

# Análise Léxica

## Especificação do diagrama de transição

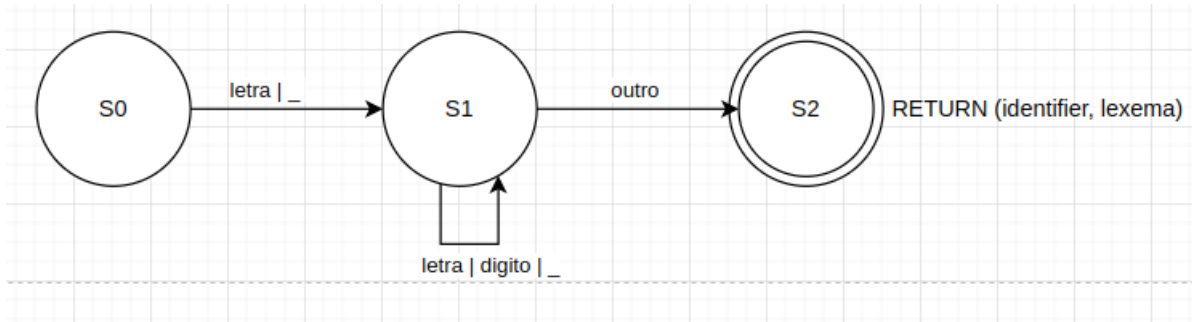


Figura 1: Diagrama de transição do token identificador

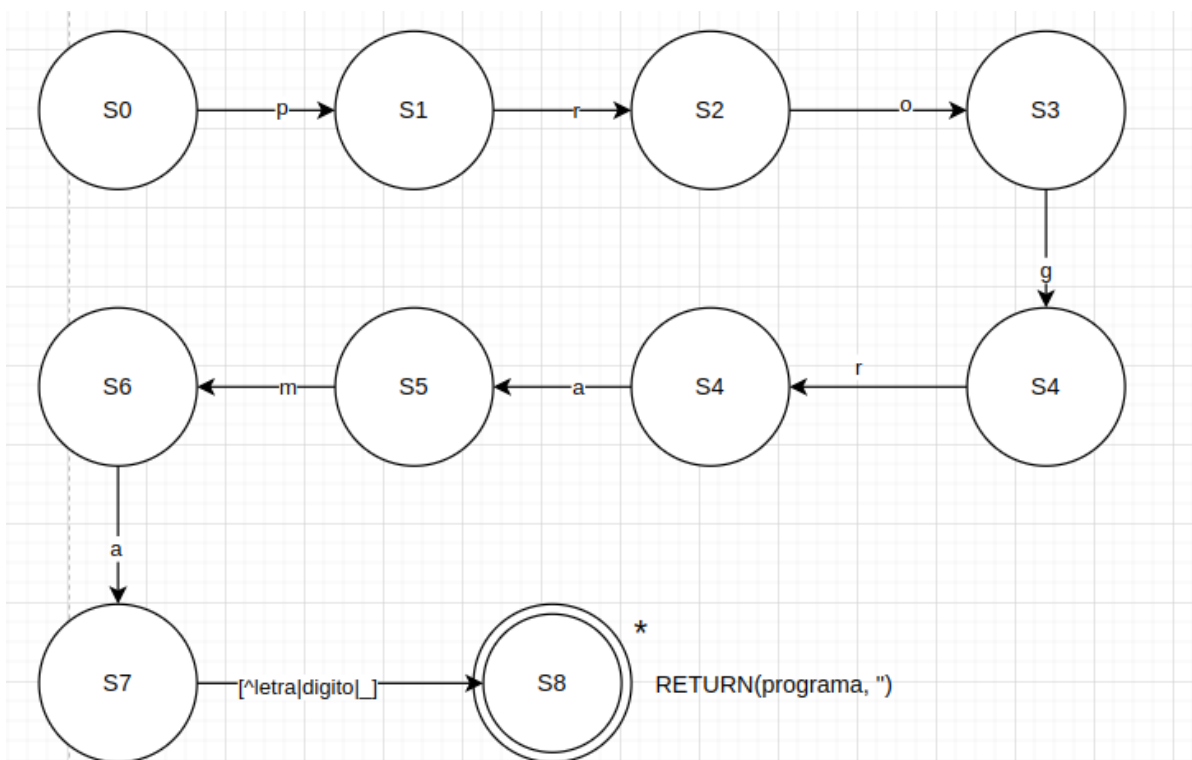


Figura 2: Diagrama de transição do token programa

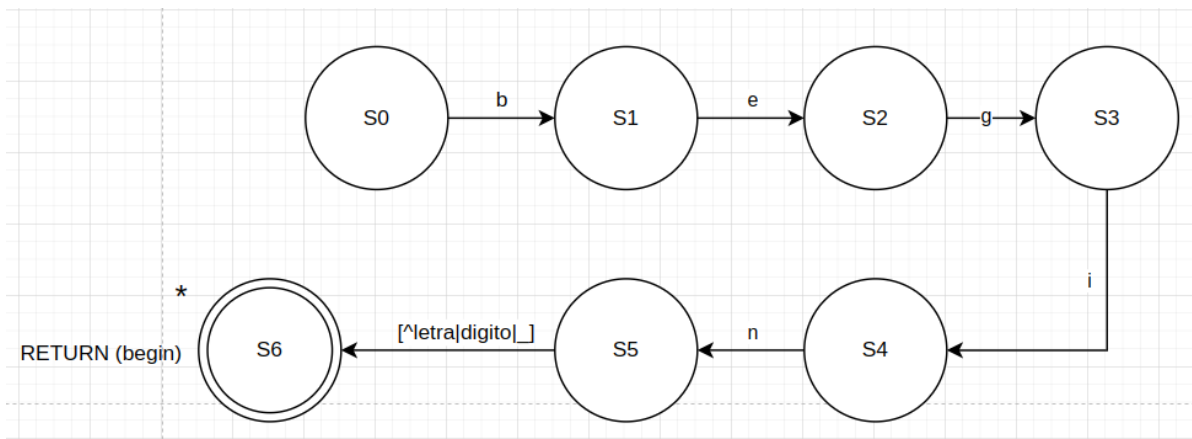


Figura 3: Diagrama de transição do token begin

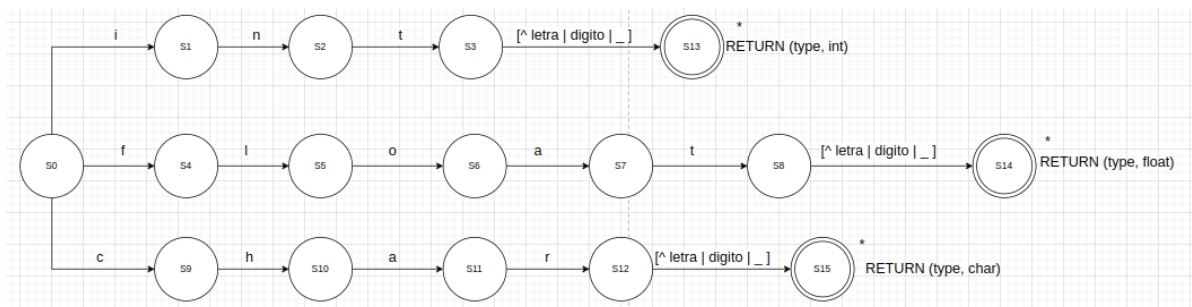


Figura 4: Diagrama de transição do token type

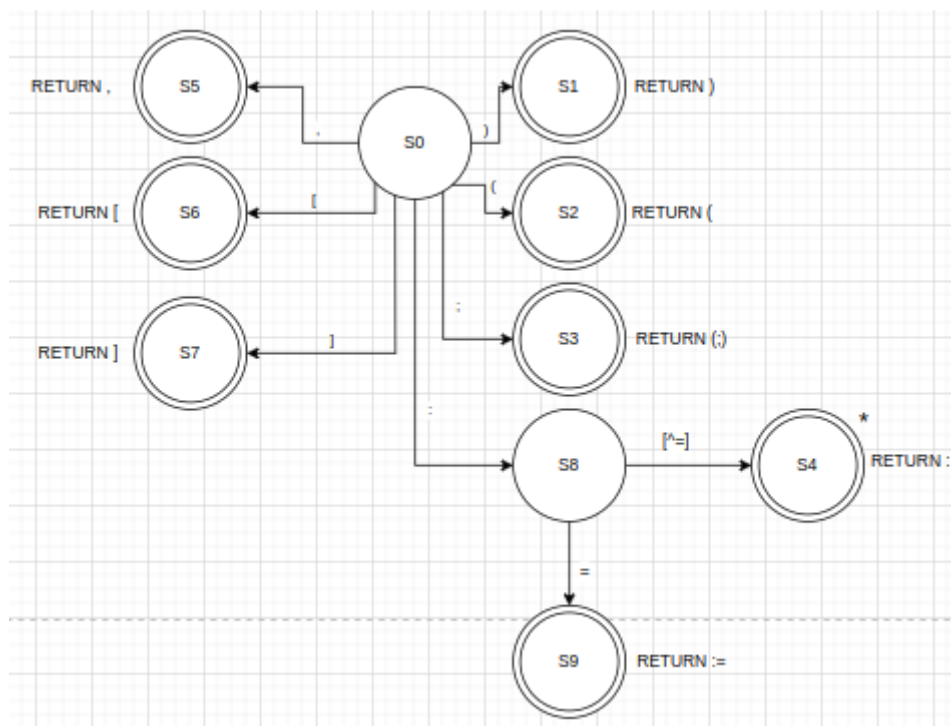


Figura 5: Diagrama de transição de tokens auxiliares

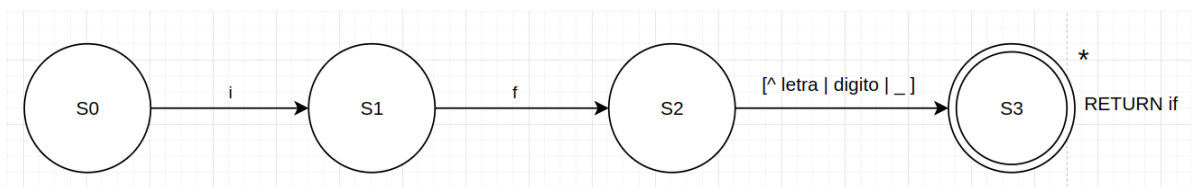


Figura 6: Diagrama de transição do token if

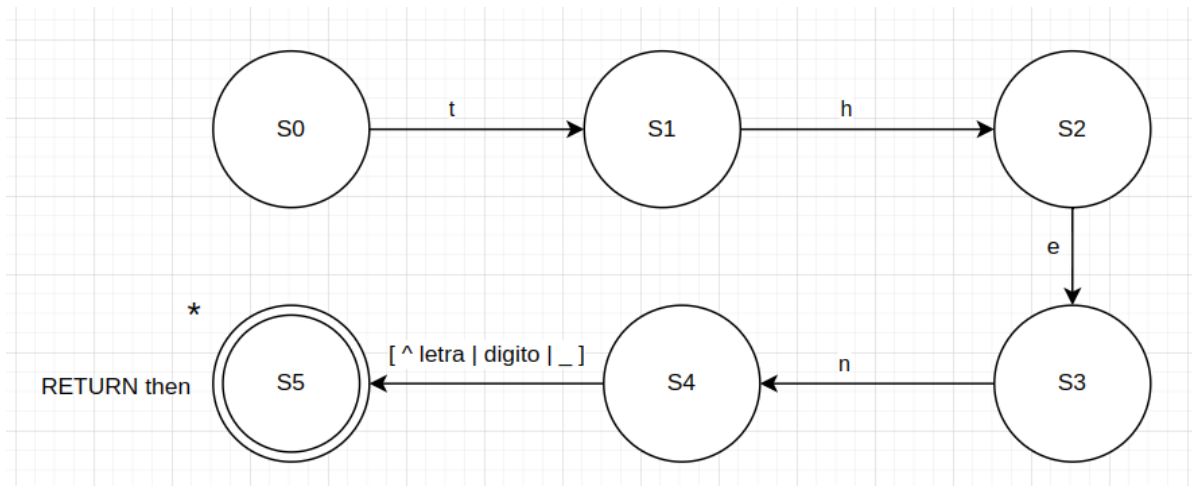


Figura 7: Diagrama de transição do token then

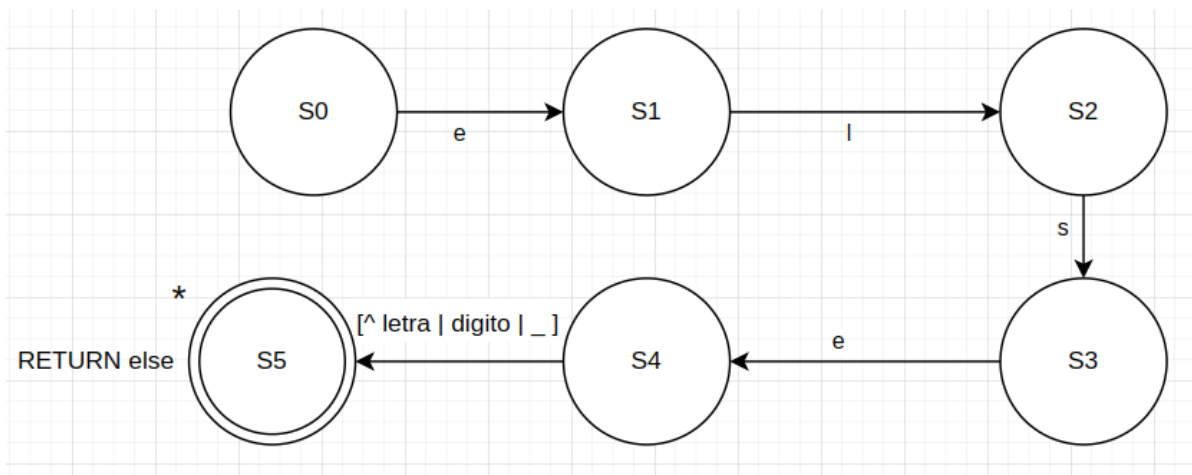


Figura 8: Diagrama de transição do token else

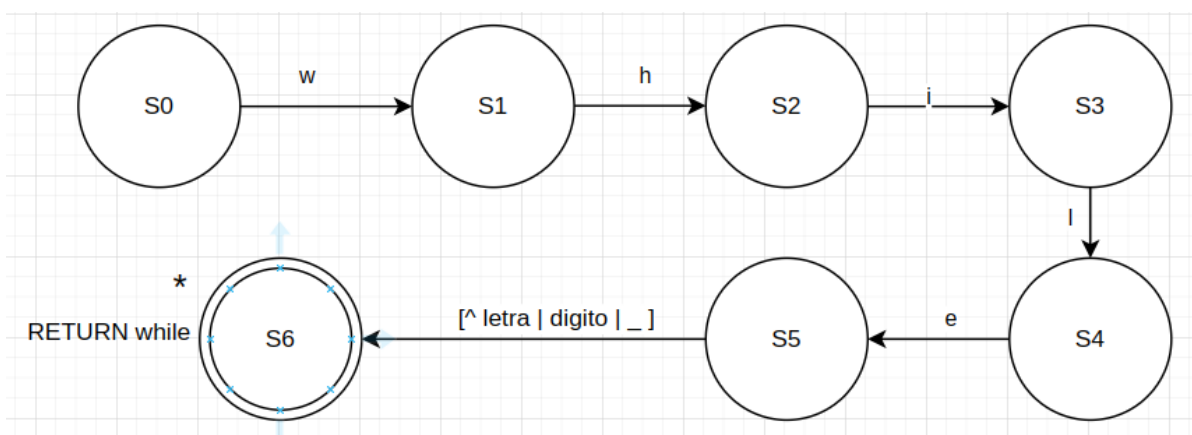


Figura 9: Diagrama de transição do token while

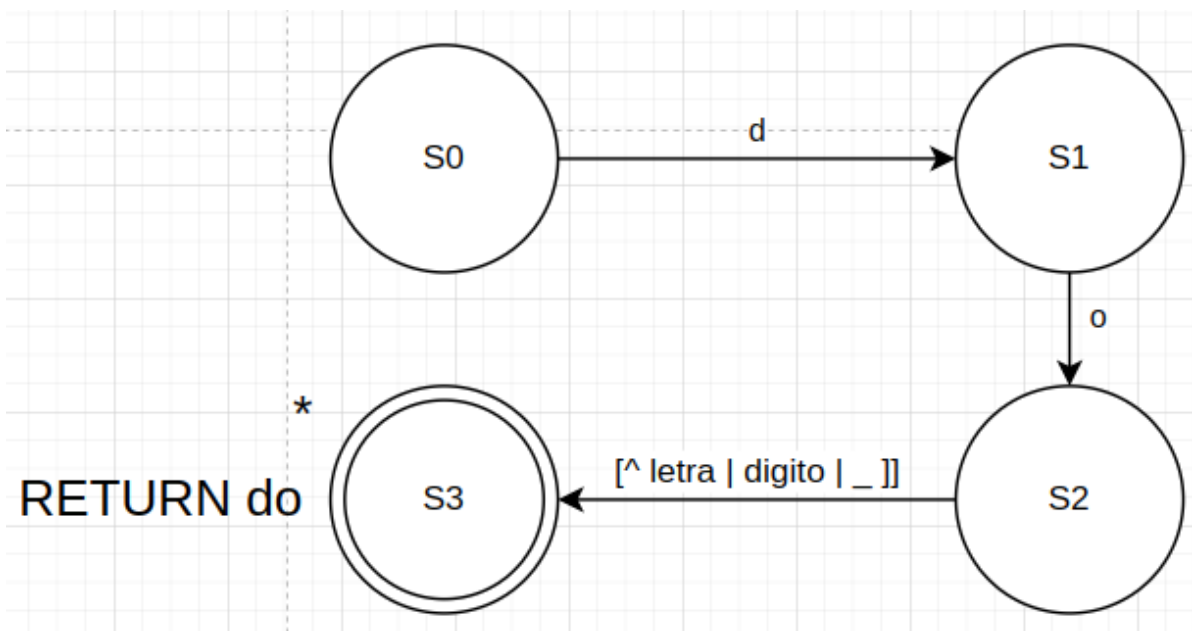


Figura 10: Diagrama de transição do token `do`

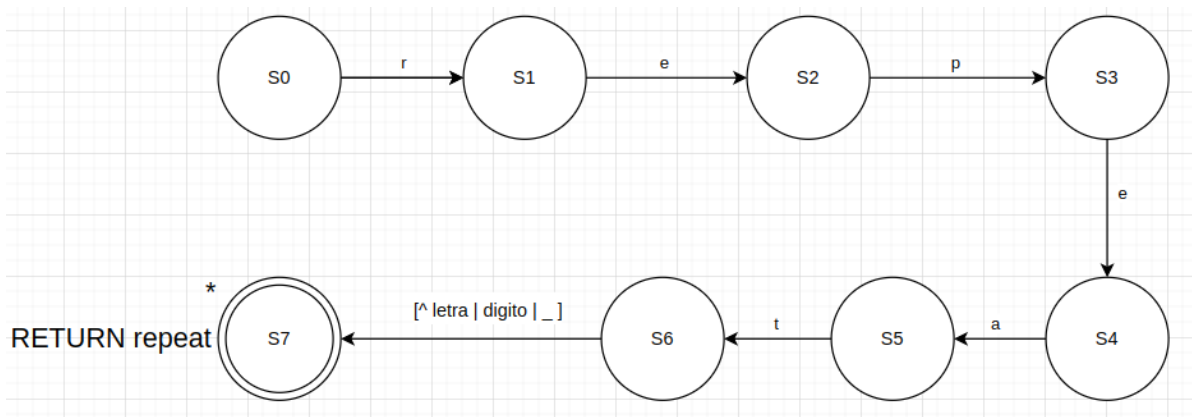


Figura 11: Diagrama de transição do token `repeat`

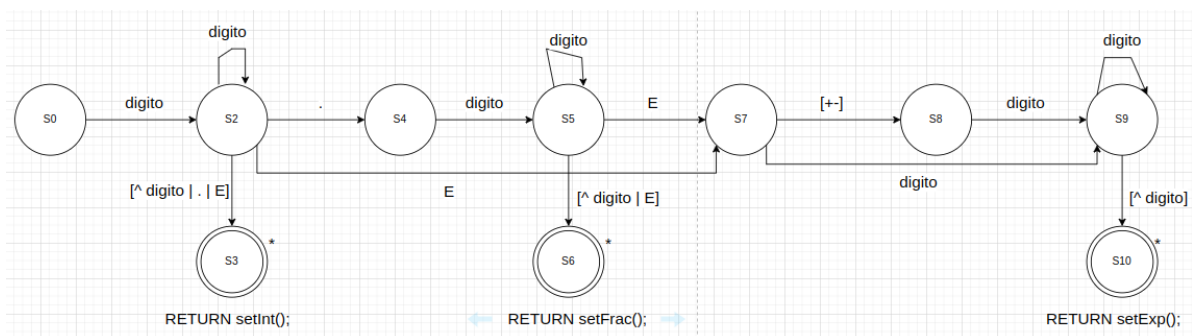


Figura 12: Diagrama de transição do token `number`

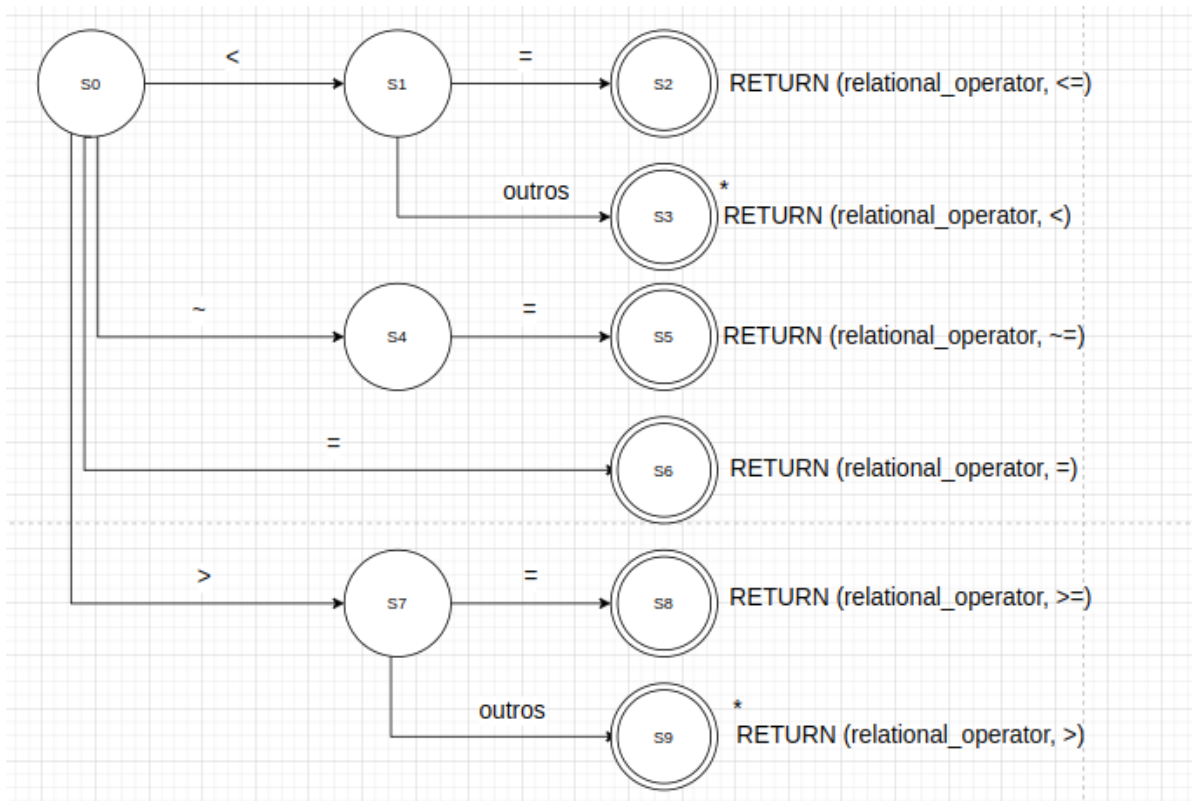


Figura 13: Diagrama de transição do token relational\_operator

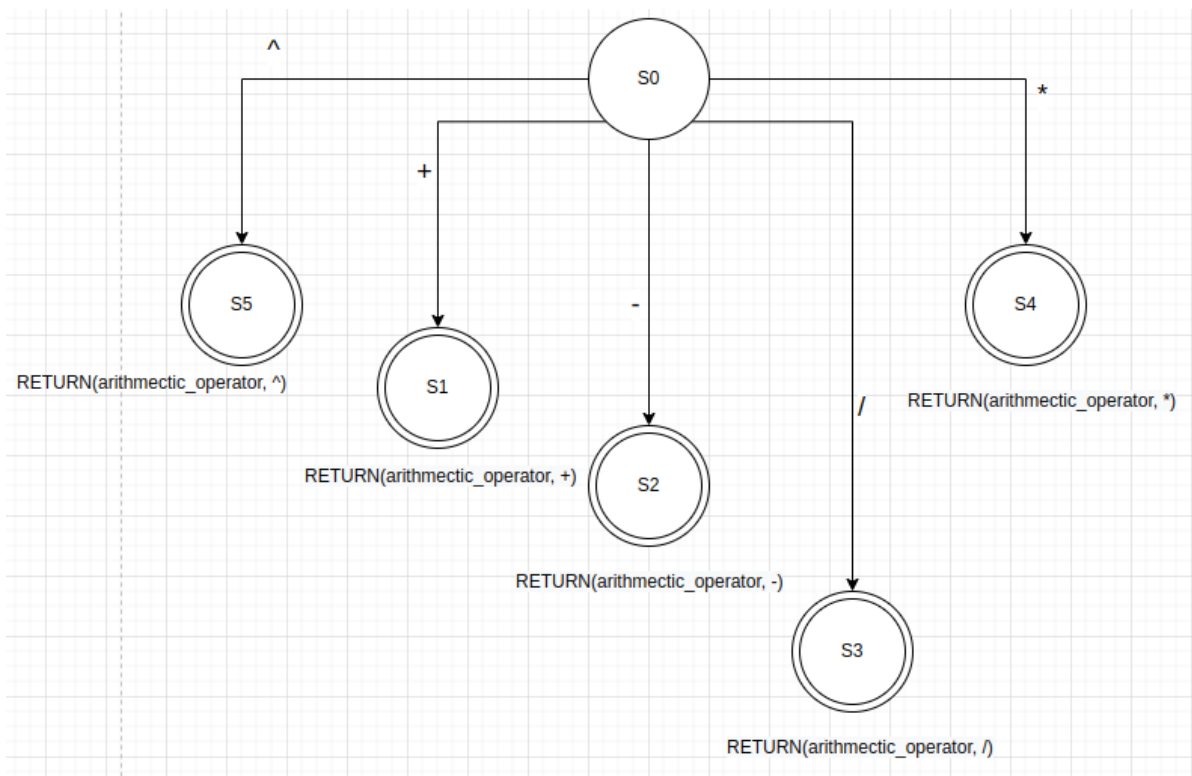
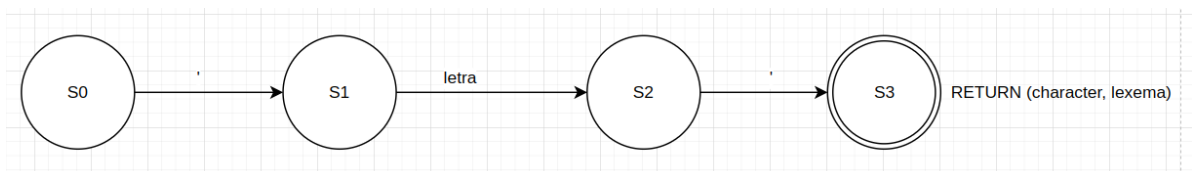
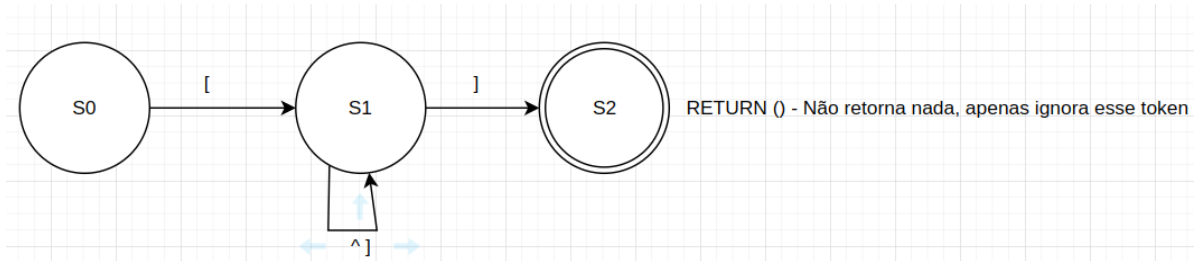


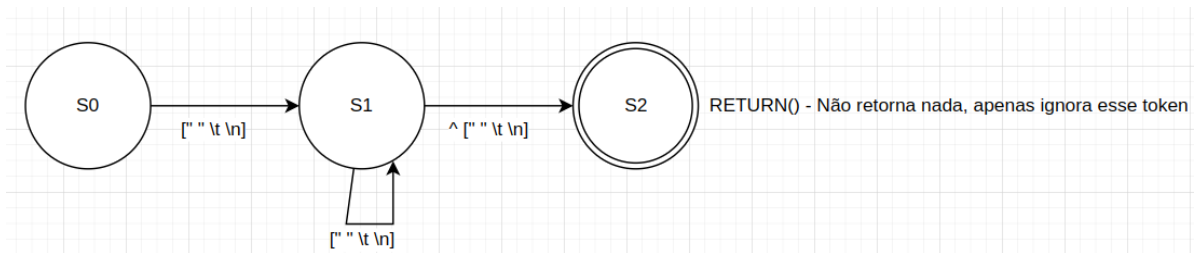
Figura 14: Diagrama de transição do token arithmetic\_operator



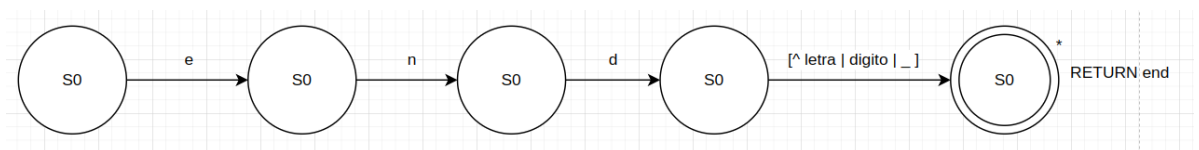
**Figura 15: Diagrama de transição do token character**



**Figura 16: Diagrama de transição do token comments**



**Figura 17: Diagrama de transição do token space**



**Figura 18: Diagrama de transição do token end**

**Unificando todos os diagramas de transição em um diagrama de transição não determinístico**



Figura 20: Diagrama de transição convertido em um AFD

## Implementação do analisador léxico

Para testar esse analisador léxico usaremos o seguinte código de teste:

```
programa meuprograma
[ Esse é um simples programa que verifica se x>y e conta]
begin
  int:x;
  int:y;
  float cont;
  x:=50;
  y:=20;

  if(x>y) then
    cont := cont+50;
  else
    cont:= cont-1;

end
```

```
Token found in position: 11 8c -> <PROGRAMA,none>
Token found in position: 11 14c -> <IDENTIFIER,1>
Token found in position: 21 15c -> <BEGIN,none>
Token found in position: 31 14c -> <TYPE,none>
Token found in position: 31 1c -> <TWO_POINTS,none>
Token found in position: 31 2c -> <IDENTIFIER,2>
Token found in position: 41 14c -> <TYPE,none>
Token found in position: 41 1c -> <TWO_POINTS,none>
Token found in position: 41 2c -> <IDENTIFIER,3>
Token found in position: 51 16c -> <TYPE,none>
Token found in position: 51 7c -> <IDENTIFIER,4>
Token found in position: 61 12c -> <IDENTIFIER,5>
Token found in position: 61 1c -> <RELOP,EQ>
Token found in position: 61 3c -> <INT,6>
Token found in position: 61 1c -> <SEMICOLON,none>
Token found in position: 71 12c -> <IDENTIFIER,7>
Token found in position: 71 3c -> <INT,8>
Token found in position: 71 1c -> <SEMICOLON,none>
Token found in position: 81 13c -> <IF,none>
Token found in position: 81 1c -> <OPEN_PARENTHESES,none>
Token found in position: 81 2c -> <IDENTIFIER,9>
Token found in position: 81 2c -> <IDENTIFIER,10>
Token found in position: 81 6c -> <THEN,none>
Token found in position: 91 23c -> <IDENTIFIER,11>
Token found in position: 91 1c -> <RELOP,EQ>
Token found in position: 91 5c -> <IDENTIFIER,12>
Token found in position: 91 3c -> <INT,13>
Token found in position: 91 1c -> <SEMICOLON,none>
Token found in position: 101 15c -> <ELSE,none>
Token found in position: 111 15c -> <IDENTIFIER,14>
Token found in position: 111 1c -> <RELOP,EQ>
Token found in position: 111 7c -> <IDENTIFIER,15>
Token found in position: 111 2c -> <INT,16>
Token found in position: 111 1c -> <SEMICOLON,none>
Token found in position: 121 6c -> <END,none>
```

Figura 21: Tokens Recebidos pelo analisador sintático



-----Symbol table-----		
Position	Token name	Lexeme
1	IDENTIFIER	meuprograma
2	IDENTIFIER	x
3	IDENTIFIER	y
4	IDENTIFIER	cont
6	INT	50
8	INT	20
16	INT	1

Figura 22: Tabela de símbolos