CS 210 Advanced Data Structures & Algorithms
Fall 2020

Homework Assignment #3

(Due: 12/3/20)

[100 points]

Part 1: Written Exercises

1. **[15 points]** ***(The Selection Problem & Algorithms and Computational Complexity: The Searching Problem)*** **Chapter 8:**
    a. What is the **average-case** time complexity of **QuickSelect**? Explain.
    b. What is the **worst-case** time complexity of **the median-of-medians MMSelect algorithm**? Explain.
    c. What is the **lower bound for searching** on a sorted array of *n* items only by comparison of keys? Explain.
2. **[10 points]** ***(Intractability: P, NP & NP-Complete problems)*** **Chapter 9: Exercise #1, #6 & #15.**
3. **[10 points]** ***(Advanced Binary Search Trees)***
    a. Draw the **AVL tree** that results when you insert items with the keys 4, 10, 3, 6, 5 and 25 in that order into an initially empty AVL tree.
    b. Draw the **splay tree** that results when you insert items with the keys 4, 9, 3, 7, 5 and 6 in that order into an initially empty splay tree.
4. **[15 points]** ***(Advanced M-way Search Trees)***
    a. Draw the **2-3 tree** that results when you insert items with the keys 5, 21, 8, 63, 69, 32, 7, 19 and 25 in that order into an initially empty 2-3 tree.
    b. Draw the **2-3-4 tree** that results when you insert (***one-pass* insertion using *preemptive splitting***) items with the keys 1, 12, 8, 2, 25, 6, 14, 28, 17, 7 and 52 in that order into an initially empty 2-3-4 tree.
    c. Draw the **2-3-4 tree** that results when you insert (***two-pass* insertion similar to the 2-3 tree insertion**) items with the keys 1, 12, 8, 2, 25, 6, 14, 28, 17, 7 and 52 in that order into an initially empty 2-3-4 tree.

Part 2: Programming Exercises*

1. **[20 points]** ***(Divide-and-Conquer)*** **Average-Case Linear Time Divide-and-Conquer-based Algorithm for the k-th Smallest Selection Problem (**Chapter 8: QuickSelect Algorithm 8.5 in Exercise #28**) & QuickSort using the median as the pivot (**Chapter 2: Algorithm2.6**).**

   Test your implementation using (1) the test case given in class and (2) your own test cases including best-cases and worst-cases.

2. **[30 points]** ***(Advanced Binary Search Trees)*** **The AVL Search Tree**

   Design and implement the AVL Search Tree (AvlST) ADT. You should include ***insert, search, showAvlST*** and ***showBF***.

   The operation ***showAvlST*** prints the *keys* in the AVL search tree as rotated counterclockwise 90 degrees from its conventional orientation using a "reverse" inorder tree traversal.

   The operation ***showBF*** prints the *balance factors* in the AVL search tree as rotated counterclockwise 90 degrees from its conventional orientation using a "reverse" inorder tree traversal.

   Test your implementation using (1) the test case given in class and (2) your own test cases via *showAvlST* and *showBF*.

*** NOTE:**

- Programming in C++ using Visual Studio
- **Pair Programming**
- Deliverables:

  ✓ **The screenshots of test results for all test cases.**
  ✓ **The Visual Studio project folder.**

*** End of Homework Assignment #3 ***