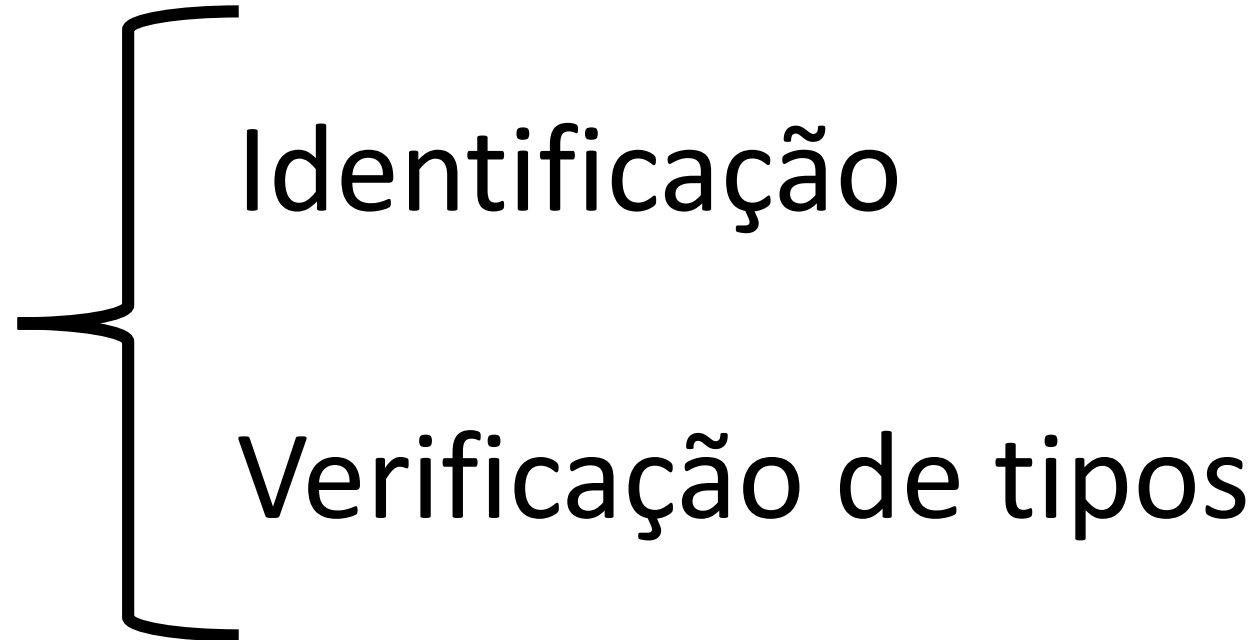


Análise de Contexto

terça-feira, 28 de abril de 2026

Análise de Contexto

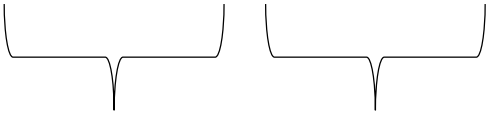


Identificação

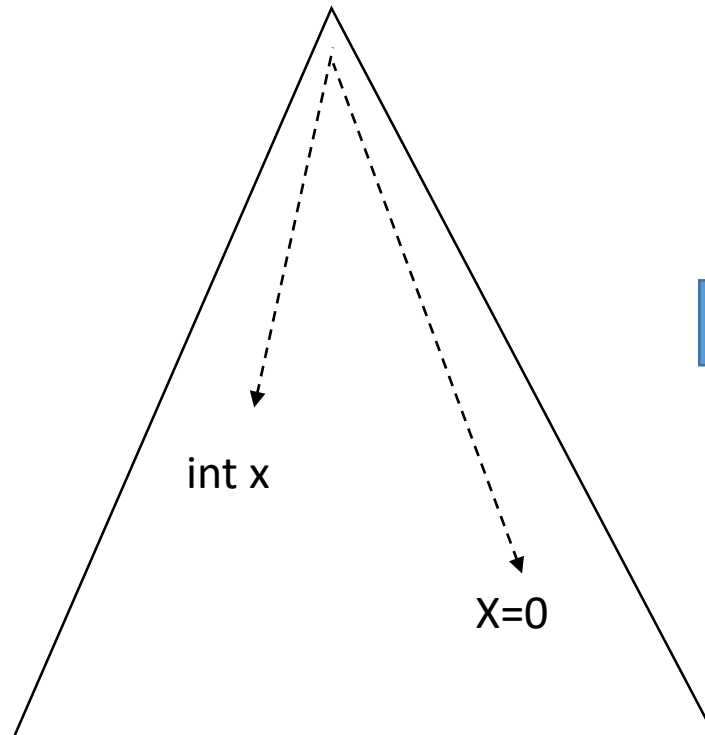
- Vincular cada uso de um nome com a respectiva declaração.
- Uso: comandos, expressões etc.
- É necessário conhecer as Regras de Escopo da linguagem.
- Escopo: região do programa em que um mesmo nome se refere a um mesmo objetivo (constante, variável, função, procedimento etc).
- Decora a AST.

Exemplo (linguagem C):

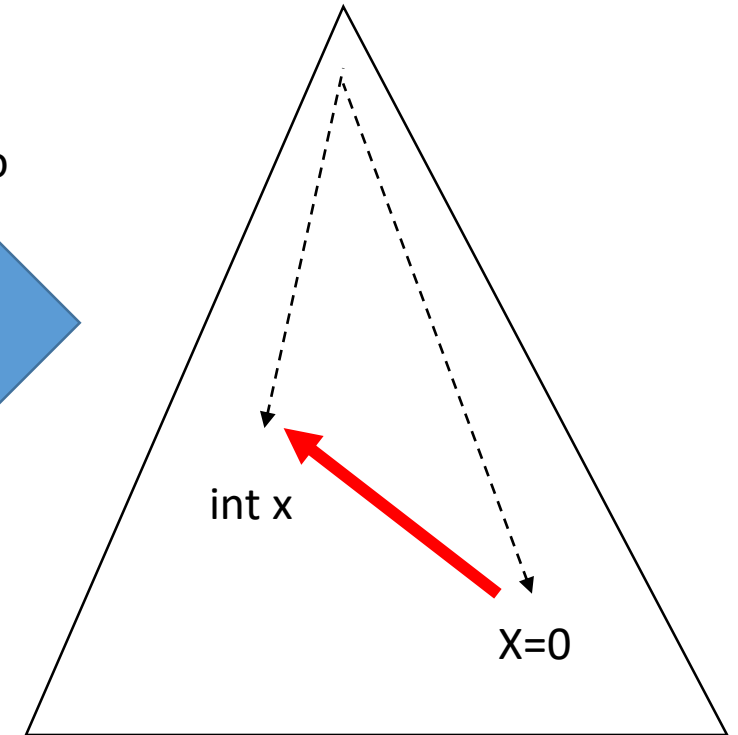
`int x; ... x=0; ...`



Declaração Uso



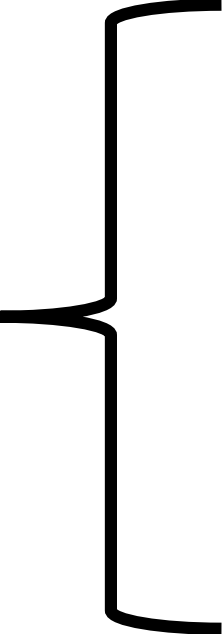
Identificação



Regras de Escopo

- Escopo Estático: quando o escopo dos nomes é fixo conhecido em tempo de compilação.
- Escopo Dinâmico: quando o escopo dos nomes depende do fluxo de execução do programa.
- Iremos considerar apenas linguagens com escopo estático.

Regras de Escopo



- Monolíticos
- Blocos planos
- Blocos aninhados

Serão consideradas somente linguagens em que o nome fica visível apenas APÓS a declaração do mesmo (e não ANTES).

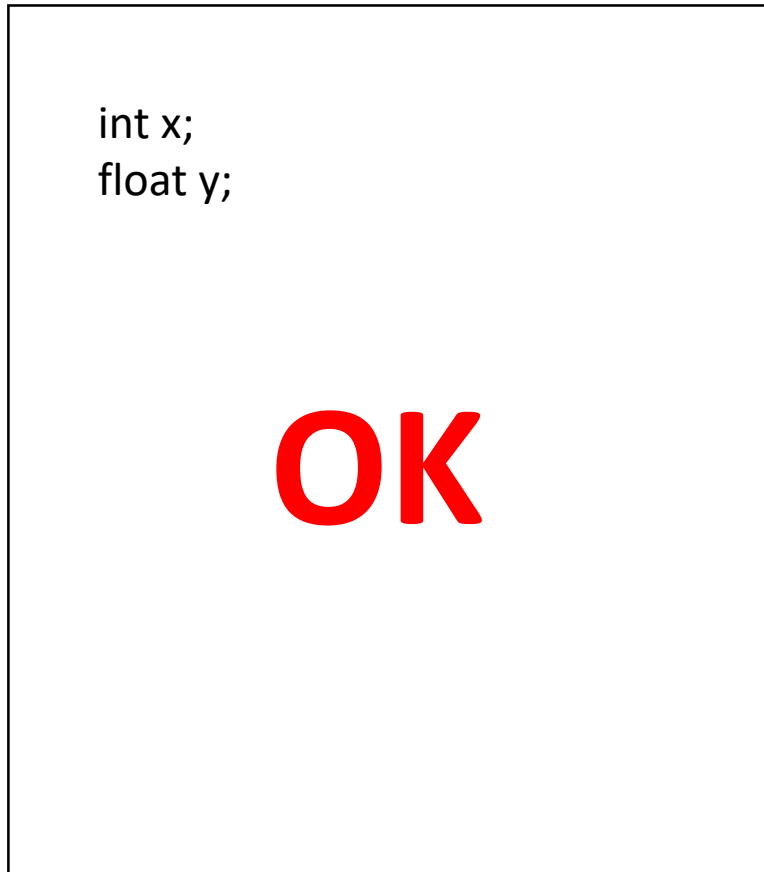
Tabela de Símbolos

- Também conhecida como Tabela de Identificação.
- Auxiliar no processo de identificação.
- Funciona como um ADT.
- Pode ser implementado como vetor, lista, árvore etc.
- Tempo crítico na execução do compilador.
- Operações básicas:
 - enter: para inserir ou novo nome na tabela e inserir o ponteiro para a declaração.
 - retrieve: para pesquisar um nome; retorna o ponteiro ou null.
- Outras operações dependem da estrutura de blocos da linguagem.
- Não insere nomes repetidos.
- Nomes não referenciados geram alertas (warnings).

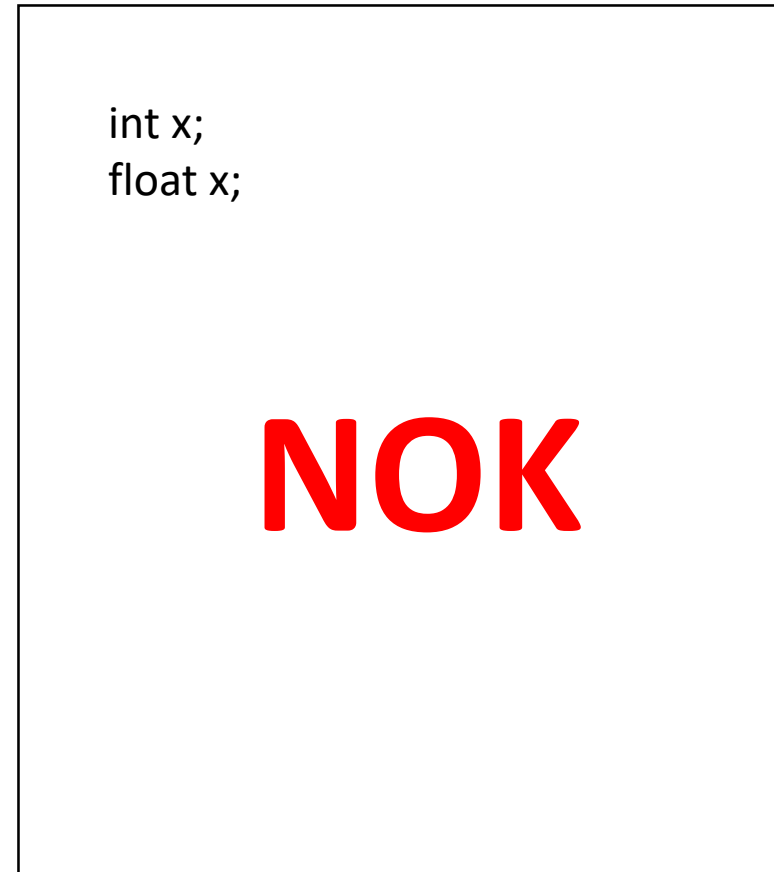
Monolíticos

- Um único bloco, correspondente ao programa principal.
- Linguagens antigas: Basic, Fortran, Cobol.
- Regras de Escopo:
 - Nomes podem ser escolhidos à vontade.
 - O mesmo nome não pode ser declarado mais de uma vez no mesmo bloco (mesmo que sejam objetos de naturezas diferentes).
- Identificação:
 - Se o nome que está sendo usado foi declarado no bloco corrente, ok.
 - Se não, erro.

Monolíticos



P



P

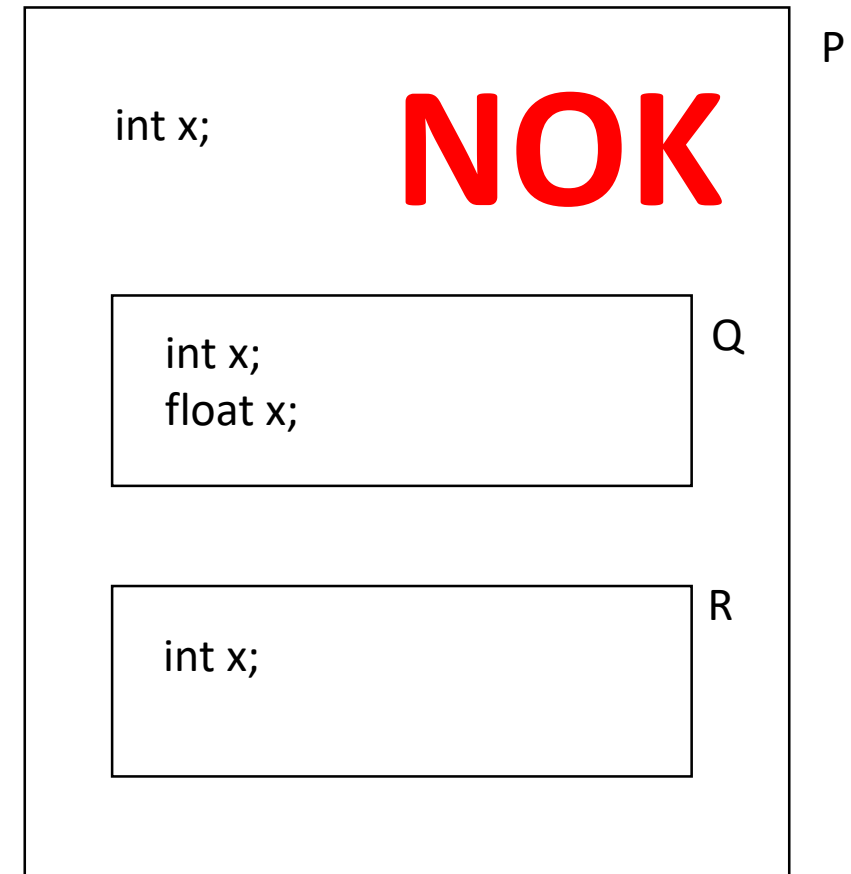
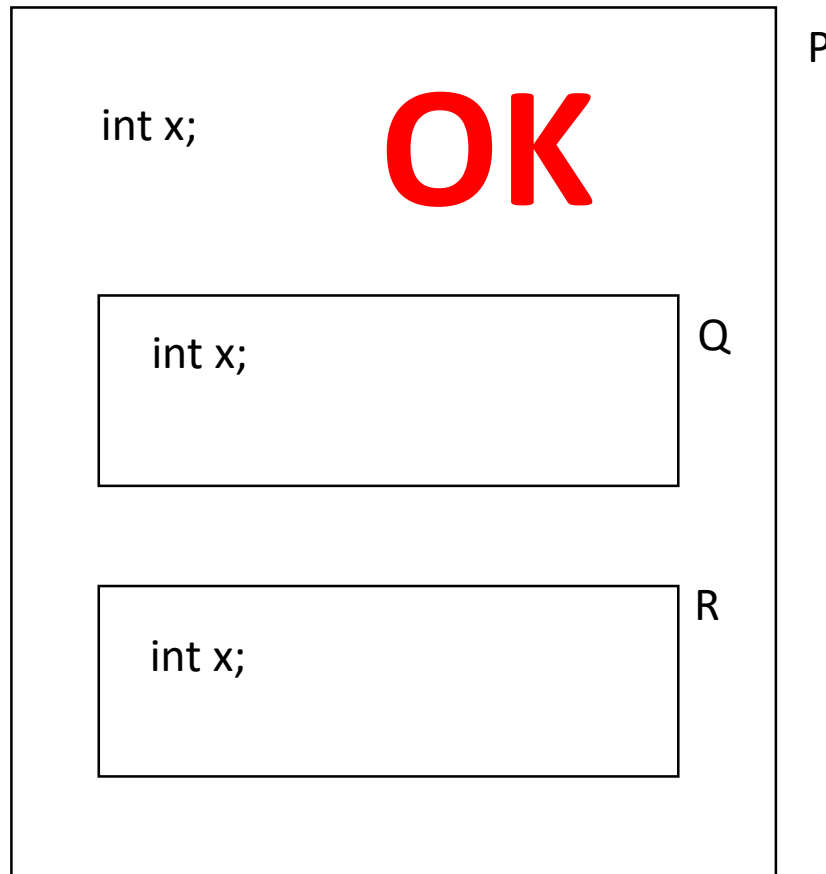
Monolíticos

- Escopo de um nome:
 - É o bloco todo.
 - $x_p = P$
 - $y_p = P$
- Identificação:
 - Basta percorrer a AST de cima para baixo, da esquerda para a direita.
 - Ao encontrar declaração, ENTER.
 - Ao encontrar uso, RETRIEVE.

Blocos planos

- Dois tipos de bloco: global e local.
- Um único bloco global, correspondente ao programa principal, com zero ou mais blocos internos (locais).
- Linguagens antigas: C (sem {}), Fortran, Cobol.
- Regras de Escopo:
 - Nomes podem ser escolhidos à vontade.
 - O mesmo nome não pode ser declarado mais de uma vez no mesmo bloco (mesmo que sejam objetos de naturezas diferentes).
 - Pode haver repetição desde que em blocos diferentes.
- Identificação:
 - Se o nome que está sendo usado foi declarado no bloco corrente (local), ok.
 - Se o nome foi declarado no bloco global, ok.
 - Se não, erro.

Blocos planos



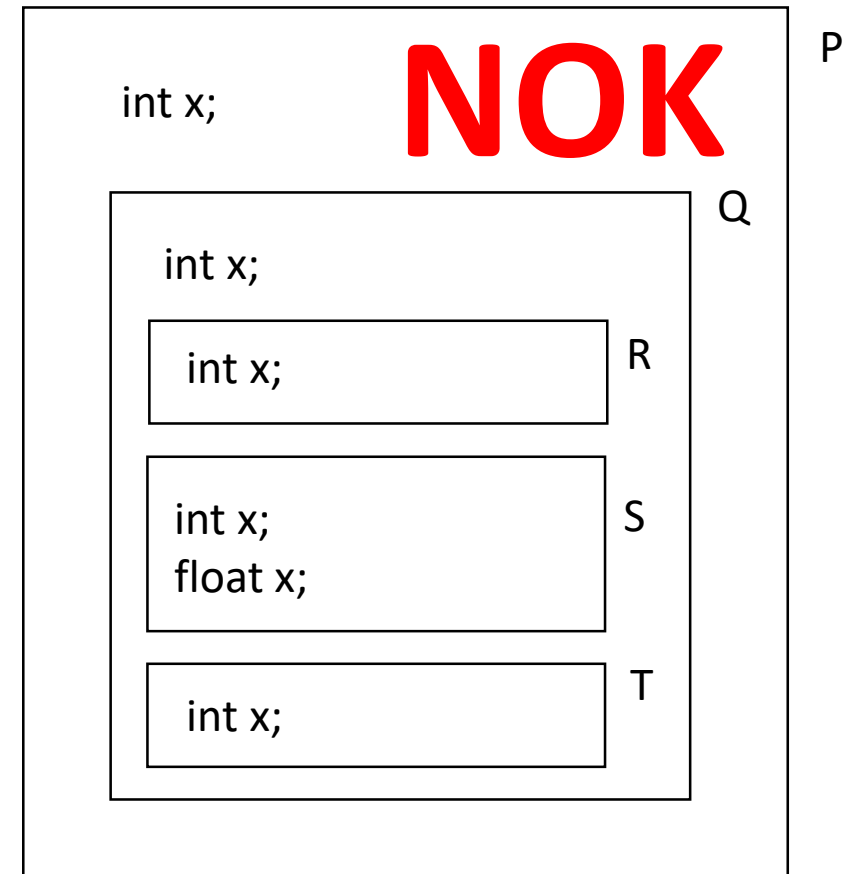
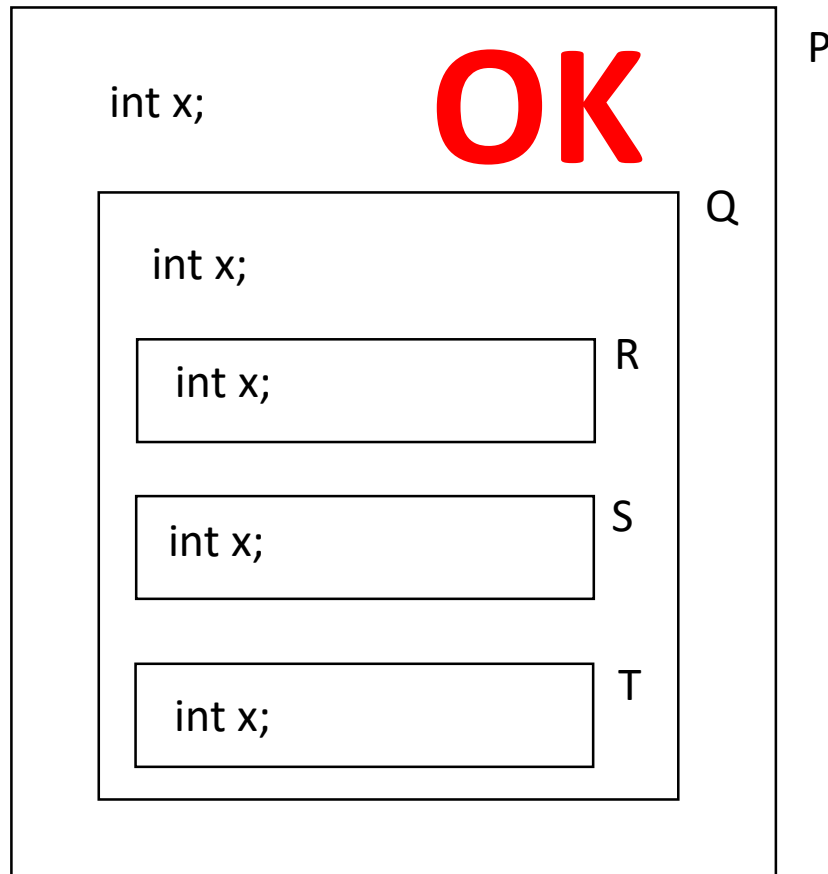
Blocos planos

- Escopo de um nome:
 - É o bloco onde o nome foi declarado (global ou local), EXCLUINDO os blocos internos (locais, se houverem) em que o nome foi redeclarado.
 - $x_p = P-Q-R$
 - $x_q = Q$
 - $x_r = R$
- Identificação:
 - Basta percorrer a AST de cima para baixo, da esquerda para a direita.
 - G para GLOBAL ou L para local são inseridos com os nomes.
 - Início de bloco, openScope muda de G para L
 - Ao encontrar declaração, ENTER.
 - Ao encontrar uso, RETRIEVE.
 - Ao final do bloco, close Scope muda de L para G.

Blocos aninhados

- Um bloco pode conter outro, em qualquer nível.
- Um único bloco global, correspondente ao programa principal, com zero ou mais blocos internos cada um destes com novos blocos ou não, e assim por diante.
- Linguagens: C (com {}), Pascal, Algol, Java etc.
- Regras de Escopo:
 - Nomes podem ser escolhidos à vontade.
 - O mesmo nome não pode ser declarado mais de uma vez no mesmo bloco (mesmo que sejam objetos de naturezas diferentes).
 - Pode haver repetição desde que em blocos diferentes.
- Identificação:
 - Se o nome que está sendo usado foi declarado no bloco corrente, ok.
 - Senão, se o nome foi declarado no bloco que envolve o bloco corrente, ok.
 - Segue assim até que não haja mais blocos.
 - Se não, erro.

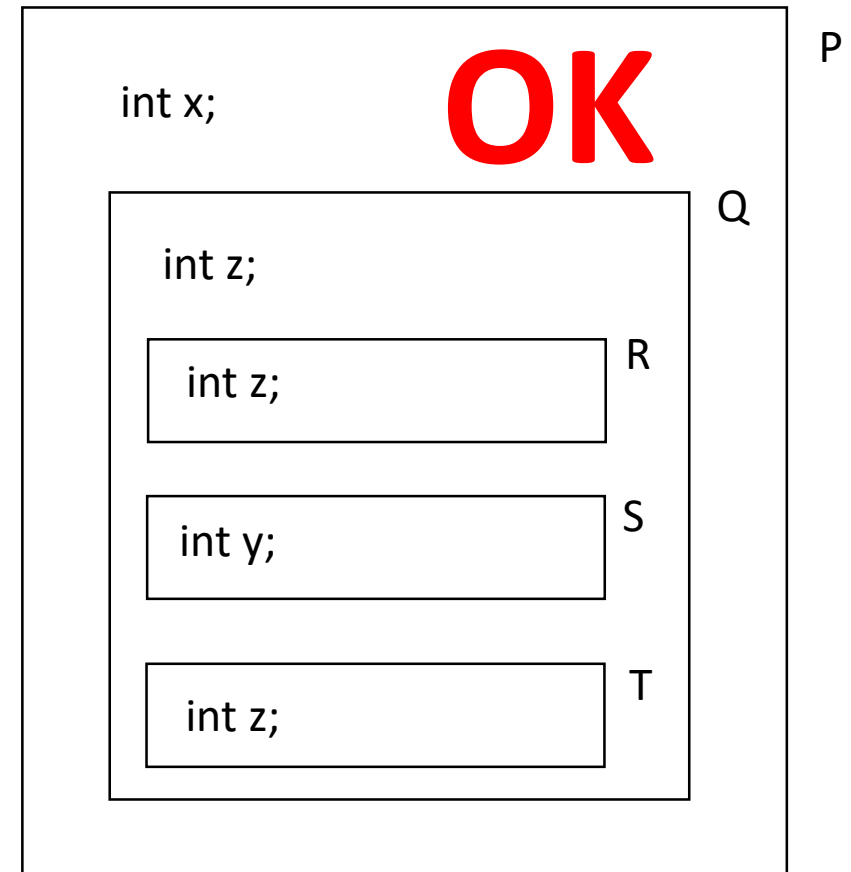
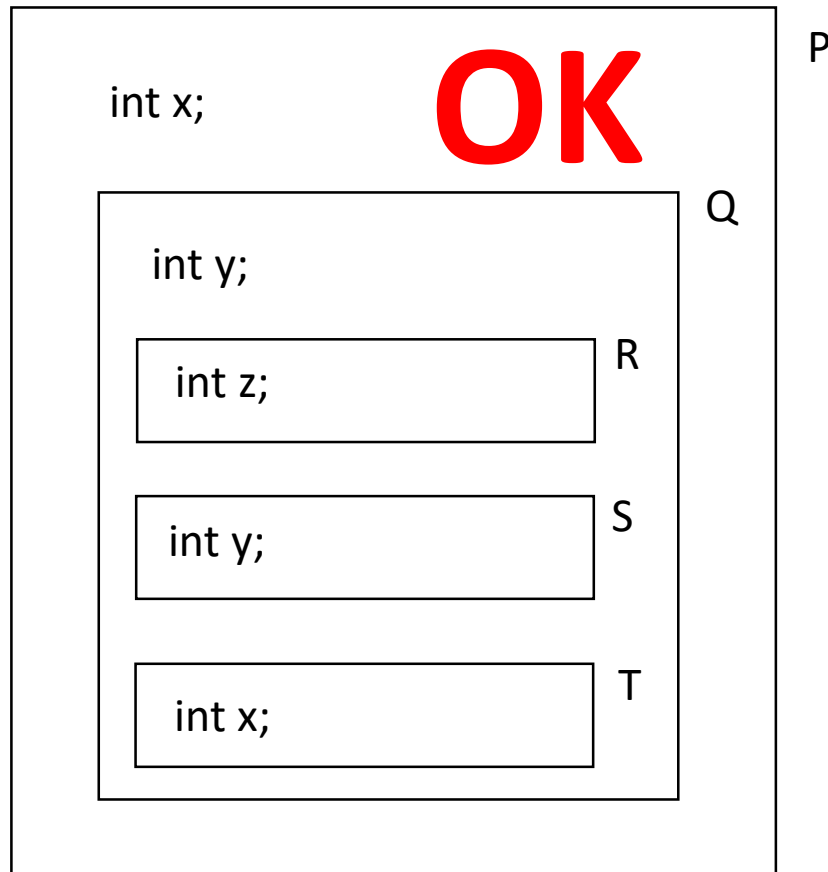
Blocos aninhados



Blocos aninhados

- Escopo de um nome:
 - É o bloco onde o nome foi declarado, EXCLUINDO os blocos internos (locais, se houverem) em que o nome foi redeclarado.
 - $x_P = P-Q$
 - $x_Q = Q-R-S-T$
 - $x_R = R$
 - $y_S = S$
 - $x_T = T$
- Identificação:
 - Basta percorrer a AST de cima para baixo, da esquerda para a direita.
 - Em vez de G/L usa-se n para indicar o nível do bloco (0 para global etc).
 - G para GLOBAL ou L para local são inseridos com os nomes.
 - Início de bloco, openScope muda de n para n+1
 - Ao encontrar declaração, ENTER.
 - Ao encontrar uso, RETRIEVE.
 - Ao final do bloco, closeScope muda de n para n-1.

Blocos aninhados



Blocos aninhados

- Esquerda:
 - $x_P = P - T$
 - $y_Q = Q - S$
 - $z_R = R$
 - $y_S = S$
 - $x_T = T$
- Direita:
 - $x_P = P$
 - $z_Q = Q - R - T$
 - $z_R = R$
 - $y_S = S$
 - $z_T = T$

Verificação de tipos

- Verificar que as operações estão sendo usadas de acordo com a sua assinatura.
- Assinatura de uma operação:
 - Nome (ou símbolo).
 - Precedência.
 - Associatividade.
 - Nº de operandos.
 - Tipo dos operandos.
 - Tipo do resultado.
 - Semântica.
- Usa as Regras de Tipos da linguagem.
- Iremos considerar apenas tipos estáticos (o tipo de um nome é conhecido em tempo de compilação).
- Tipos dinâmicos (aqueles que podem mudar durante a execução do programa) não serão considerados.