Análise Semântica

**Tabela de Símbolos** – estrutura principal da compilação: relacionada com todas as etapas.

Não é possível realizer a análise semântica sem a tabela de símbolos

Essencial para geração de código (é necessário verificar, por exemplo, o tipo de variável desejada, de acordo com cada tipo vai ser um código gerado distinto) 🡪 define o tipo de código que será gerado

Também utilizada para visualizar as palavras reservadas e símbolos especiais da linguagem.

Principais operações:

Inserir: armazena informações fornecidas pelas declarações (nome, tipo, escopo).

Verificar: recupera informação relacionado ao elemento sempre que for utilizado.

Remover: remove (ou torna inacessível) as informações a respeito de elementos que foram declarados e não são mais úteis.

*Obs: comportamento da tabela de símbolo depende fortemente das propriedades da linguagem sendo compilada.*

Toda hora que depara-se com um identificar há o acesso a tabela de símbolos:

* Verificar ou inserir um elemento;
* Verificar seu tipo, escopo ou alguma outra informação;
* Atualizar alguma informação associada ao identificador;
* Remover um elemento quando este não se faz mais necessário.

Acesso eficiente à tabela:

Implementação

- Estática

- Dinâmica: melhor opção.

Estrutura de dados

- Listas, matrizes

- Árvore de busca (exemplo: B e AVL)

Acesso

- Sequencial, busca binária, etc

- *Hashing:* opção mais eficiente

Questões de projeto:

* Tamanho da tabela:
  + não é possível saber quantas linhas terão – tipicamente, de algumas centenas a mil “linhas”
  + dependentente da forma de implementação
* Uma única tabela X várias tabelas
  + Diferentes declarações tem diferentes informações e atributos
  + Interessante usar apenas uma quando a linguagem proíbe o mesmo identificador entre tipos diferentes de declarações
* Escopo
  + Representação: várias tabelas (local, global) ou uma única.
  + Tratamento