CSE 373 Spring 2013: Midterm Practice

1. Mathematical Foundations:

   (a) Prove by mathematical induction on the length n that the following procedure to determine the length of a linked list returns the correct length. Each node has a data field and a next field that points to the next node.

   ```
   length(L: listptr): integer {
      if L == null return 0;
      else return 1 + length(L.next);
   }
   ```

   (b) Prove by mathematical induction on the height h that the following procedure to determine the height of a binary tree returns the correct height. Each node has a data field, a left pointer, and a right pointer. Start with height -1 for the basis.

   ```
   height(T: treeptr): integer  {
      if T == null return -1;
      else return 1 + max(height(T.left),height(T.right));
   }
   ```

2. Complexity

   (a) What is the complexity of inserting a new element in a stack of n elements implemented as a linked list?

   (b) What is the complexity of deleting the first element from a queue of length m implemented as a circular array?

   (c) What is the complexity of deleting a node from a binary search tree?

   (d) Analyze the complexity of the following code, first computing T(n), the giving the result in Big-Oh notation.

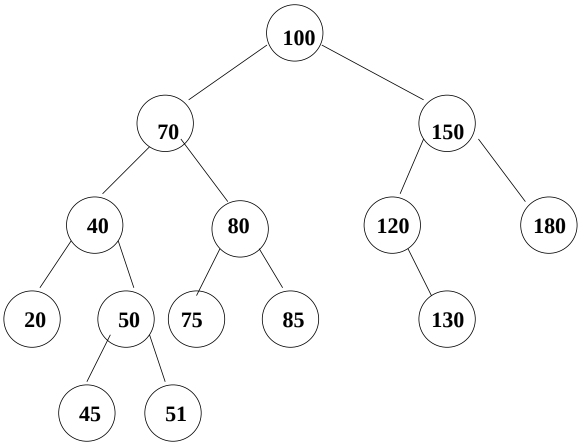
   ```
   result = 5;
   for i := 1 to n {
      result = result - 1;
      if result < 0 break;
   }
   print result;
   ```

3. Lists, Stacks, and Queues

   (a) A linked stack has nodes with fields `data` and `next`. The pointer `top` points to the top element of the stack. Write a function named `pop` that removes the top element from the stack, returns the value in its `data` field, and does so in O(1) time. You can write in pseudocode or Java.

4. Trees

   (a) Write a recursive procedure to find the maximum value in an AVl tree.

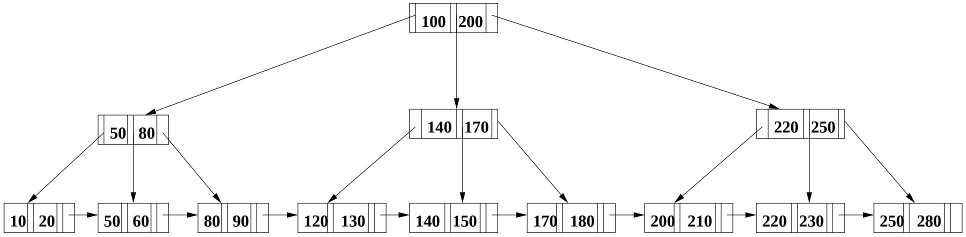


(b) You are given the above binary tree T.

a. (12 pts) Show how to insert a new node with key 41 if T is an AVL tree. (Show each step.)

b. (12 pts) Show how the original T would be reorganized after a `find(130,T)` operation if the tree is a SPLAY tree. (Show each step.)

(c) (13 pts) Suppose that you started with a B^+ tree of order 3 (with a maximum of 2 keys per internal node and 2 keys per leaf node) that looks like this:



and you want to insert a key of 25.

Show how this insertion would be done and the resultant B^+ tree.

5. Hashing

(a) What does the term “rehashing” mean?

(b) What is the worst-case complexity of storing a key in a hash table if the array is very full?

6. General: Be able to give short answer to questions about the structures and concepts we have covered.

(a) What is the advantage of a linked list over a list stored in an array? What is the disadvantage?

(b) Are splay trees always balanced? Why or why not?