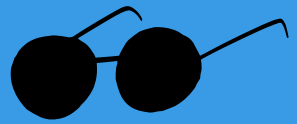
PARADIGMAS

sintaxe

DE PROGRAMAÇÃO

Derivação à Extrema Esquerda

```
<programa> → begin <lista_inst> end
<lista_inst> → <inst>
             | <inst> ; <lista_inst>
<inst> → <var> = <expressão>
<var> → A | B | C
<expressão> → <var> + <var>
            | <var> - <var>
            | <var>
```



```
<programa> => begin <lista_inst> end
           => begin <inst> ; <lista_inst> end
           => begin <var> = <expressão> ; <lista_inst> end
           => begin A = <expressão> ; <lista_inst> end
           => begin A = <var> + <var> ; <lista_inst> end
           => begin A = B + <var> ; <lista_inst> end
           => begin A = B + C ; <lista_inst> end
           => begin A = B + C ; <inst> end
           => begin A = B + C ; <var> := <expressão> end
           => begin A = B + C ; B = <expressão> end
           => begin A = B + C ; B = <var> end
           => begin A = B + C ; B = C end
```

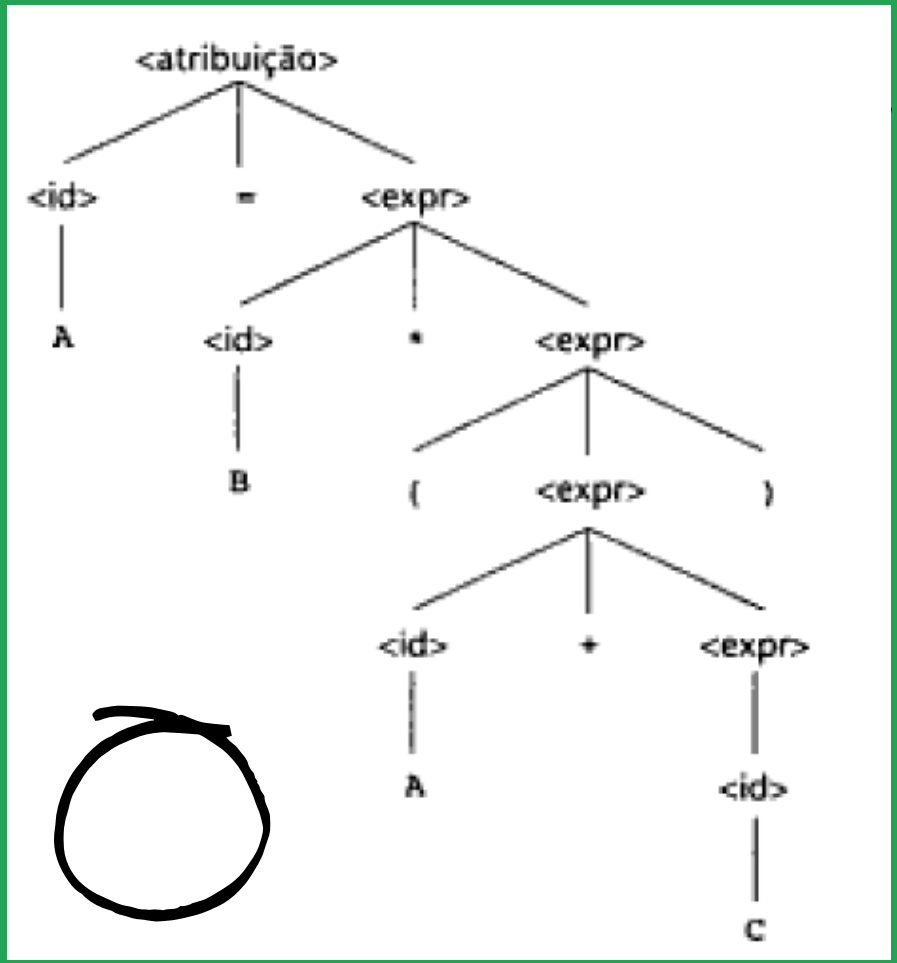
Atribuição Simples

```
<atribuição> → <id> = <expr>
<id> → A | B | C
<expr> → <id> + <expr>
       | <id> * <expr>
       | ( <expr> )
       | <id>
```

```
<atribuição> = > <id> = <expr>
             = > A = <expr>
             = > A = <id> * <expr>
             = > A = B * <expr>
             = > A = B * ( <expr> )
             = > A = B * ( <id> + <expr> )
             = > A = B * (A + <expr> )
             = > A = B * (A + <id> )
             = > A = B * ( A + C )
```

Árvores de Análise (Parse Trees)

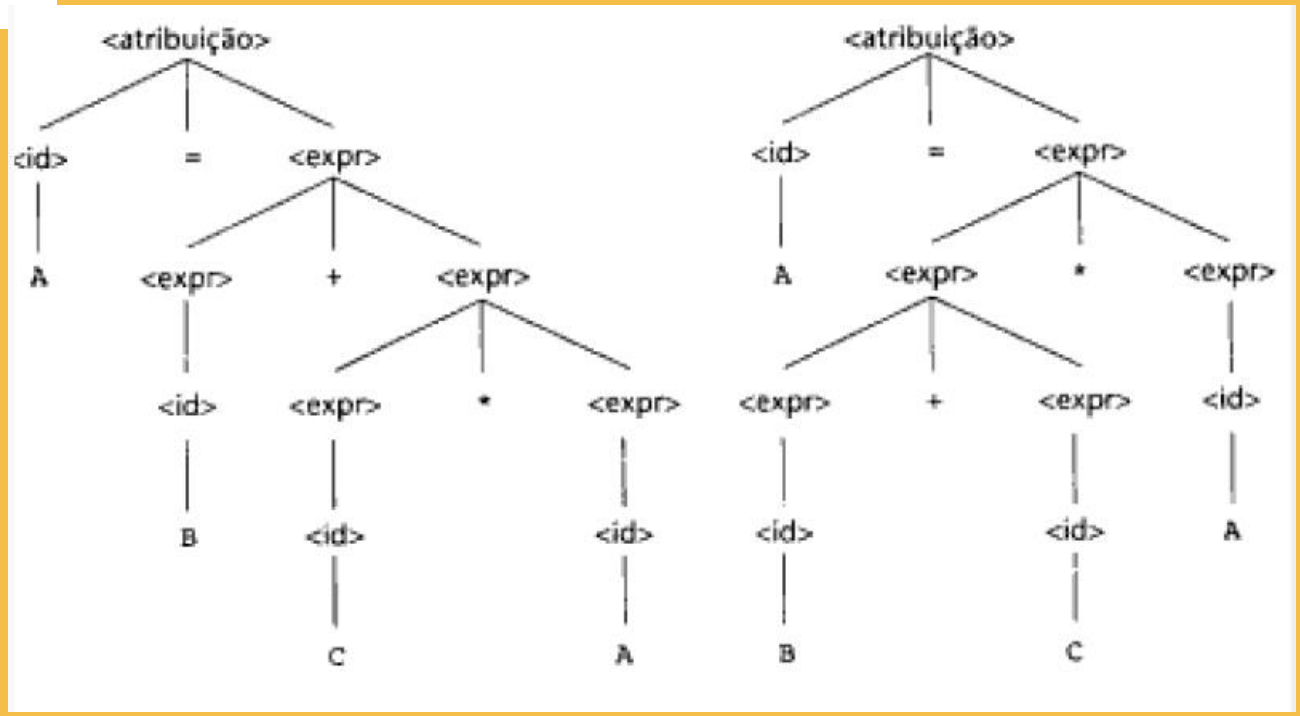
```
<atribuição> = > <id> = <expr>
             = > A = <expr>
             = > A = <id> * <expr>
             = > A = B * <expr>
             = > A = B * ( <expr> )
             = > A = B * ( <id> + <expr> )
             = > A = B * (A + <expr> )
             = > A = B * (A + <id> )
             = > A = B * ( A + C )
```



<atribuição> → <id> = <expr>
<id> → A | B | C
<expr> → <expr> + <expr>
| <expr> * <expr>
| (<expr>)
| <id>

Gramática Ambígua

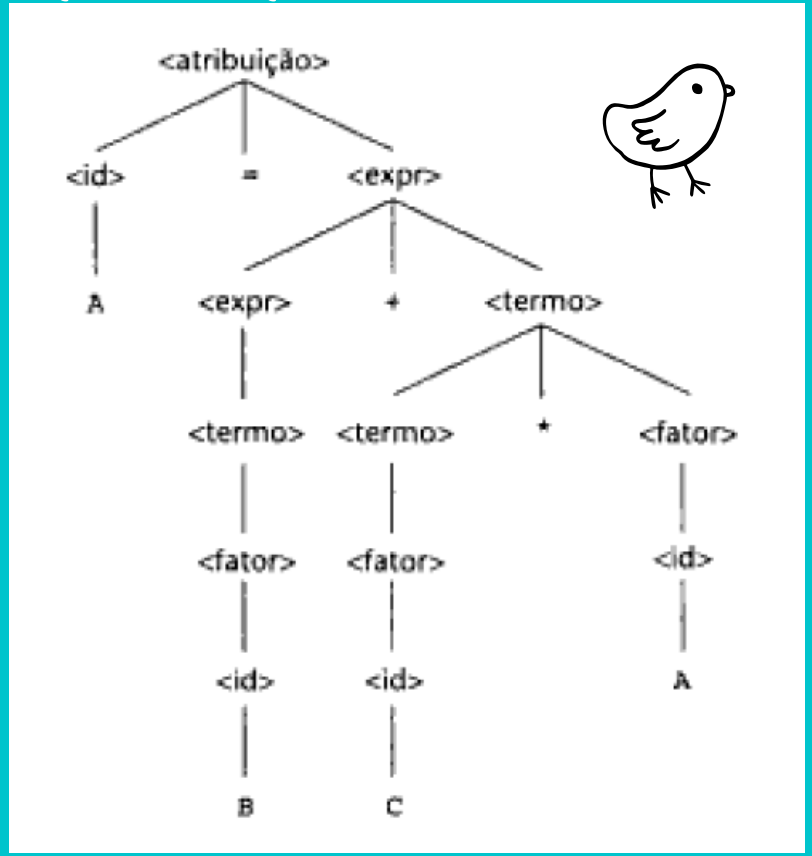
A = B + C * A



Gramática para Expressões

<atribuição> → <id> = <expr>
<id> → A | B | C
<expr> → <expr> + <expr>
| <termo>
<termo> → <termo> * <fator>
| <fator>
<fator> → (<expr>)
| <id>

<atribuição> = > <id> = <expr>
= > A = <expr>
= > A = <expr> + <termo>
= > A = <termo> + <termo>
= > A = <fator> + <termo>
= > A = <id> + <termo>
= > A = B + <termo>
= > A = B + <termo> * <fator>
= > A = B + <fator> * <fator>
= > A = B + <id> * <fator>
= > A = B + C * <fator>
= > A = B + C * <id>
= > A = B + C * A



Derivação à Extrema Direita

```
<atribuição> → <id> = <expr>
<id> → A | B | C
<expr> → <expr> + <expr>
       | <termo>
<termo> → <termo> * <fator>
        | <fator>
<fator> → ( <expr> )
        | <id>
```

```
<atribuição> = > <id> = <expr>
             = > <id> = <expr> + <termo>
             = > <id> = <termo> + <termo> + <fator>
             = > <id> = <expr> + <termo> * <id>
             = > <id> = <expr> + <termo> * A
             = > <id> = <expr> + <fator> * A
             = > <id> = <expr> + <id> * A
             = > <id> = <expr> + C * A
             = > <id> = <termo> + C * A
             = > <id> = <fator> + C * A
             = > <id> = <id> + C * A
             = > <id> = B + C * A
             = > A = B + C * A
```

LOVE!

THANK YOU!