CS 70	Discrete Mathematics and Probability Theory	
Fall 2013	Vazirani	Infinity Practice

1. **Cardinalities**
   Answer the following two questions, giving a proof in each case. You may use without proof any result covered in class provided you state it clearly.
   (a) Is the set of *pairs* of natural numbers countable or uncountable?
   (b) Is the set of irrational numbers countable or uncountable?

2. **Countability**
   (a) You are given an array of n bit strings $a_1, a_2, \ldots, a_n$, each of length n. Show how to construct another bit string b of length n such that $b \neq a_i$ for all $1 \leq i \leq n$ by only looking only at n bits in the array. (Looking at the i-th bit of a_j counts as one.)
   (b) Let X be the set of reals from 0.001 to 0.002. Show by a diagonalization argument that X is uncountable.
   (c) Show that the set of perfect powers is countable. (A perfect power is an integer x which equals d^2 where d is an integer.)

3. **Countable or uncountable?**
   Determine whether the following sets are countable or uncountable. For each countable set, display a one-to-one correspondence between the set of natural numbers and that set, or an enumeration of the set. For each uncountable set, explain why it is uncountable.
   (a) The set of binary strings which are palindromes. A string s is a palindrome if it can be written as the concatenation of some string t followed by the reversal of t.
   (b) The set of real numbers in the interval [0,1] whose decimal representation contains a single "1" digit, and all other digits are "0".
   (c) The set of real numbers in the interval [0,1] whose decimal representation contains only "0" and "1" digits (mixed in any order or combination).
   (d) The set of rooted, finite binary trees, in which trees are distinguished only by their shape (that is, you should ignore node values).

4. **Uncomputable**
   We say that two programs are equivalent if they give the same output on every input. Prove that it is impossible to write a computer program that takes as input two pieces of code, code1 and code2, and tests whether the two programs are equivalent.