06/03/2023 Sección: D02

Seminario de Solución de Problemas de Traductores de Lenguaje II

Tarea 5:

Analizador Sintáctico Con Arbol

Nombre del maestro:

López Franco Michel Emanuel

Nombre del alumno:

Ramos Calderón Christian Daniel

Código del alumno:

216577014

Índice

Contents

3 | Seminario de Solución de Problemas de Traductores de Lenguaje II

Introducción

En esta actividad demos de desarrollar el árbol de LR de tal manera que podamos ver el camino que lleva el análisis de la sintaxis y que es lo que este general al final.

Desarrollo

Así que tenemos que desarrollar tanto como el analizador léxico como el analizador Sintáctico siendo posible que el programa pueda leer la siguiente gramática que son las reglas proporcionadas por el maestro.
Básicamente para el desarrollo del programa anteriormente utilizamos la clase EP con la cual manejamos los Terminales, No terminales y el Final de la cadena, de esta manera solo cambiamos el cómo están las reglas porque teníamos otras para una gramática en especial ahora con esto podríamos integrar cualquier tipo de gramática de un lenguaje C. Y para este ultimo programa debemos de realizar el árbol LR de tal manera que podamos ver el camino que lleva el análisis de la sintaxis. De la misma manera se utilizarán las reglas que proporciono el maestro. Las reglas proporcionadas por el maestro son las siguientes:

- R1 programa ::= Definiciones
- R2 Definiciones ::= \e
- R3 Definiciones ::= Definicion Definiciones
- R4 Definicion ::= DefVar
- R5 Definicion ::= DefFunc
- R6 DefVar ::= tipo identificador ListaVar ;
- R7 ListaVar ::= \e
- R8 ListaVar ::= , identificador ListaVar
- R9 DefFunc ::= tipo identificador (Parametros) BloqFunc
- R10 Parametros ::= \e
- R11 Parametros ::= tipo identificador ListaParam
- R12 ListaParam ::= \e
- R13 ListaParam ::= , tipo identificador ListaParam
- R14 BloqFunc ::= { DefLocales }
- R15 DefLocales ::= \e
- R16 DefLocales ::= DefLocal DefLocales
- R17 DefLocal ::= DefVar
- R18 DefLocal ::= Sentencia
- R19 Sentencias ::= \e
- R20 Sentencias ::= Sentencia Sentencias
- R21 Sentencia ::= identificador = Expresion ;
- R22 Sentencia ::= if (Expresion) SentenciaBloque Otro
- R23 Sentencia ::= while (Expresion) Bloque
- R24 Sentencia ::= return ValorRegresa ;
- R25 Sentencia ::= LlamadaFunc ;
- R26 Otro ::= \e
- R27 Otro ::= else SentenciaBloque

- R28 Bloque ::= { Sentencias }
- R29 ValorRegresa ::= \e
- R30 ValorRegresa ::= Expresion
- R31 Argumentos ::= \e
- R32 Argumentos ::= Expresion ListaArgumentos
- R33 ListaArgumentos ::= \e
- R34 ListaArgumentos ::= , Expresion ListaArgumentos
- R35 Termino ::= LlamadaFunc
- R36 Termino ::= identificador
- R37 Termino ::= entero
- R38 Termino ::= real
- R39 Termino ::= cadena
- R40 LlamadaFunc ::= identificador (Argumentos)
- R41 SentenciaBloque ::= Sentencia
- R42 SentenciaBloque ::= Bloque
- R43 Expresion ::= (Expresion)
- R44 Expresion ::= opSuma Expresion
- R45 Expresion ::= opNot Expresion
- R46 Expresion ::= Expresion opMul Expresion
- R47 Expresion ::= Expresion opSuma Expresion
- R48 Expresion ::= Expresion opRelac Expresion
- R49 Expresion ::= Expresion opIgualdad Expresion
- R50 Expresion ::= Expresion opAnd Expresion
- R51 Expresion ::= Expresion opOr Expresion
- R52 Expresion ::= Termino

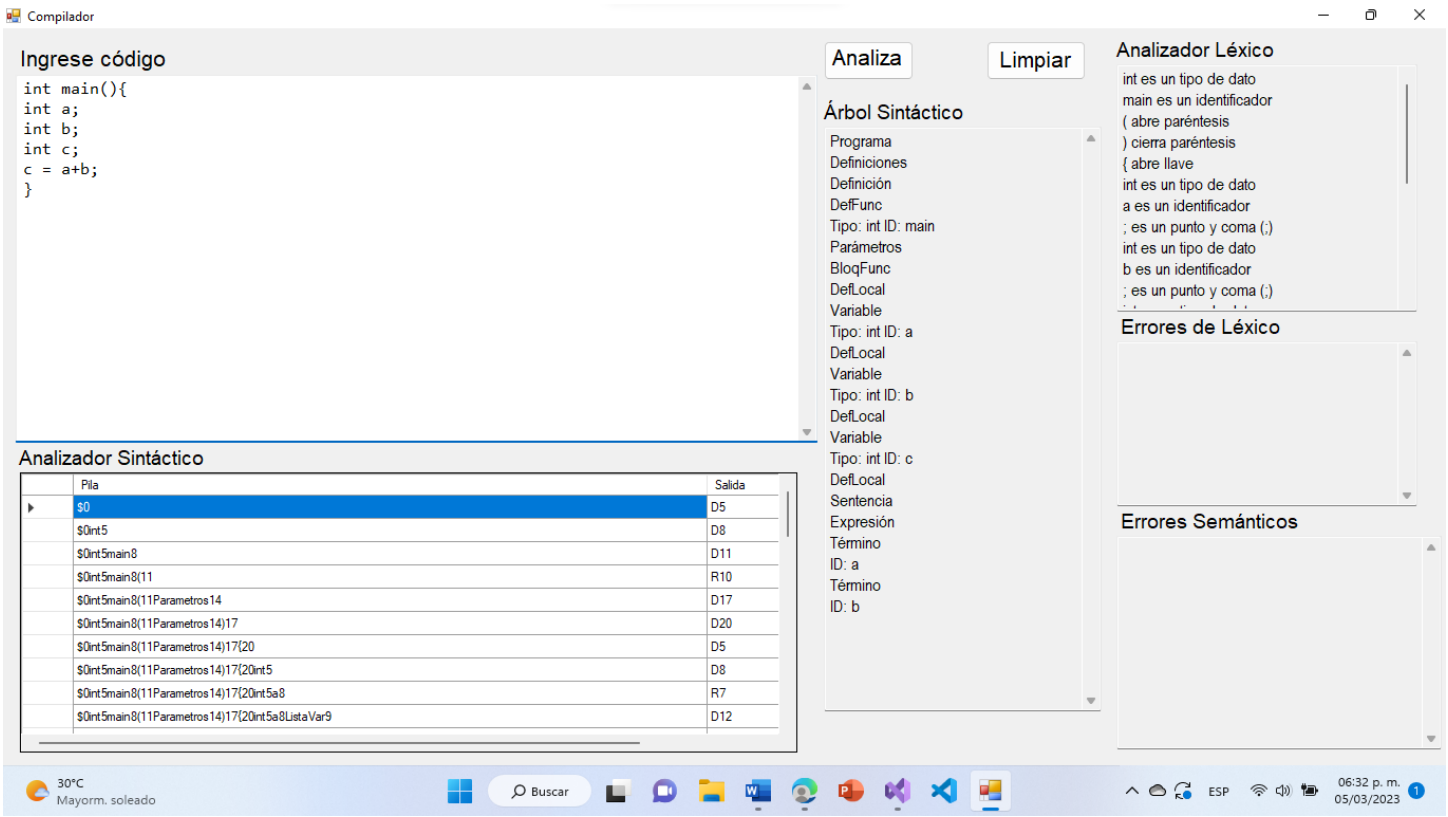
Para estos casos utilizaremos los siguientes códigos para validar lo que es la función del programa y de tal manera mostrarlo con las siguientes capturas de pantalla.
Estos son los codigos que pondremos a prueba en el analizador:

1.
```
int suma(float a,float b){
        return a+b;
}
int main(){
float a;
float b;
float c;
c = suma(a,b);
}
```

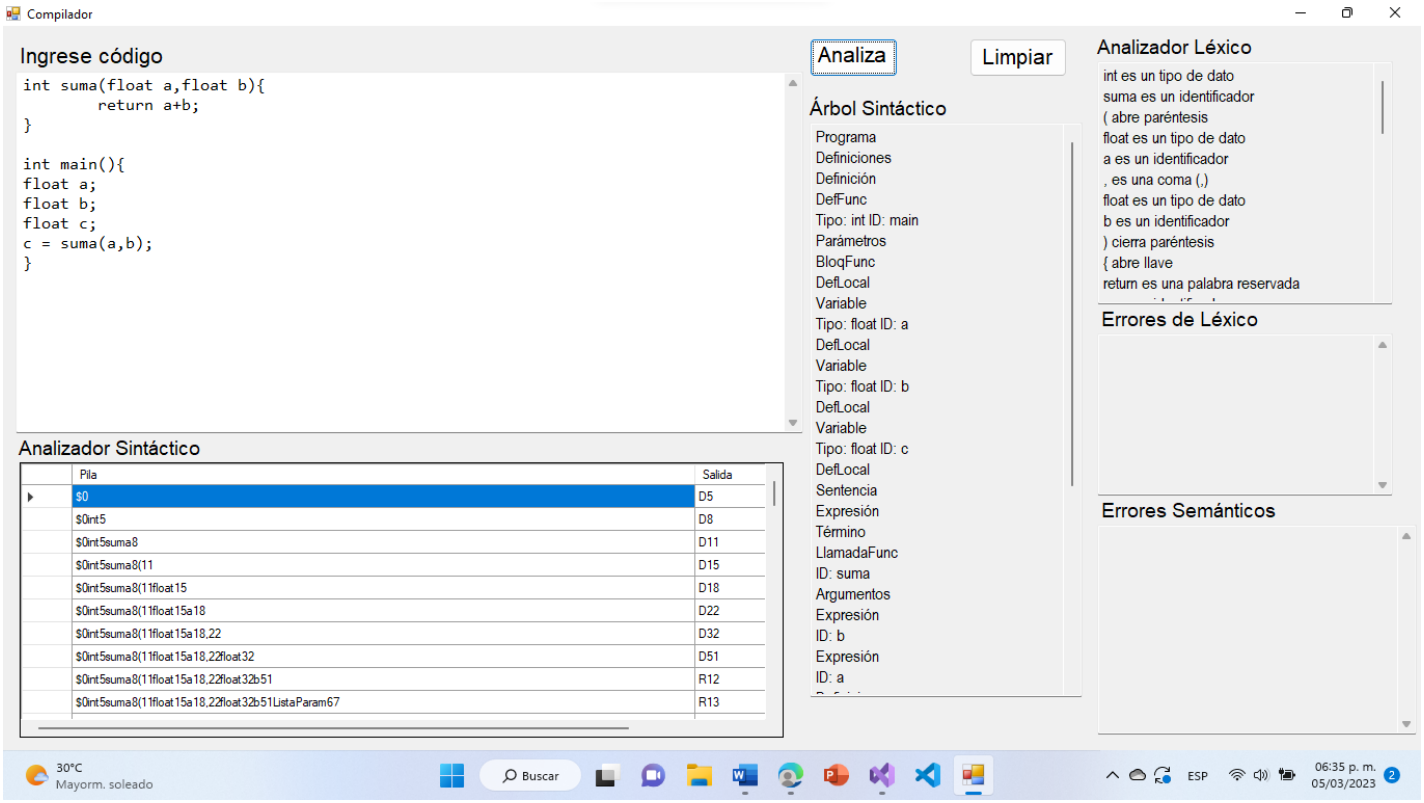
2.
```
int main(){
float a;
float b;
float c;
c = suma(a,b);
}
```

Capturas de pantalla

Primer código:



Para el Segundo código manejaremos lo que es la función de suma:



Conclusion

Con este último programa terminamos lo que es la materia, realmente lo difícil o el reto es la manera en que interpretas los datos y como los manejas, es tener un orden completo, pero a la vez debemos de ser cocientes de los procesos que se llevan de tal manera asiendo un orden. Del anterior programa a este lo que hicimos fue separar las clases para tener un programa más limpio y el uso de los nodos para general el árbol aun que la vista no es tan grafico más en escrito aun que cumple con la solución.

Bibliografía

No valida en este trabajo