Assignment # 1  
(Due February 10, 2014)

1. Write a detailed description of a 1-tape Turing machine that accepts the following language:

   \[ P = \{0^n1^n \mid n \geq 1\}. \]

Some definitions for Questions 2 and 3.

A language \( L \) is *decidable* if there is a Turing machine that always halts and accepts \( L \).

We can also use a Turing machine \( M \) to *compute* a function \( f(x) \), as follows. The Turing machine \( M \) has an input tape (read-only), an output tape (write-only), and a work tape (read/write). With a value \( x \) placed in the input tape, the Turing machine \( M \) runs and eventually halts with the value \( f(x) \) in its output tape.

2. Let \( L_1 \) and \( L_2 \) be two languages. We say that \( L_1 \) is *reducible* to \( L_2 \) if there is a Turing machine that always halts, and on any (yes or no) instance \( x_1 \) of \( L_1 \), produces an instance \( x_2 \) of \( L_2 \) such that \( x_1 \) is a yes-instance of \( L_1 \) if and only if \( x_2 \) is a yes-instance of \( L_2 \).

   Prove: Suppose that \( L_1 \) is reducible to \( L_2 \). If \( L_1 \) is undecidable, then \( L_2 \) is undecidable.

3. Consider the following language:

   \[ \text{Test} = \{(M; x, y) \mid \text{on input } x, \text{ the Turing machine } M \text{ outputs } y\}. \]

   Show that \textit{HALTING} problem is reducible to the problem \textit{Test}.

   **Remark.** Since we already know that \textit{HALTING} is undecidable, this result plus the result in Question 2 shows that automatic testing of program correctness is impossible.