Lista de Exercícios #09

Geração de Código, Representações Intermediárias

1. Considerando a gramática abaixo, crie a árvore de derivação, a árvore sintática abstrata e o grafo acíclico direcionado para as entradas $x = (b * c) + (b * c)$; e $x = a * 2 + a * 2 * b$;.

   S → x = E;
   E → E_1 + T
   E → E_1 - T
   E → T
   T → T_1 * F
   T → T_1 / F
   T → F
   F → (E)
   F → **num**
   F → **name**

2. Transforme a gramática do exercício 1. em um esquema de tradução que cria uma árvore de derivação.

3. Repita o exercício 2., criando uma árvore sintática abstrata para qualquer entrada.

4. Repita o exercício 2., criando um grafo acíclico direcionado.

5. O esquema de tradução abaixo gera uma árvore sintática abstrata. Mostre seu funcionamento para uma série de entradas válidas.

   E → T `{ R.h = T.ptr; }` R `{ E.ptr = R.s; }`
   R → + T `{` R_1`.h = geraNo('+', R.h, T.ptr); }` R_1 `{ R.s =` R_1`.s; }`
   R → - T `{` R_1`.h = geraNo('-', R.h, T.ptr); }` R_1 `{ R.s =` R_1`.s; }`
   R → ϵ `{ R.s = R.h; }`
   T → (E) `{ T.ptr = E.ptr; }`
   T → id `{ T.ptr = geraFolha(id, id.nome); }`
   T → enum `{ T.ptr = geraFolha(num, num.val); }`

6. O que é um bloco básico? Por que alguns compiladores adotam um conceito de bloco básico diferente do tradicional? Discorra sobre as vantagens de cada abordagem e seu impacto nas etapas de otimização, por exemplo.

7. Construa o grafo de fluxo de controle para o código abaixo:

```
stmt0
while (i < 100) { stmt1 }
stmt2
if (x = y) { stmt3 } else { stmt4 }
stmt5
```

8. Construa o grafo de dependência de dados para o código abaixo:

```
x = 0
i = 1
while (i < 100)
    if (a[i] > 0)
        then x = x + a[i]
    i = i + 1
print x
```

9. Qual a vantagem e a desvantagem da IR linear de código de um endereço?

10. Transforme a gramática do exercício 1. em um esquema de tradução que gere código de um endereço.

11. Traduza uma série de entradas válidas utilizando o esquema de tradução do exercício 10..

12. Altere o esquema de tradução criado no exercício 4., gerando código de um endereço. Utilize o grafo acíclico direcionado para aproveitar valores eventualmente já calculados.

13. Refaça o exercício 12., gerando código de três endereços.

14. O que é TAC?

15. Crie um esquema de tradução para gerar uma IR pós-fixada a partir de uma IR pré-fixada de expressões aritméticas. Apresente uma série de entradas válidas mostrando o seu funcionamento com uma analisador ascendente e outro analisador descendente. Explique as diferenças, caso existam, nos esquemas de tradução para cada um dos analisadores.

16. Existem três abordagens para implementar TAC em memória, ou seja, manter o código TAC em memória. Qual delas você acha mais apropriada para o projeto de compiladores?