Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

3. Análise Sintática

3.1 Os ajustes necessários para que a GLC da linguagem seja LL(1)

(1) Estrutura do Programa e Declarações:

```
S ::= <tipo> "main" "(" ")" <bloco>
<bloco> ::= "[" <lista_declaracao_local> <lista_comandos> "]"
<lista_declaracao_local> ::= <tipo> <lista_ids> ";" <lista_declaracao_local> |
ε
<lista_comandos> ::= <comando> <lista_comandos> | ε
<comando> ::= <if>
            | <while>
            | <do>
            | <for>
            | "id" <pos_id_comando>
<pos_id_comando> ::= ":=" <expressao> ";"
<tipo> ::= "void" | "int" | "char" | "float"
<lista_ids> ::= "id" <lista_ids'>
<lista_ids'> ::= "," "id" <lista_ids'> | ε
```

(2) Comandos:

```
<if> ::= "if" "(" <condicao> ")" "then" <entidade> <if_tail>
<if_tail> ::= <elsif> <if_tail>
          | <else>
          | ε
<elsif> ::= "elsif" "(" <condicao> ")" "then" <entidade>
<else> ::= "else" <entidade>
<while> ::= "while" "(" <condicao> ")" "do" <entidade>
<do> ::= "do" <entidade> "while" "(" <condicao> ")" ";"
<for> ::= "for" "(" "id" ";" <num> ";" <num> ";" <expressao> ")"<entidade>
<entidade> ::= <bloco> | <comando>
```

(3) Expressões Aritméticas:

```
<expressao> ::= <termo> <expr'>
<expr'> ::= "+" <termo> <expr'>
        | "-" <termo> <expr'>
        | ε
<termo> ::= <potencia> <termo'>
<termo'> ::= "*" <potencia> <termo'>
         | "/" <potencia> <termo'>
```

| ε
<potencia> ::= <fator> <potencia'>
<potencia'> ::= "**" <potencia>
| ε
<fator> ::= "id" | "const_int" | "const_float" | "const_char" | "(" <expressao> ")"

(4) Condições:

<condicao> ::= <operando_cond> <operador_relacional> <operando_cond>
<operando_cond> ::= "id" | "const_int" | "const_float" | "const_char"
<operador_relacional> ::= "==" | "!=" | "<" | ">" | "<=" | ">="

3.2 Cálculo do FIRST e FOLLOW para os símbolos da gramática

Não-Terminal	**FIRST**
S	{ void, int, char, float }
<bloco>	{ [}
<lista_declaracao_local>	{ void, int, char, float, ε }
<lista_comandos>	{ if, while, do, for, id, ε }
<comando>	{ if, while, do, for, id }
<pos_id_comando>	{ := }
<tipo>	{ void, int, char, float }
<lista_ids>	{ id }
<lista_ids'>	{ , , ε }
<if>	{ if }
<if_tail>	{ elsif, else, ε }

Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

Não-Terminal	**FIRST**
<elsif>	{ elsif }
<else>	{ else }
<while>	{ while }
<do>	{ do }
<for>	{ for }
<entidade>	{ [, if, while, do, for, id }
<expressao>	{ id, const_int, const_float, const_char, (}
<expr'>	{ +, -, ε }
<termo>	{ id, const_int, const_float, const_char, (}
<termo'>	{ *, /, ε }
<potencia>	{ id, const_int, const_float, const_char, (}
<potencia'>	{ **, ε }
<fator>	{ id, const_int, const_float, const_char, (}
<condicao>	{ id, const_int, const_float, const_char }
<operando_cond>	{ id, const_int, const_float, const_char }
<operador_relacional>	{ ==, !=, <, >, <=, >= }

Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

Não-Terminal	**FOLLOW**
S	{ $ }
<bloco>	{ $, if, while, do, for, id,], elsif, else, ; }
<lista_declaracao_local>	{ if, while, do, for, id,] }
<lista_comandos>	{] }
<comando>	{ if, while, do, for, id, "]", elsif, else }
<pos_id_comando>	{ if, while, do, for, id, "]", elsif, else }
<tipo>	{ main, id }
<lista_ids>	{ ; }
<lista_ids'>	{ ; }
<if>	{ if, while, do, for, id, "]", elsif, else }
<if_tail>	{ if, while, do, for, id, "]" }
<elsif>	{ elsif, else, if, while, do, for, id, "]" }
<else>	{ if, while, do, for, id, "]" }
<while>	{ if, while, do, for, id, "]", elsif, else }
<do>	{ if, while, do, for, id, "]", elsif, else }

Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

Não-Terminal	**FOLLOW**
<for>	{ if, while, do, for, id, "]", elsif, else }
<entidade>	{ if, while, do, for, id, "]", elsif, else, ";" }
<expressao>	{ ";", ")", "]" }
<expr'>	{ ";", ")", "]" }
<termo>	{ +, -, ;,),] }
<termo'>	{ +, -, ;,),] }
<potencia>	{ *, /, +, -, ;,),] }
<potencia'>	{ *, /, +, -, ;,),] }
<fator>	{ **, *, /, +, -, ;,),] }
<condicao>	{), ; }
<operando_cond>	{ ==, !=, <, >, <=, >=,), ; }
<operador_relacional>	{ id, const_int, const_float, const_char }

3.3 Enumeração de todas as produções da gramática:

Estrutura Global e Declarações

1. S ::= <tipo> "main" "(" ")" <bloco>
2. <bloco> ::= "[" <lista_declaracao_local> <lista_comandos> "]"
3. <lista_declaracao_local> ::= <tipo> <lista_ids> ";" <lista_declaracao_local>

4. <lista_declaracao_local> ::= ε
5. <tipo> ::= "void"
6. <tipo> ::= "int"
7. <tipo> ::= "char"
8. <tipo> ::= "float"
9. <lista_ids> ::= "id" <lista_ids'>
10. <lista_ids'> ::= "," "id" <lista_ids'>
11. <lista_ids'> ::= ε

Comandos

12. <lista_comandos> ::= <comando> <lista_comandos>
13. <lista_comandos> ::= ε
14. <comando> ::= <if>
15. <comando> ::= <while>
16. <comando> ::= <do>
17. <comando> ::= <for>
18. <comando> ::= "id" <pos_id_comando>
19. <pos_id_comando> ::= ":=" <expressao> ";"
20. <entidade> ::= <bloco>
21. <entidade> ::= <comando>

Estruturas de Controle

22. <if> ::= "if" "(" <condicao> ")" "then" <entidade> <if_tail>
23. <if_tail> ::= <elsif> <if_tail>
24. <if_tail> ::= <else>
25. <if_tail> ::= ε
26. <elsif> ::= "elsif" "(" <condicao> ")" "then" <entidade>
27. <else> ::= "else" <entidade>
28. <while> ::= "while" "(" <condicao> ")" "do" <entidade>
29. <do> ::= "do" <entidade> "while" "(" <condicao> ")" ";"

Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

30. <for> ::= "for" "(" "id" ";" <num> ";" <num> ";" <expressao> ")" <entidade>

Expressões Aritméticas

31. <expressao> ::= <termo> <expr'>
32. <expr'> ::= "+" <termo> <expr'>
33. <expr'> ::= "-" <termo> <expr'>
34. <expr'> ::= ε
35. <termo> ::= <potencia> <termo'>
36. <termo'> ::= "*" <potencia> <termo'>
37. <termo'> ::= "/" <potencia> <termo'>
38. <termo'> ::= ε
39. <potencia> ::= <fator> <potencia'>
40. <potencia'> ::= "**" <potencia>
41. <potencia'> ::= ε
42. <fator> ::= "id"
43. <fator> ::= "const_int"
44. <fator> ::= "const_float"
45. <fator> ::= "const_char"
46. <fator> ::= "(" <expressao> ")"

Condições

47. <condicao> ::= <operando_cond> <operador_relacional> <operando_cond>
48. <operando_cond> ::= "id"
49. <operando_cond> ::= "const_int"
50. <operando_cond> ::= "const_float"
51. <operando_cond> ::= "const_char"
52. <operador_relacional> ::= "=="
53. <operador_relacional> ::= "!="
54. <operador_relacional> ::= "<"
55. <operador_relacional> ::= ">"

56. <operador_relacional> ::= "<="

57. <operador_relacional> ::= ">="

3.4 Tabela de Análise Preditiva:

Não-Terminal/Terminais	void, int, char, float	main	id	const	(	)	[	]	;	:=	,	if, while, do, for	then, do, elsif, else	relac. (==, etc)	aritm. (+, -, *, /, **)	$
S	1															
<bloco>							2									
<lista_decl_loc>	3							4				4				
<tipo>	5,6,7,8															
<lista_ids>			9													
<lista_ids'>									11		10					
<lista_com>			12					13				12				
<comando>			18									14,15,16,17				
<pos_id_com>										19						
<entidade>			21				20					21				
<if>												22				
<if_tail>			25					25				25	23 (elsif), 24 (else)			
<elsif>													26			
<else>													27			
<while>												28				
<do>												29				
<for>												30				
<expressao>			31	31	31											
<expr'>						34		34	34						32, 33	
<termo>			35	35	35											
<termo'>						38		38	38						36, 37	
<potencia>			39	39	39											
<potencia'>						41		41	41						40	
<fator>			42	43,44,45	46											
<condicao>			47	47												
<operando_cond>			48	49,50,51												
<oper_relac>														52 a 57		