Enunciado Laboratório 01 de Compiladores.
Entrega 26/04/2025

Você deve:

- **Fazer o trabalho individualmente;**
- **Implementar e enviar um programa com a solução;**
- **Realizar os envios SOMENTE pelo Classroom, até 26/04/2025.;**
- **Explicar ao docente seu código em algum momento de aula de laboratório.**
- **O nome do(s) arquivo(s) enviado(s) deve(m) ser no seguinte formato**
  - **compiladores-lab1-A-nome-completo-do-aluno.extensao**
  - **compiladores-lab1-B-nome-completo-do-aluno.extensao**
  - **compiladores-lab1-C-nome-completo-do-aluno.extensao**
  - **etc**

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). Também não é necessário tratar quaisquer caracteres além dos comuns ASCII. Também não é necessário tratar modos especiais de strings, bastando tratar pelo menos um modo de strings. Seu programa deve identificar todos os lexemas/ tokens/unidades léxicas da linguagem (exceto aqueles excluído acima). Sugiro fortemente utilizar um gerador automático de parser léxico, tal como JavaCC, ou uma linguagem com suporte a expressões regulares, tal como Ruby.

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.

Cada aluno terá uma linguagem a ele atribuída. Para o cálculo desta atribuição deve ser calculado o valor delta igual ao resultado do resto da divisão por 22 da soma das dezenas que serão sorteadas na Loteria Quina do dia 04/abril/2025. Por exemplo, se os valores sorteados forem 10, 13, 28, 33, 77, então o valor delta será igual a 7. De posse do valor delta, a linguagem do aluno com número x será a linguagem de número (x+delta) mod 22. Por exemplo, o aluno de número 10 terá a linguagem de número 17, enquanto o aluno de número 20 terá a linguagem de número 5. Abaixo vemos as tabelas com os número dos alunos e das linguagens.

Tabela 1

0	Alexandre Julio Lima Mendes
1	Anne Karoline Dias Santos
2	Enio Filipe Miranda Souza
3	Filipe Abner Soares Melo
4	Gabriel Mendes Sizilio
5	Gustavo Henrique Alves Rocha
6	Gustavo Rafael Nunes Durães
7	Hudson Rikelme Soares Aquino
8	Ivanderlei Mendes Silva Filho
9	João Duarte Colares
10	João Vitor Pereira Amorim
11	Lucas Flávio Gabrich Marinho
12	Lucas Pereira Freitas
13	Maicon Pedro Macedo Leles
14	Marcos Dias de Andrade
15	Natã Teixeira Santos de Oliveira
16	Patrick
17	Pedro Gabriel Dias
18	Rias Alves Leal
19	Saulo Macedo
20	Thiago Evangelista dos Santos
21	Yodemis Júnior Soares Nascimento

Tabela 1

0	Ada	http://www.ada-auth.org/standards/ada22.html
1	Ansi C	http://www.cs.columbia.edu/~sedwards/papers/sgi1999c.pdf
2	C#	https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/language-specification/lexical-structure#64-tokens
3	Ceylon	http://web.mit.edu/ceylon_v1.3.3/ceylon-1.3.3/doc/en/spec/html_single/#lexical
4	D	https://dlang.org/spec/lex.html
5	Dart	https://spec.dart.dev/DartLangSpecDraft.pdf
6	Eiffel	https://www.eiffel.org/doc/eiffel/Eiffel_programming_language_syntax#Eiffel_non-production_elements
7	FreePascal	https://www.freepascal.org/docs-html/ref/refch1.html#x8-70001
8	GAMS	https://www.gams.com/49/docs/UG_GAMSPrograms.html?search=lexical
9	Go	https://go.dev/ref/spec
10	Haskell	https://www.haskell.org/definition/haskell2010.pdf
11	Icon	https://www2.cs.arizona.edu/icon/ftp/doc/lb1up.pdf
12	Java SE 14	https://docs.oracle.com/javase/specs/jls/se14/jls14.pdf
13	Kotlin	https://kotlinlang.org/spec/syntax-and-grammar.html#syntax-and-grammar
14	Lua	https://www.lua.org/manual/5.1/manual.html (Não é necessário tratar "long brackets" de strings)
15	Nim	https://nim-lang.org/docs/manual.html#lexical-analysis
16	OCaml	https://ocaml.org/manual/5.2/lex.html
17	Ruby	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
18	Rust	https://doc.rust-lang.org/stable/reference/tokens.html
19	Scala	https://www.scala-lang.org/files/archive/spec/2.11/01-lexical-syntax.html
20	Smalltalk	https://wiki.squeak.org/squeak/uploads/172/standard_v1_9-indexed.pdf
21	Swift	https://docs.swift.org/swift- book/ReferenceManual/LexicalStructure.html