**Teoría Computacional**

**Práctica 4. Analizador léxico (AFD parte 2)**

**Objetivo:** Realizar la implementación de AFD´s que aceptan expresiones regulares, y aplicarlos para el diseño e implementación de un analizador léxico. Cualquier lenguaje de programación puede ser utilizado (C, C++, C#, Java, Python).

**Desarrollo:**

**Parte I.** Diseñar e implementar un Analizador léxico que, dado un programa fuente escrito en lenguaje Java, identifique los valores numéricos (constantes) en sus diferentes tipos (int, float, double) y notaciones.



Para resolver el problema se sugiere construir un AFD que opere los símbolos especiales definidos en la tabla 2, y con tantos estados de aceptación como las diferentes formas numéricas que deben ser reconocidas (aceptadas) que en este caso son cinco: tres formas enteras y dos reales (tabla 3).





**Tabla 3. Estados de aceptación.**

Partiendo de nuestra definición formal de AFD:

AFD = ( Σ, Q, q0, δ ,F)

Tenemos que:

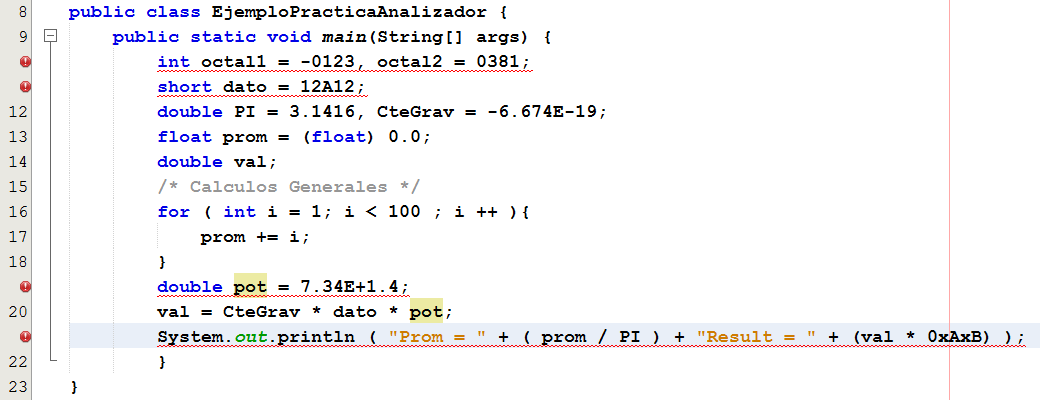
Σ = { 0, … , F, +, - , b , ; , . , E, x }

F = { C0, C1, C2, C3, C4, C5 }

Encontrar: Q, q0, δ (o el diagrama de transición).

**Entrada al Analizador léxico:** La entrada al programa será un archivo de texto con la extensión .java el cual contiene un programa fuente en Java en donde tenemos constantes numéricas.

Por ejemplo:



Programa **EjemploPracticaAnalizador.java** usado como entrada al Analizador léxico.

**Salida del analizador léxico:** La salida del programa serán mensajes de texto que indiquen si hay algún error en el uso de una constante numérica desde el punto de vista léxico.

La salida del programa, teniendo como entrada el archivo Numeros.java, es la siguiente:

**Error en línea 10.**

**Error en línea 11.**

**Error en línea 19.**

**Error en línea 21.**

Si el archivo **EjemploPracticaAnalizador.java** no tuviera ningún error la salida del Analizador sería:

**No hay errores de análisis léxico en las constantes numéricas del archivo Numeros.java.**

**Parte 2.** Agregar los siguientes estados de aceptación al AFD del analizador léxico:

|  |  |  |
| --- | --- | --- |
| **Estados de aceptación (Parte II)** | | |
| **Símbolo** | **Significado** | **Ejemplo** |
| C5 | Símbolos especiales I | Operadores de comparación (<,>,=, >=, <=, ==, !=) y operador de asignación (=) |
| C6 | Símbolos especiales II | Operadores aritméticos (+, -, \*, /. %) |
| C7 | Palabras reservadas de Java | *class, main, public, private, void, int*, etc. |
| C8 | Identificador válido en Java | *var1, dato, prom\_final, info,* etc. |
| C9 | Comentarios | Para una sola línea: //  Para más de una línea: /\* … \*/ |

**Tabla 4. Estados de aceptación Parte II.**

**Evaluación:**

Cada estado de aceptación (tabla 3 y tabla 4), que funcione correctamente, vale 1 punto.

**Presentación de la práctica:**

* Presentar el programa en ejecución: todos los estados de aceptación deben estar en el mismo programa.
* Presentar la definición del AFD usado en esta práctica:

AFD = (Σ, Q, q0, δ, F)

* El diagrama de estados del AFD se tomará como la función de transición (δ).
* El diagrama del AFD utilizado en esta práctica debe corresponder con el diseño del código.
* No utilizar expresiones regulares para esta práctica (clases de Java, Python, etc.); es requisito programar el AFD del inciso anterior.
* Sustentar un breve examen oral acerca del código y de los conceptos de Teoría Computacional empleados en esta práctica.
* No es necesario entregar reporte escrito.
* Prácticas copiadas serán canceladas.
* Presentar en forma individual o por equipo (máximo dos personas).
* Presentar práctica en laboratorio, en salón de clase, o en cubículo de profesor.

**Fecha de entrega:**

Fecha de caducidad: miércoles 9 de mayo.

**Actividad opcional para la Práctica 3**

**Hacer una versión gráfica del AFD del Analizador léxico para observar la animación o visualización de alguno de los estados de aceptación. Por ejemplo, para el estado de aceptación C0 se deberá ver la ejecución del AFD de un estado a otro, de acuerdo con el símbolo del alfabeto que lleve a cabo la transición. Al final de procesar una palabra se debe observar el estado final (aceptación o no aceptación). Valor: Hasta un máximo de 10 participaciones extra.**