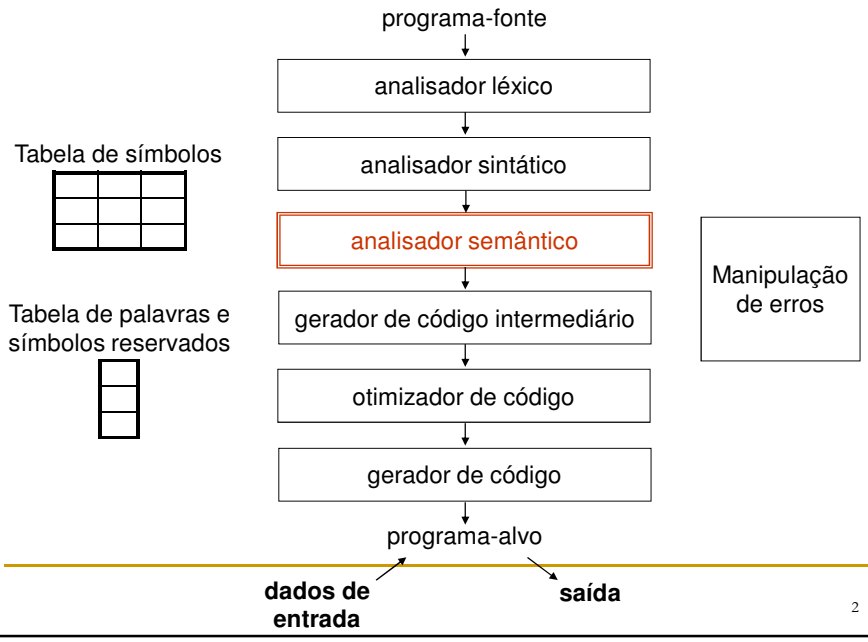
Análise semântica (continuação)

Função, interação com o compilador
Tabela de símbolos
Análise semântica

Prof. Thiago A. S. Pardo
taspardo@icmc.usp.br

1

Estrutura geral de um compilador



2

Gramática de atributos

- Gramática de atributos
  - Método usualmente utilizado para formalização semântica
  - Conjunto de atributos e regras semânticas para uma gramática
    - Cada regra sintática/gramatical pode ter regras semânticas associadas
  - Atributos associados aos símbolos gramaticais
    - Por exemplo, valor e escopo
      - x.valor, x.escopo
  - Regras semânticas que manipulam os atributos
    - Por exemplo, regra para somar os atributos valores de duas variáveis
      - x:=a+b, cuja regra é x.valor:=a.valor+b.valor

3

Cômputo de atributos

- Com base na árvore sintática explícita
  - Grafos de dependência
    - Compilador de mais de uma passagem
- *Ad hoc*
  - Análise semântica "comandada" pela análise sintática
    - Compilador de uma única passagem

4

Cômputo de atributos

- Estruturas de dados externas
  - Em vez de se armazenar os atributos na árvore sintática ou de manipulá-los via parâmetros e valores de retornos, os atributos podem ser armazenados em estruturas separadas
    - Variáveis globais
    - Listas
    - Tabelas
  - Em compilação, a **tabela de símbolos** é utilizada, em geral

5

Tabela de símbolos

- Estrutura principal da compilação
- Captura a sensitividade ao contexto e as ações executadas no decorrer do programa
- Pode estar atrelada a todas as etapas da compilação
- Permite a realização da análise semântica
- Fundamental na geração de código

6

Tabela de símbolos

- Permite saber durante a compilação de um programa o tipo e o valor de seus elementos (números e identificadores), escopo destes, número e tipo dos parâmetros de um procedimento, etc.
  - Cada token tem atributos/informações diferentes associadas

Cadeia	Token	Categoria	Tipo	Valor	...
i	id	var	integer	1	...
fat	id	proc	-	-	...
2	num	-	integer	2	...
...					

7

Tabela de símbolos

- Exemplo de atributos de identificador de variável
  - Tipo de variável (inteira, real, etc.), nome da variável, endereço na memória, escopo (programa principal, função, etc.), etc.
- Para vetor, ainda seriam necessários atributos de tamanho do vetor, o valor de seus limites, etc.

8

Tabela de símbolos

- Principais operações efetuadas
  - Inserir: armazena na tabela informações fornecidas pelas declarações no programa
  - Busca: recupera da tabela informações de um elemento declarado no programa quando esse elemento é utilizado
  - Remover: remove (ou torna inacessível) da tabela informações sobre um elemento declarado que não se mostra mais necessário no programa
- As especificidades dessas operações são dependentes da linguagem de programação em questão

9

Tabela de símbolos

- A tabela é acessada pelo compilador sempre que um elemento é mencionado no programa
  - Verificar ou incluir sua declaração
  - Verificar seu tipo, escopo ou alguma outra informação
  - Atualizar alguma informação associada ao identificador (por exemplo, valor)
  - Remover um elemento quando este não se faz mais necessário ao programa

10

Tabela de símbolos

- Estrutura da tabela de símbolos: determinada pela eficiência das operações de inserir, verificar e remover

- Várias possibilidades
  - Implementação
    - Estática
    - Dinâmica: melhor opção
  - Estrutura
    - Listas, matrizes
    - Árvores de busca (por exemplo, B e AVL)
  - Acesso
    - Seqüencial, busca binária, etc.
    - *Hashing*: opção mais eficiente
      - O elemento do programa é a chave e a função *hash* indica sua posição na tabela de símbolos
      - Necessidade de tratamento de colisões

Tabela de símbolos

- Exemplo de *hashing* com resolução de colisões para a inclusão dos identificadores i, j, tamanho e temp

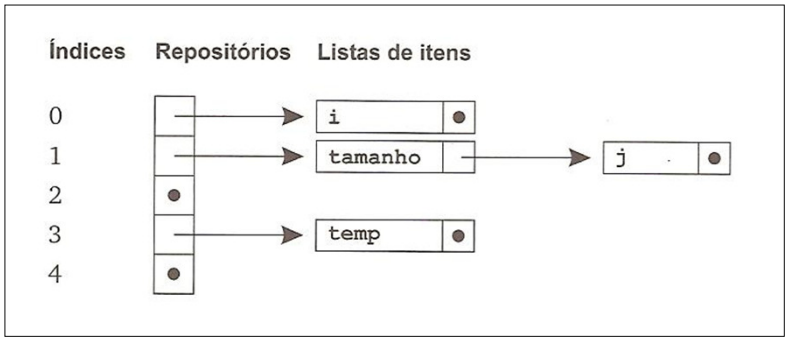


Tabela de símbolos

- Questões de projeto
  - Tamanho da tabela: tipicamente, de algumas centenas a mil “linhas”
    - Dependente da forma de implementação
      - Na implementação dinâmica, não é necessário se preocupar tanto com isso
  - Uma única tabela para todas as declarações ou várias tabelas, sendo uma para cada tipo de declaração (constantes, variáveis, tipos, procedimentos e funções)
    - Diferentes declarações têm diferentes informações/atributos (por exemplo, variáveis não têm número de argumentos, enquanto procedimentos têm)

Tabela de símbolos

- Representação de escopo de identificadores do programa: várias tabelas ou uma única tabela com a identificação do escopo (como um atributo ou por meio de listas ligadas, por exemplo) para cada identificador
  - Tratamento de escopo
    - Inserção de identificadores de mesmo nome, mas em níveis diferentes
    - Remoção de identificadores cujos escopos deixaram de existir
  - Em geral, na maioria das linguagens de programação, aplica-se a “regra do aninhamento mais próximo”

Tabela de símbolos

- Possibilidades para tratamento de escopos
  - Inclusão de um campo a mais na tabela de símbolos indicando o nível da variável no programa
    - Controle do nível durante a compilação do programa
      - Quando se chama um procedimento (ou função), faz-se nível:=nível+1
      - Quando se sai de um procedimento (ou função), faz-se nível:=nível-1
  - Associação das variáveis locais a um procedimento (ou função) à entrada da tabela para o procedimento (ou função) por meio, por exemplo, de uma lista encadeada
    - Atenção: para a checagem de tipos, deve-se saber quantos são e quais são os parâmetros de um procedimento (ou função) na tabela de símbolos
  - Tabelas diferentes para diferentes escopos

15

Tabela de símbolos

- Tratamento de escopo
  - Como diferenciar variáveis globais de locais
    - Tratamento de variáveis de mesmo nome, mas de escopos diferentes

```
program meu_prog
procedure meu_proc(x: integer)
var y: real
begin
    read(y);
    x:=x+y
end;
var x, y: integer
begin
    read(y);
    x:=x*y
end.
```

16

Exercício

- Gere a tabela de símbolos para o programa abaixo

```
program meu_prog
procedure meu_proc(x: integer)
var y: real
begin
    read(y);
    x:=x+y
end;
var x, y: integer
begin
    read(y);
    x:=x*y
end.
```

17

Tabela de símbolos

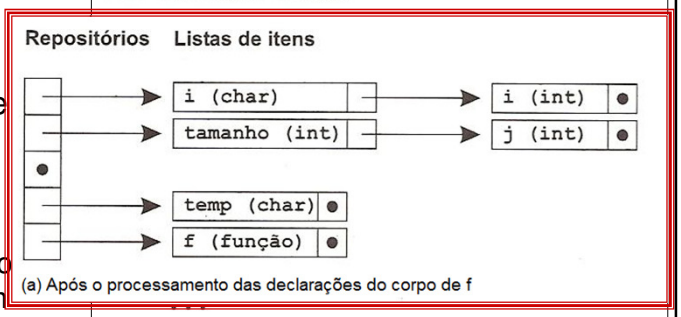
- Exemplo de tratamento de escopo
  - Sub-rotinas aninhadas
  - Variáveis globais e locais com mesmo nome

```
program Ex;
var i,j: integer;

function f(tamanho: integer): integer;
var i,temp: char;

   procedure g;
   var j: real;
   begin
      ...
   end;

   procedure h;
   var j: ^char;
   begin
      ...
   end;

begin (* f *)
   ...
end;

begin (* programa principal *)
   ...
end.
```

Tabela de símbolos

- Exemplo de tratamento de escopo
  - Sub-rotinas aninhadas
  - Variáveis glo e locais com mesmo nome

```
program Ex;
var i,j: integer;

function f(tamanho: integer): integer;
var i,temp: char;
```



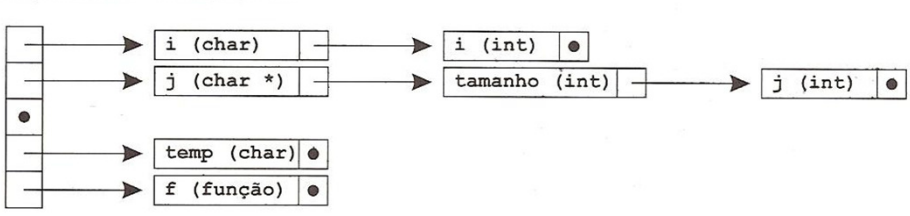
(a) Após o processamento das declarações do corpo de f

```
  end;

begin (* f *)
  ...
end;

begin (* programa principal *)
  ...
end.
```

Tabela de símbolos

- Exemplo de tratamento de escopo
  - Sub-rotinas aninhadas
  - Variáveis globais

```
program Ex;
var i,j: integer;

function f(tamanho: integer): integer;
var i,temp: char;

  procedure g;
  var j: real;
  begin
    ...
  end;

  procedure h;
  var j: ^char;
```

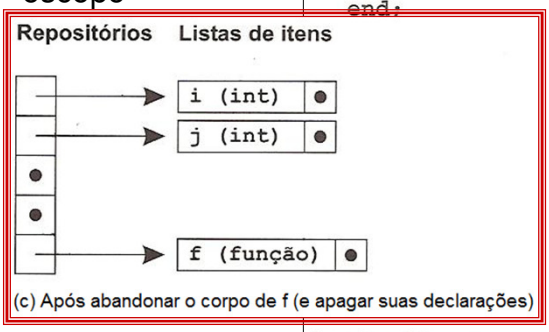


(b) Após o processamento da declaração da segunda declaração composta aninhada dentro do corpo de f

Tabela de símbolos

- Exemplo de tratamento de escopo

```
program Ex;
var i,j: integer;

function f(tamanho: integer): integer;
var i,temp: char;

   procedure g;
   var j: real;
   begin
      ...
   end;
```




```
begin (* programa principal *)
   ...
end.
```

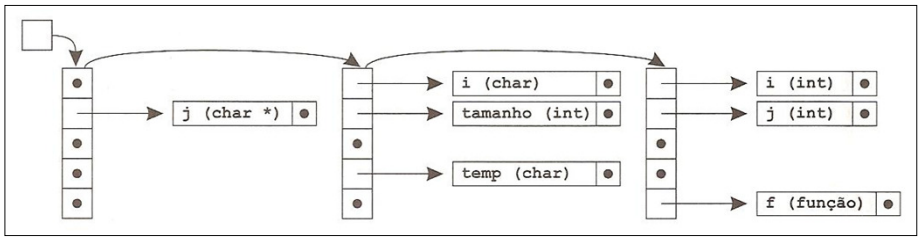
Tabela de símbolos

- Comportamento de **pilha**
  - Insere as declarações mais recentes, ocultando as antigas
  - Remove as mais recentes, voltando ao escopo anterior
  - Acesso às mais recentes

22

Tabela de símbolos

- Alternativa para tabela anterior: tabelas separadas para cada escopo
  - Mudar o escopo requer apenas a mudança do ponteiro



23

Tabela de símbolos

- A tabela de símbolos pode ser utilizada para armazenar as palavras reservadas e símbolos especiais da linguagem, podendo dispensar o uso da tabela de palavras e símbolos reservados

24

Tabela de símbolos

- Descritores
  - Registros (campos) que formam a tabela de símbolos
  - Armazenam as informações dos números e identificadores
- Diferentes identificadores têm diferentes descritores
  - Tem que se levar isso em consideração no projeto da tabela de símbolos para sua otimização
  - Possibilidade de se usar uniões para o caso de uma mesma tabela para tudo

25

Tabela de símbolos

- Inserção de elementos na tabela
  - Associação de regras semânticas às regras gramaticais
  - Verificar se o elemento já não consta na tabela
- Busca de informação na tabela
  - Realizada antes da inserção
  - Busca de informações para análise semântica
- Remoção de elementos da tabela
  - Tornar inacessíveis dados que não são mais necessários (por exemplo, após o escopo ter terminado)
  - Linguagens que permitem estruturação em blocos

26

Tabela de símbolos

- As sub-rotinas de inserção, busca e remoção podem ser inseridas diretamente na gramática de atributos
  - Explicitamente, via chamadas de sub-rotinas de manipulação da tabela
    - Compilação em mais de uma passagem, se árvore sintática é a base para a análise
    - Opcionalmente, compilação de uma única passagem, sendo a gramática de atributos utilizada apenas para a descrição da semântica

27

Tabela de símbolos

- Exemplo: decl → tipo var-lista
  tipo → **int** | **float**
  var-lista → **id,** var-lista | **id**

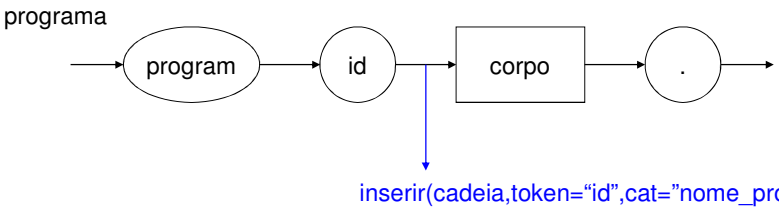
Regras gramaticais	Regras semânticas
decl → tipo var-lista	var-lista.tipo_dado = tipo.tipo_dado
tipo → int	tipo.tipo_dado = integer
tipo → float	tipo.tipo_dado = real
var-lista$_1$ → id, var-lista$_2$	id.tipo_dado = var-lista$_1$.tipo_dado var-lista$_2$.tipo_dado = var-lista$_1$.tipo_dado If busca(id)=FALSE then inserir(id,id.tipo_dado) else ERRO("identificador já declarado")
var-lista → id	id.tipo_dado=var-lista.tipo_dado If busca(id)=FALSE then inserir(id,id.tipo_dado) else ERRO("identificador já declarado")

Tabela de símbolos

- As sub-rotinas de inserção, busca e remoção podem ser inseridas diretamente na análise sintática
  - Solução *ad hoc*
    - Compilação de uma única passagem

Tabela de símbolos

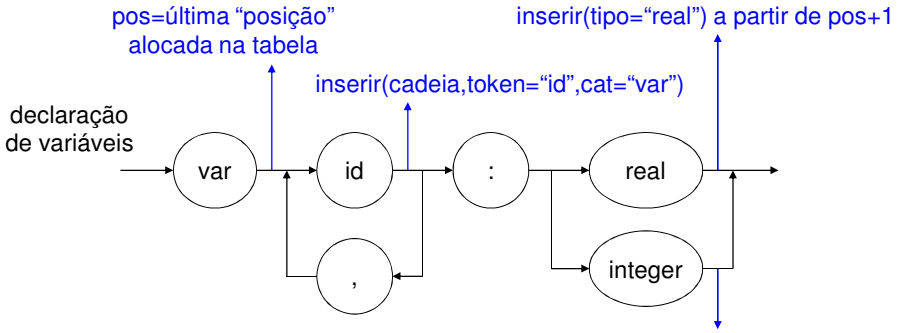
- Inserção de elementos na tabela
  - Declaração, principalmente



program meu_prog ...

Cadeia	Token	Categoria	Tipo	Valor	...
meu_prog	id	nome_prog	-	-	...

Tabela de símbolos

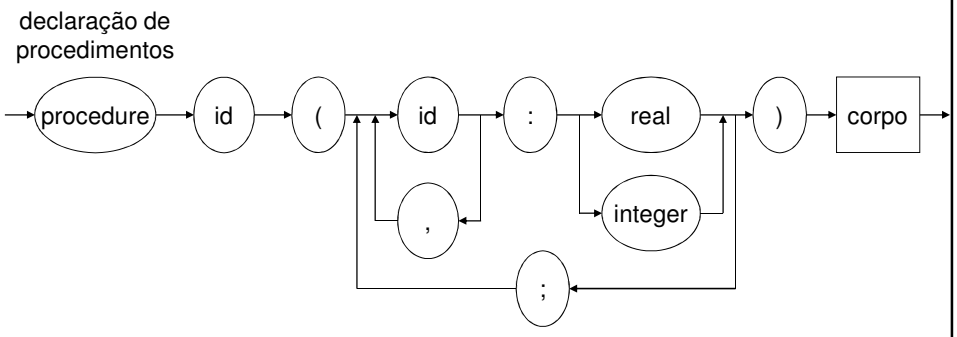


var x, y: integer

Cadeia	Token	Categoria	Tipo	Valor	...
meu_prog	id	nome_prog	-	-	...
x	id	var	integer		...
y	id	var	integer		...

Tabela de símbolos

- **Exercício**: inclua as funções adequadas



32

Tabela de símbolos

- Exercício: inclua as funções adequadas

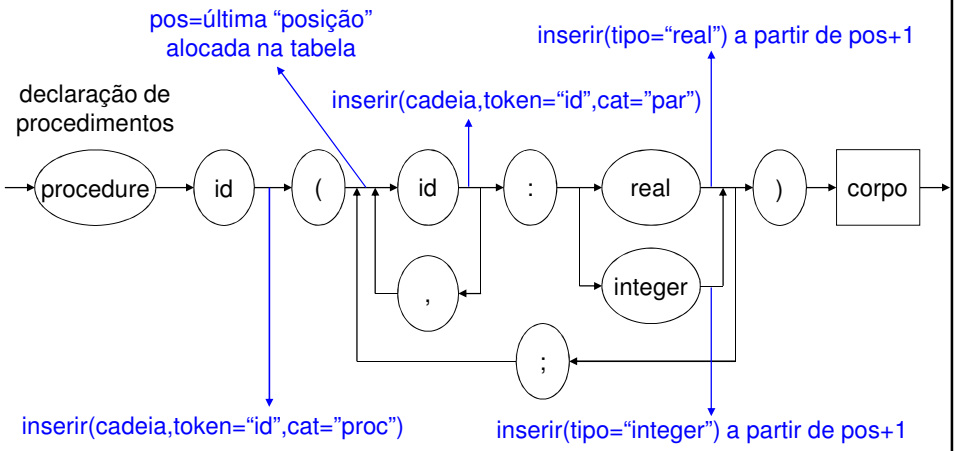


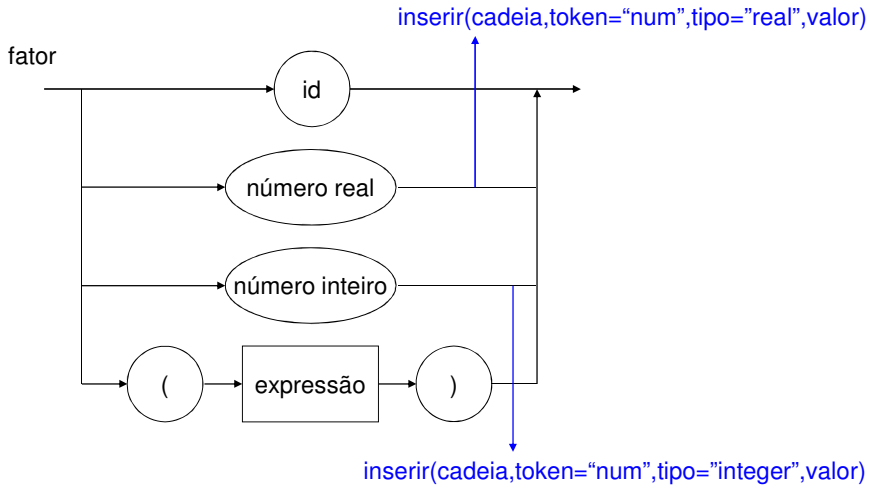
Tabela de símbolos

- Exercício: inclua as funções adequadas

procedure meu_proc(a: integer; b,c: real) ...

Cadeia	Token	Categoria	Tipo	Valor	...
meu_prog	id	nome_prog	-	-	...
x	id	var	integer		...
y	id	var	integer		...
meu_proc	id	proc	-	-	...
a	id	par	integer		...
b	id	par	real		...
c	id	par	real		...

Tabela de símbolos



35

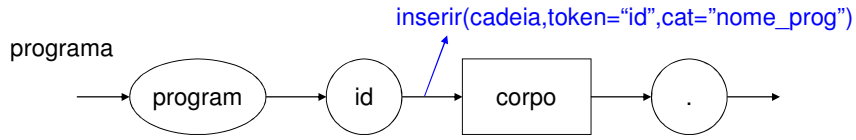
Tabela de símbolos

i:=1

Cadeia	Token	Categoria	Tipo	Valor	...
meu_prog	id	nome_prog	-	-	...
x	id	var	integer		...
y	id	var	integer		...
meu_proc	id	proc	-	-	...
a	id	par	integer		...
b	id	par	real		...
c	id	par	real		...
1	num	-	integer	1	...

36

Exemplo de procedimento




```
procedimento programa(Seg)
Inicio
        se (simbolo=program) então obtem_simbolo(cadeia,simbolo)
        senão ERRO(Seg+{id});
        se (simbolo=id) então
                inserir(cadeia,"id","nome_prog")
                obtem_simbolo(cadeia,simbolo)
        senão ERRO(Seg+P(corpo));
        corpo(Seg+{.});
        se (simbolo=simb_ponto) então obtem_simbolo(cadeia,simbolo)
        senão ERRO(Seg);
fim
```

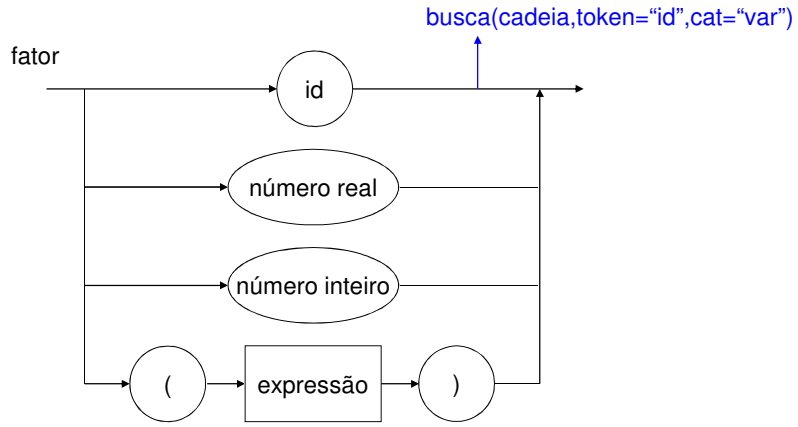
37

Tabela de símbolos

- Busca de informação
  - Sempre que um elemento do programa é utilizado
    - fator e comando
  - Verifica-se se foi declarado, seu tipo, etc.

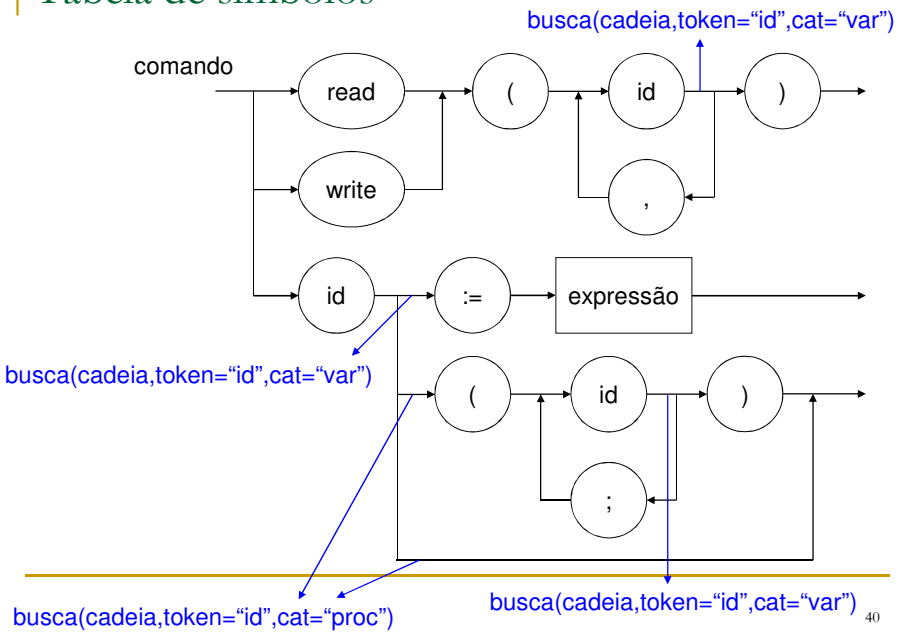
38

Tabela de símbolos



39

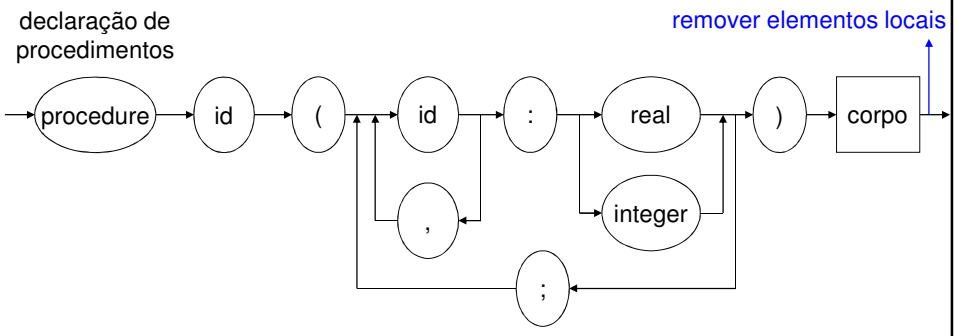
Tabela de símbolos



40

Tabela de símbolos

- Remoção de elementos da tabela
  - Variáveis locais dos procedimentos
    - Atenção: parâmetros precisam ser mantidos



41