Enunciado Laboratório 01 de Compiladores.
Entrega 04/03/2023

Você deve:

- **Fazer o trabalho individualmente;**
- **Implementar e enviar um programa com a solução;**
- **Realizar os envios SOMENTE pelo Classroom, até 04/03/2023.;**
- **Explicar ao docente seu código em algum momento de aula de laboratório.**

Escreva um programa de computador que faça análise léxica do texto de um programa de computador na linguagem sorteada para você. Nesta análise não é necessário tratar modo XML (Scala), nem validade de caractere Unicode (Go). Seu programa deve identificar todos os lexemas/tokens/unidades léxicas da linguagem. Sugiro fortemente utilizar um gerador automático de parser léxico, tal como JavaCC.

A saída do seu programa deve conter uma linha para cada lexema/token/unidade léxica encontrada. A primeira string até o primeiro espaço de cada linha deve ser uma palavra cujas letras estejam em MAIÚSCULO identificando a classe do token. Por exemplo, ID para identificador, NUM_DEC para número decimal. Você deve escolher os nomes para as classes de Tokens e explicar ao docente no momento da apresentação. Em seguida, deve vir um espaço seguido pela sequência de caracteres do arquivo de entrada que correspondem ao token correspondente a esta linha.

Sorteamos em aula 9 linguagens, como visto abaixo. As demais foram atribuídas por ordem alfabética das 4 linguagens escolhidas.

Linguagem	Aluno
Swift	Anna Júlia Costa Lauton
Java SE 14	Deise Santana dos Santos
Kotlin	Eike Stalei Vieira Neves
Ansi C	Emerson Eustáquio Santos Rodrigues Versiani
Scala	Felipe Rocha Boa-Sorte
C#	Filipi Maciel Rodrigues Jardim
FreePascal	Jefeson Martins Delazeri
Ceylon	Matteus Felippe Batista Silva
Lua	Mayara Assis Nascimento
Ruby	Pedro Henrique Soares Medeiros
C--	Talita Rodrigues de Souza
Go	Gabriel
Rust	Cauã

Abaixo, os manuais das linguagens.

	Linguagens	Manual
0	C--	https://www.cs.tufts.edu/~nr/c--/extern/man2.pdf
1	Free Pascal	https://www.freepascal.org/docs-html/ref/refch1.html#x8-70001
2	ANSI C	http://www.cs.columbia.edu/~sedwards/papers/sgi1999c.pdf
3	Go	https://go.dev/ref/spec
4	Rust	https://doc.rust-lang.org/stable/reference/tokens.html
5	Scala	https://www.scala-lang.org/files/archive/spec/2.11/01-lexical-syntax.html
6	C#	https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/language-specification/lexical-structure#64-tokens
7	Ceylon	http://web.mit.edu/ceylon_v1.3.3/ceylon-1.3.3/doc/en/spec/html_single/#lexical
8	Swift	https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html
9	Java SE 14	https://docs.oracle.com/javase/specs/jls/se14/jls14.pdf
10	Kotlin	https://kotlinlang.org/spec/syntax-and-grammar.html#syntax-and-grammar
11	Lua	https://www.lua.org/manual/5.1/manual.html (Não é necessário tratar "long brackets" de strings)
12	Ruby 1.4	https://ruby-doc.org/docs/ruby-doc-bundle/Manual/man-1.4/syntax.html#lexical https://ruby-doc.org/docs/ruby-doc-bundle/Manual/man-1.4/index.html