# Relatório Técnico MinC – Trabalho 2

Francisco Castro Del'Gaudio Junior e Hugo Gustavo Cordeiro

1 de outubro de 2025

## Sumário

# 1 Descrição do Funcionamento do Software

O software desenvolvido realiza a análise léxica e sintática de programas escritos na linguagem `MinC`.

A análise léxica continua sendo implementada com o auxílio da ferramenta `Flex`, responsável por gerar o analisador léxico a partir do arquivo `lexico.l`. Já a análise sintática foi implementada com `Bison` (`sintatico.y`), responsável por validar a estrutura gramatical e construir a árvore sintática do programa.

## Etapas de Execução do Software

A seguir, estão listados os passos para compilar e executar o compilador da linguagem `MinC`:

1. Abra o terminal na pasta onde estão localizados os arquivos `lexico.l` e `sintatico.y`.
2. Gere o código-fonte em C a partir do arquivo léxico utilizando o Flex:

   ```
   flex lexico.l
   ```

   Isso criará automaticamente um arquivo chamado `lex.yy.c`.
3. Gere o código-fonte do analisador sintático com o Bison:

   ```
   bison -d sintatico.y
   ```

   Isso criará os arquivos `sintatico.tab.c` e `sintatico.tab.h`.
4. Compile todos os arquivos necessários com o `gcc`:

   ```
   gcc main.c lex.yy.c sintatico.tab.c lista.c arvore.c -o main.exe
   ```

   Isso produzirá o executável `main.exe`.
5. Execute o programa:

   ```
   ./main.exe
   ```

6. Ao iniciar, o programa solicitará o nome de um arquivo de entrada contendo o código a ser analisado (por exemplo, `exemplo.minc`).
7. Após a análise, será exibido um menu com opções:
   - `0` - Sair;
   - `1` - Imprimir lista de palavras reservadas;
   - `2` - Imprimir lista de símbolos;
   - `3` - Imprimir árvore sintática;
   - `4` - Abrir outro arquivo.

### Funcionamento Geral

O compilador percorre o conteúdo do arquivo de entrada linha por linha e realiza as seguintes funções:

- Identificação de tokens léxicos válidos;
- Verificação da estrutura sintática;
- Construção da árvore sintática com base nas regras da gramática;
- Emissão de mensagens de erro com linha e coluna em caso de problemas léxicos ou sintáticos.

Cada token e cada produção sintática são armazenados em estruturas de dados (`Lista` e `Árvore`), permitindo posterior exibição ao usuário.

## 2 Descrição da Linguagem

A linguagem `MinC` é inspirada na linguagem C, possuindo:

- Tipos primitivos: `int`, `float`, `double`, `char`, `void`;
- Estruturas de controle: `if/else`, `for`, `while`;
- Entrada e saída padrão: `printf`, `scanf`;
- Operadores aritméticos, relacionais e lógicos;
- Strings, caracteres e macros (#include, #define).

A gramática implementada no Bison possui pelo menos 20 produções, incluindo laços de repetição, condicionais aninhados e expressões.

## 3 Tipos de Erros Tratáveis

O sistema trata erros léxicos e sintáticos.

### Erros Léxicos

- Identificadores inválidos (ex: começando por número);
- Números reais malformados (ex: 0. ou 42.a);
- Strings e caracteres não finalizados;
- Símbolos desconhecidos.

### Erros Sintáticos

- Falta de ponto e vírgula em comandos;
- Estruturas de controle mal formadas;
- Uso inesperado de tokens (ex: `else` sem `if`).

# 4 Documentação dos Principais Métodos/Funções

- `Inserir / ImprimirLista (lista.c)`: mantém a tabela de símbolos e palavras reservadas;
- `insere (arvore.c)`: adiciona nós à árvore sintática;
- `imprimirPorNivel (arvore.c)`: gera a árvore sintática em um arquivo de saída;
- `yylex (lex.yy.c)`: análise léxica gerada pelo Flex;
- `yyparse (sintatico.tab.c)`: análise sintática gerada pelo Bison;
- `main (main.c)`: coordena entrada do usuário, análise e menu.

# 5 Processo de Construção do Software

## Ambiente de Desenvolvimento

- IDE: Visual Studio Code
- Compilador: MinGW (GCC, Flex, Bison)
- SO: Windows 10

## Ferramentas

- Flex: analisador léxico;
- Bison: analisador sintático;
- GCC: compilador C.

# 6 Referências Bibliográficas


- OpenAI. *ChatGPT* (set. 2025). Assistente utilizado para apoio técnico e redação deste relatório. Disponível em: `https://chat.openai.com/`
- Aho, A. V.; Lam, M. S.; Sethi, R.; Ullman, J. D. *Compilers: Principles, Techniques, and Tools*.