CEUB
EDUCAÇÃO SUPERIOR

CENTRO UNIVERSITÁRIO DE BRASÍLIA - CEUB
Faculdade de Tecnologia e Ciências Sociais -FATECS
Curso: Bacharel em Ciências da Computação
Disciplina: COMPILADORES
Professor: Leonardo Pol

PARTICIPANTES

1.-Aluno(a): Diogo Borges RA: 22250783

2.-Aluno(a): João Pedro Issmael Vieira RA: 22252263

1ª Produção de Atividade Intelectual – PAI-Teoria (2ºbimestre - 2ªAvaliação)

Fonte: Estudo dirigido texto Cap3 "A estrutura global do Compilador"

1.- O nome *compilador*, criado nos anos 50, faz referência ao processo de composição de um programa pela reunião de várias rotinas de biblioteca; o processo de tradução (de uma linguagem fonte para uma linguagem objeto), considerado hoje a função central de um compilador, era então conhecido como *programação automática*. Nesse processo de tradução, há duas tarefas básicas a serem executadas por um compilador explique quais são?

R.- As duas tarefas básicas a serem executadas por um compilador no processo de tradução são a análise e a síntese. A fase de análise tem como objetivo compreender o código-fonte escrito pelo programador, verificando se ele está correto de acordo com as regras da linguagem, já a síntese o compilador gera o código objeto ou executável a partir da representação interna do programa.

2. Quase todos os compiladores fazem hoje uso de uma técnica chamada *tradução dirigida pela sintaxe*. Explique em que consiste esta tradução**?**

R.- A tradução dirigida pela sintaxe é uma técnica em que o processo de tradução de um programa é guiado pela estrutura sintática da linguagem. Nessa abordagem, o compilador associa ações semânticas às regras gramaticais da linguagem, de modo que, à medida que a análise sintática reconhece as construções do programa, ele executa as ações correspondentes para gerar o código intermediário ou realizar verificações semânticas

3.- Normalmente associamos a *sintaxe à* idéia de forma, em oposição a *semântica*, associada a significado, conteúdo. Assim sendo, em princípio, o que deverão descrever a sintaxe e a semântica de uma linguagem de programação?

R.- A sintaxe de uma linguagem de programação descreve as regras formais para escrever comandos corretamente, ou seja, sua estrutura e organização. Já a semântica trata do significado desses comandos, ou seja, o que eles fazem quando executados, enquanto a sintaxe garante que o código possa ser lido e interpretado pelo compilador ou interpretador, a semântica garante que ele produza o comportamento esperado.

4.- Cabe à análise léxica a separação e identificação dos elementos componentes do programa fonte; à análise léxica cabe também a eliminação dos elementos "decorativos" do programa. Explique, quais são estes elementos decorativos**?**

R.-Os elementos decorativos são partes do código que não afetam sua execução, servindo apenas para torná-lo mais legível e organizado para o programador. Durante a análise léxica, esses elementos são eliminados pelo compilador, pois não possuem função sintática nem semântica. Entre eles estão os espaços em branco, tabulações, quebras de linha e comentários, que ajudam na compreensão do código, mas são ignorados no processo de tradução do programa fonte.

FOLHA DE RESPOSTAS NÚMERO 2 – COMPILADORES

NOME:

5.- A análise sintática deve reconhecer a estrutura global do programa. Explique em que consiste o reconhecimento da estrutura global do programa**?**

R.-A análise sintática é a fase da compilação que verifica se a estrutura global do seu código-fonte segue as regras gramaticais da linguagem de programação.

Não se preocupa com o significado do código, mas sim com a sua forma e organização. Agrupa os elementos individuais do código (tokens) em construções maiores como expressões, comandos (if, for), e declarações de funções. Cria uma representação hierárquica do programa, geralmente uma Árvore de Análise Sintática (AST), que serve de "esqueleto" para as próximas etapas da compilação.

6.- Fundamentalmente, a análise semântica trata os aspectos sensíveis ao contexto da sintaxe das linguagens de programação. Por exemplo, não é possível representar em uma gramática livre de contexto uma regra como *"Todo identificador deve ser declarado antes de ser usado."*, e a verificação de que essa regra foi aplicada cabe à análise semântica. Explique como acontece a verificação dessa regra**?**

R.- A verificação de que "toda variável deve ser declarada antes de ser usada" é feita pelo analisador semântico usando uma Tabela de Símbolos.

O processo funciona em dois passos:

Na Declaração (int x;): O compilador adiciona a variável x e suas informações (como o tipo) na Tabela de Símbolos, marcando-a como válida no escopo atual.

No Uso (x = 5;): O compilador consulta a Tabela de Símbolos para ver se x já foi registrada em um escopo válido.

Se encontra, o código está correto e a análise continua.

Se não encontra, o compilador gera um erro semântico, pois a variável está sendo usada sem ter sido declarada antes.

7.- Geração de Código e Otimização

Considere o exemplo, se uma máquina tem apenas um registrador (acumulador) em que as operações aritméticas são realizadas, e apenas uma instrução para realizar cada operação (uma instrução para soma, uma para produto, ...), existe pouca ou nenhuma possibilidade de variação no código que pode ser gerado. Considere o comando de atribuição: x := a + b * c

A primeira operação a ser realizada é o produto de b por c. Seu valor deve ser guardado numa posição temporária, que indicaremos aqui por t1. (Para sistematizar o processo, todos os resultados de operações aritméticas serão armazenados em posições temporárias.) Em seguida, devemos realizar a soma de a com t1, cujo valor será guardado numa posição temporária t2. (Naturalmente, neste caso particular, o valor poderia ser armazenado diretamente em x, mas no caso geral, a temporária é necessária.) Finalmente, o valor de t2 é armazenado em x.

Escreva qual é a representação temporária armazenada em X ?

R:Vamos seguir a lógica para a expressão x := a + b * c:

Prioridade da Operação: Primeiro, o compilador respeita a ordem de precedência matemática. A multiplicação (*) tem prioridade sobre a adição (+). Portanto, a primeira operação a ser feita é b * c.

O resultado dessa multiplicação é armazenado na primeira variável temporária: t1 := b * c

Segunda Operação: Agora, a expressão original foi simplificada para x := a + t1. A próxima operação é a soma.

O resultado da soma de a com t1 é armazenado na segunda variável temporária: t2 := a + t1

FOLHA DE RESPOSTAS NÚMERO 3 – COMPILADORES

NOME:

Atribuição Final: Neste ponto, t2 contém o resultado completo de toda a expressão a + b * c. O último passo é atribuir esse valor final à variável x.

O valor contido em t2 é copiado para x: x := t2