**PRÁCTICA 22 | ÁRBOLES**

# Introducción

Después de una extensa pausa estamos volviendo a la clase de Estructura de Datos.

De acuerdo a lo que se pide en la descripción de la práctica en la actividad 2, tengo en mente agregar una clase que especialice al Árbol Binario como tal; de esta manera poder agregar métodos con menos preocupación correspondientes a la manipulación de un árbol que tiene expresiones aritméticas, dando oportunidad también para futuras extensiones en caso de necesitarse.

Tengo un poco a ventaja el haber llevado el semestre pasado matemáticas discretas, donde vimos muchos conceptos acerca de árboles como los que se piden en la actividad 3. Así pues, espero poder atender a cada uno de los puntos de la práctica de la mejor manera haciendo un correcto análisis y diseño de lo que voy a codificar.

# Desarrollo

Al momento de comenzar a pensar lo que iba a realizar me detuve un rato analizando primeramente si era buena idea lo planteado en la introducción: hacer un nuevo tipo de árbol para manejar expresiones aritméticas como la del ejemplo en la práctica. Terminé haciéndolo y aunque creo era la mejor decisión también me tomé un tiempo en pensar si era mejor usar composición o herencia para esta nueva clase. Finalmente me decidí por usar herencia y agregar esta nueva clase de nombre **ArbolExpresion.**

De acuerdo a lo que se pedía en el punto uno, que tenía relación con la identificación de operadores y operandos, recordé que había parte del código que necesitaba en una clase anteriormente hecha llamada **ExpresionAritmetica**. Así pues, hice uso de una función ya hecha ahí y agregué dos más para cubrir las necesidades de la clase ArbolExpresion, pero que también dejan abierta la posibilidad de usarse en otro lugar.

A causa de algunas confusiones, me resultó algo tardado hacer el método que solicitaba las variables al usuario y las cambiaba en un nuevo árbol, pero finalmente se logró. Los demás puntos de la práctica estuvieron sencillos de sacar, con una gran excepción para calcular la altura del árbol que fue el método que me hizo batallar más en la práctica, en especial por la “doble recursión”.

A continuación, se mencionan los archivos creados o modificados el paquete en que se ubican en el proyecto:

## ArbolBinario.java (paquete ednolineal).

## ArbolExpresion.java (paquete ednolineal).

## NodoDoble.java (paquete catalogos).

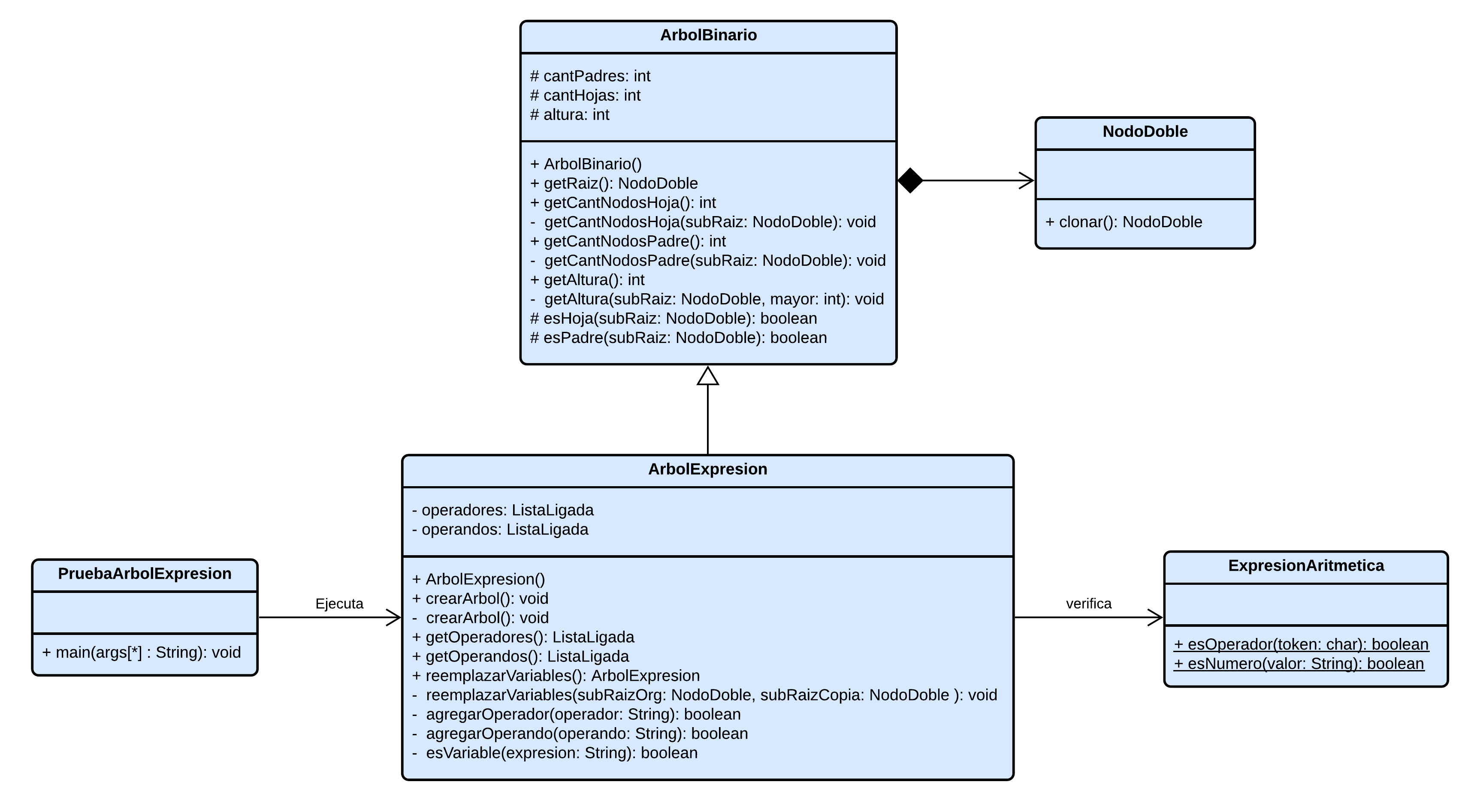
## ExpresionAritmetica.java (paquete matemáticas).

## PruebaArbolExpresion.java (paquete interfaces).

La clase anterior contiene las pruebas de esta práctica.

# Diagrama de clases de la estructura de la práctica

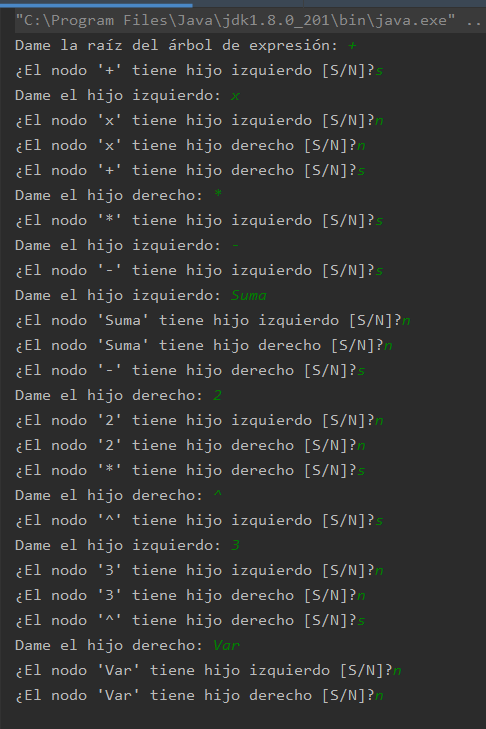
En el diagrama de clases se detalla lo que se hizo en cada una de las clases mencionadas anteriormente. Cabe destacar que para las clases que ya existían en el proyecto solamente se ilustran los métodos o atributos que fueron agregados. En cambio, para las clases agregadas se muestra en su totalidad métodos y atributos que fueron incluidos.

****

**Para mayor información sobre los métodos y las clases creadas, vaya a las siguientes rutas dentro del proyecto:**

* **/docs/ednolineal/ArbolBinario.html**
* **/docs/ednolineal/ArbolExpresion.html**
* **/docs/catalogos/NodoDoble.html**
* **/docs/matematicas/ExpresionAritmetica.html**
* **/docs/interfaces/PruebaArbolExpresion.html**

# Capturas del programa funcionando



# 

# Conclusiones

Puede notarse bastante que para manejar árboles se tiene que tener un buen dominio de la recursión, de lo contrario puede resultar muy confuso. Ese aspecto me hizo batallar mucho a la hora de obtener la altura del árbol. En cuanto a los demás puntos, la recursión no fue muy complicado comprenderla y aplicarla, me demoro más pensar qué métodos iba a necesitar para que todo quedara bien estructurado.