CS46, Swarthmore College, Spring 2014
Homework 1 (due Thursday 30 January)

This assignment consists of two parts: a written homework and a programming portion.

Part 1: Written homework

- Please read Sipser Chapter 0 before starting this homework.

1. Let A be a set of finite cardinality. Use induction on the cardinality of A to prove that $|\mathcal{2}^A| = 2^{|A|}$, i.e., prove that the cardinality of the power set of A is two to the cardinality of A.

2. A *natural isomorphism* between sets A and B is a simple bijective function, $f : A \mapsto B$, that can be considered a slight re-writing of elements from either set. For example, let $A = \{a, b, c\}$ and let $B = \{(a), (b), (c)\}$ be a set of singleton tuples. While there are many bijections between A and B, the one that maps $x \in A$ to $(x) \in B$ seems "natural". In addition to the term natural isomorphism, we define for two sets A and B the expression B^A to be the set of all functions from A to B. Let $B = \{0, 1\}$, and let $A = \{a, b, c\}$.

   (a) Give an example of one element of B^A

   (b) Write the power set of A, 2^A.

   (c) Describe a natural isomorphism between B^A and 2^A.

3. Consider an infinite grid of white squares. We could assign each grid square q a label $(i, j), i, j \in \mathbb{Z}$ to indicate the square q is located at column i, row j. On this grid we initially place n garnet squares. We then update every square's color according to the following rule: for a square q at position (i, j) if at least two of the three squares at positions $(i - 1, j), (i, j), (i, j - 1)$, i.e, the squares at q, below q and to the left of q are garnet, we color q garnet. Otherwise we color it white. Prove that if we perform n rounds of this update step, all squares will be white after the final step, regardless of the initial placement of the n squares. Hint: consider the bounding box containing all n garnet squares. What happens to the bounding box over time?

Part 2: Programming Exercise

Write a program that lists all the elements of the power set of n elements in *short lexicographical order*. You may assume that the elements are labeled $a, b, c, \ldots$ and that $n \leq 26$. Your program should take a single command line argument as the parameter n.