

Manual

Compilador en Python

Santiago Arango Upegui

Yessica Quintero Valderrama

25 de Noviembre de 2021


Introducción

1. Lista de palabra reservadas y tokens
2. Gramática BNF

Lista de palabra reservadas y tokens

Símbolo	Nombre Interno	Nombre	Uso
	nulsym		Reservado
	identsym	ID	Constante, variable e identificador de procesos
	numbersym	NUMBER	Número literal
+	plussym	PLUS	Suma en expresiones
-	minussym	MINUS	Resta en expresiones
*	multsym	TIMES	Multiplicación en expresiones
/	slashsym	DIVIDE	División en expresiones
odd	oddsym	ODD	Determina si una expresión es impar
'='	eqsym	ASSIGN	Definición de constantes, comprueba la igualdad entre dos expresiones
<>	neqsym	NE	Comprueba que dos expresiones no son iguales
<	lessym	LT	Comprueba que la expresión de la izquierda es menor que la expresión de la derecha

<=	leqsym	LTE	Comprueba que la expresión de la izquierda es menor que o igual a la expresión de la derecha
>	gtrsym	GT	Comprueba que la expresión de la izquierda es mayor que la expresión de la derecha
>=	geqsym	GTE	Comprueba que la expresión de la izquierda es mayor que o igual a la expresión de la derecha
(lparentsym	LPARENT	Comienza un elemento (factor)
)	rparentsym	RPARENT	Termina un elemento (Factor)
,	commasym	COMMA	Separa una constante, identificadores de variables en sus respectivas declaraciones
;	semmicolonsym	SEMMICOLOM	Termina declaraciones (statements)
.	periodsym	DOT	Termina un programa
:=	becomessym	UPDATE	Asignaciones de variables
begin	beginsym	BEGIN	Comienza un bloque de declaraciones
end	endsym	END	Termina un bloque de declaraciones
if	ifsym	IF	Comienza una if-then, seguido por una condición
then	thensym	THEN	Parte de if-then, que va seguido por una declaración
while	whilesym	WHILE	Comienza un ciclo while, seguido por una condición
do	dosym	DO	Parte de un ciclo while, que va seguido por una declaración
call	callsym	CALL	Llama a un procedimiento



const	constsym	CONST	Comienza declaraciones de constantes
int	intsym	VAR	Comienza declaraciones de enteros
procedure	procsym	PROCEDURE	Comienza una declaración de procedimiento
out	outsym	OUT	Salida del valor de una expresión
in	insym	IN	Pide al usuario que entre un valor y lo asigne a una variable
else	elsesym	ELSE	Sigue opcionalmente de la declaración if-then

Gramática BNF

`<program> ::= <block> .`

`<block> ::= <const-decl> <var-decl> <proc-decl> <statement>`

`<const-decl> ::= const <const-assignment-list> ; l e`

`<const-assignment-list> ::= <ident> = <number>`

`l <const-assignment-list> , <ident> = <number>`

`<var-decl> ::= var <ident-list> ; l e`

`<ident-list> ::= <ident> l <ident-list> , <ident>`

`<proc-decl> ::= <proc-decl> procedure <ident> ; <block> ; l e`

`<statement> ::= <ident> := <expression>`

`l call <ident>`

`l begin <statement-list> end`

`l if <condition> then <statement>`

`l while <condition> do <statement>`

`l e`

`<statement-list> ::= <statement> l <statement-list> ; <statement>`

`<condition> ::= odd <expression> l <expression> <relation> <expression>`

`<relation> ::= = l <> l < l > l <= l >=`

`<expression> ::= <term> l <adding-operator> <term>`

`l <expression> <adding-operator> <term>`

`<adding-operator> ::= + l -`

`<term> ::= <factor> l <term> <multiplying-operator> <factor>`



$\langle \text{multiplying-operator} \rangle ::= * \mid /$

$\langle \text{factor} \rangle ::= \langle \text{ident} \rangle \mid \langle \text{number} \rangle \mid (\langle \text{expression} \rangle)$

Notas:

1. "ε" denota el string vacío.
2. $\langle \text{ident} \rangle$ y $\langle \text{number} \rangle$ son tokens que representan identificadores y números respectivamente.

