CS 321
For Thursday Jan 21

Homework #2

1. Consider the following re-write system

$$RRR \equiv \Lambda \qquad FF \equiv \Lambda \qquad RRF \equiv FR.$$

   (a) Show that each string in $\{ F, R \}^*$ is equivalent to one of the following 6 strings:

$$\Lambda, \quad R, \quad RR, \quad F, \quad FR, \quad FRR, \quad .$$

   (b) (HINTs: Use induction.
   Is every string of length $\leq$ 2 equivalent to one of the 6 given strings?
   If a string w has length $n \geq 2$, then
   either $w = Rx$ with length of $x = n - 1$,
   or $w = Fx$ with length of $x = n - 1$.)

   (c) Show that a "MODEL" of this system is the set of Triangles with vertices labeled **A**, **B**, and **C**.
   In this MODEL, how do F and R change the labeling of the vertices of these triangles.

   (d) Show that there is a mechanical procedure (an algorithm) which determines whether or not two strings in this system are equivalent.

   (e) Show that there is a Turing nmachine which can determine whether or not two strings in this system are equivalent. (Informal description – do NOT write out instructions.)

2. **INPUT:** A number x.
   OUTPUT: n, if x is the n^{th} Fibonacci number
   NO, if x is not a Fibonacci number.
   Give an informal argument to show that a Turing machine can compute this function.

3. A **palindrome** is a string which is the same both forward and backward,

   e.g. MADAMIMADAM.

   We want you to show that the set of **palindromes** can be *recognized* by a Turing machine.
   That is,
   INPUT: A string x.
   OUTPUT: YES, if x is a palindrome
   NO, if x is not a palindrome

   Describe (in words) how your Turing machine will operate. What sequence of operations does you machine follow?
   How does your machine tell if x is a palindrome or if x is not a palindrome?

   (You do **NOT** have to write out the instructions for your Turing machine.)

4. A Knight's circuit of an $n \times n$ chessboard starts at some square H on the board, follows knight's moves, visits every square exactly once, and returns to H.

   Show that there is a Knight's circuit of the 6×6 chessboard .

   An **invariant** is a statement which is TRUE for each phase of a computation or construction.
   (In constructing a Knight's circuit the partial sequence of squares you have visited could be considered a "phase".)

   Use the idea of *invariant* to show that there **cannot** be a Knight's circuit of an $n \times n$ when n is **odd** and $n > 1$.
   HINT: LOOK at the COLORS on a checkerboard (see Purina).

5. A **TREE** is a connected graph with no cycles.

   (a) Use Induction to show: if a tree has n vertices then it has $n - 1$ edges.

(b) (HINT: Taking a tree and adding a vertex is **NOT** a valid proof because there could be trees that are NOT constructed in this manner.)

(c) (HINT: Take a tree with $n + 1$ vertices and remove something from it. (You could remove an edge or a vertex.) Then look at what remains, and apply your inductive hypothesis to these remains, and finally reverse the removal and argue that what you've found out about the remains implies that the original graph (with nothing removed) has n edges.)

6. We want the following re-write system to carry out a computation.
{ In the following the rules can be applied anywhere in a string. For example, if an A appears just in front of Z so that the whole string is $xAZy$ then the first rule *may* be applied to give the string $xZ0y$. x or y or both are allowed to be empty strings. }

$$AZ \longrightarrow Z0 \qquad BZ \longrightarrow Z1$$

$$CZ \longrightarrow Z1 \qquad DZ \longrightarrow W0$$

$$AW \longrightarrow Z1 \qquad BW \longrightarrow W0$$

$$CW \longrightarrow W0 \qquad DW \longrightarrow W1$$

$$Z \longrightarrow 0 \qquad W \longrightarrow 1$$

Your job is to assign *meaning* to this formal system.
What should you choose as "axioms" or "starting points"?
What should you choose as the "results" of the computations?
Remember that the "axioms" and "results" should be *easy* to recognize.

HINTS: You might consider strings consisting of any number of A's, B's, C's, and D's followed by a single Z as "INPUT" strings, and strings consisting of only 0's and 1's as "OUTPUT" strings.