forma de calcular la media que se necesite en cada momento. Para ello, se añadirá el siguiente método a la clase `Asignatura`, considerando que `CalculoMedia` será especificada a continuación:

- `double getMedia(CalculoMedia calc) throws AlumnoException;`

De tal forma que este método calculará la nota media de los alumnos invocando al método `calcular` proporcionado por la clase recibida como parámetro, que implementa la **interfaz** `CalculoMedia`.

Interfaz `CalculoMedia`

Se debe definir la **interfaz** `CalculoMedia` que especifique el siguiente método.

- `double calcular(Alumno[] alumnos) throws AlumnoException;`

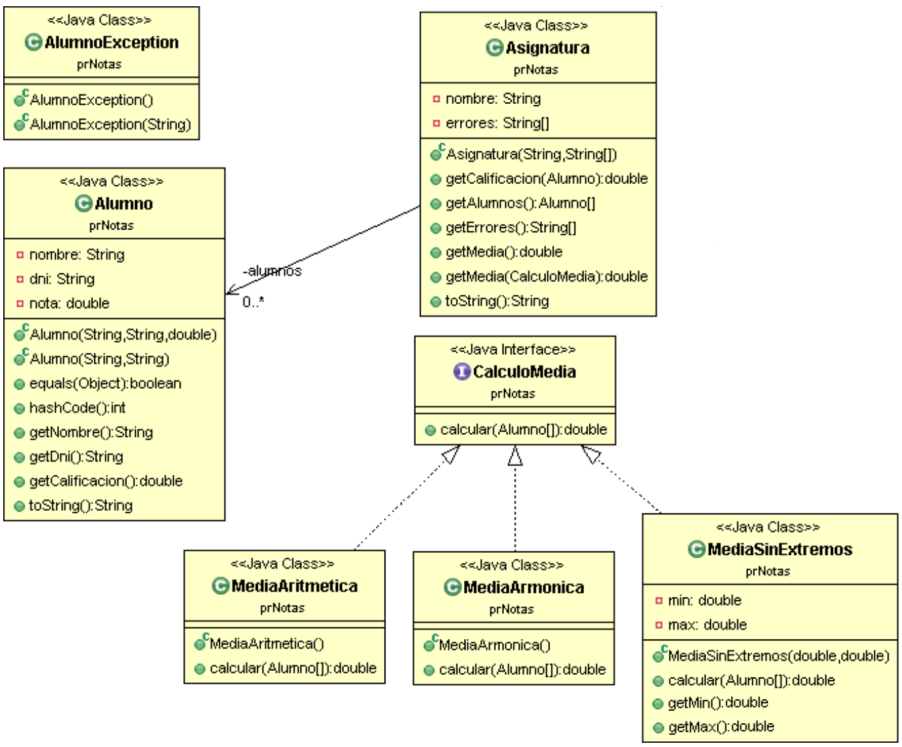

Figura 2: Diagrama de clases UML

Clases `MediaAritmetica`, `MediaArmonica` y `MediaSinExtremos`

Además, se deberán definir las clases `MediaAritmetica`, `MediaArmonica` y `MediaSinExtremos` que implementen la interfaz `CalculoMedia`, según las siguientes especificaciones:

- El método `calcular` proporcionado por la clase `MediaAritmetica` calcula la media aritmética de n alumnos según la siguiente ecuación. En caso de que no haya alumnos, lanzará una excepción `AlumnoException` con el mensaje `"No hay alumnos"`:

$$media = \frac{1}{n} \sum_{i=0}^{n-1} calificacionAlumno_i$$

- El método `calcular` proporcionado por la clase `MediaArmonica` calcula la media armónica de los k alumnos con notas superiores a 0 según la siguiente ecuación. En caso de que no haya alumnos que cumplan el requisito especificado, lanzará una excepción `AlumnoException` con el mensaje `"No hay alumnos"`:

$$media = \frac{k}{\sum_{j=0}^{k-1} \frac{1}{calificacionAlumno_j}}$$

- El método `calcular` proporcionado por la clase `MediaSinExtremos` calcula la media aritmética de aquellos valores comprendidos entre los extremos dados, ellos incluidos. En caso de que no haya alumnos que cumplan el requisito especificado, lanzará una excepción `AlumnoException` con el mensaje `"No hay alumnos"`. Los valores extremos se pasarán en el constructor de la clase y serán almacenados en sendas variables de instancia, para ser utilizados en el método `calcular`. Nótese que la clase `MediaSinExtremos` también proporciona dos métodos para consultar el valor mínimo y el valor máximo del rango.
  - `double getMin();`
  - `double getMax();`

Aplicación `Main`

Para finalizar se presenta un ejemplo de uso más completo de las clases anteriormente especificadas y la salida correspondiente.

```
import prNotas.AlumnoException;
import prNotas.Alumno;
import prNotas.Asignatura;
import prNotas.CalculoMedia;
import prNotas.MediaAritmetica;
import prNotas.MediaArmonica;
import prNotas.MediaSinExtremos;

public class Main {
    static final String[] als = {
        "25653443S;Garcia Gomez, Juan;8.1",
        "23322443K;Lopez Turo, Manuel;4.3",
        "24433522M;Merlo Martinez, Juana;5.3",
        "53553421D;Santana Medina, Petra;-7.1",
        "55343442L,Godoy Molina, Marina;6.3",
        "34242432J;Fernandez Vara, Pedro;2.k",
        "42424312G;Lopez Gama, Luisa;7.1" };
    public static void main(String[] args) {
        try {
            Asignatura algebra = new Asignatura("Algebra", als);
            try {
                Alumno al1 = new Alumno("23322443k", "Lopez Turo, Manuel");
                Alumno al2 = new Alumno("34242432J", "Fernandez Vara, Pedro");
                System.out.println("Calificacion de " + al1 + ": "
                                   + algebra.getCalificacion(al1));
                System.out.println("Calificacion de " + al2 + ": "
                                   + algebra.getCalificacion(al2));
            } catch (AlumnoException e) {
                System.err.println(e.getMessage());
            }
            try {
                CalculoMedia m1 = new MediaAritmetica();
                CalculoMedia m2 = new MediaArmonica();
                MediaSinExtremos m3 = new MediaSinExtremos(5.0, 9.0);
                System.out.println("Media aritmética: " + algebra.getMedia(m1));
                System.out.println("Media armónica: " + algebra.getMedia(m2));
                System.out.println("Media de valores en ["+m3.getMin()+", "+m3.getMax()+"]: "
                                   + algebra.getMedia(m3));
            } catch (AlumnoException e) {
                System.err.println(e.getMessage());
            }
            System.out.println("Alumnos...");
            for (Alumno alumno : algebra.getAlumnos()) {
                System.out.println(alumno + ": " + alumno.getCalificacion());
            }
            System.out.println("Errores...");
            for (String error : algebra.getErrores()) {
                System.out.println(error);
            }
            System.out.println(algebra);
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
    }
}
```

La salida al ejecutar el programa anterior es:

```
Calificacion de Lopez Turo, Manuel 23322443k: 4.3
El alumno Fernandez Vara, Pedro 34242432J no se encuentra
Media aritmética: 6.199999999999999
Media armónica: 5.83482277207447
Media de valores en [5.0, 9.0]: 6.833333333333333
Alumnos...
Garcia Gomez, Juan 25653443S: 8.1
Lopez Turo, Manuel 23322443K: 4.3
Merlo Martinez, Juana 24433522M: 5.3
Lopez Gama, Luisa 42424312G: 7.1
Errores...
ERROR. Calificación negativa: 53553421D;Santana Medina, Petra;-7.1
ERROR. Faltan datos: 55343442L,Godoy Molina, Marina;6.3
ERROR. Calificación no numérica: 34242432J;Fernandez Vara, Pedro;2.k
Algebra: { [Garcia Gomez, Juan 25653443S, Lopez Turo, Manuel 23322443K,
Merlo Martinez, Juana 24433522M, Lopez Gama, Luisa 42424312G],
[ERROR. Calificación negativa: 53553421D;Santana Medina, Petra;-7.1,
ERROR. Faltan datos: 55343442L,Godoy Molina, Marina;6.3,
ERROR. Calificación no numérica: 34242432J;Fernandez Vara, Pedro;2.k] }
```