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

CSEN1002 Compilers Lab, Spring Term 2023
Task 4: Context-Free Grammars Epsilon & Unit Rules Elimination
Due: Week starting 18.03.2023

1 Objective

For this task you will implement the algorithms for eliminating epsilon and unit rules from a given context-free grammar (CFG). 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) $\varepsilon \notin L(G)$.
- You should implement a class constructor `CfgEpsUnitElim`, and three methods; `toString`, `eliminateEpsilonRules`, and `eliminateUnitRules`.
- `CfgEpsUnitElim`, 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 a 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 set of right-hand sides—a comma-separated sequence of lexicographically sorted strings.[1] These pairs are sorted by the common left-hand side i based on the ordering of V.

[1] This is also the natural ordering of Strings in java. It is used by `Collections.sort(List<String> list)` and `Arrays.sort(String[] a)`

- For example, consider the CFG $G_1 = (\{S, A, B, C\}, \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{x}\}, R, S)$, where R is given by the following productions.

$$
\begin{array}{lcl}
S & \rightarrow & \mathtt{a}\ A\ \mathtt{b} \mid \mathtt{x}\ B \\
A & \rightarrow & B\ \mathtt{c} \mid C \mid \mathtt{c} \mid \mathtt{d} \\
B & \rightarrow & C\ A\ C\ A \mid \varepsilon \\
C & \rightarrow & A \mid \mathtt{b} \mid \varepsilon
\end{array}
$$

  This CFG will have the following string encoding.

```
S;A;B;C#a;b;c;d;x#S/aAb,xB;A/Bc,C,c,d;B/CACA,e;C/A,b,e
```

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

- `eliminateEpsilonRules` eliminates epsilon rules from the constructed CFG using the classical algorithm. For example, after invoking the method on G_1, the string returned by `toString` is the following (split for readability)

```
S;A;B;C#a;b;c;d;x#S/aAb,ab,x,xB;A/Bc,C,c,d;
B/A,AA,AC,ACA,C,CA,CAA,CAC,CACA,CC,CCA;C/A,b
```

- `eliminateUnitRules` eliminates unit rules from the constructed CFG using the classical algorithm. For example, after invoking the method on G_1, the string returned by `toString` is the following

```
S;A;B;C#a;b;c;d;x#S/aAb,xB;A/Bc,b,c,d,e;B/CACA,e;C/Bc,b,c,d,e
```

- Additionally, the above two methods can be called sequentially. Thus the result of invoking `toString` after invoking `eliminateEpsilonRules` then `eliminateUnitRules` returns the following (split for readability)

```
S;A;B;C#a;b;c;d;x#S/aAb,ab,x,xB;A,Bc,b,c,d;
B/AA,AC,ACA,Bc,CA,CAA,CAC,CACA,CC,CCA,b,c,d;C/Bc,b,c,d
```

- Important Details:
  - Your implementation should be done within the template file "`CfgEpsUnitElim.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.
  - Three CFGs will test epsilon rule elimination only.
  - Three CFGs will test unit rule elimination only.
  - Four CFGs will test epsilon rule elimination followed by unit rule elimination.
- You get one point for each correct output of `toString`; hence, a maximum of ten points.
- The evaluation will take place during your lab session of the week starting Saturday March 18.

4 Online Submission

- You should submit your code at the following link.

  `https://forms.gle/iCQhZQk2pM7Hig2p9`

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