CS 448/2405
Automata and Formal Languages
ASSIGNMENT # 1
DUE DATE: Monday, February 6

1. Give deterministic finite automata accepting the following languages over $\{0, 1\}$.
   a. The set of all strings not containing the substring 101.
   b. The set of all strings of length at least 4, and such that every block of four consecutive symbols contains at least 2 1's.
   c. The set of all strings with at least three symbols such that the third symbol from the right end is 1.

2. Prove of disprove the following for regular expressions r, s and t.
   a. $(rs + r)^* = r(sr + r)^*$
   b. $s(rs + s)^* = rr^*s(rr^*s)^*$
   c. $(r + s)^* = r^* + s^*$

3. Are the following languages regular? Prove or disprove your answer.
   a. $L = \{w \mid w = w^R\}$, where w is a string over $\{0, 1\}^*$.
   b. $L = \{0^n \mid n \text{ is not prime}\}$
   c. $L = \{0^n \mid n \text{ is even}\}$
   d. $L = \{0^n 1^m 0^n \mid m, n \geq 0\}$

4. Problem 1.57 in Sipser. NOTE: This is problem 1.42 in the OLD version of Sipser. The problem is as follows. If A is any language, let $Half(A)$ be the set of all first halves of strings in A. So x is in $Half(A)$ if there is some string y of the same length as x such that the string xy is in A. Prove that if A is regular, then $Half(A)$ is also regular.

5. Let L be any subset of $\{a\}^*$. Prove that L^* is regular.

6. This question concerns the minimization algorithm discussed in class.
   a. Prove that there exists a constant $c > 0$ such that the algorithm requires time greater than cn^2 for infinitely many DFA where n is the number of states and the input alphabet has two symbols.

b. GIve an improved algorithm for minimizing states in a DFA with a smaller runtime. In particular, try to get runtime $O(|\Sigma| n \log n)$ where $|\Sigma|$ is the size of the input alphabet, Σ. Hint: Instead of asking for each pair of states (p, q) and each input a if $\delta(p, a)$ and $\delta(q, a)$ are distinguishable, partition the states into final and nonfinal states. Then refine the partition by considering all states whose next state under some input symbol is in one particular block of the partition. Each time a block is partitioned, refine the partition further by using the smaller subblock. Use list processing to make the algorithm as efficient as possible.