CS314 Spring 2014
Assignment 1
Due Friday, January 31, **before** class

1 Problem — Rewrite Systems

Remember our "game" in the second lecture.

1. Given the same input, i.e., a sequence of characters starting with $ and ending with #, and any combination of 0s and 1s in-between, specify a set of rewrite rules that determine whether the string contains the same number of 0s and 1s.

   Here is some sample "output":

   $0011# should be rewritten as $#

   $1001# should be rewritten as $#

   $110110# should be rewritten as $11#

   $0001100# should be rewritten as $000#

   In other words, the $# indicates that the input string has the same number of 0s and 1s. If the string does not contain the same number of 0s and 1s, the resulting string shows how many more 0s or 1s there are in the input string.

2. Is there at most only a single rewrite rule that can be applied at any point in time during the rewrite process? Explain.

3. Show the steps of your rewrite system for the input strings $0110# and $00010#.

2 Problem — Regular Expressions

Write a regular expression for floating point numbers that you want to use in your new programming language. You should allow numbers of the form `0.34`, `221.5E20`, and `1.0E-5`, but **not** `.12`, `33.`, `21.5E`, or `E30`. Since this is not a total specification of all possible number patterns, make reasonable assumptions for "filling in the holes". Remember: you are a language designer here.

3 Problem — Regular Expressions

Describe the formal languages denoted by the following regular expressions using the English language (e.g.: All strings over the alphabet ... that have the property ...):

1. $((\epsilon \mid 0)\ 1^*)^*$
2. $(0|1)^*0(0|1)(0|1)$
3. $(00|11)^*\ ((01|10)(00|11)^*(01|10)(00|11)^*)^*$

4 Problem — Regular Expressions

Write a regular expression for the following languages.

1. All strings of a's, b's, and c's that contain no b's following any c's.
2. All strings of a's, b's, and c's that do not contain more than 1 b and 3 a's.

5 Problem — Finite State Automaton (FSA)

1. Specify a DFA using a transition diagram and a formal FSA specification <S, s, F, T> (see lecture 2) that recognizes the following language: "All strings of 0's and 1's that, when interpreted as a binary number, are divisible by 4. In other words, value(binary number) modulo 4 = 0."
2. Specify a DFA using a transition diagram and a formal FSA specification <S, s, F, T> (see lecture 2) that recognizes the following language: "All strings of 0's and 1's that, when interpreted as a binary number, are divisible by 3. In other words, value(binary number) modulo 3 = 0."