

## TRABALHO FINAL – parte 2: implementação do analisador léxico

Implementar o **analisador léxico** de forma que identifique, nos programas escritos na linguagem 2017.1, os *tokens* corretos, levando em consideração as especificações feitas no trabalho nº1. Deve-se implementar também **tratamento de erros** léxicos, quais sejam: símbolos que não fazem parte da linguagem em questão bem como sequências de símbolos que não obedecem às regras de formação dos *tokens* especificados.

Na implementação do analisador léxico pode ser utilizada qualquer ferramenta para geração de compiladores (GALS, JavaCC, etc.) que gere analisadores sintáticos do tipo descendente (recursivo ou preditivo tabular).

Entrada	– A entrada para o analisador léxico é um conjunto de caracteres, isto é, o programa fonte do editor do compilador.
Saída	<p>– Caso o botão <b>compilar</b> seja pressionado, a ação deve ser: executar a análise léxica do programa fonte e apresentar a saída. Um programa pode ser compilado com sucesso ou apresentar erros. Em cada uma das situações a saída deve ser:</p> <p><u>1ª situação</u>: programa compilado com sucesso</p> <ul style="list-style-type: none"><li>✓ <b>lista de tokens</b>, na área reservada para mensagens, contendo, para cada <i>token</i> reconhecido, a <u>linha</u> onde se encontra, a sua <u>classe</u> (por <u>extenso</u>) e o lexema, nessa ordem;</li><li>✓ <b>mensagem</b> (<i>programa compilado com sucesso</i>), na área reservada para mensagens, indicando que o programa não apresenta erros.</li></ul> <p>As <u>classes</u> possíveis para os <i>tokens</i> são: palavra reservada, identificador, constante inteira, constante real, constante caracter, símbolo especial.</p> <p><u>2ª situação</u>: programa apresenta erros</p> <ul style="list-style-type: none"><li>✓ <b>mensagem</b>, na área reservada para mensagens, indicando que o programa apresenta erro. Neste caso, indicar a <u>linha</u> onde ocorreu o erro e a <u>descrição</u> do erro, emitindo uma mensagem adequada. Tem-se que:<ul style="list-style-type: none"><li>• para símbolo inválido: deve ser apresentado, além da mensagem e da linha onde ocorreu o erro, o símbolo não reconhecido;</li><li>• para constante caracter inválida: devem ser apresentadas a mensagem e a linha onde ocorreu o erro;</li><li>• para comentário de linha e bloco inválidos: devem ser apresentadas a mensagem e a linha onde ocorreu o erro, sendo que, nesse caso, deve ser apresentada a linha onde <u>inicia</u> o comentário.</li></ul></li></ul> <p>As mensagens geradas por ferramentas, como o GALS, devem ser alteradas.</p> <p>Caso seja pressionado o botão <b>compilar</b> e o editor não contenha nenhuma sequência de caracteres ou contenha apenas caracteres de formatação (espaço em branco, quebra de linha ou tabulação), deve ser apresentada a mensagem <i>nenhum programa para compilar</i> na área reservada para mensagens.</p>

### OBSERVAÇÕES:

- As palavras reservadas da linguagem 2017.1 são escritas conforme segue: algoritmo até caracter e enquanto então escreva faça falso fim função início inteiro interrompa leia lógico não ou procedimento quebra real repita retorne se senão variáveis verdadeiro. As palavras reservadas devem ser especificadas como casos especiais de identificador, definido no trabalho nº1. Observa-se que palavras reservadas podem ser excluídas ou outras podem ser incluídas, quando da especificação das regras gramaticais.
- Os símbolos especiais da linguagem 2017.1 são: + - \* / \ % ^ , : ; <- = <> < <= > >= ( ) Observa-se que símbolos especiais podem ser excluídos ou outros podem ser incluídos, quando da especificação das regras gramaticais. Símbolos diferentes dos especificados nesse item constituem erro léxico.
- Os comentários (de linha e de bloco) e os caracteres de formatação (espaços em branco, final de linha e tabulação, exceto em constantes literais e comentários) devem ser reconhecidos, porém ignorados. Ou seja, não devem ser apresentados como saída do analisador léxico. Isso deve ser especificado na própria ferramenta que será usada para gerar o analisador léxico, no arquivo com as especificações léxicas (no caso do GALS, arquivo com extensão .gals). Comentários que não seguem o padrão de formação especificado devem ser diagnosticados como erro léxico.
- As **mensagens de erro** devem ser conforme nos exemplos abaixo:
  - Erro na linha 1 – @ símbolo inválido
  - Erro na linha 1 – constante caracter inválida ou não finalizada
  - Erro na linha 1 – comentário de bloco inválido ou não finalizadoNo 1º exemplo, o símbolo @ não é um símbolo especial de linguagem, portanto um erro léxico. O símbolo foi apresentado na mensagem de erro. Nos exemplos seguintes foram apresentadas a linha e a mensagem de erro. Para comentários de bloco inválidos deve ser apresentada a linha onde inicia o comentário. Nos exemplos 2 e 3 as sequências não reconhecidas não foram apresentadas na mensagem de erro.
- As especificações feitas no trabalho nº1 (e já corrigidas) devem usadas para implementação do analisador léxico. Observa-se que essas especificações devem ser adaptadas à notação da ferramenta que será utilizada para gerar o analisador léxico. Além disso, trabalhos desenvolvidos usando especificações diferentes daquelas elaboradas pela equipe no trabalho nº1 receberão nota 0.0 (zero).
- O trabalho nº1 deve ser devolvido, caso contrário, será atribuído 0.0 (zero) à implementação do analisador léxico.

- A implementação do analisador léxico, bem como da interface do compilador, deve ser disponibilizada no AVA, na **pasta da sua equipe**. Deve ser disponibilizado um **arquivo compactado** (com o nome: `lexico`) contendo: o código fonte, o executável e o arquivo com as especificações léxicas (no GALS, arquivo com extensão `.gals`).
- Na avaliação do analisador léxico serão levadas em consideração: a correta especificação dos *tokens*, conforme trabalho nº1; a qualidade das mensagens de erro, conforme descrito acima, e o uso apropriado de ferramentas para construção de compiladores.

**DATA:** entregar o trabalho até às 23h do dia 11/04/2017 (terça-feira). Não serão aceitos trabalhos após data e hora determinados.

## EXEMPLOS DE ENTRADA / SAÍDA

### EXEMPLO 1: sem erro léxico

<b>ENTRADA</b>		<b>SAÍDA (na área de mensagens)</b>		
linha		linha	classe	lexema
1	/* isso é um comentário */	3	palavra reservada	escreva
2		3	identificador	i_area
3	escreva i_area =	3	símbolo especial	=
4		5	constante caracter	"valor"
5	"valor" 01.0	5	constante inteira	0
		5	constante real	1.0
			programa compilado com sucesso	

### EXEMPLO 2: com erro léxico

<b>ENTRADA</b>		<b>SAÍDA (na área de mensagens)</b>		
linha				
1	/* isso é um comentário */	Erro na linha 5 - constante caracter inválida ou não finalizada		
2				
3	escreva i_area =			
4				
5	"valor 01.0			