

Subconjunto de la Gramática de C++ en BNFE

Diseño y Construcción de Compiladores - UNSL

<unidad de traducción> ::= { <declaraciones> }

<declaraciones> ::= <especificador de tipo> **ident** <especificador de declaración>

<especificador de tipo> ::= **void** | **char** | **int** | **float**

<especificador de declaración> ::= <definición de función> | <declaración de variable>

<definición de función> ::= ([<lista declaración de parámetros>]) <proposición compuesta>

<lista declaración de parámetros> ::= <declaración de parámetro> { , <declaración de parámetro> }

<declaración de parámetro> ::= <especificador de tipo> [**&**] **ident** [[]]

<declaración de variable> ::= <declarador init> [, <lista declaraciones init>] ;

<lista declaraciones init> ::= **ident** <declarador init> { , **ident** <declarador init> }

<declarador init> ::= [= <constante> |
[[**cons_ent**]] = { <lista de inicializadores> }]]

<lista de inicializadores> ::= <constante> { , <constante> }

<proposición compuesta> ::= { [<lista de declaraciones>] [<lista de proposiciones>] }

<lista de declaraciones> ::= <declaración> { <declaración> }

<declaración> ::= <especificador de tipo> <lista declaraciones init> ;

<lista de proposiciones> ::= <proposición> { <proposición> }

<proposición> ::= <proposición expresión>
| <proposición compuesta>
| <proposición de iteración>
| <proposición de selección>
| <proposición de retorno>
| <proposición de entrada/salida>

<proposición de iteración> ::= **while** (<expresión>) <proposición>

<proposición de selección> ::= **if** (<expresión>) <proposición> [**else** <proposición>]

<proposición de retorno> ::= **return** <expresión> ;

<proposición de entrada/salida> ::= **cin** >> <variable> { >> <variable> } ;
| **cout** << <expresión> { << <expresión> } ;

<proposición expresión> ::= [<expresión>] ;

<expresión> ::= <expresión simple> { = <expresión simple> | <relación> <expresión simple> }

<relación> ::= != | == | < | <= | >= | >

<expresión simple> ::= [+ | -] <término> { (+ | - | |) <término> }

<término> ::= <factor> { (* | / | &&) <factor> }

<factor> ::= <variable>
| <constante>
| ! <expresión>
| (<expresión>)
| <llamada a función>
| **cte_str**

<variable> ::= **ident** | **ident** [<expresión>]

<llamada a función> ::= **ident** ([<lista de expresiones>])

<lista de expresiones> ::= <expresión> { , <expresión> }

<constante> ::= **cons_ent** | **cons_float** | **cons_car**