

# Análise Léxica

Marcelo Johann

## Conteúdo da aula

- Estrutura de um Compilador
- Análise Léxica
  - Traduz seq. de caracteres em seq. de tokens
- Expressões Regulares
  - Especificam regras para formar lexemas
- Autômatos Finitos
  - Reconhedores de ERs
- Usando o gerador de scanners lex

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 2

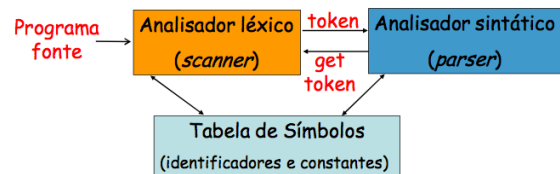
## Estrutura de um Compilador



INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 3

## Funções da Análise Léxica



INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 4

## Análise Léxica

- Identifica as palavras, ou **lexemas**, da linguagem
- Entrada: sequência de caracteres
- Saída: sequência de **tokens**
- Classifica lexemas em tokens!
- Ex: `if (a<100) x = 5;`  
[if] [ ( ] [id, a] [ < ] [num, 100] [ ( ] [id, x] [=] [num, 5] [ ; ]
- Ou  
[if, 0] [ (, 0] [id, 1] [ <, 0] [num, 2] [ (, 0] [id, 3] [=, 0]  
[num, 4] [ ;, 0]

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 5

## Expressões Regulares

- Forma de descrever linguagens simples
  - Mais simples que gramáticas, GLCs
  - Bottom-up: caracteres, construções, parênteses
- Concatenação:
  - $xy$  // x seguido de y
- Alternativas:
  - $x|y$  // x ou y
- Repetição:
  - $x^*$  // x repetido zero ou mais vezes
  - $x^+$  // x repetido uma ou mais vezes

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 6

## Exemplos de expressões regulares

- 00: representa o conjunto {00}
- (0 | 1): representa o conjunto {0,1}
- (0 | 1)(0 | 1): representa o conjunto {00,01,10,11}
- 0\*: { $\epsilon$ , 0, 00, 000, 0000, 00000, ...}
- (0 | 1)\*: representa todos os strings de 0's e 1's
- (0 | 1)\* 00 (0 | 1)\*: representa todos os strings de 0's e 1's com pelo menos dois 0's consecutivos
- Exercício: todos os strings de 0's e 1's começando por 1 e não tendo dois 0's consecutivos ?

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 7

## Exemplos de expressões regulares

- num  $\rightarrow$  digitos fracao\_opc expoente\_opc
- digito  $\rightarrow$  0 | 1 | ... | 9
- digitos  $\rightarrow$  digito digito\*
- fracao\_opc  $\rightarrow$  .digitos |  $\epsilon$
- expoente\_opc  $\rightarrow$  (E (+ | - |  $\epsilon$ ) digitos) |  $\epsilon$

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 8

## Autômatos como Reconhedores

- Autômato finito M sobre um alfabeto  $\Sigma$  é uma 5-upla  $(K, \Sigma, \delta, e_0, F)$ , onde:
  - K é o conjunto finito de estados
  - $\Sigma$  é o alfabeto dos símbolos da linguagem
  - $\delta: K \times \Sigma \rightarrow K$  é a função de transição de estados
  - $e_0$  é o estado inicial
  - F é o conjunto de estados finais
- $\delta$  é uma função parcial, pois não precisa estar definida para todos os pares  $K \times \Sigma$
- Interesse: há equivalência entre AF e ER

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 9

## Autômatos como Reconhedores

$$M = (K, \Sigma, \delta, e_0, F)$$

$$\Sigma = (d, .)$$

$$K = (e_0, e_1, e_2, e_3)$$

$$F = (e_1, e_3)$$

$$\delta(e_0, d) = e_1$$

$$\delta(e_1, d) = e_1$$

$$\delta(e_1, .) = e_2$$

$$\delta(e_2, d) = e_3$$

$$\delta(e_3, d) = e_3$$

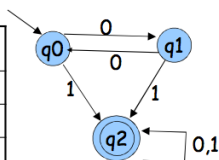
Autômato que reconhece números inteiros e reais

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 10

## Autômatos como Reconhedores

	Estado Atual	0	1
Inicial	q0	q1	q2
	q1	q0	q2
Final	q2	q2	q2



Qual expressão regular corresponde a este autômato ?

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 11

## Exemplo

– Autômato reconhedor em C com tabela

INF01147 - Compiladores - Marcelo Johann - 2010/2

Aula 02 : Slide 12

## O Gerador de Scanners lex

– Exemplos

## Leituras e Tarefas

- Ler capítulos 1 e 2 de [Price, Toscani]
- Fazer autômatos reconhededores:
  - Por estrutura de código
  - Por tabela
  - Para identificador, inteiro, real
  - Combinando Ers em um único autômato
- Usar o gerador de scanners lex
  - Para leitura dos arquivos da primeira aula
  - Para outras experiências