CSC 225 Midterm Exam

June 22, 2001

Recall that you need at least 40% (40/100) in order to write the final exam in this course.

1. [20] Assume $n = 2^k - 1$ for some integer k. Solve the following recurrence using repeated substitution: $T(n) = n + 2\,T(\frac{n-1}{2})$, $T(7) = 5$.

2. [20] Prove by induction that your solution to question #1 is correct. Or for part marks [10], apply induction to the point where you realize that your solution to #1 is incorrect, and explain what goes wrong.

   The recurrence from Question #1: $T(n) = n + 2\,T(\frac{n-1}{2})$, $T(7) = 5$. You may assume that $n = 2^k - 1$ for some integer k.

3. Circle **True** or **False** and justify your answer. **No marks will be given unless there is a correct justification.**

   (a) [5] Let a_0, a_1, a_2, and a_3 be integers where $a_i > 0$ for all $i = 0, 1, 2, 3$. Then $p(n) = a_0 + a_1 n + a_2 n^2 + a_3 n^3$ is in $\Theta(n^3)$.
   True **False**

   (b) [5] An algorithm for sorting n numbers which is $O(n \log n)$ in the worst case is always faster than an algorithm which is $O(n^2)$ in the worst case.
   True **False**

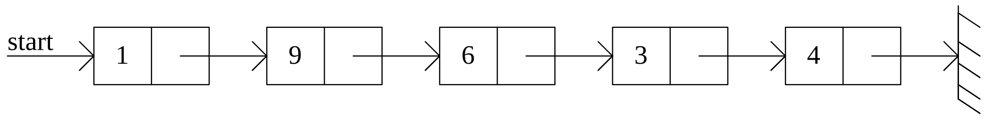
   (c) [5] It is possible to sort an array of n numbers in $O(n \log n)$ time in the worst case.
   True False

   (d) [5] Let f, g, and h be functions from the natural numbers to the positive real numbers. Then if $g \in \Omega(f)$ and $g \in O(h)$, and $f \in O(h)$ then $g \in \Theta(h)$.
   True **False**

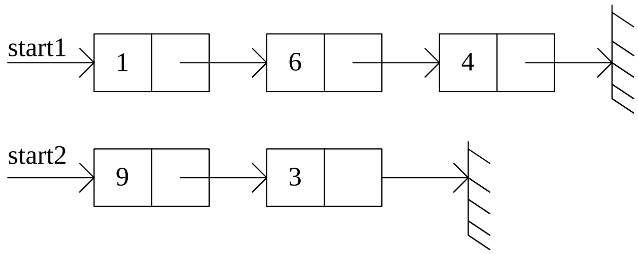
4. [20] Suppose we are given as input two linked lists L_1 (with $n1$ nodes, and start/rear pointers $start1$ and $rear1$) and L_2 (with $n2$ nodes, and start/rear pointers $start2$ and $rear2$). The objective is to create a new linked list L with n nodes, and start/rear pointers $start$ and $rear$ such that L is L_1 concatenated with L_2. Give detailed pseudocode for an algorithm for this that takes $O(1)$

time. Marks will be deducted for correct solutions that are more complex than necessary.

5. [20] Give pseudocode for an **iterative** *divide*/*split* function for merge sort which takes as input a linked list L starting at *start*, and returns as output two lists L_1 and L_2 starting at *start*1 and *start*2 respectively. The algorithm should work by placing the first cell from L on L_1, then it should place the second cell from L on L_2, then the third cell from L goes on L_1, and the fourth cell from L goes on L_2, and so on. For example, if the input is:



The output should be:



Be sure to include lots of comments in your pseudocode.