TRABALHO FINAL – parte 2: implementação do analisador léxico

Implementar o **analisador léxico** de forma que identifique, nos programas escritos na linguagem 2013.1, 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	• Caso o botão **compilar** 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: 1ª situação: programa compilado com sucesso ✓ **lista de *tokens***, na área reservada para mensagens, contendo, para cada *token* reconhecido, a linha onde se encontra, a sua classe (por extenso) e o lexema, nessa ordem; ✓ **mensagem** (*programa compilado com sucesso*), na área reservada para mensagens, indicando que o programa não apresenta erros. As classes possíveis para os *tokens* são: palavra reservada, identificador, constante inteira, constante real, constante literal, símbolo especial. 2ª situação: programa apresenta erros ✓ **mensagem**, na área reservada para mensagens, indicando que o programa apresenta erro. Neste caso, indicar a linha onde ocorreu o erro e a descrição do erro, emitindo uma mensagem adequada. Tem-se que: para palavra reservada inválida e símbolo inválido deve ser apresentado também, respectivamente, a palavra reservada e o símbolo não reconhecido; para constante literal inválida e comentário (de linha ou de bloco) inválido deve ser apresentada a linha onde ocorreu o erro. As mensagens geradas por ferramentas, como o GALS, devem ser alteradas. Caso seja pressionado o botão **compilar** e o editor não contenha nenhuma sequência de caracteres, deve ser apresentada a mensagem *nenhum programa para compilar* na área reservada para mensagens.

OBSERVAÇÕES:

- As palavras reservadas da linguagem 2013.1 são escritas conforme segue: `and false if in isFalseDo isTrueDo main module not or out return true while`. As palavras reservadas devem ser especificadas como casos especiais de `palavra reservada`, definida no trabalho nº1. Observa-se que palavras reservadas podem ser excluídas ou outras podem ser incluídas, quando da especificação das regras gramaticais. Sequências de símbolos que seguem o padrão de formação das palavras reservadas, mas são diferentes das especificadas nesse item, constituem erro léxico.
- Os símbolos especiais da linguagem 2013.1 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, tabulação, etc., 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.
- São exemplos de **mensagens de erro** adequadas:
  Erro na linha 1 – `enquantoVerdadeiro` palavra reservada inválida
  Erro na linha 1 – `@` símbolo inválido
  Erro na linha 1 – constante literal não finalizada
  Erro na linha 1 – comentário de linha incorreto
  Erro na linha 1 – comentário de bloco não finalizado
  No 1º exemplo, a sequência `enquantoVerdadeiro` segue o padrão de formação das palavras reservadas, mas não é uma palavra reservada da linguagem, portanto um erro léxico. A palavra reservada foi apresentada na mensagem de erro. No 2º 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.
- 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, no **repositório 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 18/04/2013 (quinta-feira). Não serão aceitos trabalhos após data e hora determinados.

EXEMPLOS DE ENTRADA / SAÍDA

EXEMPLO 1: sem erro léxico

ENTRADA		SAÍDA (na área de mensagens)		
linha		**linha**	**classe**	**lexema**
1	// isso é um comentário de linha	3	palavra reservada	while
2		3	identificador	i_area
3	while i_area <-	3	símbolo especial	<-
4		5	constante literal	"valor"
5	"valor" main	5	palavra reservada	main
		programa compilado com sucesso		

EXEMPLO 2: com erro léxico

ENTRADA		SAÍDA (na área de mensagens)
linha		Erro na linha 3 - iStrueDo palavra reservada inválida
1	// isso é um comentário de linha	
2		
3	while i_area <- iStrueDo	
4		
5	"valor" main	