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

CSEN 1003 Compiler, Spring Term 2020
Practice Assignment 2

Discussion: 09.02.20 - 12.02.20

Exercise 2-1

Consider the following input string: `aaabaabbbababbb`

a) and the action-augmented regular definition:

$$\begin{array}{lll} 1 & \longrightarrow & \texttt{b}^+ \quad \{\texttt{printf("1")}\} \\ 2 & \longrightarrow & \texttt{aab}^* \quad \{\texttt{printf("2")}\} \\ 3 & \longrightarrow & \texttt{a} \quad \{\texttt{printf("3")}\} \end{array}$$

Draw the state diagram of an equivalent fallback DFA with actions. What will be printed when the DFA is run of the provided input?

b) Repeat for the following action-augmented regular definition.

$$\begin{array}{lll} 4 & \longrightarrow & (\texttt{aa})^*\texttt{b}^* \quad \{\texttt{printf("2")}\} \\ 3 & \longrightarrow & \texttt{a} \quad \{\texttt{printf("3")}\} \end{array}$$

Exercise 2-2

For the following action-augmented regular definition, give a regular expression describing the language of possible outputs. Assume that all inputs are strings of 0's and 1's only.

$$\begin{array}{lll} 5 & \longrightarrow & \texttt{0} \quad \{\texttt{printf("c")}\} \\ 6 & \longrightarrow & \texttt{00} \quad \{\texttt{printf("a")}\} \\ 7 & \longrightarrow & \texttt{1} \quad \{\texttt{printf("b")}\} \end{array}$$

Exercise 2-3

Give a regular definition for non-negative integers without leading zeros. A zero is represented by a single `0`.

Exercise 2-4 (Based on an exercise from the textbook)

Write an action-augmented regular definition for a C-style string literal. A string literal starts and ends with double-quotes (") and any character in between. Any " appearing between the initial and final double-quotes must be escaped by preceding it with a backslash (\). Hence, a backslash in the string must be represented by two backslashes. The actions should produce a token $\langle \textbf{lit}, s \rangle$, where s is the string without the enclosing double-quotes and the escape backslashes.

Exercise 2-5

In this exercise, you will write an action-augmented regular definition to process sequences of Haskell-style lists of non-negative integers. A list of non-negative integers has two alternate representations:

a) A comma-separated sequence of non-negative integer literals between `[` and `]`.

b) A non-negative integer literal, followed by a `:`, followed by a list of non-negative integers.

The actions should produce a sequence of tokens for `[`, `]`, `,`, and non-negative integers, converting lists of the second form to those of the first. For example, on input `1:2:[3]`, the output should be

$$\langle \textbf{LB} \rangle, \langle \textbf{num}, 1 \rangle, \langle \textbf{comma} \rangle, \langle \textbf{num}, 2 \rangle, \langle \textbf{comma} \rangle, \langle \textbf{num}, 3 \rangle, \langle \textbf{RB} \rangle$$

Show the output on input `[12,13]4:[16][]`.