Diego Velázquez Saldaña A01177877

BabyDuck – Entrega #0

1.- Diseñar las Expresiones Regulares que representan a los diferentes elementos de léxico que ahí aparecen.

Token	Expresión Regular
id	[a-z_][a-zA-Z0-9_]*
cte_int	[0-9]+
cte_float	[0-9]+\.[0-9]+
cte_string	“(.*)?”
add	\+
sub	-
mul	*
div	/
assign	=
equal	==
notequal	!=
less	<
greater	>
semicolon	;
colon	:
comma	,
lparen	(
rparen	)
lbrace	{
rbrace	}
lbrack	[
rbrack	]
program	program
main	main
end	end
var	var
print	print
if	if
else	else
while	while
do	do

2.- Listar todos los Tokens que serán reconocidos por el lenguaje

Palabras Clave

- program
- main
- end

- if
- else
- while
- do
- print
- void
- type (int o float)

Variables

- id
- cte_int
- cte_float
- cte_string

Operadores aritméticos

- +
- -
- *
- /

Operadores de comparación

- =
- ==
- !=
- <
- >

Delimitadores

- (
-)
- {
- }
- [
-]
- ;
- ,
- :

3.- Diseñar las reglas gramaticales (Context Free Grammar) equivalentes a los diagramas.

Regla	**Definición**

Programa	program ID ; VARS FUNCS main Body end
VARS	var : Tipo ID (, ID)* ; VARS | ε
Tipo	int | float
FUNCS	void ID ([ ]) (Tipo) ID : VARS Body FUNCS | ε
Body	{ STATEMENT }
STATEMENT	ASSIGN | CONDITION | CYCLE | Print
ASSIGN	ID = EXPRESIÓN ;
Print	print (EXPRESIÓN) ;
CONDITION	if (EXPRESIÓN) Body else Body
CYCLE	while (EXPRESIÓN) do Body
EXPRESIÓN	EXP (< | > | == | != EXP)?
EXP	TÉRMINO ((+ | -) TÉRMINO)*
TÉRMINO	FACTOR ((* | /) FACTOR)*
FACTOR	(+ | -)? (ID | CTE | (EXPRESIÓN))
CTE	CTE_INT | CTE_FLOAT | CTE_STRING