CS571 HW3

Total Score : 40

1 Write the RegEx — Total Score: 10

For the following context-free grammars, write a regular expression that matches the same language, if possible. You may assume that the start nonterminal of each grammar is S.

a)

$$S \to (\ A\ )$$
$$A \to [\ B\ ]$$
$$B \to 0$$
$$B \to B\ B$$

b)

$$S \to S\ S$$
$$S \to (\ S\ )$$
$$S \to [\ S\ ]$$
$$S \to 0$$

2 Grammar Ambiguity — Total Score: 20

a) Is the following context-free grammar ambiguous? If so, prove it by providing a sentence that has two derivation/parse trees according to the grammar, and suggest a fix to the grammar that would make it unambiguous without changing the language it recognizes.

b) Would a recursive descent parser be able to recognize this grammar (answer for your corrected grammar if the original grammar was ambiguous)?

$$\begin{aligned}
Term &\to Term + Factor \\
Term &\to Factor \\
Factor &\to Factor * Literal \\
Factor &\to Literal \\
Literal &\to num \\
Literal &\to id\ (\ Params\ ) \\
Literal &\to (\ Term\ ) \\
Params &\to Param \\
Params &\to Params\ ,\ Params \\
Param &\to Term
\end{aligned}$$

You may assume that the literal *num* matches integers according to the RegEx $[0-9]+$ and that the literal *id* matches identifiers according to the RegEx $(\_?)([a-z]|[A-Z])([a-z]|[A-z]|[0-9])*$. You may further assume that the literals "+", "∗", "(", ")", and "," match only the corresponding literal strings. Under these assumptions, this grammar extends our normal expression language to include function calls. For example, note that this language includes the sentence "$1 + 2 * \text{sqrt}(3)$".

3 Syntax-Directed Translation — Total Score: 10

Normally, syntax-directed translation (SDT) rules produce an abstract syntax tree. However, SDT is more powerful than this. For this question, complete the SDT rules in the following grammar to build an *expression parenthesizer*. The parenthesizer should produce a string that adds parentheses to the input expression. For example, the input expression $1 + 5 * 3$ would produce the string "((1) + ((5) * (3)))".

$Term \rightarrow Term + Factor$	{	}
$Term \rightarrow Factor$	{	}
$Factor \rightarrow Factor * Literal$	{	}
$Factor \rightarrow Literal$	{	}
$Literal \rightarrow num$	{	}

More generally, the input expression $1 + 5 * 3$ would produce the following derivation tree under this grammar, and according to your SDT rules must have the depicted values: