Name ____________________

CS302 - Midterm 1 Practice Problems

Exam Instructions

1. The exam is xxx pages. Make sure you have all xxx pages.
2. This exam is closed book, closed notes, closed internet, and closed person EXCEPT for a two-sided, 8.5×11 piece of paper ("cheat sheet"). The content on this cheat sheet must be curated/created by you (no miniaturizing of my notes/psets/quizzes/solutions). Please turn in with your exam.
3. You have 3 hours to take the exam. Please mark your starting and ending time below.
4. You have 3.5 hours from the time you pick up the exam until the time you should return it to my office (slide under door). Sign out when you pick up the exam. Do *not* sign in when you turn it in.
5. You may use extra scratch paper, and you do not need to turn it in, but if you do, make it clear what is scratch work and what is your solution.

Please state and sign your pledge to keep the honor code PRIOR to starting the exam:

____

____

____

Time started: ____________ Time finished: ____________

1. All of the following questions are regarding the closest points in 2D algorithm.

    (a) What if, in the combine step, we looked at a region within 2δ of the midline. Could the algorithm still work? Would you have to change anything else to compensate?

    (b) What if we looked at a region within $\delta/2$ of the midline. Could the algorithm still work? Would you have to change anything else to compensate?

    (c) Why do we need to maintain separate arrays sorted by X and Y coordinates?

2. (a) Prove the following algorithm is correct:

Algorithm 1: `Maximum`(A)

Input : Array A of unique integers of size n.
Output: Maximum value in array.

if n *equals* 1 **then**
  return $A[1]$;
end
$mid = n/2$;
$m_1 =$`Maximum`$(A[1 : mid]$;
$m_2 =$`Maximum`$(A[mid + 1, n)$;
return $\max\{m_1, m_2\}$;

    (b) What is the runtime of the algorithm?

3. Suppose you have a graph T that is a binary tree, with weights on each vertex. Let T_v be the subtree with root at vertex v. Let $S(T_v)$ be the max-weight-independent set of T_v and let $W(T_v)$ be the weight of the max-weight independent set on T_v. We'll design a dynamic programming algorithm for this problem.

    (a) What are the options for $S(T_v)$ in terms of the vertex v?

    (b) For each option, write a recurrence relation for $S(T_v)$ in terms of max-weight-independent sets of subtrees of T_v.

    (c) Use this analysis describe in words (or write pseudocode) how to create a function that fills an array with the values $W(T_v)$ for each v.

4. Explain the idea of an exchange argument proof (the simple, single exchange version) in your own words.