Análise Léxica

- Descrição da tarefa -

**Objetivos:**

1 - “ler os enunciados no programa fonte, reconhecer e classificar os vários tokens” (Aula 3). Além de separar as linhas do código fonte em blocos, é preciso que cada bloco seja analisado e classificado em uma das possíveis categorias: Label, Symbol, Operator, Immediate, Register, Brackets. Caso um bloco não possa ser classificado ele é um token inválido. Neste caso o analisador léxico deve parar a execução e sinalar um erro léxico.

2 - “inicia a construção da Tabela de Símbolos” (Aula 3). Os símbolos são variáveis, procedimentos e sub-rotinas declarados pelo usuário. No caso da linguagem assembly, os símbolos são os rótulos. Essa tabela é utilizada pelo analisador léxico para verificar que o código não contém símbolos declarados mais de uma vez. Posteriormente, o analisador semântico também utiliza essa tabela para verificar que o código não contém chamadas a símbolos não definidos.

**Entradas e Saídas:**

**Entradas:**

1 - Arquivo texto (.s) contendo o código fonte.

2 - Tabela de instruções: Contém a lista de todas as instruções existentes na linguagem.

3 - Tabela de registradores: Contém a lista de todos os registradores existentes na linguagem.

4 - Tabela de diretivas: Contém a lista de todas as diretivas implementadas pelo montador.

**Saídas:**

1 - Lista de tokens: Pode ser implementada como uma lista encadeada começando no primeiro token do programa e seguindo até o último. Cada token pode ser representado por um struct com dois campos (valor e tipo). No caso da linguagem assembly, como cada instrução tem que ser escrita em uma única linha, esta informação pode ser utilizada para facilitar o trabalho dos demais analisadores. Nesse caso, podemos agrupar os tokens que são da mesma linha construindo uma lista de lista de tokens.

2 - Tabela de símbolos: Pode ser implementada como uma lista encadeada de símbolos. Cada símbolo pode ser representado por um único string.