Quiz/Homework Assignment #5
CSE326
Homework due date:
Friday, February $25^{\textbf{th}}$, beginning of class

February 19, 2000

This quiz is homework assignment #5. 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. *Explain what happens in an extendible hash table when many keys hash to similar hash values. Why won't this (necessarily) happen when many similar keys are inserted?*

   Which of the following would definitely cause an unnecessarily large directory in extendible hashing?

   (a) Many keys hash to similar hash values
   (b) Many similar keys are inserted
   (c) Keys hash evenly throughout the table
   (d) Values take up little disk space
   (e) Hash function is slow to compute

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

Use the following scenario for the next three questions.

You have the set of elements a-j and perform the following sequence of operations:

- find(d)
- union(d, a)
- union(b, c)
- union(h, j)
- find(c)
- union(h, b)
- find(j)
- union(b, a)

2. *Draw the up-trees for each step (you can just draw the parts that change if you make the pictures clear).*

   Without weighted union or path compression and choosing the root of a union by selecting the alphabetically smaller node, node `j` ends up at depth:

   (a) 0
   (b) 1
   (c) 2
   (d) 3
   (e) 4

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

3. *Draw the up-trees for each step (you can just draw the parts that change if you make the pictures clear).*

   With weighted union but without path compression and breaking ties on unions by selecting the alphabetically smaller node, node `j` ends up at depth:

(a) 0
(b) 1
(c) 2
(d) 3
(e) 4

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

4. *Draw the up-trees for each step (you can just draw the parts that change if you make the pictures clear).*

   With weighted union and path compression and breaking ties on unions by selecting the alphabetically smaller node, node `j` ends up at depth:

   (a) 0
   (b) 1
   (c) 2
   (d) 3
   (e) 4

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

5. *What operations do up-trees support? How long would it take to find an item by its key in an up-tree?*

   Which of the following does not support efficient search?

   (a) Extendible Hash Tables
   (b) Up-Trees
   (c) Treaps
   (d) Splay Trees
   (e) AVL Trees

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

Match each of the following five data structures with the best description of when you would use that data structure.

For each one that you missed, explain how it supports the desired behavior.

6. Extendible hash table

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

7. Disjoint set Union/Find Up-trees

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

8. Binomial Queue

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

9. Treap

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

10. C and D Tree

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

(a) I want to investigate an equivalence relationship among some items.
(b) I like to merge priority queues. It makes me smile.
(c) I want a dictionary data structure for large amounts of input.
(d) I want a quick to implement dictionary data structure with (usually) reasonable performance.
(e) I have no idea. You're making C and D Trees up.

11. *Why is the worst case bounded by anything? (Why isn't it infinite?) Why is the worst case $O(n)$? What is the best case for performing all the union operations that can be performed with n elements?*

    What's the worst case *total* cost of all the union operations that can ever be performed on an Up-Tree Disjoint Set Union/Find structure with n elements using weighted union and path compression?

    (a) $O(1)$
    (b) $O(n)$
    (c) $O(n\alpha(n, n))$
    (d) $O(n \log n)$
    (e) Unbounded ($O(\infty)$)

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