



ANALIZADOR DE LÉXICO Y SINTAXIS

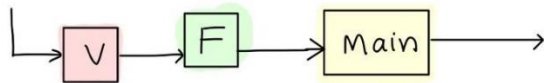
Versión 1

Dra. Norma Frida Roffe S.
7 de octubre de 2021

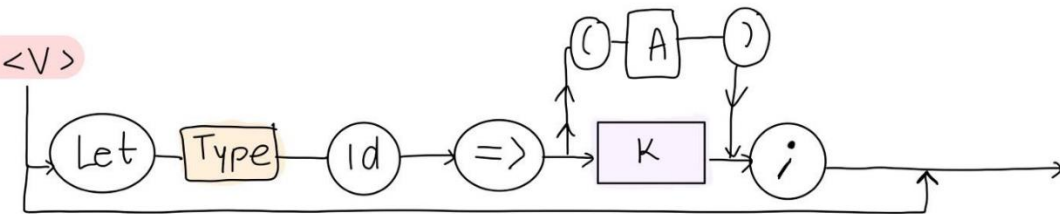
[Kenia Gabriela Sánchez Duran](#)
Lenguajes y traductores

Diagramas de sintaxis

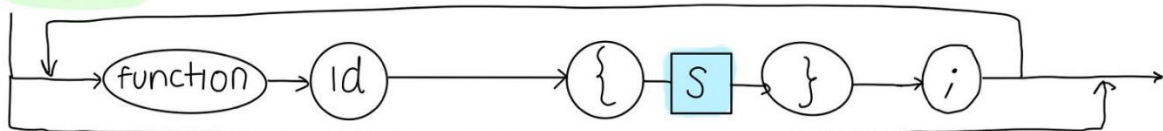
< programa >



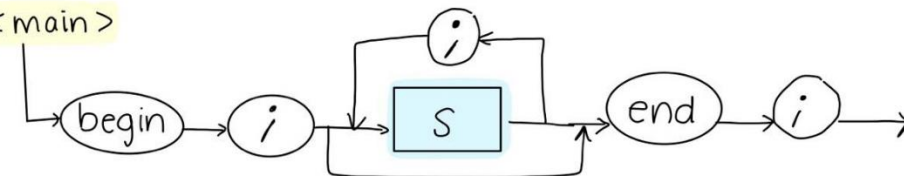
<V>



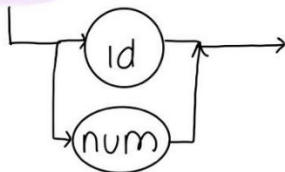
<F>



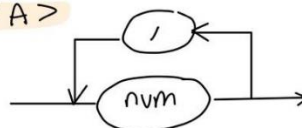
<main>



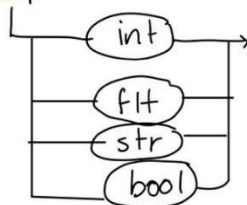
<K>



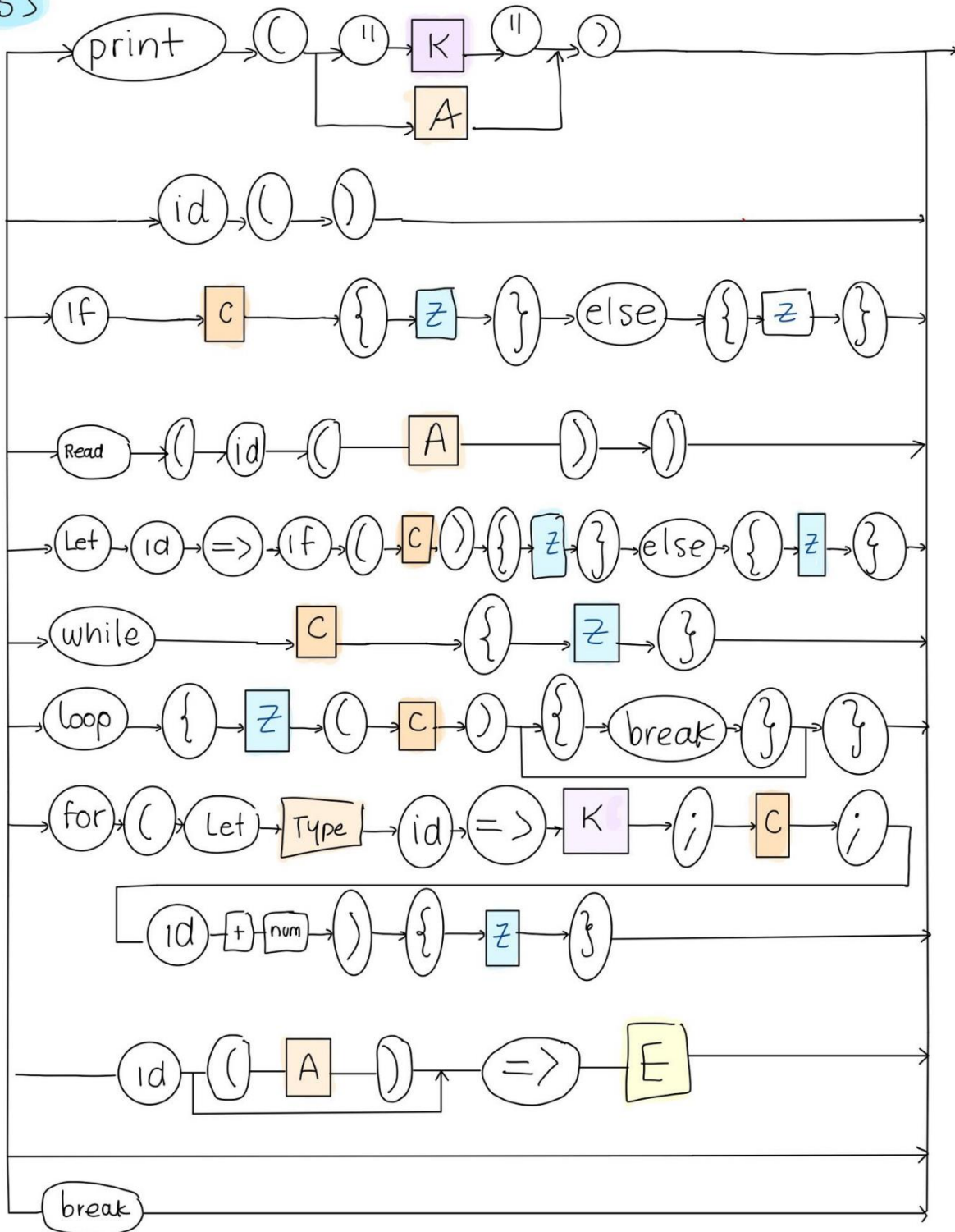
<A>

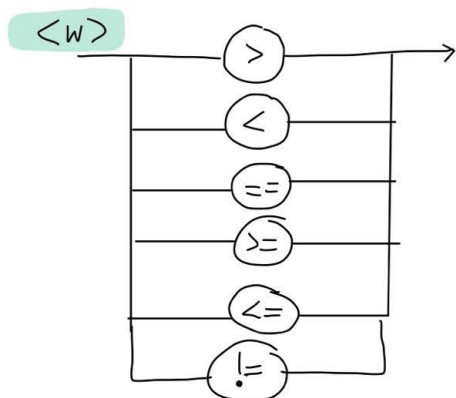
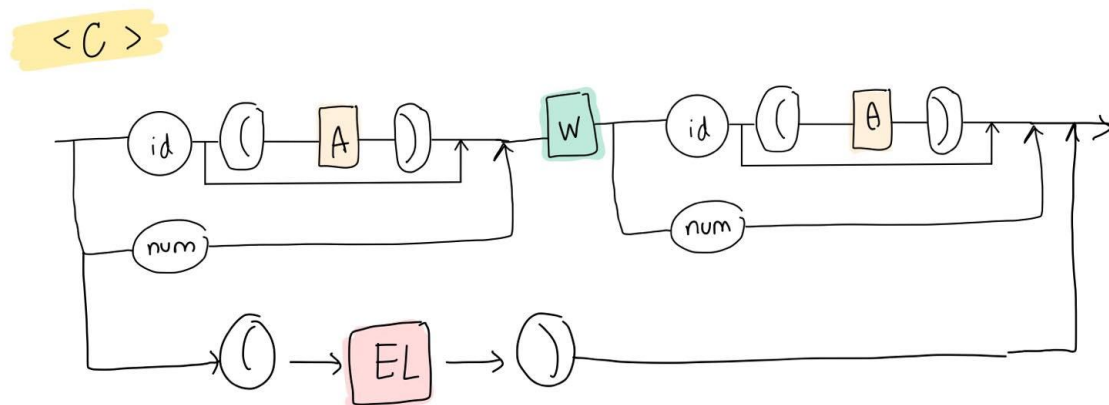
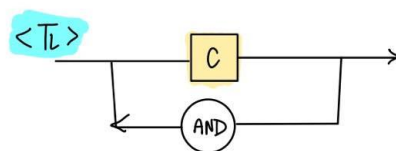
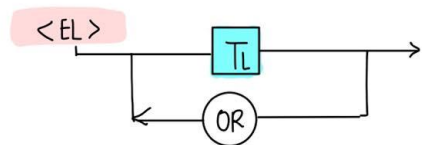
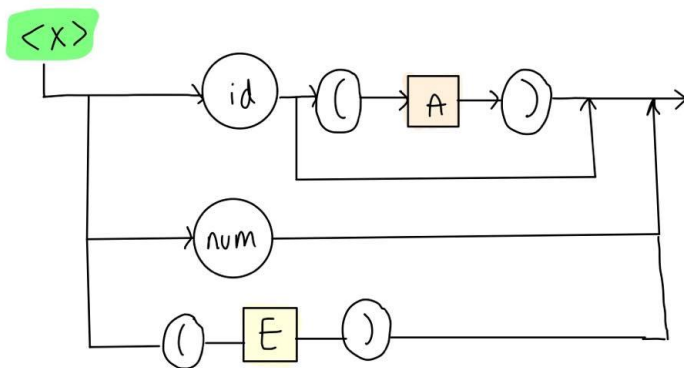
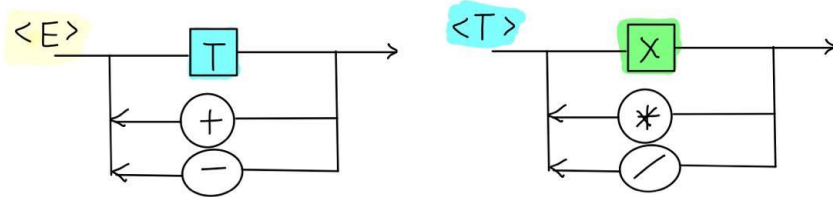


<Type>



<5>





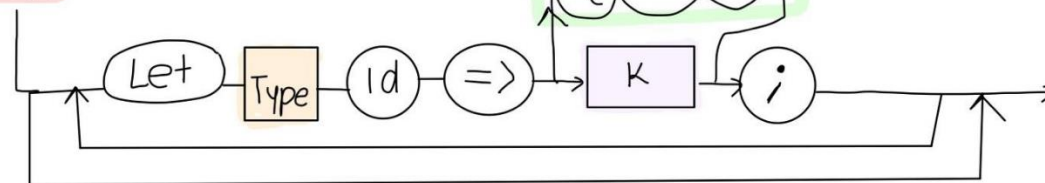
Gramática

Gramática

< programa >



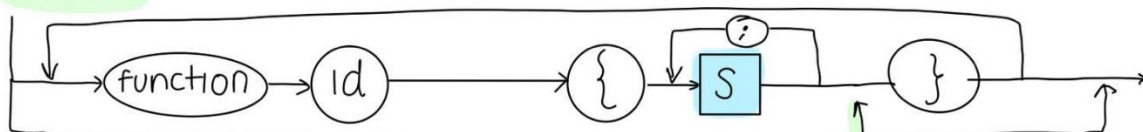
<V>



$V \rightarrow V \text{ Let type id } \Rightarrow K ; \quad V \rightarrow \epsilon \quad A \rightarrow \text{num}$

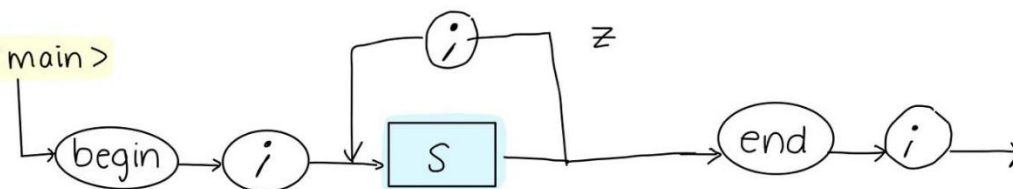
$V \rightarrow V \text{ Let type id } \Rightarrow (A) \quad A \rightarrow A, \text{num}$

<F>



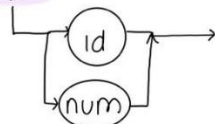
$F \rightarrow F \text{ function id } \{ Z \} \quad Z \rightarrow S ;$
 $F \rightarrow \epsilon \quad Z \rightarrow Z S ;$
 $Z \rightarrow S ;$

<main>



$Main \rightarrow \text{begin} ; Z \text{ end} ;$

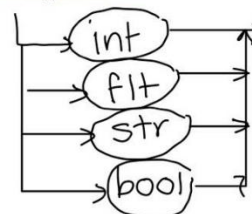
<K>

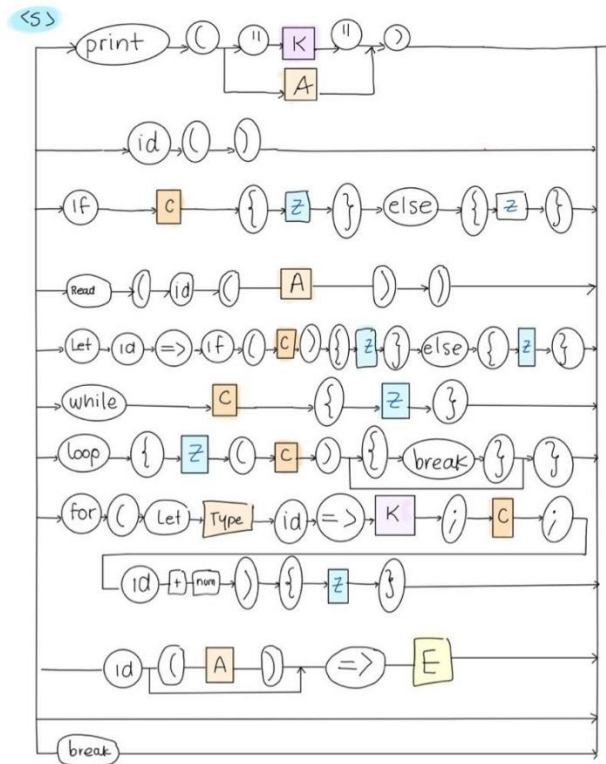


$K \rightarrow \text{id}$
 $K \rightarrow \text{num}$

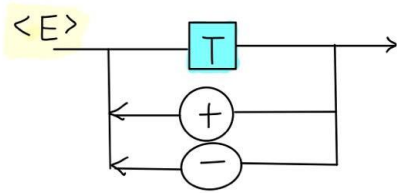
<Type>

$Type \rightarrow \text{int}$
 $Type \rightarrow \text{flt}$
 $Type \rightarrow \text{str}$
 $Type \rightarrow \text{bool}$

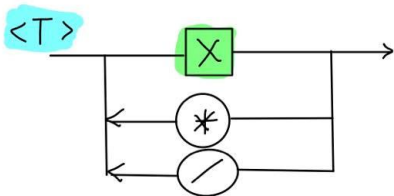




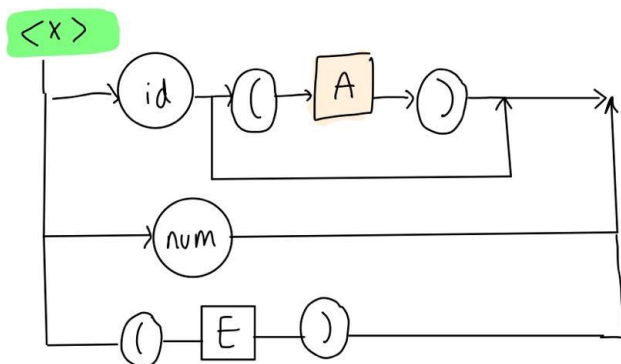
- $S \rightarrow \text{print}(M)$
 $M \rightarrow "K"$
 $M \rightarrow A$
- $S \rightarrow \text{id}(C)$
- $S \rightarrow \text{if } C \{ z \} \text{ else } \{ z \}$
- $\text{Read}(\text{id}(A))$
- $\text{Let } \text{id} \Rightarrow \text{if}(C) \{ z \} \text{ else } \{ z \}$
- $\text{while } C \{ z \}$
- $\text{Loop } \{ z \ C \ C \} B \}$
 $B \rightarrow \{ \text{break} \}$
 $B \rightarrow \epsilon$
- $S \rightarrow \text{for } (\text{Let Type id} \Rightarrow K ; C ; \text{id} + \text{num}) \{ z \}$
- $S \rightarrow \text{id } D \Rightarrow E$
 $D \rightarrow \epsilon$
 $D \rightarrow (A)$
- $S \rightarrow \epsilon$
- $S \rightarrow \text{Break}$



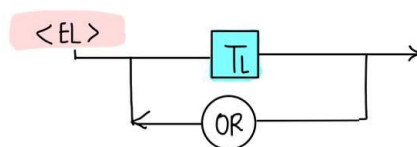
$$\begin{aligned} E &\rightarrow E + X \\ E &\rightarrow E - X \\ E &\rightarrow X \end{aligned}$$



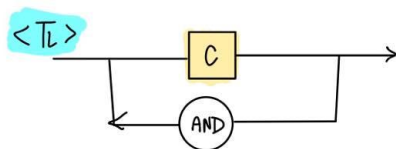
$$\begin{aligned} T &\rightarrow T * F \\ T &\rightarrow T / F \\ T &\rightarrow F \end{aligned}$$



$$\begin{aligned} F &\rightarrow id \\ F &\rightarrow id(A) \\ F &\rightarrow num \\ F &\rightarrow (E) \end{aligned}$$

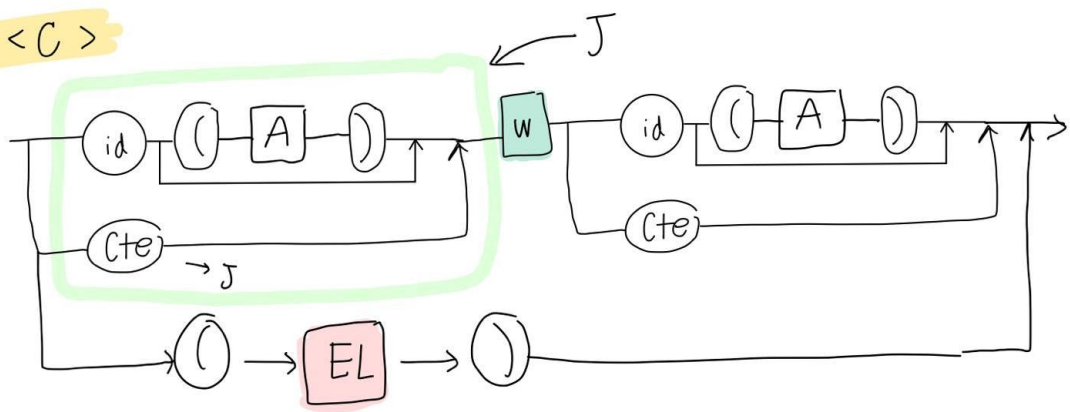


$$\begin{aligned} EL &\rightarrow TL \\ EL &\rightarrow EL \text{ OR } TL \end{aligned}$$



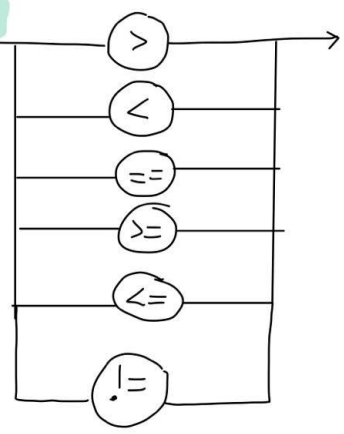
$$\begin{aligned} TL &\rightarrow C \\ TL &\rightarrow TL \text{ AND } C \end{aligned}$$

<C>



$C \rightarrow (EL)$
 $C \rightarrow JwJ$
 $J \rightarrow id$
 $J \rightarrow id(A)$

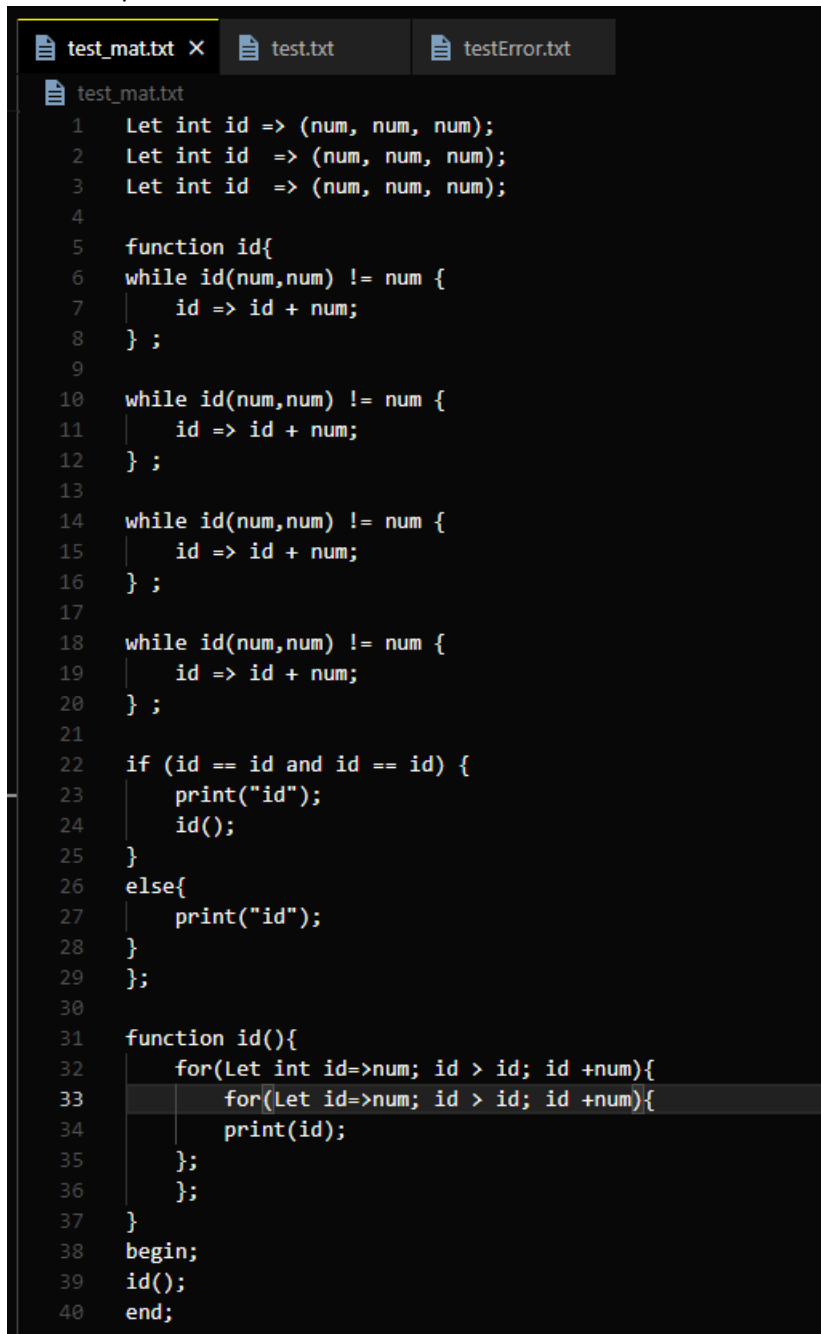
<w>



$X \rightarrow >$
 $X \rightarrow <$
 $X \rightarrow ==$
 $X \rightarrow >=$
 $X \rightarrow <=$
 $X \rightarrow !=$

Screenshots

- P.P. multiplicación de matrices.



```
test_mat.txt X test.txt testError.txt
test_mat.txt
1  Let int id => (num, num, num);
2  Let int id => (num, num, num);
3  Let int id => (num, num, num);
4
5  function id{
6  while id(num,num) != num {
7  |   id => id + num;
8  } ;
9
10 while id(num,num) != num {
11 |   id => id + num;
12 } ;
13
14 while id(num,num) != num {
15 |   id => id + num;
16 } ;
17
18 while id(num,num) != num {
19 |   id => id + num;
20 } ;
21
22 if (id == id and id == id) {
23 |   print("id");
24 |   id();
25 }
26 else{
27 |   print("id");
28 }
29 };
30
31 function id(){
32   for(Let int id=>num; id > id; id +num){
33     for(Let id=>num; id > id; id +num){
34       print(id);
35     };
36   };
37 }
38 begin;
39 id();
40 end;
```

PS C:\Users\kenya\OneDrive - Instituto Tecnológico y de Estudios Superiores de Monterrey\7mo semestre\Lenguajes y traductores\AnalizadorLexSintx> & C:/Users/kenya/anaconda3/python.exe "c:/Users/kenya/OneDrive - Instituto Tecnológico y de Estudios Superiores de Monterrey/7mo semestre/Lenguajes y traductores/AnalizadorLexSintx/anLexSin.py"

```
Let int id => (num, num, num);  
Let int id => (num, num, num);  
Let int id => (num, num, num);
```

```
function id{  
  while id(num,num) != num {  
    id => id + num;  
  } ;  
}
```

```
while id(num,num) != num {  
  id => id + num;  
} ;
```

```
while id(num,num) != num {  
  id => id + num;  
} ;
```

```
while id(num,num) != num {  
  id => id + num;  
} ;
```

```
if (id == id and id == id) {  
  print("id");  
  id();  
}  
else{  
  print("id");  
}  
};
```

```
function id(){  
  for(Let int id=>num; id > id; id +num){  
    for(Let id=>num; id > id; id +num){  
      print(id);  
    };  
  };  
}
```

```
begin;  
id();  
end;
```

```
Let  
int  
id  
=>  
(  
  num
```


- P.P. expresiones aritméticas y lógicas con paréntesis anidados.

```

test_mat.txt  test.txt  x
test.txt
1  Let int id => (num, num, num);
2  Let flt id => (num, num, num);
3  begin;
4  if (num > num){
5      id(num, num, num) => id(num,num,num) + id(
6  }
7  else{
8      for(Let int id => id; id >= num; id + num){
9          for(Let int id => id; id >= num; id +
10             id => num * num;
11         }
12     }
13 }
14
15 while (id!=id and id==id) {
16     id();
17 };
18
19 end;

```

```

OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS
PS C:\Users\kenya\OneDrive - Instituto Tecnológico y de Estudios Superiores de Monterrey\7mo semestre\lenguajes y traductores\AnalizadorLexSint\ & C:\Users\kenya\anaconda3\python.exe "c:/Users/kenya/OneDrive - Instituto Tecnológico y de Estudios Superiores de Monterrey/7mo semestre/lenguajes y traductores/AnalizadorLexSint/anlexSin.py"

Let int id => (num, num, num);
Let flt id => (num, num, num);
begin;
if (num > num){
    id(num, num, num) => id(num,num,num) + id(num,num,num);
}
else{
    for(Let int id => id; id >= num; id + num){
        for(Let int id => id; id >= num; id + num){
            id => num * num;
        }
    }
}

while (id!=id and id==id) {
    id();
};

end;

Let
int
id
=>
(
num
,
num
>
num
)
;
Let
flt
id
=>
(
num
,
num
,
num
)
;
begin
;
if

```


- Errores.

<pre> Let id => (num, num, num); Let id => (num, num, num); begin; if (num > num){ id() => id(num,num,num) + id(num,num,num); } else{ for(int id => id; id >= num; id + num){ for(Let id => id; id >= num; id + num){ id => num * ; } } } while id!=id and id==id { id(); }; end; Let id ***Incorrecto*** => (num , num , num) ; Let id </pre>	<pre> Let id => (num , num , num) ; begin ; if (num > num) { id () => ***Incorrecto*** id (num , num , num) ; </pre>
--	--