Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Lenguajes formales y de programación



**Práctica 1  
Manual Técnico**

Joshua Estuardo Franco Equité

Carné: 201708845

Sección: B-

Manual Técnico

**Requisitos del programa**

Para el correcto funcionamiento del programa es necesario tener cubiertos una serie de

requisitos, tanto hardware como software.

*Los requisitos previos de software instalado para poder ejecutar el programa serían:*

- Windows XP o superior

-.NET 3.5

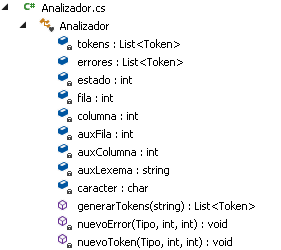
*Los requisitos mínimos de hardware serían los siguientes*:

- Procesador, al menos, Pentium 166 MHz, PowerPC 160 MHz, o UltraSparc 166 MHz

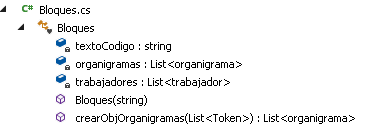
- Al menos 512 MB de memoria RAM

***Clases utilizadas***

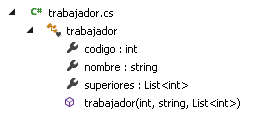
**Analizador:** Esta clase es la que analiza el texto ingresado, y caracter por caracter comprueba si pertenece a la gramática permitida por el proyecto.



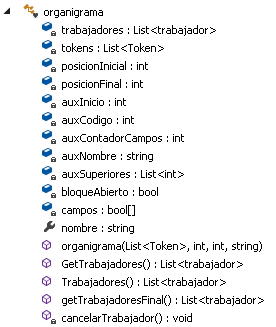
**Bloques:** Esta clase es la que separa todo el texto de entrada en distintos organigramas en el caso de que haya más de uno, además del que comprueba que los organigramas estén correctos.



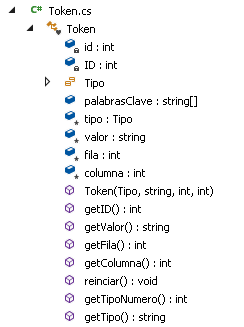
**Trabajador:** Es una clase donde se van creando objetos de tipo trabajador por cada bloque de trabajador.



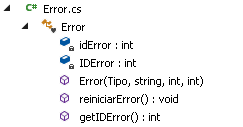
**Organigrama:** Donde se guarda el código de cada bloque organigrama, luego se evalúa el código y genera objetos de tipo trabajador donde va agregando cada nodo si cumple la sintaxis correcta.



**Token:** Cada objeto de tipo token es un elemento léxico que está en el texto de entrada, el cual se clasifica de distintas maneras.

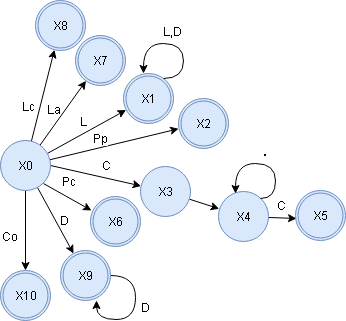


**Error:** Clase hija de token, la cual comparte varios atributos pero cambia el id debido a que no comparte el conteo con los tokens.



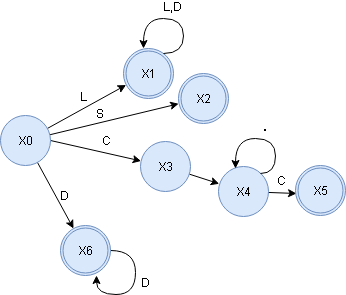
***Autómata finito determinista:***

L = {A, B, C, …, Z, a, b, c, …,z}  
D = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}  
C = { ‘”’ }  
Pc = { ‘;’ }  
Pp = { ‘:’ }  
Co = { ‘,’ }  
La = { ‘{’ }  
Lc = { ‘}’ }



Este autómata se hizo separando cada carácter como un estado distinto, teniendo en cuenta que cada signo tenía su propio conjunto, y que cada signo tenía su propio nodo de aceptación, pero este se puede simplificar.

L = {A, B, C, …, Z, a, b, c, …,z}  
D = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}  
C = { ‘”’ }  
S = { ‘;’, ‘:’, ‘,’, ‘{’, ‘}’ }



En este autómata simplificado se toman varios símbolos que solo requieren longitud uno para la aceptación y se unen en un mismo conjunto.