

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'>
```

Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

$\mid \epsilon$
<potencia> ::= <fator> <potencia'>
<potencia'> ::= "*" <potencia>

$\mid \epsilon$
<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, else }
<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. $S ::= \langle \text{tipo} \rangle \text{"main"} \text{"(" " " } \langle \text{bloco} \rangle$
2. $\langle \text{bloco} \rangle ::= \text{"[" } \langle \text{lista_declaracao_local} \rangle \langle \text{lista_comandos} \rangle \text{"}"}$
3. $\langle \text{lista_declaracao_local} \rangle ::= \langle \text{tipo} \rangle \langle \text{lista_ids} \rangle \text{";"}$
 $\langle \text{lista_declaracao_local} \rangle$

Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

4. $\langle \text{lista_declaracao_local} \rangle ::= \epsilon$
5. $\langle \text{tipo} \rangle ::= \text{"void"}$
6. $\langle \text{tipo} \rangle ::= \text{"int"}$
7. $\langle \text{tipo} \rangle ::= \text{"char"}$
8. $\langle \text{tipo} \rangle ::= \text{"float"}$
9. $\langle \text{lista_ids} \rangle ::= \text{"id"} \langle \text{lista_ids}' \rangle$
10. $\langle \text{lista_ids}' \rangle ::= \text{" , " id " } \langle \text{lista_ids}' \rangle$
11. $\langle \text{lista_ids}' \rangle ::= \epsilon$

Comandos

12. $\langle \text{lista_comandos} \rangle ::= \langle \text{comando} \rangle \langle \text{lista_comandos} \rangle$
13. $\langle \text{lista_comandos} \rangle ::= \epsilon$
14. $\langle \text{comando} \rangle ::= \langle \text{if} \rangle$
15. $\langle \text{comando} \rangle ::= \langle \text{while} \rangle$
16. $\langle \text{comando} \rangle ::= \langle \text{do} \rangle$
17. $\langle \text{comando} \rangle ::= \langle \text{for} \rangle$
18. $\langle \text{comando} \rangle ::= \text{"id"} \langle \text{pos_id_comando} \rangle$
19. $\langle \text{pos_id_comando} \rangle ::= \text{" := " } \langle \text{expressao} \rangle \text{" ; "}$
20. $\langle \text{entidade} \rangle ::= \langle \text{bloco} \rangle$
21. $\langle \text{entidade} \rangle ::= \langle \text{comando} \rangle$

Estruturas de Controle

22. $\langle \text{if} \rangle ::= \text{"if"} \text{" (" } \langle \text{condicao} \rangle \text{")"} \text{" then"} \langle \text{entidade} \rangle \langle \text{if_tail} \rangle$
23. $\langle \text{if_tail} \rangle ::= \langle \text{elsif} \rangle \langle \text{if_tail} \rangle$
24. $\langle \text{if_tail} \rangle ::= \langle \text{else} \rangle$
25. $\langle \text{if_tail} \rangle ::= \epsilon$
26. $\langle \text{elsif} \rangle ::= \text{"elsif"} \text{" (" } \langle \text{condicao} \rangle \text{")"} \text{" then"} \langle \text{entidade} \rangle$
27. $\langle \text{else} \rangle ::= \text{"else"} \langle \text{entidade} \rangle$
28. $\langle \text{while} \rangle ::= \text{"while"} \text{" (" } \langle \text{condicao} \rangle \text{")"} \text{" do"} \langle \text{entidade} \rangle$
29. $\langle \text{do} \rangle ::= \text{"do"} \langle \text{entidade} \rangle \text{" while"} \text{" (" } \langle \text{condicao} \rangle \text{")"} \text{" ; "}$

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> ::= ">"

Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

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		