Reporte Fase 2: Análisis Sintáctico

Constanza Abarca 13-10000
Pedro Maldonado 13-10790

En esta fase se construyó una gramática para el lenguaje propuesto BasicTran utilizando la librería PLY de Python3, basada en las herramientas de construcción de compiladores lex y yacc.

En primer lugar, se definió la siguiente precedencia en los operadores tomando en cuenta la tabla de precedencias que aparece en la contraportada del libro “A Logical Approach to Discrete Math” de David Gries:

```
precedence = (
('nonassoc', 'TkMayor', 'TkMenor', 'TkMayorIgual', 'TkMenorIgual'),
('left', 'TkSuma', 'TkResta'),
('left', 'TkMult', 'TkDiv', 'TkMod'),
('right', 'uminus'),
('left', 'TkConcatenacion'),
('left', 'TkPunto'),
('left', 'TkSiguienteCar'),
('left', 'TkAnteriorCar'),
('left', 'TkValorAscii'),
('left', 'TkIgual', 'TkDesigual'),
('left', 'TkConjuncion', 'TkDisyuncion'),
('right', 'TkNot')
)
```

La producción inicial está definida como “bloque” y establece que un programa debe iniciar con los tokens begin y finalizar con end u opcionalmente incluir una lista de declaraciones de variables empezando con el token with.

En general, de las producciones se deriva el símbolo no terminal “expresión” donde se pueden volver a generar más expresiones de tipo aritmética, relacional, booleanas, de caracteres, de arreglos y símbolos terminales.

El Árbol Sintáctico se construyó con una clase Nodo y se definieron distintas funciones para imprimir todos los tipos de operaciones que forman parte del lenguaje propuesto siguiendo una estructura jerárquica que representa las precedencias e indica los valores que tienen los términos que forman parte de las distintas expresiones.

Finalmente, los errores manejados son del tipo sintáctico, es decir, si en el código se encuentra alguna expresión que no cumple con las reglas indicadas en la gramática se reportará el error generado.