Descrição da Linguagem Regular

Felipe Gomes da Silva
Luis Henrique Salomão Lobato

Setembro, 2025

1 Tokens Reconhecidos

O analisador léxico da linguagem Simple C toma a decisão de aceitar ou não determinada sentença com base nos tokens destacados na Tabela 1

Tabela 1: Tabela de Tokens do Simple C

Nome do Token	Lexema(s)	Descrição
Palavras-chave: Tipos de Dados		
KEYWORD_INT	`int`	Palavra-chave para tipo inteiro.
KEYWORD_FLOAT	`float`	Palavra-chave para tipo ponto flutuante.
KEYWORD_CHAR	`char`	Palavra-chave para tipo caractere.
KEYWORD_STRING	`string`	Palavra-chave para tipo string (cadeia de caracteres).
KEYWORD_VOID	`void`	Palavra-chave para tipo vazio/nulo.
KEYWORD_BOOL	`bool`	Palavra-chave para tipo booleano.
Palavras-chave: Controle de Fluxo e Comandos		
KEYWORD_IF	`if`	Inicia uma estrutura condicional.
KEYWORD_ELSE	`else`	Bloco alternativo de uma estrutura condicional.
KEYWORD_FOR	`for`	Inicia um laço de repetição ‘for‘.
KEYWORD_WHILE	`while`	Inicia um laço de repetição ‘while‘.
KEYWORD_DO	`do`	Inicia um laço de repetição ‘do-while‘.
KEYWORD_SWITCH	`switch`	Inicia uma estrutura de seleção múltipla.
KEYWORD_CASE	`case`	Define um rótulo dentro de um ‘switch‘.
KEYWORD_DEFAULT	`default`	Define o rótulo padrão de um ‘switch‘.
KEYWORD_BREAK	`break`	Interrompe a execução de um laço ou ‘switch‘.
KEYWORD_CONTINUE	`continue`	Pula para a próxima iteração de um laço.
KEYWORD_RETURN	`return`	Retorna um valor de uma função.
Identificadores e Literais		
IDENTIFICADOR	`var`, `_x`, ...	Nome de variável, função, etc.
INT	`123`, `42`	Valor literal inteiro.
FLOAT	`3.14`, `0.5`	Valor literal de ponto flutuante.
CHAR	`'a'`, `'\n'`	Valor literal de caractere.
STRING	`"hello"`	Valor literal de string.
BOOLEAN_LITERAL	`true`, `false`	Valor literal booleano.
Operadores		
OP_SOMA	`+`	Operador de adição.
OP_SUB	`-`	Operador de subtração.
OP_MULT	`*`	Operador de multiplicação.
OP_DIV	`/`	Operador de divisão.
OP_MOD	`%`	Operador de módulo (resto da divisão).
OP_ATRIBUICAO	`=`	Operador de atribuição simples.
OP_INC_ATRIBUICAO	`+=`	Operador de atribuição com adição.
OP_DEC_ATRIBUICAO	`-=`	Operador de atribuição com subtração.
OP_MULT_ATRIBUICAO	`*=`	Operador de atribuição com multiplicação.
OP_DIV_ATRIBUICAO	`/=`	Operador de atribuição com divisão.
OP_INC	`++`	Operador de incremento.

Tabela 1: Tabela de Tokens do Simple C (Continuação)

Nome do Token	Lexema(s)	Descrição
OP_DEC	`--`	Operador de decremento.
OP_IGUAL	`==`	Operador relacional de igualdade.
OP_DIFERENTE	`!=`	Operador relacional de desigualdade.
OP_MENOR	`<`	Operador relacional menor que.
OP_MAIOR	`>`	Operador relacional maior que.
OP_MENOR_IGUAL	`<=`	Operador relacional menor ou igual que.
OP_MAIOR_IGUAL	`>=`	Operador relacional maior ou igual que.
OP_AND	`&&`	Operador lógico E (AND).
OP_OR	`||`	Operador lógico OU (OR).
OP_NOT	`!`	Operador lógico de negação (NOT).
Pontuadores e Delimitadores		
PONTO_VIRGULA	`;`	Finalizador de instrução.
DOIS_PONTOS	`:`	Usado em casos de ‘switch‘.
VIRGULA	`,`	Separador de elementos (ex: em listas).
ABRE_PARENTESES	`(`	Abre lista de parâmetros ou expressão.
FECHA_PARENTESES	`)`	Fecha lista de parâmetros ou expressão.
ABRE_CHAVES	`{`	Abre um bloco de código.
FECHA_CHAVES	`}`	Fecha um bloco de código.
ABRE_COLCHETES	`[`	Abre a declaração/acesso de um array.
FECHA_COLCHETES	`]`	Fecha a declaração/acesso de um array.

2 Descrição da linguagem regular para o analisador léxico

A linguagem regular estruturada para a aceitação de cadeias pelo analisador léxico da linguagem Simple-C pode ser descrita através das expressões regulares e definições presentes na Tabela 2. Deste modo, é garantido que variáveis não sejam declaradas com números como primeiro caractere da cadeia, além de outros erros léxicos devidamente discutidos.

Tabela 2: Expressões regulares do analisador léxico.

Expressão Regular	**Descrição**	**Tipo**
`[a-zA-Z_][a-zA-Z0-9_]*`	Reconhece identificadores válidos que começam com uma letra ou um sublinhado.	Identificador
`0|[+-]?[1-9][0-9]*`	Reconhece números inteiros.	Literal
`[+-]?([0-9]+\.[0-9]*|\.[0-9]+)`	Reconhece números de ponto flutuante.	Literal
`//.*`	Comentário de linha única, que começa com `//` e vai até o final da linha.	Comentário
`/*`	Inicia um comentário de bloco.	Comentário
`<IN_COMMENT>"*/"`	Fecha um comentário de bloco.	Comentário
`"(\.|[^"\n ])+"`	Conteúdo de uma string, incluindo caracteres escapados.	String
`'(\.|[^'\\])'`	Reconhece literais de caracteres.	Caractere
`==, !=, <=, >=, etc.`	Operadores de comparação, lógicos e aritméticos.	Operador
`;, <, >, {, }`	Símbolos de pontuação e agrupamento.	Símbolo
`{INT}{ID}`	Captura identificadores que começam com um número, que é um erro léxico.	Erro
`.`	Captura qualquer caractere que não corresponda a nenhuma regra.	Erro
`[+-]?0[0-9]+`	Captura números inteiros com zeros à esquerda, que é um erro léxico.	Erro