# CS 473: Fundamental Algorithms, Fall 2011
# HW 0 (due Monday, 23:55:00, August 29 2011)

This homework contains two problems. **Read the instructions for submitting homework on the course webpage.**

Note, that you have to submit your solution online (no paper submission).

**You also have to do quiz 0 online!**

**Collaboration Policy:** For this homework, each student should work independently and write up their own solutions and submit them.

**Read the course policies before starting the homework.**

---

1. (50 pts.) Simple, but really really slow.

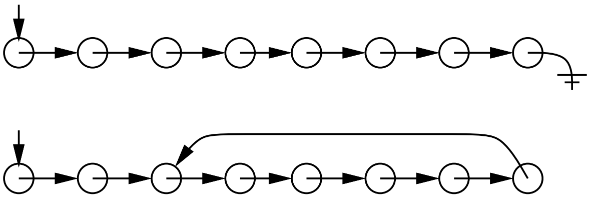
   Consider the following (somewhat bizarre) function.

```
bizzaro($x, y$):
        if $y \le 0$ then
            return $1 + x$
        if $x \le 0$ then
            return $1 + y$
        if $x$ is even and $y$ is even then
            return $2 \cdot$bizzaro($x - 1$, bizzaro($x - 1$, $2y$))+1
        else
            return $2 \cdot$bizzaro($x - 1$, bizzaro($x$, $y - 1$))+2
```

   Prove (formally) via induction that this algorithm always terminates, if given two positive integers as parameters.

2. (50 pts.) Snake or shake?

   Suppose you have a pointer to the head of singly linked list. Normally, each node in the list only has a pointer to the next element, and the last node's pointer is Null. Unfortunately, your list might have been corrupted by a bug in somebody else's code[1], so that the last node has a pointer back to some other node in the list instead.



Top: A standard linked list. Bottom: A corrupted linked list.

[1] After all, *your* code is always completely 100% bug-free. Isn't that right, Mr. Gates?

Describe an algorithm[2] that determines whether the linked list is corrupted or not. Your algorithm must not modify the list. For full credit, your algorithm should run in $O(n)$ time, where $n$ is the number of nodes in the list, and use $O(1)$ extra space (not counting the list itself).

[2] Since you've read the Homework Instructions, you know what the phrase 'describe an algorithm' means. Right?