University of Nevada, Las Vegas Computer Science 477/677 Fall 2019

Additional problems to study for the examination of September 18, 2019

Name:____

1. In each of the following situations, write O, Ω. Θ in the blank.

   (a) $\log(n!) = \_\_\_\_\_\_ (n \log n)$

   (b) $\log^2 n = \_\_\_\_\_\_ (\log \log n)$

2. Make sure you fully understand each of the following sorting algorithms.

   (a) Bubblesort.
   (b) Selection sort.
   (c) Insertion sort.
   (d) Bucket sort.
   (e) Mergesort.
   (f) Polyphase mergesort.
   (g) Quicksort.
   (h) Treesort. (Using a binary search tree.)

3. Fill in each blank.

   (a) Every comparison–based algorithm for sorting n items must make _______________ comparisons in the worst case. (Asymptotic answer.)
   (b) Of the sorting algorithms listed in problem 2, ________________________, ________________________, and ________________________ take quadratic time.

4. *Monte Carlo* algorithms and *Las Vegas* algorithms are two classes of randomized algorithms. Which of these classes does each of the following belong to?

   (a) Randomized quicksort (pick the pivot at random).
   (b) Randomized primality testing (as described in the textbook).
   (c) Randomized median finding (as described in the textbook).
   (d) The following algorithm for finding the region with the larger area. You are given two regions in the unit square, A and B. You know that the their areas differ by at least 0.01, but you do not know which has the larger area. The regions are defined in such a way that you are unable to calculate their areas using calculus. The randomized algorithm is as follows. Pick N points at random in the square, where N is a large number. Calculate a, the number of those points in A, and b, the number of those points in B. If $a > b$, you state that A has a larger area than B, otherwise you state that B has the larger area.

   Also, how large do you think N be?

5. Work problem 1.12 on page 39 of the textbook.

6. Work problem 1.13 on page 39 of the textbook.

7. Work problem 1.26 on page 40 of the textbook.

8. Work problem 1.39 on page 43 of the textbook.

9. Work problem 2.20 on page 74 of the textbook.

10. (a) Write a dynamic program to solve the maximal increasing subsquence problem: given a sequence $A = (A_1, A_2, \ldots A_n)$, of numbers, find the maximum length of any strictly increasing subsequence of A.

    For example, if $A = (5, 3, 8, 2, 6, 7, 0, 9, 4)$ then the maximum length of an increasing subsequence is 4. There are two increasing subsequences of length 4, (3,6,7,9) and (2,6,7,9).

    If $A = (5, 1, 6, 2, 7, 3, 8, 4)$ the answer is again 4, and the maximal increasing subsequences are (1,2,3,4) and (5,6,7,8).

    I don't mean that you should write C++ code. Just outline the steps of the algorithm, and be sure to make clear which data structures you use.

    (b) Modify the program above so that it returns the number of maximal increasing subsequences. For example, if $A = (1, 2, 3, 4, 5, 6, 7, 8)$ that number is 1. If $A = (5, 3, 8, 2, 6, 7, 0, 9, 4)$ that number is 2. If $A = (4, 3, 2, 1, 8, 7, 6, 5, 12, 11, 10, 9, 16, 15, 14, 13)$ that number is 256.