1) Given a (possibly empty) singly linked list of integers in nondecreasing order, write a function that creates another singly linked list that is a copy of the original list, but with all duplicate entries removed. This new list is to be in increasing order. Runtime must be linear.

2) Which of the following five statements correctly describes the relationship between the functions $f$ and $g$ defined in a)-d) below? Note that more than one of the five statements may be correct for each part. You do not need to explain your choices.

$$ f \in o(g) \quad f \in O(g) \quad f \in \theta(g) \quad g \in o(f) \quad g \in O(f) $$

a) $f(n) = n!$, \quad $g(n) = (n + 1)!$

b) $f(n) = n^2$, \quad $g(n) = n(log_2 n)^2 + 5$

c) $f(n) = 257n + 83\sqrt{n}$, \quad $g(n) = (log_2 n) + \frac{1}{6}n$

d) $f(n) = \begin{cases} n, & n \text{ odd} \\ n^2, & n \text{ even} \end{cases}$, \quad $g(n) = \frac{1}{3}n^2$

3) Let $G$ be a directed graph with vertices $v[0], \ldots, v[n-1]$ and $m$ edges. The edges are stored using the adjacency matrix implementation.

a) Write an algorithm that prints the vertex number of each vertex that is a “grandchild” of $v[0]$. That is, print $i$ if $v[0]-v[i]$ is not an edge of $G$, but there is a vertex $u$ such that $v[0]-u$ and $u-v[i]$ are both edges of $G$. Also, $v[0]$ is not considered to be its own grandchild. Your algorithm should only examine $v[0]$’s edges and the edges involving $v[0]$’s children. No vertex number should be printed more than once.

For any local data structure you use (such as stacks, queues, etc.), you may assume that the basic operations already exist (and so you don’t need to write the code for pop, enqueue, etc.).

b) State in big-theta terms the worst case runtime of your routine as a function of $n$ and/or $m$.

c) Suppose the graph had been given as adjacency lists. For this implementation, state in big-theta terms the worst case runtime of the routine as a function of $n$ and/or $m$.

For the example on the following page, the grandchildren of $v_0$ are $v_1$ and $v_4$. 