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

CSEN1002 Compilers Lab, Spring Term 2024
Task 9: ANTLR Parsing I
Due: Week starting 07.05.2024

1 Objective

For this task you will use ANTLR (`www.antlr.org`) to implement an SDD for the following problem. ANTLR documentation is available here:

`https://github.com/antlr/antlr4/blob/master/doc/index.md`

2 Requirements

- You are required to use ANTLR to implement the SDD appearing below for a CFG that generates $\{\mathtt{0}, \mathtt{1}, \#\}^+$.

$$
\begin{array}{llll}
S & \longrightarrow & F & S.\mathit{check} = F.\mathit{check} * F.m \\
F & \longrightarrow & DT & D.r = 1; \quad D.c = 1 \\
 & & & T.r = 2; \quad T.l = D.l \\
 & & & F.\mathit{check} = D.\mathit{check} * T.\mathit{check}; \quad F.m = T.m \\
T & \longrightarrow & \#N & N.r = T.r; \quad N.c = 1 \quad N.l = T.l \\
 & & & T.\mathit{check} = N.\mathit{check}; \quad T.m = N.m \\
T & \longrightarrow & \varepsilon & T.\mathit{check} = 1; \quad T.m = 1 \\
N & \longrightarrow & DT & D.r = N.r; \quad D.c = 1 \\
 & & & T.l = N.l; \quad T.r = N.r + 1 \\
 & & & N.\mathit{check} = D.\mathit{check} * T.\mathit{check}; \quad N.m = \mathit{equals}(D.l, N.l) * T.m \\
D & \longrightarrow & \mathtt{0}D_1 & D_1.r = D.r; \quad D_1.c = D.c + 1 \\
 & & & D.l = D_1.l; \quad D.\mathit{check} = (1 - \mathit{equals}(D.c, D.r)) * D_1.\mathit{check} \\
D & \longrightarrow & \mathtt{1}D_1 & D_1.r = D.r; \quad D_1.c = D.c + 1 \\
 & & & D.l = D_1.l; \quad D.\mathit{check} = \mathit{equals}(D.c, D.r) * D_1.\mathit{check} \\
D & \longrightarrow & \mathtt{0} & D.l = D.c; \quad D.\mathit{check} = 1 - \mathit{equals}(D.c, D.r) \\
D & \longrightarrow & \mathtt{1} & D.l = D.c; \quad D.\mathit{check} = \mathit{equals}(D.c, D.r)
\end{array}
$$

- The start variable S has an attribute *check* whose value is 1 if the generated string is a diagonal boolean matrix; otherwise, the value of *Check* is 0. A string over $\{\mathtt{0}, \mathtt{1}, \#\}$ represents a boolean matrix if it is of the form $r_1\#r_2\#\ldots\#r_n$, where $r_i \in \{\mathtt{0}, \mathtt{1}\}^+$ and $|r_i| = |r_j|$, for every $1 \leq i, j \leq n$; the matrix is diagonal if, in addition, the jth bit of r_i, $1 \leq j \leq |r_i|$, is 1 if and only if $j = i$, for every $1 \leq i \leq n$.

- **The only operations allowed on attributes are assignments, additions, subtraction, multiplications, and equality checks; an equality check is an expression of the form $equals(x, y)$ whose value is 1 if x is equal to y and is 0 otherwise.**
- The provided method `sCheckValue` uses the ANTLR grammar to get the value of *S.check* for a given input string. For example, for the string `10#01`, `sCheckValue` returns 1; and returns 0 for the string `11#01`.
- Important Details
  - Your implementation should be done within the template file uploaded to the CMS.
  - You are not allowed to change the grammar name, the rule name "`s`" or attribute "`check`".
  - You are allowed to write as many helper parser and lexer rules within the same grammar 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.
  - A Java file is provided in order to easily test your grammar with custom strings in addition to the public test cases.
  - Private test cases will be uploaded before your session and will have the same structure as the public test cases.

3 Evaluation

- Your SDD will be tested using ten inputs.
- You get one point for each correct output; hence, a maximum of ten points.

4 Online Submission

- You should submit your code at the following link.

`https://forms.gle/XiHwHwtrDdSSbG2r8`

- Submit one file "`Task9.g4`" containing the grammar.
- **Online submission is due by the end of your lab session.**