INSTITUTO DE CIÊNCIAS MATEMÁTICAS DE SÃO CARLOS - USP
Departamento de Ciências da Computação e Estatística
SCC 605 Teoria da Computação e Compiladores **1°Sem/2010**
Prof.: Sandra Maria Aluísio, e-mail: sandra@icmc.usp.br

Rotinas Semânticas para o Pascal Simplificado & algumas extensões

Variáveis Globais: nivelcorr, sinal, npar, K1, K2, pos, classet, End_tipo, c_id, p_id, permissão_tipo_array (inibe o encaixamento de construtores do tipo array como em: array [1..10] of array [1..20]...).
OBS: Devem **decidir** se as informações de identificadores duplicados serão inseridas ou não. Neste documento decidimos inserí-las (vejam as rotinas 0, 1, 3, 5, 6, 18).

Procedure rs(i: integer);
Begin
  Case i of

0: Se declarado(TS,s,nivelcorr) então erro('id já declarado');
insere(TS,s,ref);

0': Atualiza os campos nível, categoria, valor e tipo_c através da rotina seta_atributos com: nivelcorr, **constante**, valor obtido da rs(0'') x sinal ou rs(0''') x sinal e tipo_c obtido da rs(0'') ou rs(0''').

0'': busca(TS, s, ref, achou);
Se não achou ou (achou e categoria <> constante)
Então erro('constante não definida')
Senão obtém tipo_c e valor através da rotina obtem_atributos

0''': tipo_c = inteiro; valor = valor obtido da conversão de s para int

1: Se declarado(TS,s,nivelcorr) então erro('id já declarado');
insere(TS, s, ref); permissão_tipo_array := true;

2: Atualiza os campos nível, categoria, nbytes e Pont_tipo_elementar através da rotina seta_atributos com: nívelcorr, **tipo**, nbytes e tipo elementar com info da rs(9) ou com info do endereço do array quando tipo for array

3: Se declarado(TS,s,nivelcorr) então erro('id já declarado');
insere(TS,s,ref); permissão_tipo_array := true;

4: Atualiza os campos nível, categoria, tipo_v das variáveis na TS através da rotina seta_atributos com: nivelcorr, **variável**, o endereço fornecido pela rs(9);

5: Se declarado(TS,s,nivelcorr) então erro('id já declarado');
insere(TS,s,ref); Atualiza os campo nível e categoria através da rotina seta_atributos com: nivelcorr, **procedimento**;

npar := 0; nivelcorr:= nivelcorr + 1;

6: Se declarado(TS,s,nivelcorr) então erro('id já declarado');
insere(TS,s,ref); Atualiza os campos nível e categoria através da rotina seta_atributos com: nivelcorr, **função**;
npar := 0; nivelcorr: nivelcorr + 1;

7: busca(TS,s,ref,achou);
Se não achou ou (achou e a categoria <> tipo) então erro('tipo não definido')
Senão atualiza tipo_f;

8: elimina(TS,nivelcorr); {Não elimina as informações de tipo e passagem de parâmetros dos parâmetros de procedimentos/funções. Apagar os identificadores de parâmetros **ou** encadear a lista de parâmetros (tipo/passagem) na entrada de procedimentos para posterior checagem de tipos e passagem}
nivelcorr := nivelcorr –1;

9: busca(TS,s,ref,achou);
Se não achou ou (achou e categoria <> tipo) então erro('tipo não definido')
Senão Se permissão_tipo_array então End_tipo:= ref
Senão Atualiza os campos nbytes e Pont_tipo_elementar da entrada do array

10: Se permissão_tipo_array então
insere na TS as informações sobre o tipo de dados array;
permissão_tipo_array := false;
End_tipo := Ponteiro do descritor do array
Senão erro('array de várias dimensões não é permitido')

11: K1 := conversão de s para int;
Se conversão com sucesso então atualiza a dimensão inf do array {marcada por End_tipo}
Senão erro('overflow')

12: K2 := conversão de s para int;
Se conversão com sucesso então atualiza a dimensão sup do array e o campo nbytes,
marcados por End_tipo, com (K2 – K1 + 1)
Senão erro('overflow')
Se K2 < K1 então erro('Índice superior menor que Índice inferior')

13: permissão_tipo_array := true; {pode-se definir tipo array novamente}
pos := ref; { guarda a entrada de procedimento ou função}

14: classet := valor;

15: classet := referencia;

16 e 17: não tem para o PS, pois ele não permite procedimentos ou funções como parâmetros

18: Se declarado(TS, s, nivelcorr) então erro('id já declarado');
insere(TS,s,ref); Atualiza os campos nível, categoria e classe_transferência através da rotina seta_atributos com: nivelcorr, **parâmetro**, classet;
npar:= npar + 1;

19: rs(9);
Atualiza o campo tipo_p com ref (endereço do tipo elementar)

20: Atualiza o campo npar1 ou npar2 (do procedimento ou função que está apontado por pos);
npar:= 0;

21: Busca(TS, s, ref, achou);
Se não achou então erro('id não declarado')
Senão obtém os atributos categoria e ref; c_id:= categoria e p_id := ref

{lado direito: 22 e 22'}
22: Se c_id <> função então erro ('função não definida') {com abre-parênteses}

22': Se (c_id <> função ou c_id <> parâmetro ou c_id <> variável ou c_id <> constante) então erro('função, variável, parâmetro ou constante não definidos') {sem abre-parênteses}

23: npar := npar + 1;

24: Se a categoria do objeto apontado por p_id = procedimento
    então Se o número de parâmetros (npar1) <> npar então erro('incompatibilidade no número de parâmetros)
    senão Se o número de parâmetros (npar2) <> npar então erro('incompatibilidade no número de parâmetros);
npar:= 0;

{lado esquerdo: 25 e 25'}
25: Se c_id <> procedimento então erro('procedimento não definido') {com abre-parênteses}

25': Se (c_id <> função ou c_id <> variável ou c_id <> parâmetro ou c_id <> procedimento) então erro('função, procedimento, variável, ou parâmetro não definidos') {sem abre-parênteses}

26: Transformar a cadeia de dígitos em um tipo numérico (inteiro ou real, quando for o caso). Imprimir erro de underflow ou overflow se for o caso.