CS 181: Formal Languages and Automata Theory	**Spring 2021**
Homework 3	
Assigned: Tue 4/13	*Due: Mon 4/19, 9:00pm PDT*

Note new time: submission deadline has been extended from 6:00 to 9:00pm PDT.

Problem 1

Carry-over from last week:
Let $\Sigma = \{a, b\}$. Use the Pumping Lemma for the FSLs to show that the following language over Σ is not a FSL:

$$L = \{a^{(2n)}b^n \mid n \geq 0\}$$

Problem 2

Let $\Sigma = \{a, b, c\}$. Show a regular expression which recognizes the following language over Σ:

$$L_2 = \left\{ w \in \Sigma^+ \;\middle|\; |w| > 1, \text{ and the first and last symbols of } w \text{ are different} \right\}$$

E.g.,

- L_2 contains: $ab, ca, abcb$.
- L_2 does not contain: $\epsilon, c, bb, abcba$.

Briefly describe how your regular expression is designed to correctly represent the language.

Problem 3

Let $\Sigma = \{a, b, c\}$. Show an NFA which recognizes the following language over Σ:

$$L_3 = \{w \in \Sigma^* \mid w \text{ contains } abab \text{ or } baba \text{ (or both) and after one of those substrings there is substring } cccc\}$$

ex abbababbbccccabaa

Specify the NFA as a state diagram. It does not have to be fully specified; but if you use the blocking convention, you must use it correctly. *Be sure to clearly indicate your initial state and accepting state(s).* Part of your score will be based on demonstrating effective use of nondeterminism in the NFA model. You do not need to find an NFA with the minimal number of states, but part of your score will be based on avoiding unnecessary complexity in your NFA.

Briefly describe how your design works. Part of your score will also be based on the clarity of your description and how well the state diagram and the description match each other.

Problem 4

Let $\Sigma = \{v, +, -, (,)\}$. Let $e \in \Sigma^*$ be the string $v - v - v$. Let G_4 be the $CFG(V, \Sigma, R, E)$, with variable set $V = \{E, O\}$ and rule set R given by:

$$\begin{aligned} E &\longrightarrow v \mid EOE \mid (E) \\ O &\longrightarrow + \mid - \end{aligned}$$

a. Show a parse tree for e in G_4.

b. Show the corresponding left-most derivation in G_4.

c. Show a *different* parse tree for e in G_4.

d. Show the corresponding left-most derivation in G_4.

Problem 5

For this problem, it may be helpful to think of the symbols "`b`", "`e`", and "`s`" as representing "begin", "end", and "statement", respectively.

Let $\Sigma = \{\texttt{b}, \texttt{e}, \texttt{s}, \texttt{;}\}$. Show a context free grammar (CFG) over Σ for the language, L_5, of "begin-end blocks with the semicolon used as a statement terminator". I.e., every "statement" must be followed by a "`;`", including the outer-most "`b e`" pair. E.g.:

- L_5 contains: `bs;e;` & `bs;s;e;` & `bs;bs;e;e;` & `bbs;e;e;`
- L_5 does not contain: `s;` & `bs;e` & `bs;se;` & `bs;` & `bs;;` & `bs;s;e;e;` & `bs;bs;e;` & `be;` & `be` & `b;e` & `es;b;` & `bs;e;bs;e;`

Problem 6

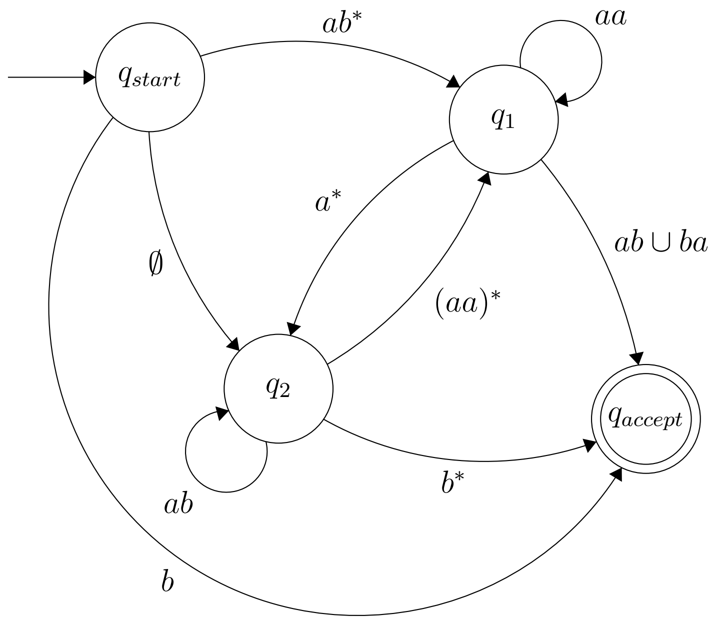
For the this problem, it may be helpful to think of the symbols "`b`", "`e`", and "`s`" as representing "begin", "end", and "statement", respectively.

Let $\Sigma = \{\texttt{b}, \texttt{e}, \texttt{s}, \texttt{,}\}$. Show a context free grammar (CFG) over Σ for the language, L_6, of "begin-end blocks with the comma used as a statement separator". I.e., two consecutive "statements" in a "`b e`" pair must have a "`,`" between them. E.g.:

- L_6 contains: `bse` & `bs,se` & `bs,bs,see` & `bbs,see`
- L_6 does not contain: `s` & `bs,s,e` & `bs,s` & `b,se` & `be` & `bs,se,` & `bs,see` & `bs,bse` & `b,e` & `es,sb` & `bse,bse`

Problem 7

Refer to the GNFA diagram in Figure 1.61 on page 70 of the Sipser textbook, presented here for convenience with names given to all four of the states to make it easier to write your answers:



Show two different ways that the GNFA can accept the string "aaab". For each way, list the sequence of states and transitions and show the portion of the input string matched for each transition.