Compilers: Theory and Practice Homework I

Due: 09/08/2020 (8 am EDT) (submit via Canvas) Total Points : 100

Guidelines:

- Georgia Tech Honor Code will be enforced.
- Answers should be concise, complete and precise.

1. Rewrite the grammar in the lesson 1 slides, “Parser: Grammar Rules”, to incorporate multiple declarative statements followed by multiple statements. The modified grammar should handle any length legal programs with respect to the above condition. Show only the modified grammar rules. Do not show the unmodified ones. (15 points)
2. Give the state diagrams of the DFAs for the following languages. You can directly give DFA or you can first do a NFA and convert it to DFA. (35 points)
    1. {w | w starts with 1 and has an odd length EXCLUSIVE OR w starts with 0 and has an even length}, alphabet = {0, 1}
    2. {w | w is any string not in b*a* that can be derived from {a, b}}
    3. {w | w is any string that can be derived from the alphabet except 111}, alphabet = {0, 1}
    4. All the strings of lengths 5 or higher that start with the letter ‘b’ and end with a letter ‘d’ or ‘m’. The alphabet is [a-z].
    5. Unsigned integers in which the digits occur in non-decreasing order. For example, 1145 is a legal string, but 121 is not.
3. Consider the following language:
   {w | w ends in “01” or “10”}, alphabet = {0,1}
   In this language, “00010001” and “1111010” are legal strings. “11” and “000” are not legal strings. (26 pts)
    1. Devise a guessing NFA for the language.
    2. Convert the NFA into a minimal DFA. Use Brzozowski’s algorithm from lecture P1L3, “DFA Minimization Parts 1.”
4. For the following languages, determine if they are regular or not. If the language is regular, then implement it using regular expressions. If the language is not regular, then informally show why not. That is, explain what the machine needs to do to recognize the expression and why it cannot do that. (24 points)
    1. L(k) = { a^{k} | $*^{k}$ represents strings made out of a’s such that the number of a’s in them can range from 0 to k, k is a fixed positive integer for a given language L(k)}
    2. L = { a Ω b | Ω represents strings made out of 3 or less a’s followed by 4 or less b’s, a string may contain 0 a’s and/or 0 b’s}
    3. L = { s | strings made out of a’s whose length is a perfect multiple of 6, strings are non-empty}
    4. L = { s | strings made out of a’s whose length is a perfect square, strings are non- empty}