

Compiladores – Análise Semântica

Fabio Mascarenhas – 2017.2

<http://www.dcc.ufrj.br/~fabiom/comp>

Análise Semântica

- Muitos erros no programa não podem ser detectados sintaticamente, pois precisam de *contexto*
 - Quais variáveis estão em escopo, quais os seus tipos
- Por exemplo:
 - Todos os nomes usados foram declarados
 - Nomes não são declarados mais de uma vez
 - Tipos das operações são consistentes

Escopo

- Amarração dos *usos* de um nome com sua *declaração*
 - Onde nomes podem ser variáveis, funções, métodos, tipos...
- Passo de análise importante em diversas linguagens, mesmo linguagens “de script”
- O *escopo* de um identificador é o trecho do programa em que ele está visível
- Se os escopos não se sobrepoem, o mesmo nome pode ser usado para coisas diferentes

Declarações e escopo em TINY

- Vamos adicionar declarações de variáveis em TINY no início de cada bloco, usando a sintaxe:

```
CMDS -> CMDS ; CMD
      | VAR CMD
VAR   -> var IDS ;
      |
IDS   -> IDS , id
      | id
```

- O escopo de uma declaração é todo o bloco em que ela aparece, incluindo outros blocos dentro dele!
- Uma variável pode ser redeclarada em um bloco dentro de outro, nesse caso ela *oculta* a variável do bloco mais externo

Exemplo - escopo

- Qual o escopo de cada declaração de x no programa abaixo, e qual declaração corresponde a cada uso?

```
var x;  
read x;  
if x < 0 then  
  var x;  
  x := 5;  
end;  
write x;
```

The diagram illustrates the scope of the variable x in the provided code. Red circles highlight the following elements: the first x in `var x;`, the x in `read x;`, the x in `if x < 0`, the x in the inner `var x;`, the x in `x := 5;`, and the x in `write x;`. Red arrows indicate the flow of scope resolution: an arrow from the first `var x;` points to the `read x;` and the `if` condition; another arrow from the inner `var x;` points to the `x := 5;` assignment. A large red bracket on the left side of the code block encompasses the entire program, indicating the global scope.

Analizando escopo

- Fazemos a análise do escopo usando uma *tabela de símbolos*
- Uma tabela de símbolos mapeia um *nome* a algum *atributo* desse nome (seu tipo, onde ele está armazenado em tempo de execução, etc.)
- Cada escopo tem sua própria tabela, derivada da tabela do escopo onde está incluso
- Existem duas operações básicas: *inserir* e *procurar*, usadas na declaração e no uso de um nome
- Essas operações implementam as regras de escopo da linguagem

Tabelas de Símbolos - repeat

```
var x;  
read x;  
if x < 0 then  
    var x;  
    x := 5  
end;  
repeat  
    var y;  
    y := x;  
    write y;  
    x := x - 1  
until y = 0  
write x
```