Técnicas de Compilación

Trabajo Práctico Nro. 1

Profesor: Maximiliano A. Eschoyez

Alumnos: Cantarelli, Sofía
Figueroa, Javier

Consigna

Dado un archivo de entrada en lenguaje C, se debe generar como salida el Árbol Sintáctico (ANTLR) correcto. Para lograr esto se debe construir un parser que tenga como mínimo la implementación de los siguientes puntos:

1. Reconocimiento de un bloque de código, que puede estar en cualquier parte del código fuente, controlando balance de llaves.
2. Verificación de:
   • declaraciones y asignaciones,
   • operaciones aritméticológicas,
   • declaración/llamada a función.
3. Verificación de las estructuras de control if, for y while.
4. Ante el primer error léxico o sintáctico el programa deberá terminar.

Desarrollo de la solución propuesta

La idea principal para abordar la solución a la problemática abordada es la generación de una aplicación que posea un generador Parser, utilizando ANTLR para poder construir y recorrer árboles de sintaxis. La aplicación reconoce una serie de símbolos e identifica estructuras de control como if, while y for; además de asignaciones, declaraciones, funciones y llamadas a funciones. Estas reglas se declaran con un orden por ejemplo, un programa esta compuesto de instrucciones y estas instrucciones presentan una instrucción que pueden ser las mencionadas con anterioridad.

prog : instrucciones;

instrucciones : instruccion instrucciones
|
;

instruccion : declaracion PYC
| asignacion PYC
| bucles
| funcion
| llamadaFuncion PYC
;

Los símbolos que reconocemos son separadores y aritméticos-lógicos, ademas de palabras reservadas propias del lenguaje C, como tipos de datos, nombres de las estructuras de control. También se filtraran espacios en blanco, saltos, fin de oración. Los tipos de datos trabajados son int, char, double y void.

Acrónimos

- asig : utilizada para asignar el valor
- exprLogica: expresión Lógica
- exprComparab: expresión comparable
- operanComparab: operación utilizada para la comparación
- exprArit: expresión aritmética

Ejemplos utilizados para demostrar la funcionalidad de nuestro programa:

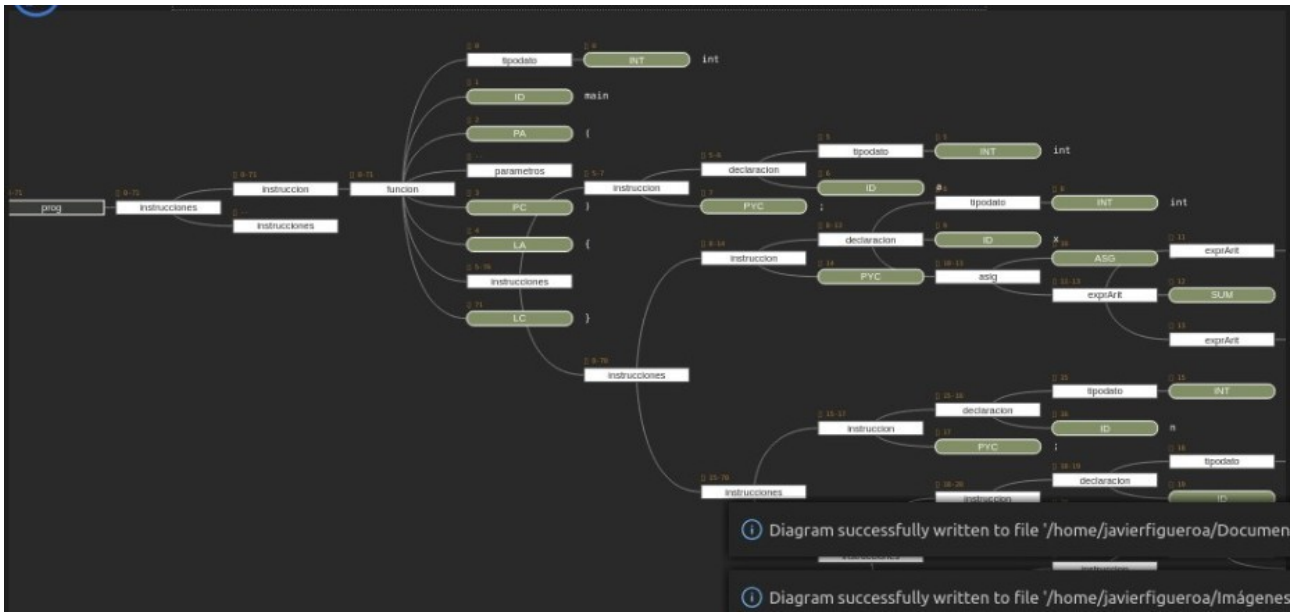
1.

```
void main () {
	int a;
	int x = 5+1;
	int n;
	double g;
	for(int m = 0; m <=5; m++){
		m = n + 2 ;
		if (m<0) {
			int h;
		}
		if (n< 1) {
			a = 3+5;
		}
	}
}
```

2.

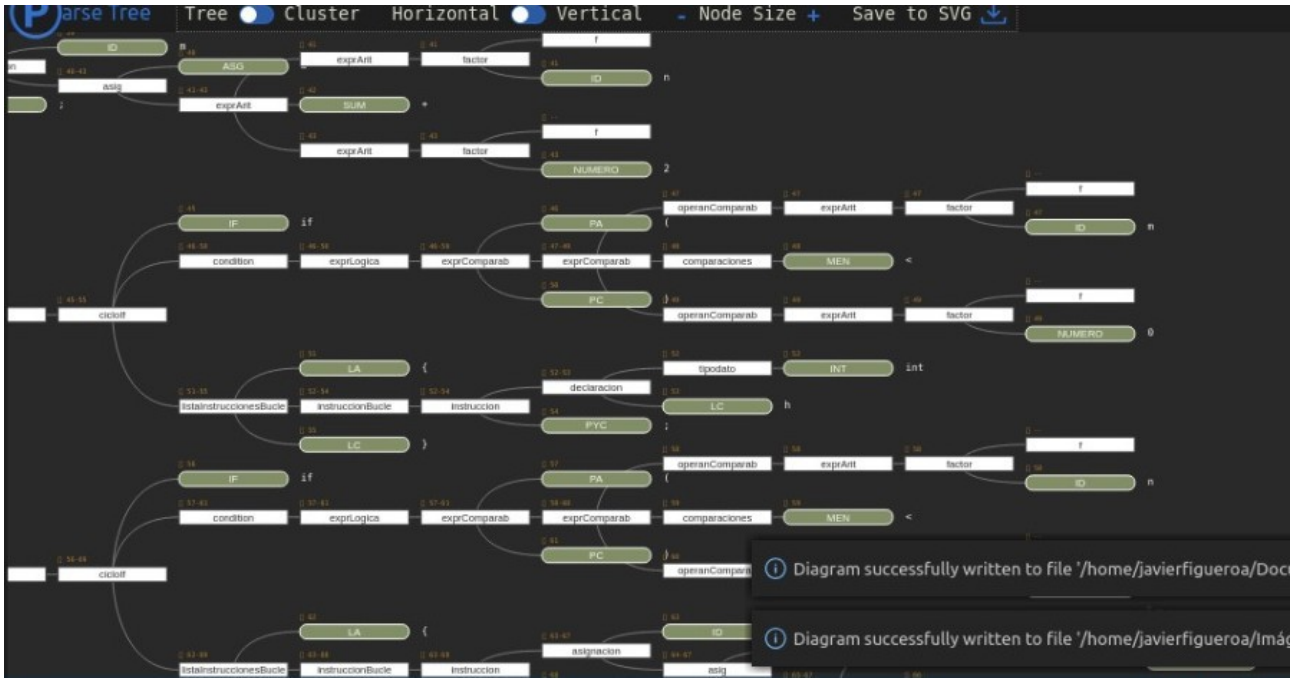
```
int main () {
	int a;
	sumar();
}

void sumar(){
	int b = 2+4;
}
```

Anexo


prog
instrucciones
instruccion
funcion
tipodato
INT
int
ID
main
PA
parametros
IPC
LA
instrucciones
LC
declaracion
PYC
asig
ASG
exprArit
SUM
Diagram successfully written to file '/home/javierfigueroa/Documen
Diagram successfully written to file '/home/javierfigueroa/Imágenes


FOR
for
PA
declaracion
ID
ASG
asig
exprArit
factor
NUMERO
PYC
INT
int
DOUBLE
tipodato
condition
exprLogica
exprComparab
operanComparab
comparaciones
MENIG
bucles
cicloFor
INCREMENT
LA
instruccionBucle
instruccion
asignacion
Diagram successfully written to file '/home/javierfigueroa/Docu
Diagram successfully written to file '/home/javierfigueroa/Imág


Diagram successfully written to file '/home/javierfigueroa/Docun
Diagram successfully written to file '/home/javierfigueroa/Imáge


Parse Tree
Tree
Cluster
Horizontal
Vertical
Node Size
Save to SVG
Diagram successfully written to file '/home/javierfigueroa/Doc
Diagram successfully written to file '/home/javierfigueroa/Imág

Bibliografía

- http://lsi.vc.ehu.es/asignaturas/FdIc/labs/a1/htm/asig.html
- http://www.lcc.uma.es/~galvez/ftp/libros/Compiladores.pdf
- http://ce.sharif.edu/courses/94-95/1/ce414-2/resources/root/Text%20Books/Compiler%20Design/Alfred%20V.%20Aho,%20Monica%20S.%20Lam,%20Ravi%20Sethi,%20Jeffrey%20D.%20Ullman-Compilers%20-%20Principles,%20Techniques,%20and%20Tools-Pearson_Addison%20Wesley%20(2006).pdf
- http://oa.upm.es/32288/1/PFC_JOSE_BARBERA_TORRALVO.pdf
- http://www.lsi.us.es/~troyano/documentos/guia.pdf