

# **Universidade Federal de Uberlândia**

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

Prof. Luiz Gustavo Almeida Martins

Lucas de Paula Martins – 12221BCC001

Cauê Grassi Ribeiro da Silva - 12221BCC019

## **3<sup>a</sup> Etapa do Projeto**

Uberlândia, Minas Gerais

10 de fevereiro de 2026

# Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

## 1. Projeto da Linguagem

### 1.1. Gramática Livre de Contexto (GLC)

$G = (V, T, P, S)$

Definições:

- $G$  = gramática livre de contexto (GLC) da linguagem
- $T$  = conjunto de símbolos terminais (Tokens)
- $V$  = conjunto de símbolos não terminais (Variáveis Sintáticas)
- $P$  = conjunto de produções (Regras de Transição)
- $S$  = símbolo inicial da gramática (Símbolo Sentencial)

#### Conjunto de Terminais (T)

$T = \{"void", "int", "char", "float", "main", "if", "then", "elsif", "else", "while", "do", "for", "+", "-", "**", "/", "**", "==", "!=" , "<", ">" , "<=", ">=" , ":" , "(" , ")" , "[" , "]" , "," , ";" , "id", "const_int" | "const_float" | "char_const"\}$

#### Conjunto de Variáveis (V)

$V = \{\langle\text{programa}\rangle, \langle\text{bloco}\rangle, \langle\text{lista_comandos}\rangle, \langle\text{comando}\rangle, \langle\text{decl}\rangle, \langle\text{lista_decl}\rangle, \langle\text{tipo}\rangle, \langle\text{lista_ids}\rangle, \langle\text{atribuicao}\rangle, \langle\text{if}\rangle, \langle\text{lista_elsif}\rangle, \langle\text{elsif}\rangle, \langle\text{else}\rangle, \langle\text{entidade}\rangle, \langle\text{while}\rangle, \langle\text{do}\rangle, \langle\text{for}\rangle, \langle\text{expressao}\rangle, \langle\text{termo}\rangle, \langle\text{fator}\rangle, \langle\text{condicao}\rangle, \langle\text{operando_cond}\rangle, \langle\text{operador_relacional}\rangle, \langle\text{constante}\rangle\}$

#### Símbolo Inicial S

$S = \{\langle\text{programa}\rangle\}$

#### Conjunto de Produções (P)

##### (1) Estrutura do Programa

```
<programa> ::= <tipo> "main" "(" ")" <bloco>
<bloco> ::= "[" <lista_declaracao> <lista_comandos> "]"
<lista_comandos> ::= <comando> <lista_comandos> | ε
<comando> ::= <if>
            | <while>
            | <do>
            | <for>
            | <atribuicao>
```

##### (2) Declarações

# Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

<lista\_declaracao> ::= <declaracao> <lista\_declaracao> | ε

<declaracao> ::= <tipo> <lista\_ids> ;

<tipo> ::= "void" | "int" | "char" | "float"

<lista\_ids> ::= "id" | "id" , <lista\_ids>

## (3) Comandos

<atribuicao> ::= <id> := <expressao> ;

<if> ::= if "(" <condicao> ")" then <entidade> <lista\_elsif>

<lista\_elsif> ::= <elsif> <lista\_elsif> | <else> | ε

<elsif> ::= elseif "(" <condicao> ")" then <entidade>

<else> ::= else <entidade>

<entidade> ::= <bloco> | <comando>

<while> ::= while "(" <condicao> ")" do <entidade>

<do> ::= do <entidade> while "(" <condicao> ")" ;

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

## (4) Expressões Aritméticas

<expressao> ::= <expressao> + <expressao>

| <expressao> - <expressao>

| <expressao> \* <expressao>

| <expressao> / <expressao>

| <expressao> \*\* <expressao>

| "id" | "const\_int" | "const\_float" | "char\_const"

## (5) Condições

<condicao> ::= <operando\_cond> <operador\_relacional> <operando\_cond>

<operando\_cond> ::= "id" | "const\_int" | "const\_float" | "char\_const"

<operador\_relacional> ::= == | != | < | > | <= | >=

## 1.2. Identificação dos Tokens da Linguagem

### (1) Palavras-chave

Token	Lexema	Atributo	Descrição
T_VOID	void	—	Tipo void
T_INT	int	—	Tipo inteiro
T_CHAR	char	—	Tipo caractere
T_FLOAT	float	—	Tipo float
T_MAIN	main	—	Função principal
T_IF	if	—	Comando condicional
T_THEN	then	—	Parte do if
T_ELSIF	elseif	—	Else condicional
T_ELSE	else	—	Else simples
T WHILE	while	—	Laço while

# Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

T_DO	do	—	Início do bloco do-while
------	----	---	--------------------------

## (2) Operadores Aritméticos

Token	Lexema	Atributo
T_SOMA	+	—
T_SUB	-	—
T_MULT	*	—
T_DIV	/	—
T_POT	**	—

## (3) Operadores Relacionais

Token	Lexema	Atributo
T_IGUAL	==	—
T_DIF	!=	—
T_MENOR	<	—
T_MAIOR	>	—
T_MENOR_IG	<=	—
T_MAIOR_IG	>=	—

## (4) Símbolos e Separadores

Token	Lexema	Atributo
T_ABREPAR	(	—
T_FECHAPAR	)	—
T_ABREBLOCO	[	—
T_FECHABLOCO	]	—
T_VIRG	,	—
T_PV	;	—
T_ATRIB	:=	—

## (5) Constantes e Identificadores

Token	Exemplo	Atributo
T_ID	idade	string (lexema)
T_NUM_INT	123	valor inteiro
T_NUM_FLOAT	3.14, 0.1E-2	valor float
T_CHAR_CONST	'A'	caractere

# Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

## (6) Comentários e Tokens Especiais

Token	Padrão	Atributo
T_COMENT	{% ... %}	ignorado
T_WS	espaços, tabs, quebras	ignorado
T_EOF	fim do arquivo	—
T_ERROR	qualquer sequência inválida	string (lexema)

### 1.3. Expressões Regulares de Cada Token

#### Palavras-chave

"void" → T\_VOID  
"int" → T\_INT  
"char" → T\_CHAR  
"float" → T\_FLOAT  
"main" → T\_MAIN  
"if" → T\_IF  
"then" → T\_THEN  
"elsif" → T\_ELSIF  
"else" → T\_ELSE  
"while" → T WHILE  
"do" → T\_DO  
"for" → T\_FOR

#### Operadores Aritméticos

"+" → T\_SOMA  
"-" → T\_SUB  
"\*\*" → T\_MULT  
"/" → T\_DIV  
"\*\*\*" → T\_POT

#### Operadores Relacionais

"==" → T\_IGUAL

# Universidade Federal de Uberlândia

Bacharelado em Ciência da Computação

GBC071 Construção de Compiladores

"!+"	→ T_DIF
"<="	→ T_MENOR_IG
">="	→ T_MAIOR_IG
"<"	→ T_MENOR
">"	→ T_MAIOR

## Símbolos / Separadores

"("	→ T_ABREPAR
")"	→ T_FECHAPAR
"["	→ T_ABREBLOCO
→ T_FECHABLOCO	
" , "	→ T_VIRG
" ; "	→ T_PV
" := "	→ T_ATRIB

## Identificadores e Constantes

[A-Za-z_]	[A-Za-z0-9_]*	→ T_ID	
[0-9]+		→ T_NUM_INT	
[0-9]+	. [0-9]+	(E [+ -]?)? [0-9]+)	→ T_NUM_FLOAT
'[A-Za-z]'		→ T_CHAR_CONST	

## Comentários

\{%.(\.|\\n)\*?%\} → T\_COMENT

## Espaços e Separadores (ignorados)

[ \\t\\r\\n]+ → T\_WS

## Tokens especiais

EOF → T\_EOF

. → T\_ERROR