German University in Cairo
Department of Computer Science
Assoc. Prof. Haythem O. Ismail

CSEN1002 Compilers Lab, Spring Term 2024
Task 5: Context-Free Grammars Left-Recursion Elimination
Due: Week starting 02.04.2024

1 Objective

For this task, you will implement context-free grammar (CFG) left-recursion elimination algorithm introduced in Lecture 3 of CSEN1003. Recall that a CFG is a quadruple (V, Σ, R, S) where V and Σ are disjoint alphabets (respectively, containing *variables* and *terminals*), $R \subseteq V \times (V \cup \Sigma)^*$ is a set of *rules*, and $S \in V$ is the *start variable.*

2 Requirements

- We make the following assumptions about input CFGs for simplicity.
  a) The set V of variables consists of upper-case English letters.
  b) The start variable is the symbol S.
  c) The set Σ of terminals consists of lower-case English letters (except the letter `e`).
  d) The letter "`e`" represents ε.
  e) We only consider CFGs with no cycles and no ε-rules.
- You should implement a class constructor `CfgLeftRecElim`, and two methods; `toString`, and `eliminateLeftRecursion`.
- `CfgLeftRecElim`, a class constructor, takes one parameter which is a string description of a CFG and constructs a CFG instance. A string encoding a CFG is of the form $V\#T\#R$.
  - V is a string representation of the set of variables; a semicolon-separated sequence of upper-case English letters, starting with S.
  - T is a string representation of the set of terminals; a semicolon-separated sequence of alphabetically sorted lower-case English letters.
  - R is a string representation of the set of rules. R is a semicolon-separated sequence of pairs. Each pair represents the largest set of rules with the same left-hand side. Pairs are of the form i/j where i is a variable of V and j is a string representation of the set of right-hand sides—a comma-separated sequence of strings. These pairs are sorted by the common left-hand side i based on the ordering of V.
- For example, consider the CFG $G_1 = (\{S, T, L\}, \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{i}\}, R, S)$, where R is given by the following productions.

$$\begin{array}{lcl} S & \rightarrow & S\,\texttt{c}\,T\,\texttt{i} \mid L\,\texttt{a} \mid T\,\texttt{i} \mid \texttt{b} \\ T & \rightarrow & \texttt{a}\,S\,\texttt{b} \mid L\,\texttt{a}\,\texttt{b}\,S \mid \texttt{i} \\ L & \rightarrow & S\,\texttt{d}\,L \mid S\,\texttt{i} \end{array}$$

This CFG will have the following string encoding.

```
S; T; L#a; b; c; d; i#S/ScTi, La, Ti, b; T/aSb, LabS, i; L/SdL, Si
```

- `toString` returns a string representation of a CFG. This string representation is the same as the one used for the input to the constructor.

- `eliminateLeftRecursion` eliminates left recursion in the constructed CFG where a newly-introduced variable, for the elimination of immediate left-recursion for variable A, is the string A'. The letter `e` denotes the empty string. Newly added rules appear in the order indicated in Slides 33 and 34 of Lecture 3. For example, after invoking the method on G_1, the string returned by `toString` is the following (split for readability)

```
S;T;L;S';L'#a;b;c;d;i#S/LaS',TiS',bS';T/aSb,LabS,i;
L/aSbiS'dLL',iiS'dLL',bS'dLL',aSbiS'iL',iiS'iL',bS'iL';S'/cTiS',e;
L'/aS'dLL',abSiS'dLL',aS'iL',abSiS'iL',e
```

- Important Details:
  - Your implementation should be done within the template file "`CfgLeftRecElim.java`" (uploaded to the CMS).
  - You are not allowed to change package, file, constructor, or method names/signatures.
  - You are allowed to implement as many helper classes/methods within the same file (if needed).
  - Public test cases have been provided on the CMS for you to test your implementation.
  - Please ensure that the public test cases run correctly without modification before coming to the lab to maintain a smooth evaluation process.
  - Private test cases will be uploaded before your session and will have the same structure as the public test cases.

3 Evaluation

- Your implementation will be tested on ten CFGs.

- You get one point for each correct output of `toString` following a call to `eliminateLeftRecursion`; hence, a maximum of ten points.

- The evaluation will take place during your lab session of the week starting Tuesday, April 02.

4 Online Submission

- You should submit your code at the following link.

  `https://forms.gle/X3ia9JRmhgfB3Fja9`

- Submit one Java file (`CfgLeftRecElim.java`) containing executable code.
- **Online submission is due by the end of your lab session.**