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

CSEN1002 Compilers Lab, Spring Term 2022
Task 3: Regular Expressions

Due: Week starting 19.03.2022

1 Objective

For this task you need to implement Thompson's construction for converting a regular expression to an equivalent NFA. Description of Thompson's construction can be found in Chapter 3 of the textbook and at https://en.wikipedia.org/wiki/Thompson's_construction.

2 Requirements

- We make the following assumptions for simplicity.
  a) The alphabet Σ of the regular expression is always the binary alphabet $\{\texttt{0}, \texttt{1}\}$.
  b) Regular expressions do not include $\varnothing$.
  c) The empty string ε is represented by `e`.
  d) $\circ$ is represented by `.` and $\cup$ by `|`.
  e) Regular expressions are represented in *postfix* notation.
  f) States of the resulting NFA are numbers.
  g) For a postfix regular expression R, states *introduced* by NFA equivalent to a prefix of R are smaller (as numbers) than states *introduced* by NFA equivalent to longer prefixes of R. For operators (such as concatenation and *) which introduce a start and an accept state, the start state is smaller (as a number) than the accept state.
- You should implement a class constructor `RegToNFA` and a method `toString`.
- `RegToNFA` takes one parameter which is a string description of a postfix regular expression and constructs the equivalent NFA as per Thompson's construction.
- `toString` returns a string describing the NFA resulting from Thompson's construction. A string describing the NFA resulting from Thompson's construction is of the form $N\#I\#F\#Z\#O\#E$.
  – N is the number of states of the NFA.
  – I is the initial state.
  – F is the final state.
  – Z, O, and E, respectively, represent the `0`-transitions, the `1`-transitions, and the ε-transitions.

  - Z, O, and E are semicolon-separated sequences of pairs of states; each pair is a comma-separated sequence of two states. A pair i, j represents a transition from state i to state j; for Z this means that $\delta(i, \texttt{0}) = j$, similarly for O and E. These pairs are sorted by the source state and (if multiple pairs share the same source state, due to non-determinism) then by the destination state.
  - For example, `toString`, being invoked on a a `RegToNFA` object representing the regular expression `01|`, should return the string `6#4#5#0,1#2,3#1,5;3,5;4,0;4,2`

- Important Details:
  - Your implementation should be done within the template file "`RegToNFA.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 by ten input regular expressions.
- 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 19.

4 Online Submission

- You should submit your code at the following link.

  https://forms.gle/Mq3HPYHGpiSxWG279

- Submit one Java file (`RegToNFA.java`) containing executable code.
- Online submission is due on Thursday, March 24th, by 23:59.