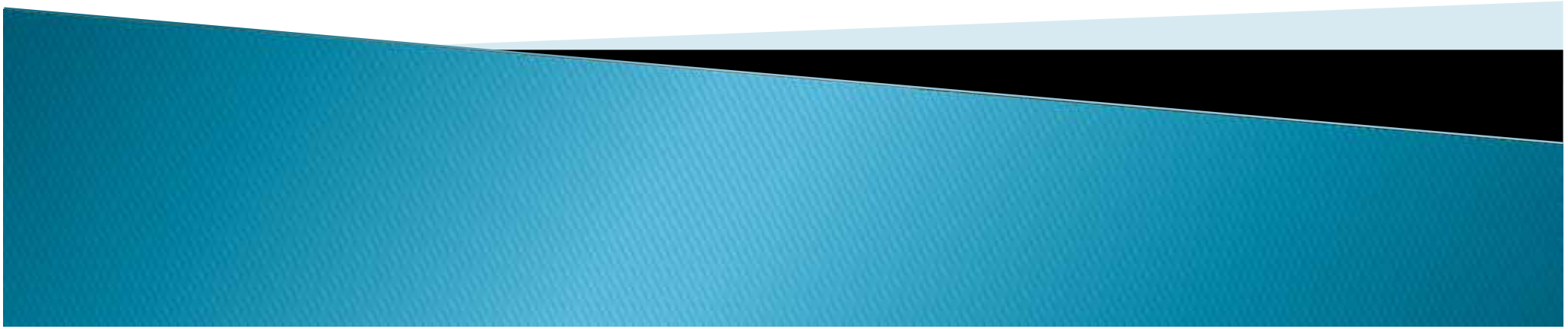
Compiladores

Análise Léxica – Expressões Regulares

Prof. Tales Viegas
https://fb.com/ProfessorTalesViegas



Expressões Regulares

- Gramáticas regulares também podem ser descritas através de expressões regulares.

- Possuem uma notação textual, com símbolos próprios para definir derivações.

Expressões Regulares

- Alguns símbolos

Símbolo	Significado
.	Qualquer caractere (apenas 1)
*	0 ou mais ocorrências (opcionalidade)
+	1 ou mais ocorrências (obrigatoriedade)
[abcd]	1 dos elementos do conjunto delimitado por colchetes (a, b, c ou d)

Expressões Regulares – Exemplos

Exemplo	Significado
aa	Somente a palavra aa
ba*	Palavras que iniciam com b e possuem 0 ou mais a
0-9[0-9]*	Palavras que começam com um dígito e posuem 0 ou mais dígitos
a-z[a-z0-9]*	Palavras que começam com uma letra minúscula e possuem 0 ou mais letras minúsculas ou dígitos
[ab]+	Qualquer combinação de a e b (pelo menus um deles)
[ab]*aa[ab]*	Qualquer expressão com a e b que contenha 2 a em sequência
a*ba*ba*	
[ab]*[(aa)(bb)]	
a?b+	

Exercício 3

- Para cada um dos padrões abaixo descubra a expressão regular para reconhecê-lo:
  - Números Reais – Ex: 99.99, 99999.99, -123.12, +123.4
  - Strings – Ex: “Tales Bitelo Viegas”
  - Comentários – Ex: /* */
  - Números Inteiros – Ex: 1234, 567, 9, 0, -123

Exercício 4

- Crie uma expressão regular que reconheça identificadores que começam com uma letra e tenha pelo menos um sublinhado e após qualquer sequência de letras ou números

Exercício 5

- Crie um autômato finito equivalente a seguinte expressão regular:

  a[a(bc)]*b[ab]*

Implementando um Analisador Léxico

- Funções de um analisador léxico:
  - Localizar e abrir o código-fonte
  - Separar os tokens
  - Classificar os tokens
  - Eliminar comentários
  - Eliminar brancos
  - Gerar uma lista de tokens classificados
  - Fechar o arquivo

Análise Léxica

Supondo o trecho de programa abaixo:

if (5 = MAX) go to 100

A lista devolvida pelo léxico poderia ser:

if, (, [const,341], =, [ident,729],), go to, [label,554]

Índices para a tabela de símbolos: contém informações sobre constantes, variáveis e cédulas.

Tabela de Símbolos

- Controla as informações de escopo e de amarrações dos nomes
- É pesquisada toda vez que um nome é encontrado no código-fonte
- É necessário permitir a entrada de novos nomes e a obtenção daqueles já inseridos
- Podem ser implementados por:
  - Listas lineares
  - Tabelas hash

Tabela de Símbolos

- Supondo *n* entradas:
- **Lista Linear**
  - Fácil implementação
  - Desempenho pobre para *n* grandes

- **Tabela hash**
  - Melhor desempenho
  - Mais esforço de programação
  - Mais espaço para armazenamento

Tabela de Símbolos

- Entradas
  - Declaração de nomes
  - Formato não necessita ser uniforme
  - Implementação: registro com uma sequência de palavras

- Tipos de Entradas
  - Palavras reservadas: inseridas inicialmente na tabela (antes da análise léxica ser iniciada)
  - Identificadores: Fortemente relacionadas com o papel de um nome no contexto do código-fonte
  - Importante: o mesmo nome pode definir objetos distintos

Operações sobre a Tabela

- Verificar se um dado nome está na tabela
- Inserir um nome da tabela
- Acessar as informações associadas a um nome
- Adicionar informações associadas a um nome
- Eliminar um nome

Considerações

- É comum termos mais do que uma tabela de símbolos
- Se o formato das entradas de informação não variar, uma única tabela pode bastar
- Dependendo de como a análise léxica é implementada, pode ser útil inicializar a tabela com as palavras reservadas
- Se a linguagem *reserva* palavras, então é essencial que as palavras reservadas sejam introduzidas na tabela

Exemplo

- São símbolos especiais ou reservados:
  - todos os operadores e separadores de uma gramática: +, -, *, /, :=, :, =, >, >=, <, <=, &, ; etc
  - as palavras reservadas da linguagem : BEGIN, END, FOR etc

Símbolo	Código
<=	1
<	2
:=	3
:	4
=	5