Data Structures: cs271
Denison University
Professor Thomas C. Bressoud

Jan. 26, 2012
Homework Set 2 DRAFT

Homework Set 2 DRAFT

This problem set is due **Wednesday February 8**
Analysis (in hard copy) by **class time**; Programming by **11:59PM**.
Solutions should be submitted in PDF form using LaTeX.
Remember, your goal is to communicate. Full credit will be given only to the correct solution which is described clearly. Convoluted and obtuse descriptions might receive low marks, even when they are correct. Also, aim for concise solutions, as it will save you time spent on write-ups, and also help you conceptualize the key idea of the problem. Demonstration of your work/reasoning can result in partial credit.

---

1. Prove Theorem 3.1 on page 48:

   Theorem 1 (3.1)**.** *For any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.*

2. Show that if $d(n)$ is $O(f(n))$, and $e(n)$ is $O(g(n))$, then the product $d(n)e(n)$ is $O(f(n)g(n))$.

3. Argue the following using the definitions of O, Ω, and Θ.

   (a) $2^{n+1} = O(2^n)$

   (b) $\ln n = \Theta(\log_2 n)$

   (c) $n^\epsilon = \Omega(\lg n)$ for any $\epsilon > 0$

4. Read Section B.5 (in Appendix B of CLRS). Use induction to show that a nonempty binary tree with n nodes has height at least $\lfloor \lg n \rfloor$.

5. Use induction to show that a complete binary tree with height h contains 2^{h+1} - 1 nodes.

6. Consider a binary search tree T whose keys are distinct. Prove that if the right subtree of a node x in T is empty and x has a successor y in T, then y is the lowest ancestor of x whose left child is also an ancestor of x. (Recall that every node is its own ancestor.)

7. Show that if a node in a binary search tree has two children, then its successor has no left child and its predecessor has no right child.

8. Implement a Binary Search Tree of integers. Your binary search tree must support the following public operations, where `T` is a `typedef` for an `int`, or is a template type parameter.

```
typedef int T;

// Binary Tree Node
class BTNode {
 public:
  BTNode() : key(0), parent(NULL), left(NULL), right(NULL) {}
  BTNode(const T & e1, BTNode *p = NULL, BTNode *l = NULL, BTNode *r = NULL) :
  key(e1), parent(p), left(l), right(r) {}
  T key;
  BTNode * parent;
  BTNode * left;
  BTNode * right;
};

// Binary Search Tree
class BST {
 public:
  BST() : root(NULL) {}                // Constructor
  ~BST();                              // Destructor
  void Clear();                        // Remove all nodes from BST
  void Insert (const T & el);          // Insert element el into BST
  void Delete (const T & el);          // Find and remove node with key el
  void DeleteNode (BTNode * & node);   // Remove BTNode node from BST
  bool Empty() const;                  // Is BST empty?
  BTNode * Search(const T & el) const; // Search for BTNode with key el
  T * Minimum() const;                 // Return pointer to min key
  T * Maximum() const;                 // Return pointer to max key
  BTNode * Successor(BTNode *) const;  // Given node, find successor
  BTNode * Predecessor(BTNode *) const;// Given node, find predecessor
 protected:
  // Helper functions
  void Clear(BTNode * &);
  BTNode * Search(BTNode *, const T &) const;
  BTNode * Minimum(BTNode *) const;
  BTNode * Maximum(BTNode *) const;
 private:
  BTNode * root;
```

9. In teams of two and three, generate test case scripts and accompanying expectation check files and post them on Piazza for all class members to use in evaluating their implementations. This means that test cases need to be in place at least 4 hours in advance of the due date/time of the program itself.

10. A list can be sorted by inserting the elements into a binary search tree and then extracting them in an in-order traversal.

    (a) What is the worst-case running time of this algorithm?

    (b) Implement this sorting algorithm using the BST class specified above. Using a variety of inputs and input lengths, compare this sort to the other sorting algorithms implemented in HW01.