Lenguajes de Programación y Procesadores de Lenguaje

Práctica 3

Generación de código para una calculadora usando ANTLR

Supongamos una pequeña calculadora que realiza operaciones aritméticas sencillas según la gramática:

ecuacion	→ **id** = expresion ;
expresion	→ termino ("+" termino | "-" termino)*
termino	→ factor ("*" factor | "/" factor)*
factor	→ "(" expresion ")" | dato
dato	→ **num** | **id** | **-** **num** | **-** **id**

Se pide implementar un traductor mediante la herramienta ANTLR que traduzca las expresiones aritméticas a código de 3-direcciones, representado mediante cuádruplos de la forma `(operador, resultado, arg1, arg2)`. Los tipos de operadores son:

(ASSIGN, *result*, *arg1*, NULL)	Asigna *arg1* a *result*
(ADD, *result*, *arg1*, *arg2*)	Suma *arg1* , *arg2* y lo almacena en *result*
(SUB, *result*, *arg1*, *arg2*)	Resta *arg1* ,*arg2* y lo almacena en *result*
(MULT, *result*, *arg1*, *arg2*)	Multiplica *arg1* , *arg2* y lo almacena en *result*
(DIV, *result*, *arg1*, *arg2*)	Divide *arg1* , *arg2* y lo almacena en *result*
(NEG, *result*, *arg1*, NULL)	Multiplica por (-1) *arg1* y lo almacena en *result*
(HALT, NULL, NULL, NULL)	Final del programa

Por ejemplo, para la entrada:

```
a = 3 + 2 ;
b = a * 2 ;
c = (a + b) + (2 * 6) ;
d = -1 + a ;
```

Obtenemos la salida:

```
( ADD, t0, 3, 2 )
( ASSIGN, a, t0, NULL )
( MULT, t1, a, 2 )
( ASSIGN, b, t1, NULL )
( ADD, t2, a, b )
( MULT, t3, 2, 6 )
( ADD, t4, t2, t3 )
( ASSIGN, c, t4, NULL )
( NEG, t5, 1, NULL )
( ADD, t6, t5, a )
( ASSIGN, d, t6, NULL )
( HALT, NULL, NULL, NULL )
```