Relatório Técnico: Análise do Código Léxico

1. Structs

1.1 Symbol

- Propósito: Representa um identificador ou palavra reservada.

- Campos:

  - lexeme[100]: Armazena o nome do identificador ou da palavra reservada.

1.2 Token

- Propósito: Representa uma unidade léxica identificada no código de entrada.

- Campos:

  - TokenName token: Tipo de token (identificador, palavra-chave, número, operador, etc.).

  - char lexeme[100]: O valor real do token, como o identificador ou número.

  - int line: Linha do código onde o token foi encontrado.

  - int column: Coluna onde o token começa.

---

2. Funções

2.1 addSymbolToTable(Token token)

- Propósito: Adiciona um novo símbolo (identificador) à tabela de símbolos, evitando duplicatas.

- Comportamento: Percorre a tabela de símbolos existente para verificar duplicatas. Caso o símbolo ainda não esteja presente, ele é adicionado à tabela.

2.2 writeSymbolTable(FILE *file)

- Propósito: Exibe e grava a tabela de símbolos no arquivo fornecido.

- Comportamento: Percorre a tabela de símbolos e grava os identificadores armazenados no arquivo de saída.

2.3 logLexicalError(FILE *file, char invalidChar, int line, int column)

- Propósito: Grava erros léxicos, como caracteres inválidos, no arquivo de saída.

- Comportamento: Escreve uma mensagem de erro contendo o caractere inválido e sua posição (linha e coluna).

2.4 writeTokenToFile(FILE *file, Token token)

- Propósito: Grava os tokens identificados no arquivo de saída.

- Comportamento: Converte o tipo de token para uma string correspondente e grava o lexema, linha e coluna no arquivo. No caso de identificadores (ID), também os adiciona à tabela de símbolos.

2.5 is_identifier(char *lexeme)

- Propósito: Verifica se um lexema é um identificador válido.

- Comportamento: Verifica se o primeiro caractere é uma letra ou sublinhado e se os demais são letras, dígitos ou sublinhados.

2.6 is_integer(char *lexeme)

- Propósito: Verifica se um lexema representa um número inteiro válido.

- Comportamento: Confirma se todos os caracteres do lexema são dígitos.

2.7 is_float(char *lexeme)

- Propósito: Verifica se um lexema representa um número flutuante válido.

- Comportamento: Conta pontos decimais e garante que haja exatamente um ponto e que todos os

outros caracteres sejam dígitos.

2.8 recognizeOperatorOrSymbol(char *input, int *position, int line, int column)

- Propósito: Reconhece operadores e símbolos a partir da entrada.

- Comportamento: Identifica operadores como +, -, =, >=, e símbolos como {, }, (,), gravando-os como tokens.

2.9 recognizeIdentifier(char *input, int *position, int line, int column)

- Propósito: Identifica identificadores e palavras reservadas na entrada.

- Comportamento: Coleta o lexema e verifica se corresponde a uma palavra reservada ou identificador válido.

2.10 recognizeNumber(char *input, int *position, int line, int column)

- Propósito: Identifica números inteiros ou flutuantes na entrada.

- Comportamento: Coleta dígitos inteiros e verifica a presença de um ponto decimal para determinar se o número é flutuante.

2.11 lexicalAnalysis(char *input, FILE *file)

- Propósito: Função principal de análise léxica, que percorre a entrada e identifica tokens.

- Comportamento: Divide a entrada em tokens (identificadores, números, operadores, etc.) e grava esses tokens em um arquivo de saída.

2.12 main(int argc, char *argv[])

- Propósito: Função principal que coordena a execução da análise léxica.

- Comportamento: Abre o arquivo de entrada fornecido pelo usuário, executa a análise léxica em cada linha e grava a tabela de símbolos e tokens em um arquivo de saída.