Written Assignment 1

This assignment asks you to prepare written answers to questions on regular languages and finite automata. Each of the questions has a short answer. You may discuss the assignment with other students and work on the problems together. However, your write-up should be your own individual work.

Please print your name and email address on your homework! We need this information so that we can give you credit for the assignment and so that we can return it to you.

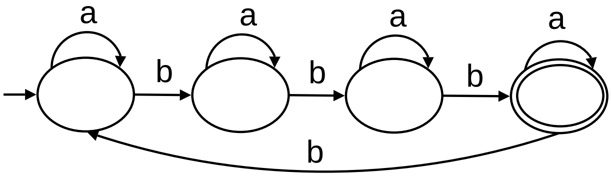
1. Consider the following languages over the alphabet $\Sigma = \{a, b\}$.

   - L_1 : All strings that contain at least three a's.
   - L_2 : All strings that contain at most one b.
   - L_3 : All strings that contain at least three a's but at most one b.
   - L_4 : All strings that contain no b's.

   Give a deterministic finite automaton (DFA) for the languages L_1, L_2, L_3 and L_4.

   Aside: This example illustrates that regular languages are closed under intersection and complementation. Note that $L_3 = L_1 \cap L_2$ and $L_4 = \Sigma^* - L_2$ where Σ^* is the language of all strings over the alphabet Σ.

2. Consider the following DFA over the alphabet $\Sigma = \{a, b\}$.



   Give a one-sentence description of the language recognized by the DFA. Write a regular expression for the same language.

3. Consider the following languages:

- L_1 is all strings over the alphabet $\Sigma = \{x, y\}$ where either x occurs an odd number of times or y occurs an odd number of times (or both).
- L_2 is all strings over the alphabet $\Sigma = \{x, y, z\}$ where either x occurs an odd number of times or y occurs an odd number of times or z occurs an odd number of times (or both, or all three).

Give a non-deterministic finite automaton (NFA) for the languages L_1 and L_2.

Aside: Non-deterministic finite automata are no more powerful than DFAs in terms of the languages they can describe. They can be exponentially more succinct than DFAs, however.

4. Determine whether or not the following languages are regular. Explain why in one or two sentences.

   - L_1 is all strings over the alphabet $\{(,)\}$ where the parentheses are balanced. For example, $(()(())) \in L_1$ but $(() \notin L_1$.
   - L_2 is all words *Programming Language Pragmatics* by Michael L. Scott.
   - L_3 is all 10-digit numbers that are prime.
   - L_4 is the Ocaml language (as described in its reference manual). The alphabet is the set of all tokens and the language is the set of all valid Ocaml programs. L_4 is not regular; give two reasons why. **Aside:** This explains why we cannot use a lexer to *parse* languages like Cool or Ruby or C.

5. Give one advantage and one disadvantage of system described in Backus' *Speedcoding* paper.

Don't forget to print your name and email address on each page of your writeup.