

Memoria Final

Entrega (Parcial)

Analizador lenguaje BSL

Realizado por:

Marcelino Tena Blanco; NIA: 100383266

Índice

1. Introducción
2. Construcción del analizador léxico
   1. Comentarios
   2. Operadores Aritmeticológicos
   3. Declaración de variables e inicialización
   4. Declaración de funciones
   5. Bucles y sentencias
   6. Estructuras structs
3. Introducción

El siguiente documento trata sobre que la realización y el procedimiento para llegar a la creación de un analizador léxico para el lenguaje BSL, el cual está especificado su formato en aula global. Para explicar el proceso llevado para crear el analizador léxico, iré comentándolo mediante estructuras según el siguiente orden:

* Los dos tipos de comentarios
* Operadores Aritmeticológicos
* Declaración de variables e inicialización de variables
* Declaración de funciones
* Bucles y sentencias
* Estructuras structs

Una vez que realicé el analizador léxico, he estado realizando parte del analizador sintáctico pero todavía no es funcional, ya que faltan bastantes estructuras. Por ahora reconoce hasta declaración de variables e inicialización, aunque no hace nada con ellas, pero puede reconocer su estructura y saber si está bien o no.

Para acabar con la introducción, hay que añadir que se han implementado los ejemplos dados en el enunciado como pruebas de la práctica. Conozco que puede ser que me deje algunas estructuras más explícitas sin comprobar, pero por varias circunstancias no he podido agregar más. El programa principal de esta entrega parcial es AnalizadorLexico, el cual es el programa dado en el enunciado para comprobar el correcto funcionamiento del analizador léxico sin tener que ejecutar el parser del cup.

1. Construcción del analizador léxico

A continuación iré estructura por estructura detallando cada una de las partes compuestas por el analizador léxico:

* 1. Comentarios

Para los comentarios debemos tener en claro que existen dos formas para realizarlos, una es mediante el símbolo # que marca como comentario todo lo que queda de línea, y la otra es mediante un símbolo de abrir comentario, que es un conjunto de símbolos (<!--) y un símbolo de cerrar comentario (- ->). Para que el analizador léxico sea capaz de reconocer tanto uno como otro, tenemos una expresión regular que tiene dos opciones que son dos expresiones regulares, donde una es la que permite que sea comentario todo lo que falte de línea a partir del símbolo # y el otro solo comprende que es comentario lo que hay dentro de los símbolos antes dichos de abierto y cerrado.

Esta es la forma de reconocerlo, en el caso de que tenga que hacer algo, directamente, no hace nada, ya que es un comentario.

* 1. Operadores Aritmeticológicos

En el caso para que el analizador reconozco los símbolos aritmeticológicos lo único que he realizado es agregarlos cada uno como tokens, ya que no es necesario hacer otra cosa. En el caso de, por ejemplo, exista un igual mayor o un similar, será el analizador sintáctico quien tendrá la capacidad de reconocer este conjunto de símbolos, pero en el reconocedor léxico solo los añadimos como nuevos tokens por separado. Los tokens añadidos en este caso son:

* Símbolos aritméticos
  + “+”, “-“, “/”, “\*”, “=”
* Símbolos lógicos
  + “<”, “>”, “NOT”, “AND”, “OR” , “!”, “&”, “|”
  1. Declaración de variables e inicialización
  2. Declaración de funciones
  3. Bucles y sentencias
  4. Estructuras structs