CS 457, Data Structures and Algorithms I
Third Problem Set

November 5, 2016

Due on November 16. Collaboration is not allowed. Contact me directly with any questions.

1. (10 pts) Solve Exercise 9.3-7 on Page 223 of your textbook.
2. (15 pts) Solve Problem 9-2 on Page 225 of your textbook.
3. (18 pts) Solve Problem 12-3 on Page 304 of your textbook.
4. (10 pts) Solve Exercise 12.2-8 on Page 294 of your textbook.
5. (20 pts) Solve Problem 13-3 on Page 333 of your textbook.
6. (15 pts) You are given a red-black tree T with 15 internal nodes (nodes that hold key values) that form a *full* binary tree of height 3 (i.e., a full binary tree of height 4 if you include the NIL leaves). Can you assign colors to the nodes so that a call to RB-INSERT(T, z) for *any* new key value $z.key$ will cause RB-INSERT-FIXUP(T, z) to change the color of the root to red before switching it back to black? The initial assignment of colors needs to obey the red-black properties. If such a color assignment exists, then provide a sequence of 15 numbers whose insertion (in that order) would lead to such a tree, along with a figure of the resulting tree. If not, then explain why such an assignment cannot exist, using the fact that the tree needs to satisfy the red-black properties.
7. (12 pts) Provide an algorithm that, given a graph $G = (V, E)$ as input, decides whether G is bipartite. Also, provide a tight bound regarding the worst-case running time of the algorithm (show how you got this bound), and explain why your algorithm is correct.