Quiz/Homework Assignment #4
CSE326
Homework due date:
Wednesday, February 16th, beginning of class

February 11, 2000

This quiz is homework assignment #4. Any question you answered correctly in class are considered correct for the purposes of the assignment. Any question you did not answer or which you answered incorrectly is now a short answer question for the homework assignment due in class on Wednesday. We will hand back the graded quiz on Friday.

In *italics* are instructions on how to answer each question for homework. The correct letter answer is listed after each question.

Good luck!

1. *Describe the (most precise) asymptotic relationship relationship between each pair. Explain why (a) is the answer and why each of (b)-(e) breaks one of the conditions.*

   In which of the following is

   - $f(n) \in o(n)$
   - $g(n) \in \omega(n \log n)$ and
   - $f(n) > g(n)$ for some finite $n \geq 1$

   (a) $f(n) = \sqrt{4n}$, $g(n) = n^2$
   (b) $f(n) = n$, $g(n) = n \log n$
   (c) $f(n) = 2n$, $g(n) = n \log n$
   (d) $f(n) = 3$, $g(n) = 16^n$
   (e) $f(n) = 10$, $g(n) = 8$

   Correct answer(s): **(a)**

2. *Explain why none of (b)-(d) actually is a flaw. In terms of the definition of Ω time, explain why (a) is a flaw.*

   The following is a *flawed* best case analysis of merging skew heaps in terms of n, the total size of the heaps. Which option below describes the flaw?

   > If $n = 4$ and both heaps are leftist heaps with half of the elements, neither one's root will have a right child. Therefore, the merge will take $\Omega(1)$ time.

   (a) The analysis does not generalize to arbitrary n.
   (b) The analysis does not generalize for different sizes of each heap with the same n.
   (c) Best case analyses derive tight bounds and upper bounds, not lower bounds
   (d) The analysis improperly assumes that the skew heaps are leftist heaps.
   (e) None of the above. The analysis is correct.

   Correct answer(s): **(a)**

3. *Give the time required by the naive version of each data structure's "build" operation.* **Briefly** *explain and give the running time of the best "build" operation for each data structure.*

   We can build each of the following structures from a set of initial data by repeatedly calling insert until the data structure is complete, taking $O(n)$ times the time for an insert.

However, we can do better with several of these data structures. Which of the following *cannot* be built asymptotically faster than the naive method described here?

Hint: remember that any binary heap is a leftist heap!

(a) AVL tree
(b) Hash table
(c) Splay tree
(d) Skew heap
(e) Leftist heap

Correct answer(s): **(b)**

4. *Explain what properties each type of tree/heap does ensure. Explain why those properties result in a nearly balanced tree for (a), (b), and (d) and why they do not for (c).*

   Which of these does *not* ensure a balanced or nearly balanced tree?

   (a) binary heap
   (b) d-Heap
   (c) leftist heap
   (d) AVL Tree
   (e) All of these ensure at least a nearly balanced tree

   Correct answer(s): **(c)**

5. *Give an example of a tree for which the given testing method would succeed but the tree is still not a BST. Explain what testing method is needed instead.*

   Can we ensure that a tree is a Binary Search Tree just by checking that every node's left child has a key less than the node's key and every node's right child has a key greater than the node's key? (Choose the most general answer.)

   (a) Yes, always.
   (b) No, not for any type of BST.
   (c) No, not if it's also an AVL tree.
   (d) No, not if it's also a splay tree.
   (e) No, not on Wednesday.

   Correct answer(s): **(b)**

6. *Review heaps and the heap-order property! Explain why all of these have the heap-order property (that is, what advantage do they gain from having heap-order?).*

   Which of these heaps enforces the property that a node's children's keys must have larger priority values than the node's key?

   (a) Binary heap
   (b) Leftist heap
   (c) Skew heap
   (d) d-Heap
   (e) All of the above

   Correct answer(s): **(e)**

7. *List how long (asymptotically) merge does take for each of these. Explain why the splay tree join operation cannot be used to support general merging.*

   Which of the following data structures supports a worst-case merge in $O(\log n)$ time?

   (a) Hash table
   (b) Binary search tree
   (c) Splay tree
   (d) AVL tree
   (e) None of these

   Correct answer(s): **(e)**

8. *Explain how find minimum must be performed in a hash table. What's another operation that performs (similarly) poorly on hash tables?*

   Assuming we have a comparator for keys, hash tables support the find minimum operation in:

   (a) $O(1)$
   (b) $o(\log n)$
   (c) $\Theta(\log n)$

(d) $O(n + tableSize)$

(e) Not at all. This operation cannot be performed on a hash table.

Correct answer(s): **(d)**

9. *Explain why open hashing is always better for an insert under the condition in (c) if resizing is not allowed. Explain how open hashing could be worse for the conditions in (a) and (b).*

   An insert into an open hashing table (without resizing) is always better than an insert into a closed hashing table (without resizing) under which of the following conditions:

   (a) $0 \leq \lambda < \frac{1}{2}$

   (b) $\frac{1}{2} \leq \lambda < 1$

   (c) $1 \leq \lambda$

   (d) Open hashing is always better than closed hashing.

   (e) Open hashing is never better than closed hashing.

   Correct answer(s): **(c)**