York University
CSE 2001 – Unit 3.0 DFA Machines
and Simple loop Java programs, NFA, Regular Expressions, and Extended Regular Expressions

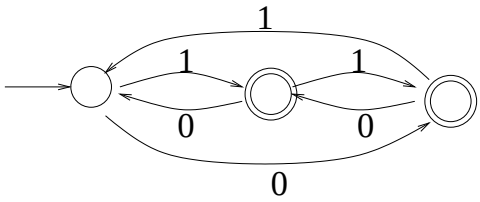
Instructor: Jeff Edmonds

Read Jeff's notes. Read the book. Go to class. Ask lots of question. Study the slides. Work hard on solving these questions on your own. Talk to your friends about it. Talk to Jeff about it. Only after this should you read the posted solutions. Study the solutions. Understand the solutions. Memorize the solutions. The questions on the tests will be different. But the answers will be surprisingly close.

1. Suppose that the CPU of a DFA is required to remember the current value of x and of y where x is a 100 digit number and $y \in \{1, 2, 3, 4, 5\}$. How many states does this DFA require?

2. Give an NFA for the language $L_1 = \{\omega \in \{0,1\}^* \mid \omega \text{ ends in } 00\}$. Also give a regular expression for it.

3. Consider the following DFA



   (a) If the number of 0's in the input string is 28 and the number of 1's is 10, does this DFA: Definitely-Accept Definitely-Reject Depends-on-the-order-of-the-characters

   (b) Denote the language accepted by the above machine with L_2. Describe this language.

4. Let $L = \{\alpha \in \{a,b\}^* \mid \alpha \text{ ends in a block of } a\text{'s and this block has odd length }\}$. For example $\alpha = aabaabaaa$ is accepted because the last block of a's has length three which is odd. However $\alpha = aabaab$ does not end in a block of a's and if you wanted to say that it does then this hypothetical block would have length zero which is even. Either way, this string is rejected. Build a DFA for this.

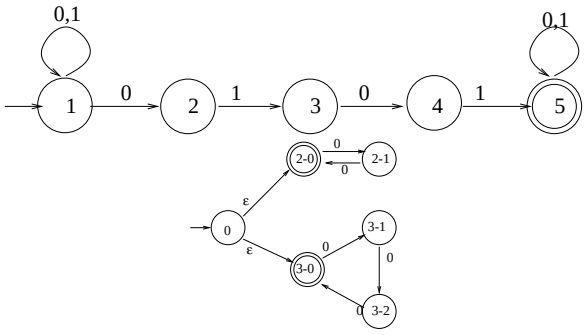
   (a) A DFA is defined as $M = \langle Q, \Sigma, \delta, q_{start}, F\rangle$. For your DFA, what are each of these: Q, Σ, δ, q_{start}, and F?

   (b) For each state, what does the the CPU "knows" about the input string read in so far.

   (c) Give a regular expression of L.

5. Consider a new kind of finite automaton called an *all-paths-NFA*. An all-paths-NFA M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ that recognizes $x \in \Sigma^*$ if *every* possible computation of M on x ends in a state from F. Note, in contrast, that an ordinary NFA accepts a string if *some* computation ends in an accept state.

   (a) Consider the following two machines. What language is accepted if these machines are viewed as all-paths-NFA?



(b) Prove that all-paths-NFA recognize the same class of languages as regular NFA. I.E. Given an all-paths-NFA M, covert it into a regular NFA M' such that $L(M) = L(M')$. Similarly, given a regular NFA M', covert it into an all-paths-NFA M such that $L_{\text{all-paths-NFA}}(M) = L(M')$.

6. One proves $(L_1^* \cup L_2^*) = (L_1 \cup L_2)^*$ by letting L_1 and L_2 be arbitrary yet unspecified languages and letting ω be an arbitrary yet unspecified string. Then you argue that if $\omega \in (L_1^* \cup L_2^*)$, then $\omega \in (L_1 \cup L_2)^*$. Conversely you prove that if $\omega \in (L_1 \cup L_2)^*$, then $\omega \in (L_1^* \cup L_2^*)$.

   One disproves $(L_1^* \cup L_2^*) = (L_1 \cup L_2)^*$ by providing concrete languages L_1 and L_2 and a concrete string ω and either proving that ($\omega \in (L_1^* \cup L_2^*)$ and $\omega \notin (L_1 \cup L_2)^*$) or proving that ($\omega \in (L_1 \cup L_2)^*$ and $\omega \notin (L_1^* \cup L_2^*)$). Your example should be as simple as possible.

   Prove or disprove the following.

   (a) $(L_1^* \cup L_2^*) = (L_1 \cup L_2)^*$.

   (b) $(L_1 \circ L_2)^* = (L_1 \cup L_2)^*$.

   (c) $(L_1 \cup L_2) \circ L_3 = (L_1 \circ L_3) \cup (L_2 \circ L_3)$

7. Every subset of a regular language is regular.