HW 3 (due Monday, at noon, February 11, 2013)

CS 473: Fundamental Algorithms, Spring 2013 Version: **1.0**

Make sure that you write the solutions for the problems on separate sheets of paper. Write your name and netid on each sheet.

Collaboration Policy: The homework can be worked in groups of up to 3 students each.

---

1. (50 PTS.) The vampire strike back.

Several years back vampires took over Champaign, IL (this story did not get sufficient coverage in the lamestream media). It turns out that there are several kinds of vampires. The only way to decide if two vampires V_1 and V_2 are of the same kind, is to ask them to bite each other (**biteEachOther**(V_1, V_2)) – if they both get sick they are of different kinds, otherwise, they are of the same kind. This check takes constant time. Specifically, **biteEachOther**(V_1, V_2) returns true if V_1 and V_2 are of the same kind, and false otherwise.

As is usual in such cases, there is one kind that is dominant and form the majority of the Vampires. Given the n vampires $V_1, \ldots, V_n$ living in Champaign, describe an algorithm that uses the **biteEachOther** operation and discovers all the vampires that belongs to the majority type. Formally, $n \geq 4$, and there are at least 3 different kinds of vampires, and you have to discover all the vampires that belong to the kind that has at least $n/2$ members. You have to describe a deterministic algorithm that solves this problem in $O(n \log n)$ time (a faster algorithm is possible, but it is hard). An algorithm that runs in $O(n^2)$ time (or slower) would get at most 25% of the points, but you might want to first verify you know how to do it with this running time bound.

2. (50 PTS.) Liberty towers.

A German mathematician developed a new variant of the Towers of Hanoi game, known in the US literature as the "Liberty Towers" game[1]. Here, there are n disks placed on the first peg, and there are $k \geq 3$ pegs, numbered from 1 to k. You are allowed to move disks only on adjacent pegs (i.e., for peg i to peg $i+1$, or vice versa). Naturally, you are not allowed to put a bigger disk on a smaller disk. Your mission is to move all the disks from the first peg to the last peg.

(A) (15 PTS.) Describe a recursive algorithm for the case $k = 3$. How many moves does your algorithm do?

(B) (15 PTS.) Describe a recursive algorithm for the case $k = n + 1$. How many moves does your algorithm do? (The fewer, the better, naturally. For partial credit, the number of moves has to be at least polynomial in n.)

(C) (20 PTS.) Describe a recursive algorithm for the general case. How many moves does your algorithm do? (The fewer, the better, naturally.) In particular, how many moves does your algorithm do for $n = k^2$? A good solution should yield a solution that is as good as your solution for the (A) and (B) parts.

[1] From Wikipedia: "During World War I, due to concerns the American public would reject a product with a German name, American sauerkraut makers relabeled their product as "Liberty cabbage" for the duration of the war."

This question is somewhat more open ended – we have no specific solution at mind – do your best – but don't spend the rest of your life on this problem.