COMP 2119A Introduction to Data Structures and Algorithms
Test 1
Date: 04 November 2016 (Friday)
Time allowed: 100 minutes (3:30pm – 5:10pm)

[Note that for the questions involving algorithm design, before you present your algorithm, try to describe your idea first if you think that would help the marker to understand your solution! More marks will be given to faster algorithms.]

1) [12%] Prove or disprove each of the followings.
   (i) $20n = O(0.01n^{1.01})$.
   (ii) If $f(n) = \omega(g(n))$, then $\log(f(n)) = \omega(\log(g(n)))$.
   (iii) $O(0.5) + O(n^{0.001}) = O(1000000)$.
   (iv) Let $f(n) = \begin{cases} 10 & n = 1 \\ f(n/2) + n & n > 1 \end{cases}$. $f(n) = O(n)$, where $n = 2^k$ for some integer k.

2) [8%] Assume that we have 27 coins of which one of them is counterfeit and is known to be heavier than others. You are given a balance, which can only tell which side of the balance is heavier or indicate that both sides are of equal weight. Design an algorithm using as few weightings as possible to identify the counterfeit coin.

3) [15%] F(n) ($n \geq 1$) is defined as follows. F(1) = 5, F(2) = 6, and F(n) = F(n-1) + F(n-2) for $n > 2$. Design a recursive algorithm to compute F(n) and analyze the time complexity of your algorithm.

4) [10%] Given a *sorted* array of *distinct* integers $A[1..n]$ in decreasing order, find out whether there is an index i for which $A[i] = i$.

5) [10%] Consider the ADT stack. In addition to the operations, **Push**, **Pop**, **Top**, we have to support a new operation, **FindAve**, which returns the average value of all elements in the stack. Design a data structure to support these operations such that each operation takes constant time. Analyze the time complexity of these four operations. [No need to check the overflow and underflow conditions and no need to give the procedures for **Empty** and **Full**. [Hint: use an extra stack.]

6) [15%] Describe how you can represent an n-vertex directed multi-graph using an $n \times n$ adjacency matrix. Design algorithms to answer the following questions and analyze their worst case time complexities.
   (a) Determine whether a *source* – a vertex with edges going to every other vertex and in-degree = 0 exists.
   (b) Report a pair of vertices (i, j) with the maximum number of edges going from i to j.
   (c) Report the vertex with the maximum number of neighbors..

5) (a) [9%] Consider inserting the keys 30, 52, 61, 35, 45, 58, 47, 118, 89 into an (initially empty) hash table of length $m = 13$ using open addressing with

primary hash function $h'(k) = k \bmod m$. Illustrate the result of inserting these keys using linear probing, using quadratic probing with $c_1 = 1$ and $c_2 = 3$, and using double hashing with $h_2(k) = 1 + (k \bmod (m - 1))$. [You do NOT need to show the intermediate steps.]

(b) [11%] In double hashing ($h(k, i) = (h_1(k) + ih_2(k)) \bmod m$), prove that if $h_2(k) \neq 0$, and $h_2(k)$ is relatively prime to m, then the probe sequence is a permutation of $[0..m\text{-}1]$.

6) [10%] Show how to use two queues to implement a stack and analyze the time complexities of your algorithms.