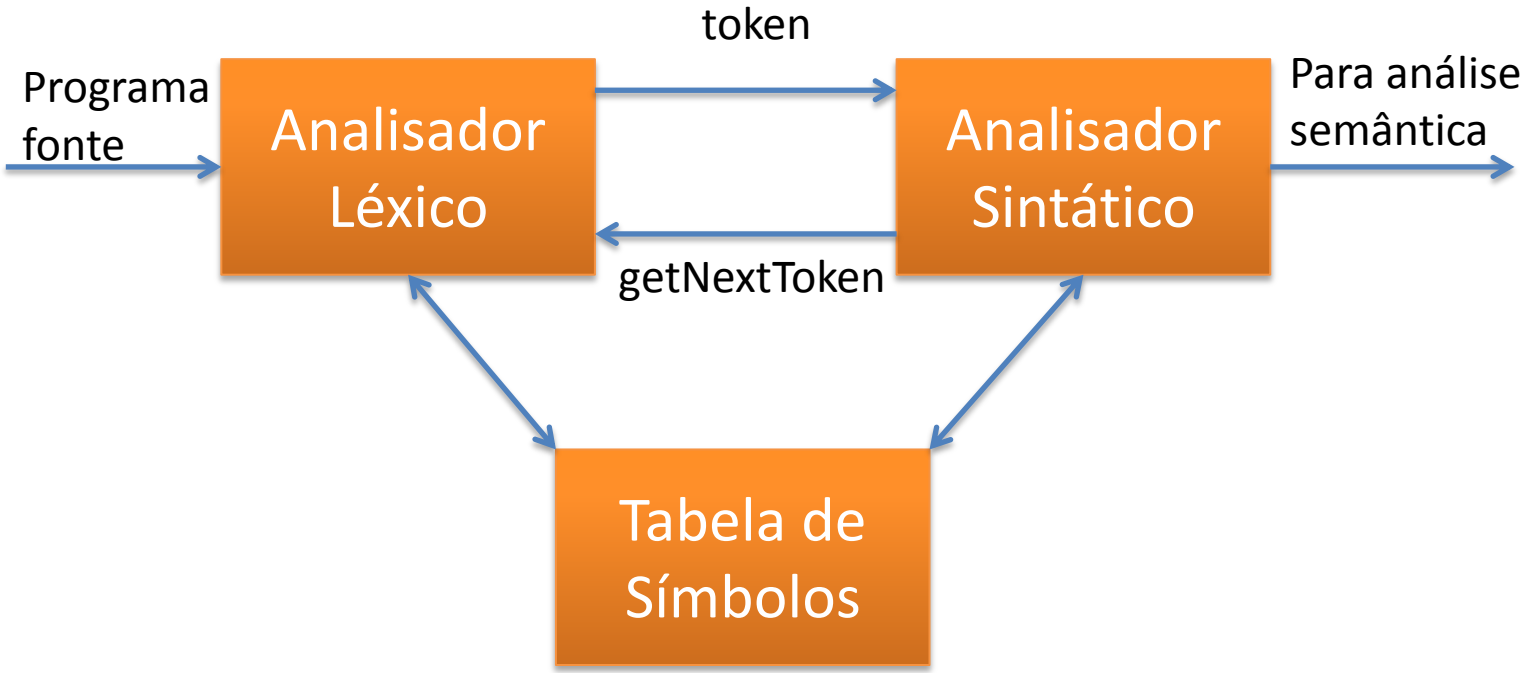
Análise Léxica

Prof. Leandro I. Pinto

Análise léxica

- Na primeira fase do compilador, o analisador léxico lê os caracteres de entrada do programa fonte



- Também remove comentários e o espaço em branco
  - Correlaciona mensagens de erro com o número de quebras de linha

Tokens e Terminais

- Em um compilador, o analisador léxico lê os caracteres do programa fonte;
- Agrupa-os em unidades lexicamente significativas;
  - Chamadas lexemas
- Produz como saída tokens representando esses lexemas;
- Token = um nome e um valor de atributo;
- Tokens (ou terminais) são símbolos abstratos utilizados pelo analisador sintático;
- O valor do atributo, se houver, aponta para a tabela de símbolos, que contem informações adicionais sobre o token;
  - Essas informações não fazem parte da gramática

Tokens, padrões e lexemas

- Token: é um par consistindo de nome e um valor de atributo opcional;
  - O nome representa um tipo de unidade léxica (ex. palavra chave);
- Padrão: é uma descrição da forma que os lexemas de um token podem assumir
  - Palavra chave: ‘int’
  - Número: Expressão regular [0-9]*
- Lexema: uma sequência do programa que casa com o padrão de algum token
  - E é identificada como uma instância desse token

Exemplos

```
printf("Total = %d\n", score);
```

Token	Descrição Informal	Exemplos de Lexemas
`if`	Caracteres i e f	if
`else`	Caracteres e, l ,s, e	else
`cmp_lessthan`	Caractere <	<
`id`	Letra seguida por letras e dígitos	pi, score, D2
`number`	Qualquer constante numérica	3.14159, 0, 6.02e23
`literal`	Qualquer caractere diferente de “, cercado por “s	“Compiladores show”

Atributos de tokens

- Quando mais de um lexema casar com um padrão, o analisador léxico precisa fornecer mais informações;
  - Ex.: Qual lexema foi encontrado realmente;
  - 567 pode retornar o token CONST e um atributo pode conter o valor 567;
- O nome do token influencia na análise sintática;
- O atributo influencia a tradução dos tokens após o reconhecimento;

Exercício prático

1. Construa um autômato que aceite a seguinte linguagem:
   1. $L = \{(\varepsilon|d^*|dd^*.dd^*|o)\ tal\ que\ d\ é\ um\ dígito\ e\ o \in \{+, -, *, /, '\backslash 0'\}\}$
   2. Ex.: o, d, dd, dd.d, dd.ddd
2. Qual a gramática da linguagem desse autômato?
3. Implemente uma função que retorne os tokens de uma expressão aritmética;
   - int lexica(char *str, int *pos);
   - A função recebe uma string e uma posição nesta string. A partir de pos, a função procura pelo próximo token e o retorna, incrementando pos até a posição após o token encontrado.
   - Ex.: Entrada = 50 + 20
     - Chamada 1 Retorna: CONST
     - Chamada 2 Retorna: OPERADOR
     - Chamada 3 Retorna: CONST

Exercício prático - Teste do autômato anterior

d	Accept
d . d	Accept
d d d . d d d d d	Accept
o	Accept
o .	Reject
.	Reject
d .	Reject
o d	Reject
d o	Reject
o o	Reject
λ	Accept

Referências

- AHO, Alfred V et al. (). **Compiladores:** principios, técnicas e ferramentas. 2. ed. São Paulo: Pearson/Addison Wesley, 2007. 634 p. ISBN 9788588639249 (broch.).