CSE 326 DATA STRUCTURES
HOMEWORK 6

Due: **Friday, Feb 26, 2010** at the beginning of class. Please put your quiz section in addition to your name at the top of your homework.

Problem 1. Unions

Show the result of the following sequence of instructions:

```
hw6-union(3,4), hw6-union(3,5), hw6-union(2,1), hw6-union(1,7),
hw6-union(6,3), hw6-union(9,8), hw6-union(14,15), hw6-union(1,8),
hw6-union(3,10), hw6-union(11,3), hw6-union(3,12), hw6-union(3,13),
hw6-union(1,14), hw6-union(16,0), hw6-union(14,16), hw6-union(1,3),
```

Where `hw6-union`(x,y) differs from normal union by not assuming that x and y are roots. `hw6-union` will first do a `find` operation on the parameters x and y to find their roots, and will then do a regular union on those two roots as follows:

```
hw6-union(x, y) {
  u = find(x);
  v = find(y);
  union(u, v);
}
```

Use figures such as those on page 296 of Weiss. As usual, showing some steps will allow for the possibility of partial credit. If the two trees containing x and y have the same size/weight (b), or height (c), make the root of the set containing x the root.

(a) Perform all hw6-union(x, y) operations by making the root of the tree containing x the root. (In otherwords, the root of the tree containing the first parameter should be the root of the new tree.)

(b) Perform all the regular union calls by size/weight.

(c) Perform all the regular union calls by height.

(d) For the tree you constructed in part a), do a find with path compression on the deepest node. If there is more than one "deepest node" do a find on the one with the smallest key value.

Problem 2. Maze Creation

Design an algorithm that generates a maze that contains NO path from start to finish, but has the property that the removal of a *prespecified* wall creates a unique path. (Hint: You may use the pseudocode given to you in the slides from class. If you do, you'll only need to add roughly two lines of pseudocode and change one existing line.)

[Note: edited 2/24/2010] Equivalently, define a method, `maze(Cell` a`, Cell` b`)`, that computes a maze with a unique path from a start to an end (you can select start and end) only if you remove the *prespecified* wall, (a,b).

Note that you can, if you wish, wait until after your algorithm has completed to declare which cells are the start and end cells. If you choose to follow this idea, you don't need specify the selection of start and

end cells in your algorithm, but explain how they could be selected after your pseudocode. (You should use pseudocode similar to what is used on the slides from class to describe maze generation.)

Problem 3. Deunion

Suppose we want to add an extra operation, `deunion`, which undoes the last `union` operation that has not been already undone.

(a) Show that if we do union-by-height and `finds` without path compression, then `deunion` is easy and a sequence of M `union`, `find`, and `deunion` operations takes $O(M \log N)$ time.

(b) Why does path compression make `deunion` hard?