CMSC 420:Fall 2019 Dave Mount

Midterm Exam 1

This exam is closed-book and closed-notes. You may use one sheet of notes (front and back). Write all answers on the exam paper. You may use any algorithms or results given in class. If you have a question, either raise your hand or come to the front of class. Total point value is 100 points. Good luck!

Your Name:	StudentID:

I pledge on my honor that I have not given or received any unauthorized assistance on this examination.

Your Signature:

Problem 1. (40 points) Short answer questions. Except where noted, explanations are not required but may be given for partial credit.

(1.1) (5 pts) Given an *extended binary tree* with n *internal nodes*, how many *total nodes* (both internal and external) does this tree have?

(1.2) (5 pts) When deleting a key from a standard (unbalanced) binary search trees, we mentioned that the *replacement node* may either be the inorder successor or inorder predecessor of the deleted node. What is the danger of using one of these *exclusively* (say, the inorder successor) over choosing *randomly* between these two options?

(1.3) (5 pts) When deleting a key from a *treap*, is it advantageous to *randomize* the choice of the replacement node (as in (1.2)), or does it not matter? **Explain briefly.**

(1.4) (5 pts) You are given a 2-3 tree of height h. As a function of h, what is the *minimum* number of *red nodes* that might appear on any path from a root to a leaf node in corresponding AA tree? What is the *maximum* number?

Minimum	Maximum

(1.5) (10 pts) Recall that in a skip list, each node at some level is promoted to the next higher level with probability 1/2. We showed that the expected number of hops made at each level of the skip list is 2. Suppose that we change the probability from 1/2 to some value p, where $0 < p < 1$. Will the expected number of hops per level change or remain the same? If it changes, what is the new expected value? **Explain briefly.**

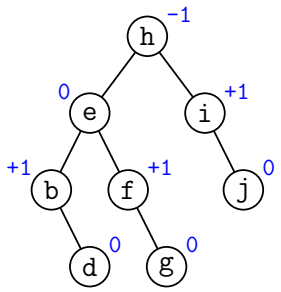
(1.6) (10 pts) **True or false?** In every extended binary tree having n external nodes, there exists an external node of depth at most $\lceil \lg n \rceil$. **Explain briefly.**

Recall that the *depth* of a node in a rooted tree is the length (number of edges) from the root to this node.

Your Initials: _____

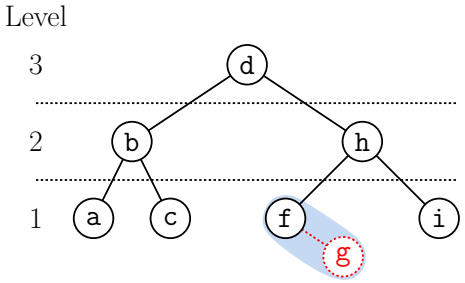
Problem 2. (20 points)

(2.1) (10 points) Consider the AVL tree in the figure below. (We have labeled each node with its balance factor.) Show the result of performing `insert("c")` on this tree.



Show both the **final tree** and the **balance factors** for all the nodes. (Intermediate results can be given for partial credit.)

(2.2) (10 points) Consider the AA tree in the figure below. (Note that all the nodes are black except for `g`, which is red). Show the result of performing `insert("e")` on this tree.



Show both the **final tree** and indicate the **levels**. (Intermediate results can be given for partial credit.)

Your Initials: _____

Problem 3. (10 points) Suppose that in addition to the key-value pair (`p.key` and `p.value`) and pointers to the node's left and right children (`p.left` and `p.right`), each node of a binary search tree stores a sibling pointer, `p.sibling`, which points to `p`'s sibling, or `null` if `p` has no sibling.

Recall the following code for performing a right rotation of a node in a binary search tree:

```
BinaryNode rotateRight(BinaryNode p) {
    BinaryNode q = p.left;
    p.left = q.right;
    q.right = p;
    return q;
}
```

Modify the above code so that, in addition to performing the rotation, the sibling pointers are also updated. (You may *not* assume the existence of other information, such as parent pointers or threads. Your function should run in constant time.)

Problem 4. (10 points) You are given a binary search tree where, in addition to the usual fields `p.key`, `p.left`, and `p.right`, each node `p` has a *parent link*, `p.parent`. This points to `p`'s parent, and is `null` if `p` is the root. Given such a tree, present pseudo-code for a function

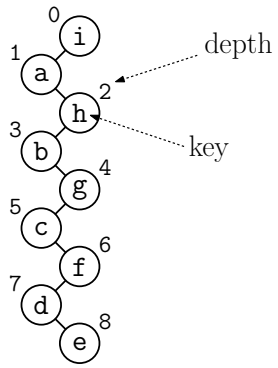
`BinaryNode preorderSuccessor(BinaryNode p)`,

which is given a non-null reference `p` to a node of the tree and returns a pointer to `p`'s *preorder successor* in the tree (or `null` if `p` has no preorder successor). Your function should run in time proportional to the height of the tree. Briefly explain how your function works.

Your Initials: _____

Problem 5. (25 points) A *zig-zag tree* is defined to be a binary search tree having an odd number of nodes that consists of a single path, alternating between right- and left-child links. An example is shown in the figure below, where we have also labeled each node with its *depth*, that is, the length of the path from the root.

(5.1) (5 points) Draw the final tree that results from executing `splay("e")` on the tree of the figure below. (Intermediate results can be given for partial credit.)



(5.2) (10 points) Let T be a zig-zag tree with n nodes, and let T' be the tree that results after performing a splay operation on T's deepest leaf. Consider a node p at level k in T, for $0 \leq k \leq n-2$. What is the depth of p in T'? Express your answer as a function of k and n. Your formula should apply to every node of the tree, *except the node that was splayed*.

(5.3) (10 points) Give a short proof justifying the correctness of your formula. (**Hint:** Think about how a node's depth changes during each zig-zag rotation.)

Your Initials: _____

Scratch-work area: