# The grammar of the ALGO syntax

```
PARSE ::= ALGO . EOF

ALGO ::= "ALGORITHM" . "(" . IDENT . ")" . BLOCK

BLOCK ::= "{" . SEQUENCE . "}"

SEQUENCE ::= ( STATEMENT )*

STATEMENT ::=
    | DECL ";"
    | EXPR . opt_ASSIGNMENT ";"
    | IFTE
    | RETURN ";"
    | WHILEDO

RETURN ::= "return" . EXPR

EXPR ::= opt_PRE_POST_OP . E1 . opt_BINOP_EXPR

opt_BINOP_EXPR ::=
    | epsilon
    | (BINOP | PTR) . EXPR

opt_ASSIGNMENT ::=
    | BINASSIGN . EXPR
    | epsilon

DECL ::= SIGNAGE . IDENT . opt_DECL . ("," . IDENT . opt_DECL)*

SIGNAGE ::= (SIGN | epsilon) . (PRE_TYPE | epsilon) . TYPE . (PTR)*

opt_DECL ::=
    | epsilon
    | BINASSIGN . EXPR

IFTE ::= "if" . "(" . EXPR . ")" . (BLOCK | EXPR . ";") . opt_ELSE

opt_ELSE ::=
    | epsilon
    | "else" . (BLOCK | EXPR . ";")

WHILEDO ::= "while"  . "(" . EXPR . ")" . BLOCK

E1 ::= (
    | "(" . EXPR . ")"
    | VAR_FUNC
    | (PTR)+ . IDENT
    ) . opt_PRE_POST_OP

FUNCTION ::= IDENT . "(" . (epsilon | ARGS) . ")"

VAR_FUNC ::= IDENTIFIER . ( "(" . ARGS . ")" | epsilon ) . opt_PRE_POST_OP

ARGS ::= epsilon | EXPR . ("," . EXPR)*

opt_ARGS ::=
    | epsilon
    | "," . ARGS

opt_PRE_POST_OP ::=
    | PRE_POST_OP
    | ("[" . EXPR . "]")*

PRE_POST_OP ::= "++" | "--" | "!"

PTR ::= "*" | "&"

IDENT ::= (LOWERCASE | UPPERCASE | "_")+ . (LOWERCASE | UPPERCASE | "_" | DIGIT)*

VALUE ::=
    | INTEGER
    | FLOAT
    | CHAR
    | STRING
    | BOOLEAN

INTEGER ::= (DIGIT)+

FLOAT ::=
    | (DIGIT)+ . "." . (DIGIT)*
    | (DIGIT)* . "." . (DIGIT)+

CHAR ::= "'" . (ASCII)* . "'"

STRING ::= """ . (ASCII)* . """

ASCII ::= All characters in the ASCII table !! WITH "\\" AND NOT "\" !!

BOOLEAN ::= "true" | "false"
```

```
BINOP ::= "/" | "!=" | "==" | "|" | "+" | "-" | "%" | "<"
    | ">" | "<=" | ">=" | "&&" | "||" | "<<" | ">>"

BINASSIGN ::= "=" | ":=" | "+=" | "-=" | "*=" | "&="  | "|=" | "<<=" | ">>="

TYPE ::=
    | "bool"
    | "int"
    | "char"
    | "string"
    | "long"
    | "double"
    | "float"

SIGN ::= "unsigned" | "signed" | epsilon

PRE_TYPE ::= "long" | "short" | epsilon
```