CONSTRUÇÃO DE COMPILADORES I

Checkpoint 01: Analisador Léxico

Esta atividade pode ser feita em grupo (até 4 integrantes) e consiste em fazer alterações no código do analisador léxico feito em sala de aula, disponível em https://gitlab.com/maelso/mini_compiler na *branch main*.

Importante: Esta atividade corresponderá a **2 pontos** na **primeira nota da disciplina**.

Obs. O aluno pode escolher implementar a solução em qualquer linguagem, como Java, Python, C/C++, etc. Porém, o analisador léxico deve possuir sua estrutura semelhante à que foi mostrada em sala de aula. ***Não é permitido o uso de ferramentas automatizadas de análise léxica.***

Tendo como base o código implementado em sala e enviado para a branch acima mencionada, faça as seguintes alterações:

1. (1 ponto) Reconhecimento de identificadores:
   a) (a-z | A-Z | _)(a-z | A-Z | _ | 0-9)*

2. (1 ponto) Adicione suporte ao analisador léxico para reconhecer os seguintes operadores matemáticos:
   a) soma(+)
   b) subtração (-)
   c) multiplicação (*)
   d) divisão(/).

3. (1 ponto) Adicione suporte ao analisador léxico para reconhecer o operador de atribuição (=).

4. (1 pontos) Adicione suporte ao analisador léxico para reconhecer os seguintes operadores relacionais:
   a) maior que (>)
   b) maior ou igual que (>=)
   c) menor que (<)
   d) menor ou igual que (<=)
   e) diferente (!=)
   f) igual (==)

5. (1 ponto) Adicione suporte ao analisador léxico para reconhecer:
   a) parêntese esquerdo (‘(’)
   b) parêntese direito (‘)’)

6. (1 ponto) Adicione suporte no analisador léxico para reconhecer constantes numéricas com ponto decimal:
   Exemplos válidos: 123, 123.456, .456

   Exemplos inválidos: 1., 12., 156.

Expressão regular: ((0-9)*.)?(0-9)+

7. (1 ponto) Adicione suporte no analisador léxico para reconhecer as seguintes palavras reservadas:
   a) int
   b) float
   c) print
   d) if
   e) else

   Obs.: Use uma tabela de palavras reservadas e, antes de criar um novo token, verifique se é uma palavra reservada, caso positivo, retorne da tabela.

8. (1,5 pontos) Suporte a comentários:

   Comentário de linha única: iniciados por #, válidos até o final da linha (\n ou \r).

   Comentário de múltiplas linhas**:** delimitados por /* e */, podendo ocupar várias linhas.

   Comentários devem ser **ignorados** (não geram tokens).

9. (1.5 ponto) Ao encontrar qualquer erro léxico retorne uma mensagem de erro com a linha e a coluna do código fonte que foi encontrado o símbolo gerador do erro. Obs.: Caracteres não permitidos (@, `, ´, ç, ¨, etc, etc) também devem gerar um erro léxico.

Entrega

Os trabalhos devem ser entregues até o dia **18/09/2025**, impreterivelmente até as 12 horas (**meio-dia**). O envio deverá ser feito via **BlackBoard**. Para grupos, **apenas um integrante faz o envio**, porém os nomes dos demais integrantes devem estar no código (na classe Main), obrigatoriamente.

Deve ser enviado o código fonte compactado, junto com instruções para execução (possíveis dependências para instalação).

Atraso na entrega do trabalho acarretará nota zero.

A nota final dependerá da sua defesa oral em sala.