Trabalho De Compiladores: Resumo Completo Sobre As Três Fases De Análise De Um Compilador: Análise Léxica, Sintática E Semântica

Rian Beskow Friedrich

1 Análise Léxica:

1.1 O Que Ela Faz:

A análise léxica é a primeira etapa do processo de compilação, nela, se divide e converte uma sequencia de caracteres em uma sequência símbolos léxicos (tokens), preparando-o para a análise sintática.

1.2 O Que Ela Gera:

É gerado um fluxo de tokens, que é uma sequência de símbolos que representam as partes do código.

1.3 O Que É Esperado No Final:

No final, espera-se que não existam caracteres inválidos ou sequências ilegais de símbolos. Se algo não faz parte da linguagem, ele é detectado aqui, para assim, não ter nenhum problema sequencialmente na compilação.

1.4 Exemplo Explicativo:

Digamos que tenhamos um código-fonte como o do exemplo a seguir:

int x = 5 + 2;

A Análise Léxica a separa em tokens:

[int] [identificador:x] [=] [número:5] [+] [número:2] [;]

2 Análise Sintática:

2.1 O Que Ela Faz:

A análise sintática recebe os tokens da etapa anterior e verifica se eles estão na ordem correta segundo a gramática da linguagem**,** ou seja, se a estrutura é válida. Para isto, é montado uma árvore sintática**,** que representa a estrutura hierárquica do código.

2.2 O Que Ela Gera:

É gerado uma árvore sintática, que mostra como os tokens se organizam em expressões e comandos.

2.3 O Que é Esperado No Final:

No final, espera-se que o programa esteja estruturado corretamente, sem erros de como ausência de parênteses, ponto e vírgula e entre outros.

2.4 Exemplo Explicativo:

Suponha que em um programa tenha uma sequência de tokens recebidos, como o do exemplo anterior:

[int] [identificador:x] [=] [número:5] [+] [número:2] [;]

Esta sequência pode ser conferida com uma árvore sintática simplificada:

Atribuição

- **Tipo: int**
- **Variável: x**

Expressão

- **Número: 5**
- **Operador: +**
- **Número: 2**

3 Análise Semântica:

3.1 O Que Ela Faz:

A análise semântica verifica se o significado do código faz sentido, conferindo tipos de dados, declarações de variáveis, escopos, compatibilidade de operações e entre outros.

Por exemplo: não é permitido somar um número com uma string, ou usar uma variável que não foi declarada.

3.2 O Que Ela Gera:

É gerado uma árvore anotada com informações semânticas, como tipos de variáveis, contexto, escopo e vários outros.

3.3 O que é Esperado No Final:

No final, espera-se que o código faça sentido logicamente e semanticamente.

3.4 Exemplo Explicativo:

Vamos usar o código usado anteriormente:

int x = 5 + 2;

Com este código, é possível fazer uma árvore semântica anotada:

Declaração

Tipo: int

Identificador: x

[Tipo: int]

Atribuição

Operador: =

Expressão

Número: 5

[Tipo: int]

Operador: +

Número: 2

[Tipo: int]