

## Trabalho Prático

A construção de um compilador para uma linguagem imperativa simplificada

### Prática No.3 – Verificação de unicidade, classes e tipos

Nesta fase, o analisador sintático construído na segunda parte deverá ser transformado em um esquema de tradução que fará a verificação semântica do programa-fonte. Esta fase é um passo intermediário para a obtenção do tradutor (compilador completo), que será capaz de gerar código Assembly para a linguagem L.

1. Modifique a tabela de símbolos, acrescentando os seguintes campos:
  - **Classe**, que poderá assumir os valores *classe-var* e *classe-const*. Estes valores serão definidos para cada identificador, no momento de sua declaração;
  - **Tipo**, que poderá assumir os valores *tipo-inteiro*, *tipo-lógico*, *tipo-byte* e *tipo-string*. Estes valores serão definidos para identificadores de variáveis e constantes, no momento de sua declaração.
2. Verifique a unicidade dos identificadores: cada identificador não pode ser declarado mais de uma vez, nem pode ser usado sem ter sido antes declarado.
3. Verifique a compatibilidade de classes de identificadores nas regras da gramática. Por exemplo, um comando de atribuição só pode atribuir valores a variáveis.
4. Verifique a compatibilidade de tipos das expressões, em cada comando. Lembre-se que os tipos inteiro e byte podem ser misturados nas expressões, valendo as seguintes regras:
  - Uma constante numérica é um byte se estiver no intervalo [0,255], caso contrário é considerada inteira.
  - Um valor negado (-) se transforma em inteiro.
  - Operações entre inteiro e byte resultam em inteiro.
  - Para a divisão, os operandos devem ser convertidos para inteiro.
  - Podem ser feitas comparações entre inteiros e bytes; o resultado é sempre do tipo lógico.
  - Strings só podem ser concatenados (+) e comparados quanto à sua igualdade (==).

O tipo das expressões pode ser verificado recursivamente, com o auxílio de um atributo sintetizado que é passado como parâmetro por referência.

5. Teste o analisador semântico com exemplos de programas-fontes corretos e exemplos de erros semânticos.

As mensagens devem ter os seguintes formatos (onde *nn* é o número da linha onde o erro foi detectado e *lex* é o lexema encontrado):

*nn*:identificador nao declarado [*lex*].

*nn*:identificador ja declarado [*lex*].

*nn*:classe de identificador incompatível [*lex*].

*nn*:tipos incompatíveis.

**O que entregar:**

- Códigos-fontes
- Esquema de tradução

**Obs: Leia as especificações gerais contidas no documento “Descrição do trabalho”.**