

# Analizador sintático e léxico para a linguagem C- - Demonstrando o software

Victor E. Almeida    Marco A. G. Pedroso

UNIOESTE

11 de abril de 2022

# Conteúdo

- 1 Introdução
- 2 lex/Flex
- 3 Analisador léxico
- 4 Conclusão

# Softwares utilizados

- **Sistema para compilar:** GNU make e gcc,
- **Linguagem de programação:** C11,
- **Gerador de analisador léxico:** GNU flex;
- **Gerador de analisador sintático:** GNU bison.

# Exemplo mínimo do flex

---

```
1      int num_lines = 0, num_chars = 0;
2
3      %%
4      \n      ++num_lines; ++num_chars;
5      .      ++num_chars;
6      %%
7
8      int main() {
9          yylex();
10         printf("lines = %d, chars = %d\n",
11                num_lines, num_chars);
12         return 0;
13     }
```

---

# Funcionamento do lex/flex

---

```
1  EXPRESSION_BEGIN "("
2  {EXPRESSION_BEGIN} {
3      doLog (
4          LOG_TYPE_INFO,
5          "inicio de expressao [(] encontrado"
6      );
7      is_open_expression++;
8  }
```

---

# Lista de Tokens reconhecidos I

- **comando para o preprocessador:** `"#"(.*)`
- **palavras reservadas:**  
`"if" | "else" | "const" | "for" | "while" | "struct"`
- **tipos de dados:**  
`"int" | "float" | "double" | "char"`
- **atribuição:**  
`("+" | "-" | "*" | "/" | "%" | "<<" | ">>" | "&" | "|" | "^")? "="`
- **operador aritmético:**  
`"+" "+"? | "-" "-"? | "/" | "*" | "sizeof" |  
"[" {INTEGER_LITERAL} "]"`
- **operador relacional:**  
`"&&" | "||" | "!" | ("=" | "!") "=" | ("<" | ">") "="?`

# Lista de Tokens reconhecidos II

- fim de expressão: ;
- início de bloco: {
- fim de bloco: }
- abre parenteses: (
- fecha parenteses: )
- inteiro literal: {DIGIT}+
- float literal:  
{INTEGER\_LITERAL} "." {INTEGER\_LITERAL}
- string literal: \"[^\n\"]+
- char literal: '\\[?\\.\\']
- identificador: ({LETTER})( {ALPHA\_NUM} | \_ )\*

# Mão na massa!!





# Agradecimentos

Perguntas?



Obrigado pela atenção